SIMULATION OF ROBOT SWARMS USING PHEROMONE ALGORITHM

A REPORT SUBMITTED TO THE UNIVERSITY OF MANCHESTER
FOR THE DEGREE OF BACHELOR OF SCIENCE
IN THE FACULTY OF SCIENCE AND ENGINEERING

2024

Chittesh Kumar Singore

Supervised by Dr Clare Dixon

Department of Computer Science

Contents

Word Count: 12345

List of Tables

List of Figures

Abstract

Pheromones are chemical substances released by ants to communicate with each other. Swarm Robotics offers a promising approach for achieving goals which requires distributed intelligence and adaptability. This project is a bio-inspired single virtual pheromone algorithm for swarm robots providing in-direct communication and to achieve goals together. By inducing principles of chemical communication between ants into the robots, they deposit virtual pheromone trails in search of their target exploring their environment. This project also focuses that unlike other foraging algorithms, this has no explicit leader guiding all other robots, depending on a decentralized robot swarm to make efficient decisions. The proposed system supports swarm intelligence by deploying robots and reacting to pheromones simultaneously. It uses single-type 'attractive' pheromone for investigating the swarm's coordination and adaption in challenge tasks. The results show how virtual pheromones can be used to improve robotics swarm efficiency and open the door to more sophisticated uses in automated agriculture[1], environmental monitoring[13], and search and rescue mission[2].

Declaration

No portion of the work referred to in this report has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Copyright

Acknowledgements

I would like to thank my supervisor Professor Clare Dixon. The project was highly influenced by her valuable advice and suggestions, along with her support. I am also grateful for my Family and Friends for moral support.

Chapter 1

Introduction

1.1 Motivation

Inspired by nature swarm robotics harnesses the power of collective behavior representing a viable method for tackling various challenging tasks. Pheromone-based communication is known to be a simpler and more flexible coordination mechanism than the one that exists in other biological swarms. Ants and other insects leave pheromone trails that direct their nestmates and create a distributed memory in the environment that greatly impacts decision-making. Ants and other insects leave pheromone trails that direct their nestmates and create a distributed memory in the environment that greatly impacts decision-making[15].

Compared to other coordinating methods in biological systems, pheromone-based communication proves to be more simpler and has better flexibility . This makes it especially attractive for robotic applications where the capacity to adapt to changing conditions is critical[38]. Moreover, simulations are essential to swarm robotics because they offer a platform for the investigation and improvement of algorithmic techniques before their deployment in real systems[5]. They support both algorithm development and energy optimisation, which is becoming more important given the limitations of real-world applications.

The aim of the thesis is to use pheromone algorithms to simulate robot swarms and to understand the relationship between swarm behaviour and pheromone-mediated communication therefore developing a realistic tested by incorporating virtual pheromones and observing their distribution, evaporation, and robot detection. The simulations proposed helps in the study of various pheromone tactics and examine the outcomes due to the changes in parameters such as evaporation time and diffusion rate on the swarm's capacity for exploration, foraging, and adaptation to changing environmental conditions

Moreover the aim also includes to find the most efficient pheromone-tuning techniques for real-world situations with limitations like poor communication or unstable terrain . Lastly, the thesis focus on improving swarm coordination algorithms, which could result in increased effectiveness in a variety of applications, such as focused medical interventions using nanoscale robots, environmental monitoring, and search and rescue operations.

The term "swarm intelligence" was dedicated to a group of cellular robotic systems in the 1980s[3]. Although the phrase was originally associated with robotic applications, it has now expanded to include a wide range of study. From social insect studies to optimisation algorithms which has also occasionally lost its direct correlation with robotics. The application of swarm intelligence ideas to physically embodied robotic systems has been referred to as "swarm robotics" in recent years[35].

Swarm robotic systems can be used for the tasks which too laborious for humans to perform, like distributed sensing, environmental monitoring, disaster response, and area exploration. These systems are made to function with little control and direct communication, depending instead on indirect cooperation mechanisms like stigmergy, which is an example of environmental communication demonstrated by ant pheromone trails[38]. To summarise, completing challenging tasks can be done without the need for human intervention because swarm robotics provides a dependable, and adaptable method for managing large groups of robots by utilising emergent behaviours, resulting in effectiveness in a variety of applications, including focused medical interventions using nanoscale robots, environmental monitoring, and search and rescue operations.

1.2 Aim

This project aims to:

1. Design, develop and evaluate a simulation of robot swarm using pheromone-based foraging algorithm with an emphasis on adaptation and energy optimizations.

2. Compare and analyze the effect of pheromone-based and different algorithms on the swarm behaviour.

1.3 Objectives

The objectives of this project are:

1. Research, understand the state-of-the art swarm robotics and intelligence[12].

2. Examine how pheromone algorithms function in both artificial and natural settings.

3. Divide the project in structured phases, from foundational elements to complex algorithms like random and pheromone-based algorithms.

4. Choose the right tools and technologies for simulation.

5. Design, develop and understand the working of the differential drive robots work in the chosen simulator.

6. Implement the random walk of the robots

7. Implement the collision avoidance mechanism of the robots (among themselves and the walls, corners)

8. Design and Develop the simulation of the random foraging robots according to the Bristol Robotics Laboratory model[27].

9. Design and Develop the simulation of the pheromone-based foraging robots according to the University of Lincoln model[25].

10. Experiment and Evaluate the simulation of foraging robots which uses the random and pheromone-based algorithms against the experiments performed in [27, 25] respectively.

11. Compare the performance of the proposed pheromone-based algorithm by inspiring from the experiments performed in [25].

12. Compute and Compare the resulting net energy consumption by random and pheromone-base algorithms.

13. Visualize the experiments to assess the growth and development of the algorithms, and examine robot behaviour.

1.4 Outcomes

The project was successfully able to implement single virtual pheromone foraging algorithm to a swarm of robots.

1.5 Report Structure

- **Chapter-1 Introduction:** Establishes the focus of the project, outlines the motivation behind the work, and defines aims and objectives.
- **Chapter-2 Background:** Presents essential background information on swarm intelligence, swarm robotics and different algorithms including pheromone-based.
- **Chapter-3 Design:** Details the design of the pheromone simulation system, including DRUL system, and mechanisms for pheromone deposit and decay. Outline methodologies for robot decision-making behaviour.
- **Chapter-4 Implementation:** Bridges the gap between concept and reality.
- **Chapter-5 Experiment and Evaluation:** A series of experiments designed to investigate and analyze the effectiveness of pheromone-based foraging and random foraging algorithms
- **Chapter-6 Conclusion:** Summarizes the key finding of the project and reflect on the success in achieving the outlined objectives of the project
- **Chapter-7 Future Work:** Suggests potential areas of future research and development in this area.

Chapter 2

Background

2.1 Introduction to Swarm Robotics & Intelligence

Swarm Intelligence is a field which is inspired by the natural swarming behaviours in species such as ants, bees and fishes. The core principle of swarm intelligence is well explained in the book review of "Swarm Intelligence: from Natural to Artificial" [4, 39] by Barbara Webb where it is said that simple agents following simple rules and "collective intelligence" of the group can solve complex behavioural problems.

A very important question can be asked why and how social insects inspire for problem solving. It is addressed in the book [4], which claims the critical concepts are *self-organization* and *stigmery*. The idea of Self-organization is some kind of global coherence in structure of behaviour can emerge purely from local interaction. Insect swarms provide a potential of self-organising systems with different kinds of local interactions that can be implemented, and seems to be emergent result of local actions. For instance, termites provides a deep understanding of how they build their well-structured nests and ants alternating their activity dynamically to increase colony efficiency. Stigmery provides an important mechanism for self-organization, how agents' have indirect communication helping each others' behaviour. These concepts becomes alternative to more conventional approaches to problem solving. The book [4] covers different example of insect social behaviour such as foraging, task allocation, brood sorting, self-organization and structured nest building and communicative transport, which solves analogous problems such as communication network or collective robotics tasks.

Kennedy and Eberhart (1995) introduced Particle Swarm Optimization (PSO) [21], a pivotal algorithm in swarm intelligence for optimisation of functions for collective exploration of the operational space, inspired by the social behaviour observed in bird flocking or dish schooling. Another way robot swarm is used for a tasks is for foraging, hence, Dorigo and Di Caro (1999) developed the Ant Colony Optimization (ACO) algorithm [10], inspired from the foraging behaviour of ants and demonstrate power of the swarm in finding optimal paths through graph structures solving complex bio-inspired problems.

Swarm intelligence (SI) continues to evolve, with researchers pushing boundaries through new techniques and applications. Researchers are integrating SI algorithms with techniques like machine learning, fuzzy logic and evolutionary computation to create more powerful approaches for complex-problem solving, for instance, a Chaotic PSO algorithm combined with Support Vector Regression (SVR) for enhanced electric load forecasting model [18]. Multi-swarm strategies are being developed for complex multi-objective optimization problem to achieve multiple goals with dynamic environment such as multi-objective particle swarm optimizer [34]. SI is proving increasingly valuable for large-scale problems, particularly in data analysis for clustering and feature extraction [19].

2.2 Foraging Algorithms in Swarm Intelligence

Foraging is a biological metaphor for cooperative behavioural strategies in swarm robotics research. However, some researchers emphasize that competition for space in the environment between the agents affects group's efficiency. Krieger and Billter [22, 23] take a threshold-based approach to allocate their robots to each task: resting or foraging. In their experiment each robot has to be characterized with a different randomly chosen threshold in order to regulate the activity of the team, this strategy aims to mitigate crowding effects and optimize resource collection. Labella, Dorigo, & Deneubourg [24] designed adaptive task allocation mechanism for robot swarms focusing on robots dynamically transitioning between resting and foraging states to improve the performance by adjusting probability of leaving home, however, there is absence of knowledge about other robots. Guerrero and Oliver [16] explored how to efficiently allocate tasks to groups of robots (coalitions) in dynamic, real-time situations, however, it demands of communication between robots which constrains the scalability of the swarm.

Swarm intelligence draws inspiration from natural systems, giving rise to a powerful class of optimization algorithm known as foraging algorithm. Foraging is a biological metaphor for cooperative behavioural strategies in swarm robotics research. The algorithm constitutes of two main principles - exploration and exploitation.

- **Exploration** - Like an ant wandering aimlessly across the floor (arena), this algorithm is used to search for a solution space for potential areas of interest.
- **Exploitation** - Upon discovering a promising region (eg. food source for ants) algorithm shifts concentration on maximising the yield of the discovered solution.

The balance between the two is essential for optimizing outcomes in complex and uncertain environments.

Some researchers emphasize that competition for space in the environment between the agents affects group's efficiency. Krieger and Billter [22, 23] take a threshold-based approach to allocate their robots to each task: resting or foraging. In their experiment each robot has to be characterized with a different randomly chosen threshold in order to regulate the activity of the team, this strategy aims to mitigate crowding effects and optimize resource collection. Labella, Dorigo, & Deneubourg [24] designed adaptive task allocation mechanism for robot swarms focusing on robots dynamically transitioning between resting and foraging states to improve the performance by adjusting probability of leaving home, however, there is absence of knowledge about other robots. Guerrero and Oliver [16] explored how to efficiently allocate tasks to groups of robots (coalitions) in dynamic, real-time situations, however, it demands of communication between robots which constrains the scalability of the swarm.

Communication plays a vital role in mimicking foraging behaviour. Whether through the intricate waggle dance of bees or pheromone trails of ants, information sharing helps the swarm collectively focus its effort on the most optimal resource [36, 17]. Algorithms integrates mechanisms for agents to communicate findings and collaboratively guide the search process which can be direct (explicitly exchange of data), or indirect (resembling stigmery), where interaction with the environment shape subsequent actions[11]. Foraging algorithms strive to capture various resilience of dynamic environment, integrating mechanisms that identify when exploited solution is no longer optimal and intelligently trigger a resumption of exploration. This enables to tackle problem where the target solution itself shifts and evolve under changing conditions.

Foraging algorithms has diverse range of applications from logistical to analytical tasks. Routing and Path Optimization, it streamlines the routes for delivery networks, optimize data flow within communication systems and address complex challenges like Travelling Salesman Problem. The algorithm provides technique for resource allocation and scheduling, intelligently assign tasks, distributing resources, and managing timeline for efficient use in situation with constraints. With data analysis, foraging algorithm power data clustering to group similar data points within massive dataset to reveal insightful patterns. Moreover, multi-swarm foraging models explore collaboration and competition, seeking increased efficiency when problem solutions require resources beyond single swarm models.

When several robots work together to complete the forage task, the group's performance should be enhanced. Information sharing (stigmery as discussed above) and labour division are two instances of the cooperative tactics[41]. Ant colonies exhibit division of labour, with each member carrying out a task if a stimulus reaches a predetermine threshold. The Bristol Robotics Laboratory foraging model[27] uses division of labour technique. The robots continue to explore until the exploring time limit has been archived. Even if they are unable to locate food, they return to the nest to rest and recharge. They resume exploring when the resting time is over. With this technique robot are able to save more battery energy and avoid overcrowding.

2.3 Pheromone Algorithm

Among the many ways to solve problems found in nature, the intricate communication mechanisms of social insects stand out as powerful models for optimization. Ant colonies demonstrate remarkably simple local interactions among individuals leading to emergence of efficient collective behaviours. Pheromone Algorithm, a prominent communication methods used in swarm intelligence with core principle of ant communication - laying and following of pheromone trails - into a computational framework for far-reaching applications in robot swarms. Modeled after pheromone markings left by ants to guide their others, pheromone algorithm utilizes indirect communication known as stigmery. Robots within the swarm interact with the environment by depositing “virtual pheromones”. These digital markers, analogous to the scent trails laid by ants, encode information about paths traveled or discoveries made for other robots that influences their behaviour often biasing their movement decisions towards the more strongly concentrated paths.

Pheromone algorithm incorporates crucial mechanisms to mimic natural processes of evaporation and diffusion. The processes of evaporation is time-sensitive for the decay of pheromone markers serving two main purposes; preventing outdated information from misleading the swarm ensuring there is no dominance of established routes in the solution space, and promotes continuous exploration as gradual fading encourages to explore more area.

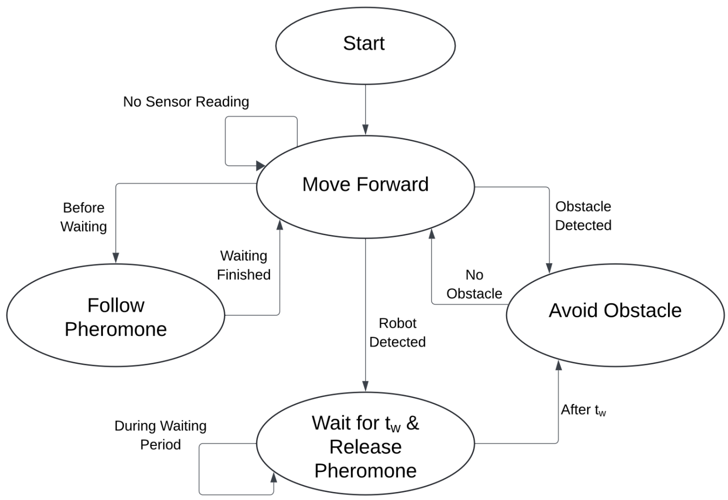

Figure 2.1: State Diagram of General Pheromone Algorithm[30]

Pheromone algorithm come in two primary variations: single-pheromone and multi-pheromone systems.

- Single-pheromone Systems have been utilized extensively in the simulation of ant behaviour, particularly in the Ant Colony Optimization (ACO) algorithms [11]. These models mimic how ants deposit a single type of pheromone to communicate indirectly and find the shortest paths to food sources. The algorithm establishes a singular communication medium, allowing for straightforward implementation in computational algorithms. While this system is simple and effective for certain tasks, often lack the complexity needed for dynamic and varied environments. They can be limited in their ability to handle multiple simultaneous objectives or adapt to changing conditions, as they rely on a singular type of signal.

- Multi-pheromone Systems introduces a richer communicative landscape, where robots can deposit distinct pheromone types, enabling them to communicate a wider range of information. The system helps in more complex situations like task allocation to multiple targets, ability to distinguish between different types of resources or assigned specialized roles withing the swarm. Algorithm like DACO was created to make better and enhanced robot swarm for optimization [42]. By depositing unique pheromone "scents", robots leave a multifaceted informational trail for other robots in the swarm which empowers the swarm to make more sophisticated decisions. With this system, swarms can pursue multiple goals simultaneously, and teams of robots might be guided by distinct pheromones towards targets of varying importance with search and rescue scenario. Furthermore, the robots can prioritize and distinguish high-value resources to optimize the collective effort.

Pheromone algorithm might be carefully implemented, attention must be paid to parameters such as rate of pheromone deposition and evaporation, and how strongly robots react to pheromone concentrations. If evaporation is too slow, the swarm might stagnate, hindering its ability to adapt to changes and potentially leading to premature convergence on suboptimal solution. Consequently, if evaporation is too fast, the swarm's memory may be insufficient to guide them efficiently towards the best solution.

Research into pheromone algorithm remains a dynamic field. Pheromone algorithms are increasingly integrated with evolutionary techniques [29], or using machine learning to fine-tune pheromone depositions and evaporation rates, predict promising paths. Additionally, recent advancement have explored the use multiple pheromones, where different types of pheromones serves diverse roles from foraging to alarm systems. The introduction of multiple-pheromone communication system (ColCOSϕ) [25] marks a significant step forward. This system enhance the complexity and efficiency of swarm behaviours, offering more nuanced interaction patterns and decision-making capabilities. This system promotes the release of two different types of pheromones: attractive and repulsive, to assist the pheromone tracking and avoidance feature of the robots, and also alert the robots about any warning. We will explore a similar system to ColCOSϕ system and its implication in greater details in the following chapter.

2.4 Computation of Net Energy Consumption

The simulation of foraging robots, the efficiency of the swarm may be objectively evaluated by calculating the net energy. This metric measures the equilibrium between the energy used for foraging and the energy obtained from the things foraged, reflating the swarms overall energy dynamics.

Definition of Net Energy

Net Energy, written $E(n)$, is the difference between the total energy the robots used to forage and the total energy they obtained from the food they found. It is computed at each time step. The net energy energy sheds light on the long-term sustainability and effectiveness of the swarms foraging approach.

Formula for Net Energy

The net energy at each time step is calculated using the following equation:

$$E(k+1) = E(k) + E_p(k+1) - E_c(k+1) \tag{2.1}$$

where;

$E(k)$ is the net energy at time step k,

$E_p(k+1)$ is the energy produced at time step k+1,

$E_c(k+1)$ is the energy consumed at time step k+1.

Calculation of Energy Produced The quantity of foraged food items found during a time step is directly correlated with the energy produced during that phase which is represented by $E_p(n)$. It computed as follows:

$$E_p(n) = F(n) \cdot E_f \tag{2.2}$$

where;

$F(n)$ is the number of food items foraged at time step n,

E_f is the energy value of each food item.

Calculation of Energy Consumed Whether the robots are in active or resting condition determine how much energy they use, which is represented by $E_c(n)$. It computed as follows:

$$E_c(n) = E_r \cdot N_R(n) + \alpha \cdot E_r \cdot (N - N_R(n)) \tag{2.3}$$

where;

E_r is the energy consumed by a robot is a resting state per time step,

N is the total number of robots in the swarm,

$N_R(n)$ is the number of robots resting at time step n,

α is a coefficient representing the increased energy consumption when robot are active (typically $\alpha = 1$).

Integrating all the components, the updated formula for the net energy consumption will be:

$$E(k+1) = E(k) + F(n) \cdot E_f - (E_r \cdot N_R(n) + \alpha \cdot E_r \cdot (N - N_R(n))) \tag{2.4}$$

Therefore, this formula helps to keep track and optimize the swarm's energy efficiency over time. It becomes an important criteria to evaluate different foraging algorithms and can be seen in Chapter-5.

Chapter 3

Design

This chapter discusses the design of the simulation of robot swarms using pheromone-based and random foraging algorithm as per the objectives. The concepts introduced in the previous chapter outlines the requirements for the simulation.

3.1 Design in Phases

The behaviour of each robot is designed based on different requirement in the objectives of this project. The whole project was divided into four main phases as mentioned below. Each phase was experimented with certain criteria to examine the performance of the robot swarm as explained in Chapter-5.

- Phase-1: Designing, implementing and evaluating basic robot design and features like differential robot drive, collision avoidance,etc.
- Phase-2: Designing, implement and evaluate random foraging algorithm inspired from Bristol Robotic Lab model[27].
- Phase-3: Design, implement and evaluate single-pheromone based algorithm inspired from the system by Lincoln University[25].

Individual robot behaviour is modeled using a probabilistic finite state machine (as depicted in figure 2.1). Simulation requirements representing the project's objectives were carefully analyzed to determine the core classes needed: Robot Initialization, State, PID Controller, Robot Controller and the Pheromone Algorithm. The project utilizes objected-oriented design. Arrays are employed for the efficient storage and management of multiple robots and food items.

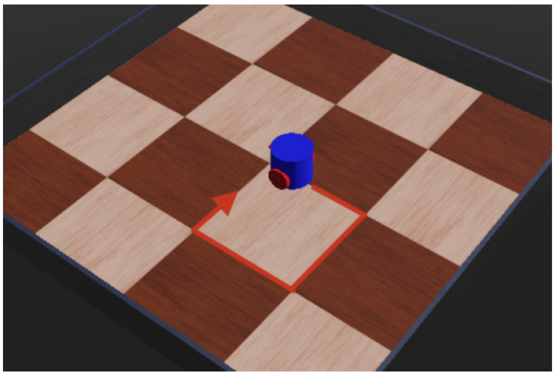

Figure 3.1: Differential Drive Robot Moving in Polygon Shape

ADD UML DIAGRAM HERE

3.1.1 Phase-1: Foundation

This phase lays groundwork for the robotic development journey. The journey focuses is to learn the design and implement the fundamental aspects of robot's design and functionality. Using this phase, the core features were established empowering the robot with:

- Mobility - used to design and implementation of a differential robot drive system[6]. The system provide the robot with the ability to navigate its environment effectively.
- Collision Avoidance - the robot was also programmed to detect and avoid obstacles in its path with appropriate sensors, to ensure safe operations.

The robot, programmed to move in square shape (1m x 1m), exhibited deviations from its intended path several (as shown in the figure). This pattern is observed when the robot's movement is in *open loop system*[28]. In open-loop system, the robot lacks real-time feedback about its position and changing environment leading to inaccuracies overtime. Closed-loop system addresses this issue by incorporating sensors for continuous feedback. This allows the robot to adjust its course in real-time, ensuring it stays on the desired path despite potential environmental changes or accumulated errors. This experiment with basic implementation of differential drive robot provided valuable insights. It helped to gain a deeper understanding of the limitations of open-loop systems and necessity of the closed-loop systems for precise robot navigation. It was beneficial for learning the principles of differential drive robots, including the

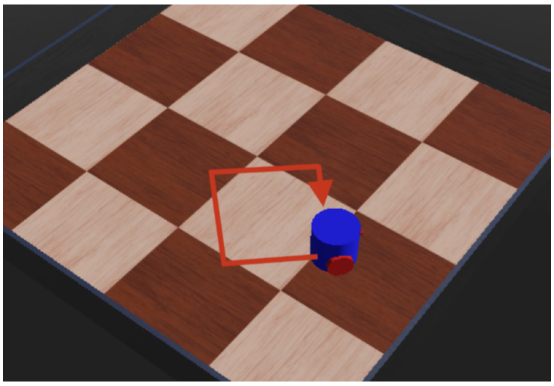

Figure 3.2: Differential Drive Robot in Open-Loop System

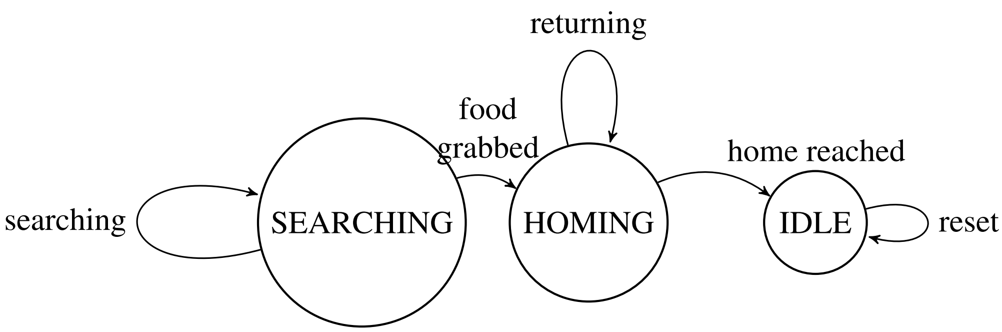

Figure 3.3: State Machine Diagram For Random Foraging

calculations deciding the movement and rotation. This phase established the groundwork for implementing timed maneuvers, which are essential for obstacle avoidance and complex path planning.

3.1.2 Phase-2: Random Foraging Algorithm

This phase is inspired from foraging algorithm is based on probabilistic finite state machine for swarm foraging developed by Bristol Robotics Laboratory [27], which uses a simpler state machine for random foraging (figure 3.3). The robot operates within an environment where obstacles are present and utilizes a combination of reactive and goal-oriented behaviours.

The algorithm: The algorithm centered around three core behaviours: searching, food detection, and homing. Algorithms 1,2 and 3 provides pseudocode for each component of this algorithm.

- *Searching*: In this primary state, the robot employs a random movement pattern. To prevent immediate collisions, it leverages distance sensor reading from eight direction. These sensor readings are incorporated into pre-defined coefficient table to adjust motor speeds dynamically.

Algorithm 1 Pseudocode for Searching

```
Initialize the supervisor and sensors
Set time step from simulation environment
Initialize the and able all sensors
Initialize motor and set initial motor position and velocities
Define initial state as SEARCHING
while True do
    Reset actuator values
    Get sensor inputs
    Blink LEDs to indicate activity
    Map current robot trajectory
    Calculate movement speeds based on sensor input
    Update robot state based on current conditions
    if robot state is SEARCHING then
        if wall or cliff is detected then
            Adjust speeds to avoid obstacles
        end if
        Check if food is grabbed
    end if
end while
```

- *Food Detection*: The robot is equipped with camera to continuously capture the environment. This function enables the robot to detect and grab the target resource (food in this case).

Algorithm 2 Grabbing Function Pseudocode

```
function GRABBING
    is_red ← GETCAMERAINPUT
    if is_red and 65000 < number_pixels < 517140 then
        senses_food ← true
        print ”seeing food”
    else
        senses_food ← false
    end if
    print ”FOOD GRABBED”, senses_food
    return senses_food
end function
```

- *Homing* : Upon detecting food, the robot transitions from searching to homing. A vector, to determine the difference between its current position and the starting location, is calculated to point the robot back towards its nest (home). If the distance falls below specific threshold, the robot transitions to an idle state and stop its motors.

Algorithm 3 Homing Function Pseudocode

function HOMING
    start_position ← GETROBOTPOSITION
    current_position ← GETROBOTPOSITION
    print "start", $start_{p}osition$
    print "current position:", *current_position*
    distance_to_start ← DISTANCETOHOME(*current_position*)
    print "distance to start:", *distance_to_start*
    if $0.05 \leq round(distance\_to\_start, 2) \leq 0.06$ **then**
        print "Reached home. Stopping!"
        Set motor velocities to 0
        exit()
    else
        print "Not yet at home."
        if $round(\text{angle\_to\_start}, 2) > 0.1$ **then**
            TURNLEFT
        else
            TURNRIGHT
        end if
        GOFORWARD
    end if
end function

This random foraging algorithm demonstrates a simple but effective approach for robots to locate resources within an arena and subsequently return to a designated home base. Experiment have been conducted on a few criteria which have been discussed in Section 5.2.

3.1.3 Phase-3: Foraging Using Pheromone Algorithm

WRITE ABOUT DataSSS This phase is inspired from multi-pheromone system ColCOSϕ developed by the researchers at the University of Lincoln [42]. The Single-Pheromone algorithm is designed to release single type of 'attractive' pheromone comprehend the ColCOSϕ system, trying to simulate complex behaviours observed in nature such as those exhibited by ants when foraging for food. The algorithm is embedded within robotic control system that employs a blend of artificial techniques and sensor feedback to navigate and accomplish task effectively. Table 3.1 shows why localisation of virtual pheromone values can be considered for a multi-pheromone system.

Substance	**Method**	**Multiple Pheromone**	**Description**
Optics	1. Localization system and projector[37], 2. Localization system and screen[30]	Possible Yes	1. Unstable to ambient light, 2. Controllable, stable , flexible to modify parameter of evaporation, diffusion
Data Information	1. IR communication[33], ✓ 2.Localization based Virtual Environment[25]	Possible	1. Cannot implement all the properties of pheromone, 2. Pheromone information stored elsewhere, No direct interactions with robots.
Chemical substances	Ethanol [14]	Yes	Not very controllable, impractical for micro-robots.

Table 3.1: Comparison Of Implementation of Virtual Pheromone[25]

Pheromone algorithm can be divided into four key steps involved in information exchange to guide robots in a swarm - *Deposit, Sense, Adjust Behaviour, Evaporate*. Following is the explanation of the these steps with a strategy of following the higher concentration of pheromone;

1. **Deposition** The pheromone-deposit function (as illustrated by algorithm 4)plays a vital role in simulation how individual robots can leave virtual "pheromone trails" within the arena as they explore. Robots initiate in a default SEARCHING state, depositing pheromones exclusively during active exploration to ensure the immediate relevance of their markings. The Algorithm 4, shows the pheromone gets deposited behind the robots as they move and gets stored using two data structures: pheromone grid for an individual robot and a global pheromone grid for entire team of robots (refer table 3.2). Furthermore, the implementation employs two pheromone "intensities", every third deposit utilizes

a stronger pheromone marking. This potentially adds dynamic elements to the signaling system, allowing robots to emphasize certain paths or location.

A movement detection mechanism has been implemented for preventing the deposition at stationary position. Only when a robot has moved a significant distance (0.0015 m) it leaves a pheromone trace. The state of the pheromone map is periodically written to a file '*pheromone_grid.txt*'. This serves a crucial purpose for analysis and visualization during the research process. It allows for the observation of how pheromone deposition patterns change over time and their influence on robot behaviour.

Algorithm 4 Pseudocode for depositing pheromone based on robot state and movement

```
function DEPOSIT_PHEROMONE(robot_name, robot_pos)
    Initialize or open a file for logging pheromone data
    if robot's state is SEARCHING then
        pheromone_intensity ← exploration_intensity
    else if robot's state is HOMING then
        pheromone_intensity ← homing_intensity
    end if
    current_time ← retrieve system or simulation time
    previous_position ← retrieve position from robot_positions[robot_name]
    if significant movement from previous_position to robot_pos then
        new_entry ← (pheromone_intensity, robot_pos, current_time)
        Update pheromone_grids[robot_name] with new_entry
        Log "Pheromone deposited by robot_name at robot_pos with intensity pheromone_intensity at time current_time"
    else
        Log "No significant movement detected for robot_name, no pheromone deposited"
    end if
    Save the updated pheromone grid to file
    return updated pheromone grid for robot_name
end function
```

2. **Sensing Pheromones**]

The robots can actively sense and process the pheromone markings of others at the heart of effective coordination using the algorithm 5. Algorithms 4 and 5 are interlinked to embody the design principles enabling this vital behaviour. The algorithm findNeighbours intelligently filters pheromone deposits, emphasizing

Table 3.2: Comparison of Shared Pheromone Grid vs. Individual Pheromone Grids

Feature	Shared Pheromone Grid	Individual Pheromone Grids
Objective	Collaborative information sharing	Individual perception and memory
Pheromone Grid	Single grid updated and sensed by all robots	Separate grids assigned to each robot
Robot Interaction	High degree of cooperation and coordination among robots	Robots exhibit more independent behavior
Usefulness	Scenarios where collective effort and coordination are crucial	Scenarios requiring unique exploration or decision-making by each robot
Robot Autonomy	Lower individual autonomy due to reliance on shared information	Higher individual autonomy with personal perception and decision-making
Exploration/Decision-Making	Based on collective sensing and mapping	Based on individual robot's sensors and algorithms
Data Structure Format	Nest Dictionary with keys as robot names and values as an array mapping time steps to tuples of intensity and position	Similar Nest Dictionary structure with individual record for each robot, allowing independent updates and access.

those that are both nearby and recent. This focus on 'fresh' pheromones promotes coordination in a dynamic environment. The algorithm $sense_pheromone$ utilizes the findNeighbour function to gather relevant deposit comparing with current pheromone level. A *DRUL (Directing Robot Using Localisation) Pheromone Sensing System*, implemented to provide robots within the swarm with localized pheromones (neighbours), has been integrated to the $sense_pheromone$ algorithm which classifies them as 'front', 'back', 'left', and 'right' to expand the potential for sophisticated decision-making.

3. **Adaptive Robot Behaviour**

   A central theme in the proposed multi-robot system is the ability of individual robots to dynamically adapt their behaviour in response to a variety of sensory data. A key design principle is the differentiation between 'SEARCHING' and 'HOMING' states, ensuring the robot engages in the appropriate decision-making based on its current objective. While in 'SEARCHING' state, the robot analyzes pheromone data, identifying the direction with the strongest pheromone

Algorithm 5 Pseudocode for sensing pheromone by a robot

```
function SENSE_PHEROMONE(robot_name, robot_pos)
    Initialize an empty list to store nearby pheromones
    pheromones_nearby ← []
    for all entries in pheromone_grids[robot_name] do
        position ← entry.position
        intensity ← entry.intensity
        if distance(position, robot_pos) ≤ sensing_threshold then
            Append entry to pheromones_nearby
        end if
    end for
    if pheromones_nearby is not empty then
        Analyze pheromones to determine direction and strength
        Make decisions based on pheromone information (e.g., follow path)
    else
        Continue current behavior or switch to search mode
    end if
    return pheromones_nearby
end function
```

intensity and can follow two different strategies as explained in the section later. The two strategy enables the robots to follow their teammates for task accomplishment or keeps exploration for an efficient foraging. When the robot is in 'HOMING' state and withing a close range of its starting position, it engages in finer adjustments to ensure accurate final positioning.

A PID (Proportional-Integral-Derivative) controller[43] calculates precise steering adjustments, allowing direct navigation toward either the target source (food) or its starting location. Additionally, sensor readings are integrated to modulate these speeds, ensuring the robot gracefully navigates around obstacles in its environment.

4. **Evaporation** Inspired by the behaviour of pheromones released by ants, the evaporation algorithm 7, introduces crucial element of decay. The algorithm has simple mechanism with profound implications for the overall dynamics and emergent collaborations patterns within the robot team. The core principle is the gradual fading of pheromone signals over time. It considers time elapsed since a pheromone has been deposited, along with customizable evaporation rate (0.1 seconds). Initially, it employs a mathematical exponential decay model (see equation below) to ensure that the intensity of a pheromone diminishes over

Algorithm 6 Adjust Robot Behavior Based on Sensed Pheromones

```
function ADJUST_ROBOT_BEHAVIOR(sensed_pheromone, robot_name)
    pid_controller ← initialize PID controller with appropriate parameters
    max_intensity ← 0
    max_direction ← None
    for each direction, values in sensed_pheromone do
        for each value in values do
            if value[1][0] > max_intensity then
                max_intensity ← value[1][0]
                max_direction ← direction
            end if
        end for
    end for
    v, turn_adjustment ← PID_UPDATE(pid_controller, target_parameters)
    if max_direction =' left' then
        adjusted_left_speed ← decrease speed to turn left
        adjusted_right_speed ← increase speed to assist turn
    else if max_direction =' right' then
        adjusted_left_speed ← increase speed to assist turn
        adjusted_right_speed ← decrease speed to turn right
    else
        Set both speeds to v (forward movement)
    end if
    Adjust speeds based on obstacle proximity using distance sensors
    Set motor speeds to adjusted_left_speed and adjusted_right_speed
    return adjusted_left_speed, adjusted_right_speed
end function
```

time, with the older pheromones having a progressively weaker influence on the robots behaviour.

In the proposed pheromone-foraging system, the decay factor (known as evaporation rate) plays a crucial role in determining how long the trail lasts and how the robots make decision.

Algorithm 7 Evaporate Pheromone Intensity Over Time

function EVAPORATE_PHEROMONE($robot\_name, elapsed\_time, intensity, evaporation\_rate$)
  $decay\_factor \leftarrow e^{-evaporation\_rate \times elapsed\_time}$
  $decayed\_intensity \leftarrow intensity \times decay\_factor$
  Update pheromone grid for $robot\_name$ by setting intensity to $decayed\_intensity$
  print "Updated intensity after evaporation: ", $decayed\_intensity$
  return $decayed\_intensity$
end function

Formula for exponential decay:

Given:

- Initial pheromone intensity I
- Evaporation Rate r
- Elapsed Time t since pheromone was deposited

The decayed intensity I_{new} after time t can be calculated using:

$$I_{\text{new}} = I \cdot e^{-r \cdot t}$$

3.2 Two Approaches to Pheromone Algorithm

The first approach to pheromone-based navigation gave priority to well-traveled paths. This methods is predicated on the idea that pheromone traces serves as trustworthy marks, directing the robots towards location most likely to hold resources. The advantage is that there is a chance for greater productivity, quicker goal completion and saves energy.

However, there is a great deal of appeal to a different approach. Consider a situation where robots upon coming upon a pheromone trail, they would go along their selected router rather mindlessly copying. This allows the robot two evaluate a new trail with "fresh" pheromones. Benefits of exploring new paths includes faster exploration, resource mapping and redundancy to find alternative routes or resources in some abrupt situations.

With the time constraint, the project could only focus on trail-prioritizing pheromone strategy. However, the potential benefit of a more distributed approach may be useful and efficient.

3.3 Breadcrumb Technique

Inspired by the Breadcrumb concept, pheromones released will be released in distinctive manner: every third value has a higher pheromone intensity to provide a clear navigation for robots within the arena. Moreover, this concept helps in pheromone decay mechanism by gradually dissipating pheromone in such as way that the higher-intensity pheromones fade slower than the lower-intensity to provide information of the pheromone trail.

3.4 Design Challenges & Limitations

Challenge 1: Virtual Grid Representation. The initial design included a virtual grid environment for robots and the simulation platform where as Webotes does not provide built-in grid representation in its code. The experiment was conducted with different grid sizes in a 2D array to solve this. Identifying an appropriate size that stoped the value overflowing,therefore allowing for smooth robot movement within the simulated arena.

Challenge 2: Pheromone Sensing and the DRUL System The aim is to build the DRUL system which have feature for navigation strategy that allows for seamless movement. However, the early implementation had drawbacks. The observations indicated that Left and Right movements were not always adequately recognised due to the design of the grid and how robots navigated it.

Solution 1: Use a 2D array. Effective way to describe the virtual grid environment was to maintain the 2D array notion. This enabled accurate tracking of pheromone intensity at each position in the arena.

Solution 2: Refine the DRUL system. By taking a more refined approach to pheromone sensing, the DRUL system's capacity to recognise Left and Right movements may be enhanced.The robot might detect pheromone levels not just in its current location but also slightly ahead in adjacent grid cells (diagonally or on the sides).

Solution 3: Using a Hash Map for Pheromone Deposition Formalisation: Hash Maps are excellent for storing and retrieving key-value pairs quickly. In this case, the value would be the current pheromone strength at that point, and the key would be a unique identifier indicating the robot's position inside the grid (such as its X and Y coordinates).

The advantages of hash maps: Every position has a unique key linked to its pheromone value guarding against data overwriting. Based on the robot's present position (key), effective access to particular pheromone concentrations are offered by hash maps called effective retrieval.

Chapter 4

Implementation

The chapter bridges the gap between concept and reality. Firstly, it explores the technological toolkit selected to bring the design to life. It then dives into the practical implementation phase, where the system's functionality was built in accordance with the requirements, design principles, and foraging algorithms (random & pheromone) previously discussed.

4.1 Technologies

Balancing a focus on novel algorithms with the need for a suitable development environment posed an early challenge. The core focus was on foraging algorithms (pheromone & random), collision avoidance, random walks, and experimentation. The key decision was to decide which simulating software to be used.

Early in the implementation phase, a key decision which was accompanied by the exploration of existing simulation software. The goal was to identify a platform that would accelerate development, allowing a greater focus on the core foraging and pheromone algorithms. While simulators like Gazebo, NetLogo, Unity, etc offered potential benefits to facilitate the development of the foraging algorithm and its integrated pheromone-based communication system, ***Webots*** was ultimately chosen due to its simplicity, beginner friendliness, ease of customization, features and community support [8]. The other advantage of using this simulator is realistic robot movement and realistic 3D rendering already, also allows users to create multiples worlds.

However, the lack of experience with the simulator, as well as the amount of time dedicated to learning how to use it would be the main disadvantage of this approach.

Conversely, developing the simulation from the ground up using a familiar programming language would necessitate a substantial investment in building foundational elements.

Table 4.1: Comparison of Simulation Software

Software	**Short Description**	**Advantages**	**Disadvantages**
NetLogo[40]	Agent-based modeling environment for simple simulations and conceptual exploration.	Easy to learn, large community, good for abstract or high-level models.	May lack realism or physical accuracy needed for complex robotics simulations.
Webots [8]	Robot simulator with physics engine, focused on realistic robot behavior.	allows rapid prototyping with a vast library of pre-defined robots (eg: E-puck[7]), sensors and actuators, supports programming languages: Python or C/C++ or JAVA.	Steeper learning curve than NetLogo, potentially less specialized for swarm behavior.
Gazebo [31]	Powerful robot simulator with advanced physics, excellent for detailed and realistic environments.	High realism, extensive sensor support, can handle complex robot models.	Very complex, steep learning curve, didn't support Python.
Unity[20]	General-purpose game engine, can be adapted for simulations.	Ultimate flexibility, visually stunning environments possible.	Overkill for most robotics simulations, significant time investment, requires strong programming skills.

The decision was made to develop the simulation in Python programming language. This choice leverages the language's familiarity, simple syntax, and strong community support.

4.1.1 Class Types in Webots

Webots provides two main controller classes for robot simulations: ***Robot*** and ***Supervisor***. While Robot class provides basic functionality for regulating a robot's behaviour within the simulation, the Supervisor class grants new degree of control over the entire simulation environment.

Table 4.2: Webots Supervisor and Robot Classes

Class Type	Description
Supervisor Classes	
Supervisor	Main class to access the simulation world and nodes.
Field	Interact with fields of nodes in the simulation world.
Robot Classes	
Robot	Base class for creating and managing robot controllers.
Motor	Control the motors of the robot.
DistanceSensor	Measure distances from the robot to other objects.

4.2 The Simulation World

4.2.1 The Environment - Arena

The arena was designed to influence robot behaviour presented an unexpected challenge: robots became trapped in a repetitive turning loop at the sharp corners. Upon approaching a corner, robots would attempt to turn, sense the adjacent wall as a new obstacle, and initiate another turn. Understanding the importance of the environment, the arena was redesigned to mitigate the issue of robots getting stuck. The arena of size 2m x 2m with walls (as shown in fig. 4.1) was used for wall-avoidance training for behavioural optimization.

To address the corner navigation problem, a hexagonal arena design was considered. While Webots offers ability to adjust the size of its rectangular arena, a hexagonal arena might introduce complexities because;

- Webots primarily focuses on building robots within its environment. Creating shapes like hexagon might require workarounds or external tools, which was time-consuming compared to modifying the built-in square arena.
- Webots' physics engine is optimised for rectangular environments.

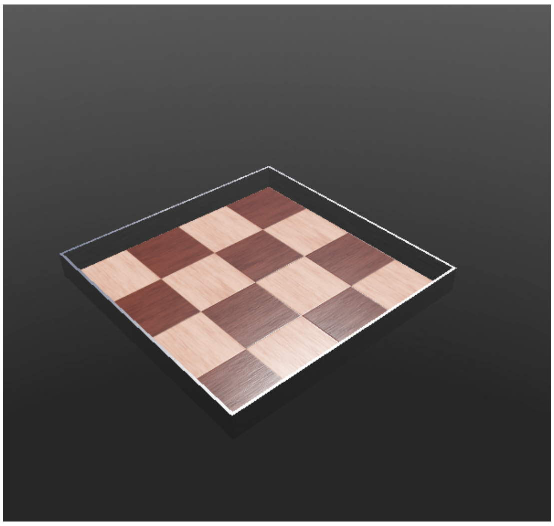

Figure 4.1: Arena

Thus by increasing the size of the rectangular arena and an improved collision avoidance mechanism addressed the problem of corner navigation issue, as the introduction of this mechanism provide robots more space to maneuver successfully. Experimental simulations with this mechanism validated this improvement, demonstrating a significant enhanced navigational abilities for the robots.

The choice of shape and size of arena highlights the intricate relationship between the simulation environment and robot behaviour.

4.2.2 Robot, Sensors and Nest: Building Blocks of Simulation

The Robot

The simulation utilizes the built-in sample robot - E-puck robot - a popular mobile robotics developed by GCtronic, to investigate swarm behaviour in a foraging context. The e-puck's blend of physical gestures and simulation capabilities makes it a valuable tool for this research domain. The e-puck's combination of physical components and robust simulation support make it uniquely well-suited for this domain of study. Physically, the e-puck's suite of sensors, including infrared proximity sensors, a camera, microphones, and an accelerometer, provides the foundation for environment perception. Each sensor modality could potentially be employed for tasks like obstacle avoidance, object recognition, or even sound localization. Furthermore, the robot's differential drive system, powered by stepper motors, grants precise movement control. This control is essential for navigating complex environments ad achieving the

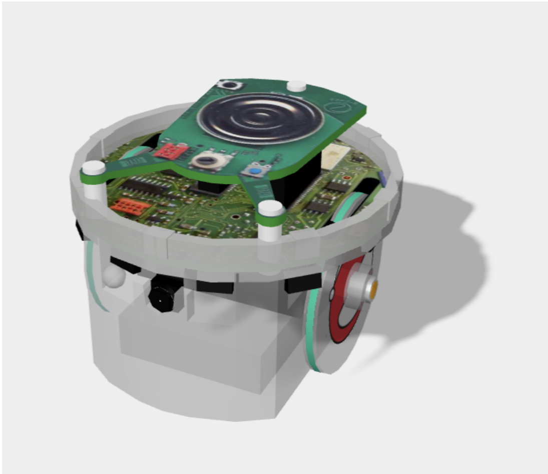

Figure 4.2: E-Puck Robot [7]

fine grained coordination often required on swarm scenarios. The e-puck came with its source file named $E_puck.proto$, .proto is an extension used in Webots to allows users to extend the set of nodes by adding their own nodes to any object.

Importantly, Webots provides a highly realistic simulation model of the e-puck robot. Thus, e-puck becomes the perfect fit for testing swarm behaviour because;

- The e-puck's pre-built nature shifts the focus towards core research on swarm algorithms and emergent behaviours.
- The Webots model balances realism with computational demands, enabling realistic simulations of multi-robot swarm scenarios.
- The e-puck's expansion ports preserve flexibility for future research, potentially exploring advanced sensing, robot-to-robot communication, or even morphological adaptations.

Collision Avoidance System: The e-puck robot employs a combination of specialized sensors and reactive behaviours for collision avoidance.

1. *Edge Detection*

   To prevent itself from falling off edges, it utilizes a set of ground sensors. These sensors detect changes in the ground surface beneath the robot. A function named `cliff_detected()` processes this sensory data; if a sensor reading falls below a specific threshold (< 100), it signals a potential drop-off, triggering an evasive maneuver.

2. *Wall Detection*

   For navigation around walls and other obstacles, the e-puck relies on eight distance sensors placed in different directions (as shown in fig. 4.3). These sensors emit infrared light and measure its reflection to gauge proximity to objects. A function named `wall_detected()` function analyzes these sensor readings; if any reading exceeds a predefined threshold (> 80), the robot infers that a wall is present and initiates avoidance behaviours.

```
def cliff_detected(self):
    for i in range(self.GROUND_SENSORS_NUMBER):
        if self.ground_sensors[i] and self.ground_sensors[i] < 100.0:
            return True
    return False

def wall_detected(self):
    # Check if any distance sensor reading indicates a wall
    for i in range(self.DISTANCE_SENSORS_NUMBER):
        if self.distance_sensors_values[i] > 80.0:
            return True  # Wall detected
    return False  # No wall detected
```

Figure 4.3: Code for Cliff and Wall Avoidance

3. *Movement-Based Collision Avoidance*

   One of the interesting mechanism included in the collision avoidance system is that the robot's movement patterns themselves contribute to obstacle avoidance. The random movement loop (fig 4.4) incorporates distance sensor readings to subtly bias the robot's trajectory. The closer an obstacle appears, the more significant its influence on the robot's speed. This results in the robot turning away from potential collision. A array of coefficient are use to fine-tune this behaviour, potentially allowing more complex steering responses.

```
self.coefficients = [[0.942, -0.22], [0.63, -0.1], [0.5, -0.06],  [-0.06, -0.06], [-0.06, -0.06], [-0.06, 0.5], [-0.19, 0.63], [-0.13, 0.942]]
#random movement of the robot (kind of avoids obstacles)
for i in range(2):
    robot.speeds[i] = 0.0
    for j in range(8):
        robot.speeds[i] += robot.coefficients[j][i] * (1.0 - (robot.distance_sensors_values[j]/ (1024/2)))
```

Figure 4.4: Code for Movement-Based Collision Avoidance Mechanism

Overall, the collision avoidance system demonstrates a resource-efficient design. It leverages the robot's sensors reactive movement adjustment capabilities for navigation, minimizing the need for computationally expensive path planning or obstacle mapping algorithms.

Sensors

The robot possesses a diverse sensor suite (figure 4.5), empowering it to perceive and interact with its environment;

- Motion Tracking: The *'left wheel motor'* and *'right wheel motor'* drive the robot's movement. Coupled with *'left wheel sensor'* and *'right wheel sensor'*, the robot can achieve precise motion control and estimate its position overtime.

- Obstacle Detection and Navigation: The robot relies on an array of eight proximity sensors (or distance sensors) (*'ps0'* to *'ps7'*), providing short-range awareness of its surroundings.

- Light Sensors: Eight light sensors (*'ls0'* to *'ls7'*) can be used to identify light sources, offering further navigational cues.

- LEDs : Multiple LEDs (*'led0'* to *'led9'*) offers visual signalling communication with other robots or the environment.

- Camera: The robot includes a *camera* to expand its sensory capability for object recognition, color-based tracking, or more complex analysis.

- Ground Sensors: Specialized ground sensors (*'gs0'*, *'gs1'*, *'gs2'*) enable the robot to detect changes in the ground surface, a critical safety feature for fall prevention and integral to the previously explained Collision Avoidance System.

Figure 4.5 shows how the sensors are arranged around the robot's body. As a future improvement, each robot can have multiple sensors including GPS sensor in order to gather more information about the position of the robot and its environment, to navigate more efficiently throughout the arena.

IMAGE HIGHLIGHTING SENSORS IN THE ROBOT WITH BOXES

Nest

The home base for robots is known as nest, represented by the area where each robot starts foraging. The pheromones are deposited by each robot after 5 secs of starting..... After foraging the robots return back to the home base (nest) and as they reach its perimeter, they progress to resting state. The robot carries the knowledge of its starting position.

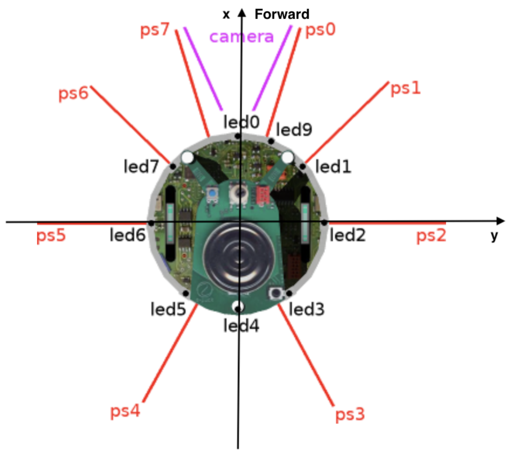

Figure 4.5: Sensors In E-Puck Robot [7]

The Target

The target source (or food) are represented by static bodies which have random positions and shape of a cube (0.1m x 0.1m x 0.1m). They are constrained to be inside the arena and not overlap with the nest.

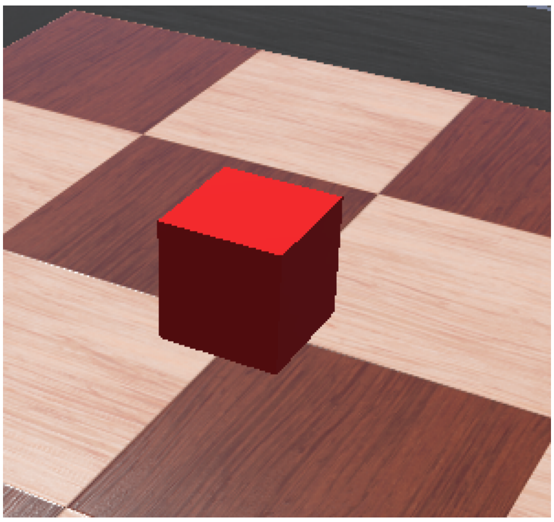

Figure 4.6: Screenshot of Static Target Source Food

4.3 The Simulation

The simulation builds a world with core components such as an arena, robots, and food for dynamic interaction. The key difference between phases is how the robots are controlled, Phase-2 uses random foraging, while Phase-3 implements pheromone-based control.

Firstly, the arena is established with the boundaries and constraints within which the robot operates. Its size, layout, and the potential presence of obstacles influence foraging strategies and the overall flow of the simulation. Robots, the primary agents in this simulation, are equipped with movement capabilities tailored to the environment. Their integrated sensor suites are essential for gathering information. Actuators enable actions such as resource collection and manipulation. Importantly, the robots' sensors and logic governing their decision-making processes are tightly intertwined within the contest of foraging task.

In phase-3, robot utilizes pheromones as their primary communication medium within the group of robot. The algorithm governs the robots' state-based behaviours, determining how they transition between searching, depositing, and homing, based on prior events (pheromone trails, target detection) in the previous time step. The step() function progresses the information of the simulation, while sensor data and pheromone detection plays a pivotal role in guiding the robots' actions.

4.3.1 Visualizing Robot Behaviour

To better understand the complex interactions within the simulated environment, the project incorporates visualization tools that illustrate both robot trajectories and pheromone concentrations. A *mappingTrajectory* function has been created to track each robot's path, operating real-time. At each time step, the robot's current position is retrieved and its coordinates are recorded. These coordinates are then used to progressively construct a line graph depicting the robot's entire movement history. The plot provides a dynamic and evolving picture of robot's search patterns, area of focus, and potential navigational patterns. Moreover, a *mappingConcentration* function made to focus on visualising the evolving pheromone landscape. The function stores data about pheromone deposition in a text file to create heatmap revealing the intensity of pheromones deposited across the arena. The real-time graphs reveal emergent patterns, laying the groundwork for in-depth analysis within Chapter-4 Experiments.

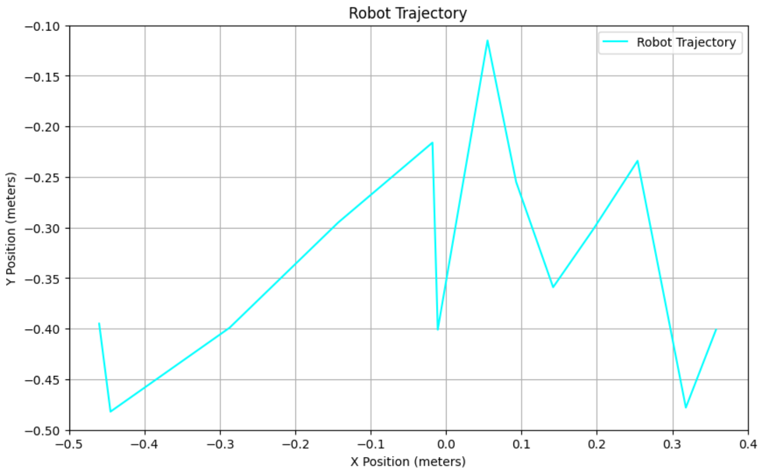

Figure 4.7: Robot Trajectory Recorded

4.4 Summary

The foraging simulation's practical implementation is described in detail in this chapter. Webots was chosen after a thorough assessment of simulation tools due to its user-friendlies and robot customisation. When designing the e-puck, sensors, and arena ,the surrounding environment had a big impact on the algorithms. Python was used to create the simulation included all the behaviours for the robot swarm. Understanding the dynamics of the simulation made easier by the seeing the sample robot trajectory. Now, the next chapter will discuss the experiments and evaluations for random and pheromone-based algorithms.

Chapter 5

Experimentation And Evaluation

After discussing the implementation procedure in the previous chapter, this chapter presents a series of experiments designed to investigate and analyze the effectiveness of pheromone-based foraging and random foraging algorithms in robot swarm focuses. Several key factors will be analysed, ranging from arena configuration to algorithm design and its metrics, to determine their impact on foraging performance. In order to assess the performance of the robot swarm, the results will be compared to those achieved in [27, 25] respectively.

The simulation was evaluated against the objectives stated in section 1.3. The setup of simulation with all the components is same as discussed in the previous chapter containing a bounded arena with food item (static body), robots with appropriate sensors and obstacles. The experiments evaluates algorithms mainly based on net energy consumption (using formula from Chapter-2) of robot swarm along with other parameters, and compares the results from the referred papers to understand the implementation of the algorithms.

The supervisor class has been used for e-puck robots to have a better control over the environment as explained in section 4.1.1.

5.1 Experiment: Impact of Arena Shape and Size

This section discuses how initial experiment with a default 1m x 1m rectangular arena revealed a limitation of robots frequently became stuck in corners while lacking explicit wall detection functionality. This posed obstacle for evaluating core foraging algorithms developed in each phase.

To solve corner conflicts, different options were explored and one of them was a hexagonal arena. While Webots documentation offer guidance, its complexity ultimately proved excessive for the project's scope. A sample world with hexagonal arena within Webots was discovered but encountered implementation constraints and errors. To prioritize advancement without unnecessary customization, I employed a two-pronged strategy was implemented:

1. Arena Resizing: The rectangular arena's dimension were doubled to 2m x 2m, creating a more spacious environment and reducing the frequency of corner collisions.

2. Algorithm improvement: The collision and wall detection algorithm (detailed in Section 4.2.2) were included. This enhanced the robots' capacity to navigate and avoid obstacle regardless of the arena shape.

While a hexagonal arena might theoretically minimises corner conflicts even more, its implementation complexity posed a barrier to rapid prototyping and algorithmic assessment. Table 5.1 discusses why modified rectangular arena was considered to be

Arena Shape	**Size**	**Advantages**	**Disadvantages**	**Rationale**
Rectangular	1m x 1m	Simple & default	Regular corner clashes	Barriers to algorithm assessment
Hexagonal	Any	✓ Diminished corner clashes	Difficult to execute, mistakes made	Implementation exceeded planned scope
✓ **Modified Rectangular**	2m x 2m or above	✓ More spacious, algorithm-focused	Default shape	✓ Prioritizes rapid prototyping and testing

Table 5.1: Table: Impact of Arena Shape and Size

utilized for the project.

5.2 Bristol Robotics Lab Foraging vs. Random Foraging

5.2.1 Experiments by Bristol Robotics Laboratory (BRL)

The Bristol Robotics Laboratory (BRL) study[27] conducted experiments using the Player/Stage sensor-based simulation tool. The foraging task was carried out by eight robots. Each of them had three bumper sensors for detecting collisions with other robots or the arena's edges, three light sensors for determining the best path back to the nest, and a camera for detecting food items placed at random in the arena. The trails adjusted the resting duration for each robot. It was gradually increased by 40 seconds at a time, from 0 to 200. The experiment lasted 20000 seconds for each single resting time value and was repeated ten times [27]. The primary results gained after conducting the studies and will be utilised for comparison were:

- The increase in net energy over time is approximately linear.
- The resting time parameter effects the net energy value. In the studies, the value net energy increased as the resting direction increased to 160 seconds, but declined as the resting time extended to 200 seconds. As a result, given the other simulation parameter, the best resting value was 160 seconds.

Limitation in Reproducing Experiments by Bristol Robotics Laboratory

A few factors make it impossible to exactly reproduce the experiments conducted in Bristol Robotics Laboratory, making the comparison more difficult:

- The robots used in this project is different than from those utilised. Different set of senors are used in the experiments by BRL that covers a wider range and posses more refined collision detection capabilities.
- The extensive duration of the BRL experiments, each lasting 20,000 seconds and is repeated 10 times, allowing more thorough assessment. The project's timeframe and resource does not support the extended duration of testing.

5.2.2 Experiments for Random Foraging Algorithm

The Random Foraging Algorithm developed in phase-2 of this project, while inspired from the algorithm by the Bristol Robotics Laboratory, incorporates different design elements. Hence, the results may differ widely due to these algorithmic variations. Nonetheless, the comparison sets a valuable benchmark for testing the implementation and evaluating with different parameters. The performance of the robot swarm simulated in this project is measured by using the same net energy computation method as the Bristol Robotics Laboratory.

With the experimental limitations, it is intended to conduct experiments comparable to those mentioned above. While differences in some parameters may produce an incomparable final net energy, the comparison seeks to determine qualitative and similar conclusions. Furthermore, this project's trials do not account for robot resting time; net energy consumption is computed exclusively based on the energy consumed during the food collection and return operation.

Example setup of the environment: A modified rectangular arena is used with size 2m x 2m with tile sized 0.5m x 0.5m (as shown in Fig. 5.1). The arena contains two food sources (red static bodies) and a couple of obstacles (wooden box).

The simulation ran for 12,000 seconds to assess the swarm's efficiency over time and to compare the growth rate of net energy to that of the Bristol Robotics Lab studies. Table 5.2 displays the settings utilised to run the experiment. Figure 5.2 depicts the graph of the swarm's net energy against time while performing the simulation with parameters comparable to those from BRL[27]. It can be seen that the graph is not linear because the parameter were chosen to show that the algorithm is successful in foraging task.

The analysis of the graph 5.2 states that the net energy decreases gradually at some instances. For example, there is a steep drop in the net energy at 9032nd second. This shows that the exploration time is too long. As a result, robots are more likely to crash with one another during that period. Collision results in same amount of energy consumption as searching for food. This is fixed in phase-3 while implementing pheromone-based algorithm, this experiment can use learning that these robot collision is also a problem thus fixed it.

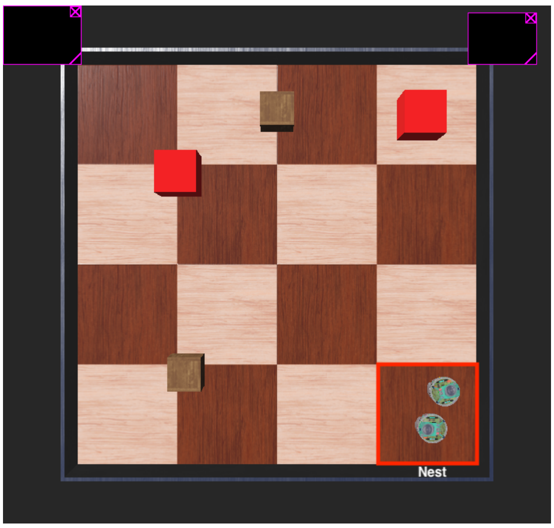

Figure 5.1: Example Setup: Two Static Food Bodies, Two Wooden Boxes (as obstacles) and Robot Nest

Parameter	Value
Arena Width	5m x 5m
Number of Robots	8
Number of Food items	10
Nest Size	0.5m x 0.5m
Energy Consumed for each state	10 units
Energy Provided by Food	2000 units
Time Exploring	200 seconds

Table 5.2: Parameters For Random Foraging Simulation

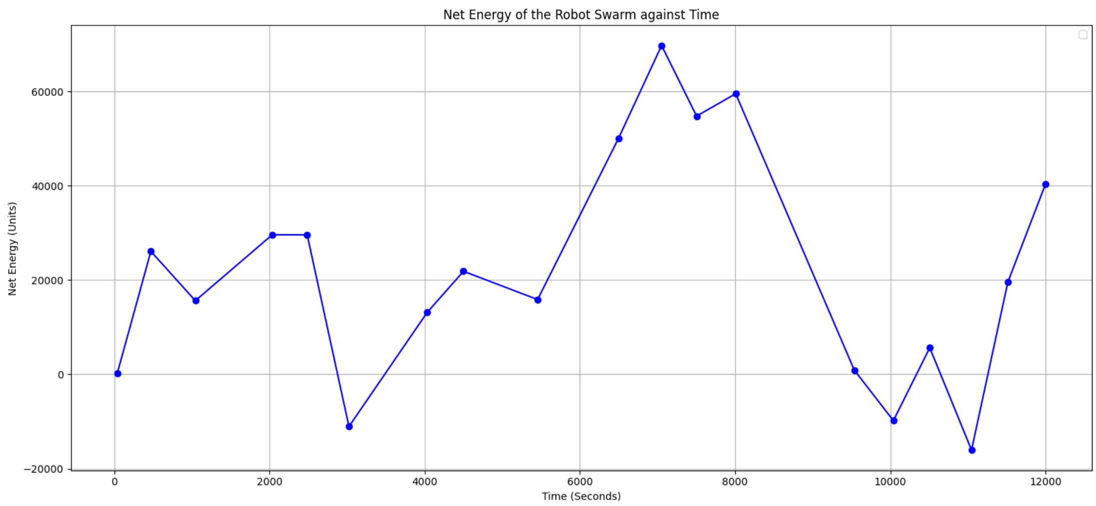

Figure 5.2: Net Energy of the Robot Swarm against Time

5.3 ColCOSϕ System vs. Proposed Pheromone-Based Foraging

5.3.1 Experiments by University of Lincoln (UoL)

The University of Lincoln proposed ColCOSϕ[25], an artificial multiple pheromone system, to support swarm intelligence research by allowing numerous robots to deploy and react to multiple pheromones simultaneously. The ColCOSϕ system simulates pheromone communication in a controlled virtual environment, providing a fresh technique to investigate swarm intelligence. A large LCD screen serves as a platform for miniature e-puck robots equipped with colour sensors to identify ‘pheromone’ trails exhibited on the screen. An above camera detects robot movement and allows them to leave their own pheromone traces. This arrangement replaces actual chemical with optical signals, allowing for fine control and manipulation of numerous pheromone kinds while replicating real-world behaviours like evaporation.

A series of experiments were performed to verify the efficiency and performance of the ColCOSϕ system[25].

- Experiment-1: Pheromone Emulation and Platform Release

  This experiment was designed with three robots with different IDs. Each robot releases on of three different types of pheromones (coloured: red, blue and green) in a constant time interval of 3 seconds.

  Parameters: Pheromone field resolution = 1920 × 1080, PC metrics (for 3 robots) = 1107ms and 219MB memory, time delay resolution = 1920 × 1080.

  Results: The proposed system simulates the pheromone releasing function and the dynamic evaporation and diffusion features correctly using the appropriate equations mentioned in the paper.

- Experiment-2: Pheromone Perception and Robot Reactive Control

  This experiment considers one robot is put on a row of color blocks displayed on the screen, with the colours specified using an 8-bit RGB colour space.

  Results: The findings from the this experiment states that the system features respond to different colours accurately due to LCD’s natural properties with tiny acceptable errors.

- Experiment-3: Gradient Tracking and Avoidance

  This experiment originally keeps the robots in random position at the edge area of the attractive pheromone or within the repellent pheromone field. As a result, the behaviour of tracing the appealing pheromone or avoiding the disagreeable pheromone is triggered. The system showed exact behaviour as expected.

- Experiment-4: Pheromone Trail Following Performance

  This experiment investigates the capabilities of a robot’s ability to follow trails of varying complexity. The researchers designed four distinct patterns: circle, straight line, fork and tortuous path. The performance of the robots were evaluated using position error metric to obtain the accuracy. The experiment showed compelling results of the system's effectiveness and precision in following trails, and suggested that there is a potential to investigate more.

Apart from these experiment a few case studies were conducted to further examine the capabilities of the ColCOSϕ system. The case studies mainly discussed about two major topic; how multiple pheromone improve efficiency of food recruitment task and how different pheromone type (attractive or repulsive) modulate the swarm behaviour. The two case studies demonstrates the ability of the system to replicate natural behaviour. In the first case study, the system successfully managed the release of numerous pheromones, allowing robots to distinguish between them. This reflects the complicated food recruitment tactics by ants. The second case study showed the system's efficacy and stability. The system's exact tracking of robot placements, along with its capacity to instantly update the virtual pheromone trails, allowed it to provide real-time, accurate duration to the robot swarm. Overall, the ColCOSϕ system successfully replicate the pheromone0based communication, which is crucial for swarm behaviour.

A direct comparison between this project's pheromone-based algorithm and the sophisticated ColCOSϕ system is challenging due to multiple considerations.

- The ColCOSϕ system uses a unique setup with a large LCD screen to simulate pheromone trails. Specialised hardware and software are needed for this setup, which are not easily accessible.

- The system makes an extensive and complex experiments that goes beyond project's scope and would have take significant amount of time. These experiment explores a variety of swarm behavior features, such as attractive and repulsive pheromones modulating swarm behaviour.

- The simulation framework does not support simultaneous deposition and detection of multiple-type pheromones as efficiently as the ColCOSϕ system. This limitation affects the ability to replicate the study of attractive and repulsive pheromones.

5.3.2 Experiments for Proposed Pheromone-Based Algorithm

The Pheromone-based Foraging Algorithm developed in phase03 of this project had all improvement learnt in past two phases from learning how a differential drive robot works (phase-1) to making a more improved collision avoidance mechanism (phase-2).

This algorithm is also inspired from the ColCOSϕ system incorporating different design elements. Nonetheless, the comparison sets a valuable benchmark for testing the implementation and evaluating with various scenarios. The performance of the robot swarm simulated in this project is measured by using the same net energy computation mentioned in Background.

With the experimental limitation, the design of the experiments aims to closely replicate the experiments mentioned in the above the paper[25]. Firstly, the net energy consumption is monitored to compare the results from the food recruitment case study from the paper. A modified approach has been used to overcome the challenge of implementing attractive and repulsive pheromones. Inspired by the Breadcrumb concept, pheromones released will be released in distinctive manner: every third value has a higher pheromone intensity to provide a clear navigation for robots within the arena.

Experiment setup of the environment: A modified rectangular arena is used with size 7m x 7m with tile sized 0.5m x 0.5m (as shown in Fig. 5.3). The arena contains two food sources (red static bodies) and a couple of obstacles (wooden box). The simulation ran for 14,000 seconds to assess the swarm's efficiency over time and to compare the growth rate of net energy.

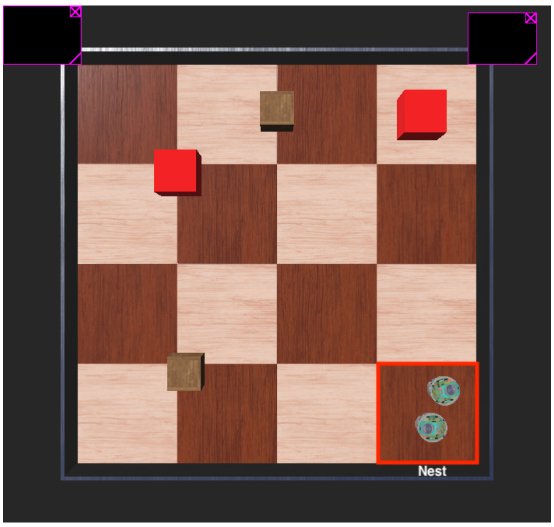

Figure 5.3: Example Setup: Arena=5mx5m, 2 Food items, 2 obstacles and Robot Nest

Net Energy Consumption Over time

Table 5.3 displays the settings utilised to run the experiment. Figure 5.4 depicts the graph of the swarm's net energy against time. It can be seen that the graph is not linear because the parameter were chosen to show that the algorithm is successful in foraging task. However, we can see that it performs better than the random foraging algorithm from phase-2.

The analysis of the graph states that the net energy decreases gradually at some instances. For example, there is a steep drop in the net energy at 8000th second. This shows that the exploration time is still long due to large size of the arena. With the graph, it can also be concluded that the collision from phase-2 has been improved drastically. However, the performance can be improves by adjusting the DRUL system as discussed later.

Parameter	Value
Arena Width	7m x 7m
Number of Robots	10
Number of Food items	15
Nest Size	1m x 1m
Energy Consumed for each state	10 units
Energy Provided by Food	2000 units
Time Exploring	400 seconds

Table 5.3: Parameters For Pheromone-based Foraging Simulation

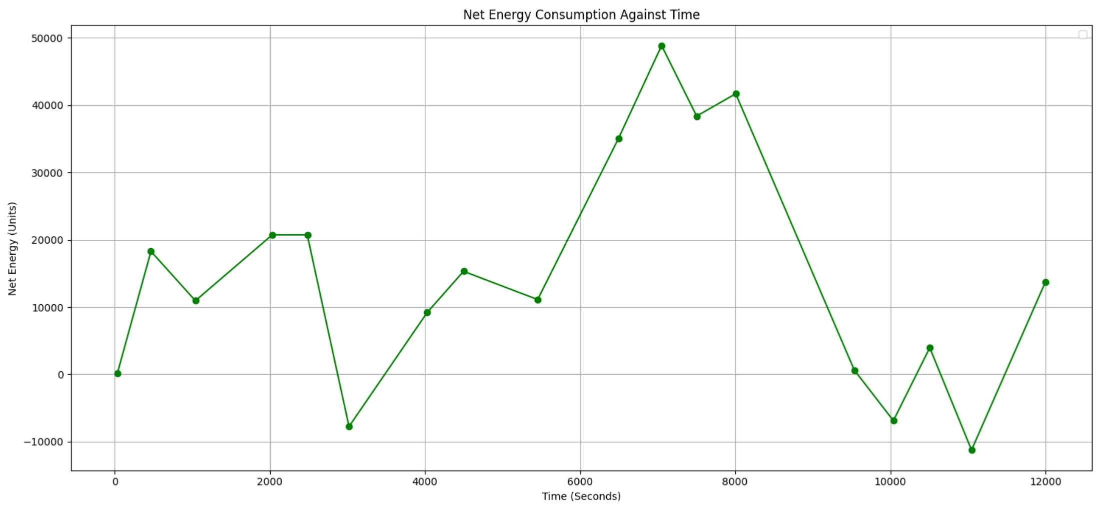

Figure 5.4: Net Energy of the Robot Swarm against Time

Pheromone Sensing and Decision-Making

This experiment investigates the effectiveness of the DRUL (Direction Robot Using Localisation) system for sensing pheromones in a grid-based environment. The system is critical for decentralised decision-making, enabling each robot to detect the direction of high-intensity pheromone and respond to it. This experiment establishes a framework for further exploration of the robots' ability to use these sensed pheromone levels for successful navigation, which will be examined in the following experiment.

After running the simulation for 14,000 seconds, the results (or output) indicates that pheromone concentration at current, front and back positions is successfully recorded. At the simulation's beginning, pheromone levels in positions on the left and right were not recorded. This shows that the robots' initial random movement patterns and lack of contact with any other pheromone trail.

For instance, during the early exploration phase, at timestamp of 9.12 second, a robot 'e2' sensed high pheromone concentration of 20.0 units both in its current and front cells of its position:

```
{'current': [[9.12, (20.0, [0.366, -0.440, -6.41e-05])]],
 'front': [[9.12, (20.0, [0.366, -0.440, -6.41e-05])]],
 'back': [[9.056, (1.0, [0.369, -0.441, -6.41e-05])]],
 'left': [], 'right': []}
this is highest, no change needed
```

Another instance by the series of the outputs captured over consecutive time steps shows higher pheromone concentration getting selected, as shown below:

```
...
{'current': [[8.736, (20.0, [0.383, -0.447, -6.44e-05])]],
 'front': [[8.736, (20.0, [0.383, -0.447, -6.44e-05])]],
 'back': [[8.672, (1.0, [0.385, -0.448, -6.45e-05])]],
 'left': [], 'right': []}
this is highest, no change needed
...
```

These results demonstrate the robot's behaviour in reaction to pheromone levels and a confirms that the DRUL system's role in directing the robots to places with higher pheromone concentrations. The lack of data in the 'left' and 'right' cells was observed which results in more energy consumption, and it was hypothesised that as the experiment advanced and additional robots deposited pheromones, these sensors would offer data that may affect the robots' navigational decisions.

Adaptive Trail Following

This experiment aims to examine the flexibility of robots moving in a pheromone-infused environment. The focus shift towards understanding whether robots can recognise pheromone trails and decide whether to follow them, becoming more central to the discussion. Most importantly, this experiment shows how robots may dynamically adjust their behaviour in response to environmental changes by examining whether they can prioritise paths based on pheromone intensity.

While performing the experiment for pheromone sensing, it was also observed that once the pheromone concentrations were calculated, the function for adjusting behaviour considers these results to modify robot's behaviour according to one of the following condition :

- if high-intensity pheromone was detected in current or front cells, no change was needed as the front and current pheromone level may be same.
- if high-intensity pheromone was detected in left or right cells, the speed of one of the wheel was set to maximum while other was adjusted using turning adjustment calculated from PID controller to have smoother turns, according to the direction the robot is about to make the turn.

From the graph 5.4, several key observations were made:

- Robot showed a preference for regions with large changes (red dots), indicating that these places were source of high-pheromone concentration.
- Robot with linear trajectory remained on course when high intensity was detected in current or front cell.
- The PID controller's turn adjustment helped the robot to take smooth turn at abrupt corners.

These observations provides a significant understanding of the robots' decision-making and adaptive capacities in a pheromone-based environment. The results of this study offers a convincing assessment the possibility that pheromone-based navigation techniques can promote effective and flexible foraging behaviour in robot swarms. The additional use of a PID controller results in an extra degree of complexity. After choosing to follow a pheromone trail, this controller dynamically modifies the robot's speed and turning angle. Improving the robot's pheromone trail following behaviour's efficiency and smoothness is the aim of the PID controller.

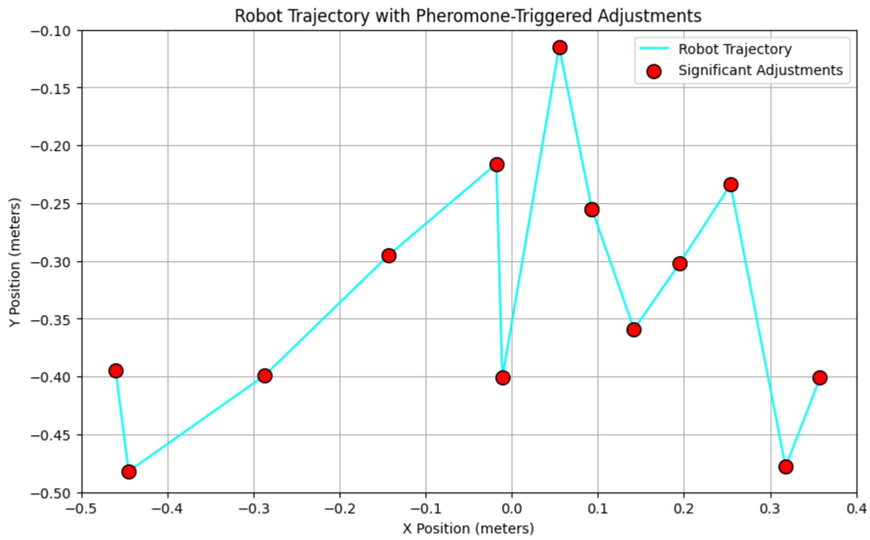

Figure 5.5: Robot Trajectory with Pheromone-Triggered Adjustments Using PID

Pheromone Evaporation Dynamics

This experiment investigates the effects of different evaporation rates of pheromones influencing the navigational strategies and overall efficiency of the robot swarm. In the proposed pheromone-foraging system, the decay factor is inspired by the equation used in [25] to determine how long the trail lasts (more discussed in section 3.2.3).

The evaporation rate may also depend on the size of the arena, if size is too small like 1mx1m with the rate of 5sec then entire arena lead to confusion. Hence, to avoid any confusion and inefficient behaviour a constant arena size (5mx5m) was used and different evaporation rates were used to determine ideal rate.

Evaporation Rate of 2 seconds

- 9 out of 15 food items were collected on average by robots, exhibiting a significant rate of evaporation.
- Robots used 1600 units of energy on average, referring to frequent changes in route due to rapid dissipation of the trails leading to high energy consumption.

Evaporation Rate of 5 seconds

- 12 out of 15 food items were collected on average by robots. Robots could be guided more successfully with more steady trails produced by moderate rate of evaporation.

- Robots used 1500 units of energy on average, suggesting a traversal with fewer unnecessary movement.

Evaporation Rate of 10 seconds

- All food items were collected by the robots, due to slower evaporation rate the trails were persistent for effective foraging routes.

- With persistent and prominent trails, robots showed energy-efficient behaviour and used 1400 units of energy on average.

Therefore, the experiment demonstrated that rapid disappearance of pheromone trails at small evaporation rates (2 seconds) results in inefficient robot behaviour. The swarm's foraging behaviour improves with longer the evaporation rate as seen by the rates of 5 and 10 seconds. The pheromone trail becomes persistent and detectable for longer periods (such as 10 seconds) to ensure most efficient navigation and foraging behaviour.

Pheromone-based robot swarm's foraging efficiency is strongly impacted by the rate of evaporation. It can be concluded that the ideal evaporation rate is the one that keep trails open for just long enough to enable efficient navigation without creating confusion. Within the constraint of this experiment, the evaporation rate of 10seconds showed the most efficient setting for achieving a balance between energy efficient and trail persistence.

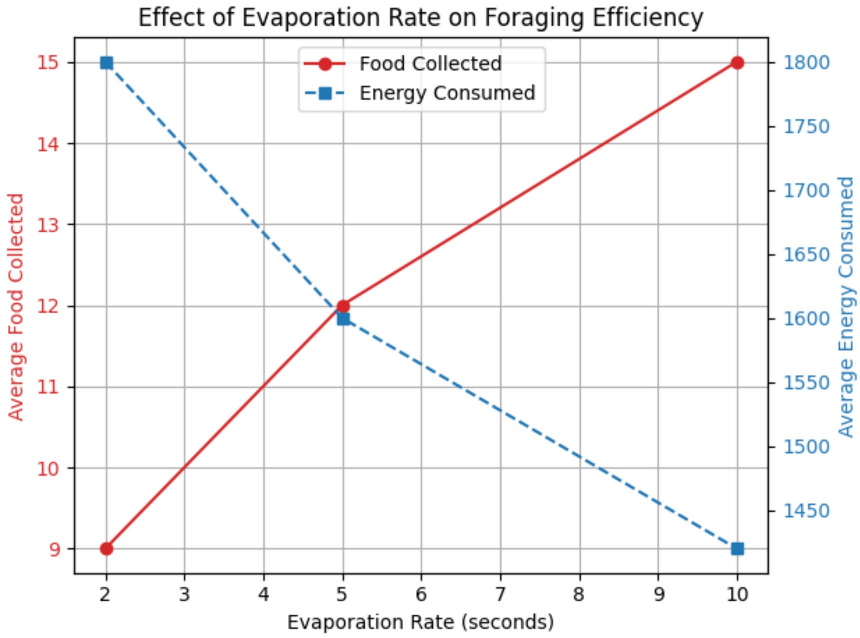

Figure 5.6: Effect of Evaporation Rate on Foraging Behaviour

5.4 Summary

The goal of this section is to assess the project in relation to its original aims and closes the chapter on Experiments and Evaluation. The chapter focused on evaluating pheromone-based and random foraging algorithms with different experiments. It investigates how different elements, including arena configuration, algorithmic metrics, and simulation parameters, affect the effectiveness of foraging. The experiments are designed in a way that makes it easy to evaluate these algorithms by comparing their net energy usage and other performance indicators to benchmarks that have been set by earlier research.

The influence of arena size and shape is one of the main studies, where it was discovered that increasing the arena's dimensions improved algorithm evaluation and decreased corner conflicts. The chapter also explores the research that compare the results with those from Bristol Robotics Laboratory, nothing the constraints resulting from the Bristol studies' longer trial duration and varied robot configurations.

The application and assessment of the proposed pheromone-based foraging algorithm and advanced ColCOSϕ system created by the University of Lincoln are covered in more detail in later parts. A comprehensive analysis of the multi-pheromone system was conducted, a visual signal-based robot communication system, using a range of experimental setups. In comparison to the proposed pheromone-based algorithm, trials showed that the strategy significantly enhances foraging efficiency and robot navigation, especially with optimised evaporation rates.

Interestingly, pheromone-based algorithm illustrates an improved efficiency when comparing the net-energy consumption over time against random foraging algorithm. The figure 5.7 depicts a graph demonstrating that the pheromone-based arrangement uses less energy and more steadily then the more unpredictable and greater energy consumption patterns seen in random foraging.

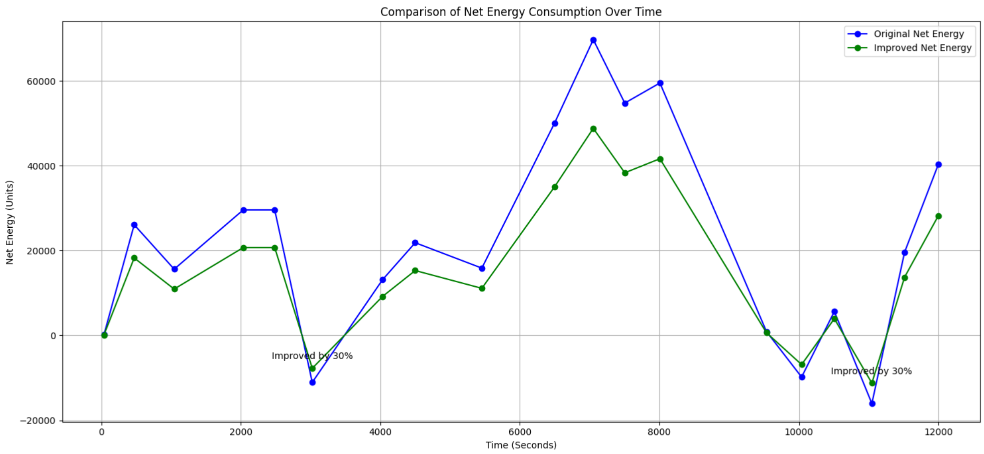

Figure 5.7: Comparison of Net Energy Over Time

The chapter concludes overall that although the pheromone-based algorithm performs better and uses less energy, there are considerable implementation issues. Pheromone-based techniques have the ability to improve the operational performance of robot swarms, as demonstrated by the collective experiments, presents a strong analytical case for adopting them over traditional random foraging techniques.

Chapter 6

Conclusion

In sum up, working on this project has been challenging and rewarding experience that has resulted in substantial academic and personal improvement. It concentrated on simulating a single-type pheromone algorithm to simulate robot swarm. Implementing and testing a bio-inspired pheromone system for indirect robot-to-robot communication to improve foraging capabilities was the project's main goal. Th design, development, and testing of a simulation that showed the useful implementation of the algorithm in a controlled setting allowed for the successful accomplishment.

This project has produced many successes. First off, the project's main accomplishment was the creation of the pheromone-based foraging algorithm, which demonstrated how intricate biological communication techniques might be incorporated into robotic systems. This technique allowed the robots to work together as a swarm by facilitating efficient indirect interactions between them. Second, the project performed exceptionally well in the area of energy efficiency, where several parameters, including diffusion and evaporation rates, were carefully adjusted to maximise the swarm's operating effectiveness and flexibility. Finally, the simulation environment was made possible through the utilisation of the Webots platform. It made it possible to follow and visualise each robot's behaviour in great detail, which gave obvious insights into the dynamics of swarm behaviour under different circumstances.

The pheromone algorithm's development requires a thorough comprehension of biological processes and the computational models that represent them which includes an innovative approach to handling challenging issues. Through this process, I improved my algorithmic design skills and was reminded of the value of approaching research and development methodically. Moreover, it took some time to become proficient with the Webots simulation programme, but it resulted in improving technical abilities and offered a reliable platform for testing and improving the algorithm.

The project's research component presented significant prospects for advancement as well. A greater knowledge of the scientific method was made possible by carrying out experiments and evaluating the results, the importance of empirical evidence and data-driven conclusions was highlighted The experiences helped in shaping research and analytical abilities which have also helped me get ready for future aspirations in the robotics and artificial intelligence fields.

In the long run, the initiative establishes a strong basis for additional swarm robotics research. Future improvements might focus on enhancing the robots' homing and sensing capabilities, including more complex computing frameworks like ROS, and investigating the use of self-healing mechanisms. The limitation has been pushed in the field of autonomous robotic systems as these innovations have the potential to dramatically improve the robustness and efficiency of robot swarms.

At last, this research not only met its initial goals, but also added new insights to the field of swarm robotics by demonstrating the practical uses and benefits of a pheromone-based algorithm. The skills and knowledge gained from this endeavour will surely benefit my future academic and professional endeavours, establishing this project as a watershed moment in my educational career.

Chapter 7

Future Work

To improve the suggested decentralized foraging robot swarm which is guided by indirect pheromone-trail communication, this chapter helps to address crucial aspects of their future development topics. The discussed issues will be resolved and will improve the swarm's effectiveness, adaptability, and resilience in a real-world setting.

Along with its technique of indirect pheromone-trail communication, the decentralized foraging robot swarm described in this paper has great potential to overcome individual errors which is critical to long-term success.

7.1 Optimizing Sensing and Homing Features

7.1.1 Refining the DRUL Pheromone Sensing System

The initial implementation of the DRUL (Direction Robot Using Localisation) system, described in section 3.2.3, which correctly detects pheromone levels at the robot's current location and nearby cells. Despite the fact that the fundamental limitation stems from the existing pheromone deposition approach, in which the robot leaves behind pheromones as they move. Accurate directional guiding is hampered by the intrinsic difficulty of accurately detecting pheromones immediately in front of the robot.

Several possible enhancements should be considered to address this issue and raise the DRUL system's effectiveness. The pheromone deposition patterns should be adjusted inconsideration of forward-facing pheromone placement. Furthermore, investigation for releasing pheromones in a restricted area surrounding the robot instead of merely behind will be examined. One of the other possible improvement is changing the DRUL sensing system's thresholds, and when a faint trail is found, the detection parameters may be adjusted to boost sensitivity.

7.1.2 Improving Homing with Machine Learning

The decentralized swarm's homing method is based on robots tracing their initial location back to a self-deposited pheromone trail that has resulted in outfitting with the Breadcrumb Technique (see Section 3.4). Despite the fact that this method gives fewer useful application, there are a few obvious drawbacks. First and foremost, the pheromones left behind on the homing trip may mask the important pheromone pathways made during the outbound foraging stage. The robot may start to rely too much on its own historical trajectory, which could reinforce less-than-ideal or ineffective routes.

To improve path prediction and optimisation during the homing process, the integration of machine learning techniques should be implemented **[source]**. Path optimisation algorithms can help in forecasting effective courses using the gathered data during both the foraging and homing phases, while reinforcement learning could teach an agent to optimise based on rewards. By suggesting quicker and more efficient pathways, this would enhance the current pheromone system and maximise the swarm's ability to carry resources.

7.2 Integrating ROS Framework

For real-world practical applications, the Robot Operating System (ROS) framework [32] exhibits a compelling future step to move this research forward and to provide considerable advantages for subsequent growth. In spite of the fact that Webots and alike simulation environments can be probably used for a large portion of the initial development of swarm algorithms, ROS offers a well-built framework that speeds up the process and makes new seam capabilities possible.

Webots is a useful tool for designing algorithms, conducting controlled testing as well as building exceptional prototypes. The ability to quickly iterate within Webots is important at the time of early stages of swarm behavior development. Nevertheless, ROS offers a number of benefits in terms of practical application. Moreover, ROS provides numerous benefits in terms of practical application. Initially, the swarm can enable to smooth integration a greater variety of robot platforms, sensors, and actuators due to the hardware abstraction layer. Therefore ensuring that the basic swarm algorithms focus on high-level decision-making, with ROS filling in the details regarding the underlying hardware. This happens because of its modular nature, ROS can divide the functions of the swarm into more manageable and reusable parts. Therefore making it easier to create, test, and maintain the system, especially when adding new features or algorithms.

A custom controller can be used to switch the swarm simulation from Webots to a ROS-based environment, acting as a ROS node[9]. It can assist the simulation and the ROS framework by implementing Webots and ROS libraries. This type of custom controller can introduce complexity to the swarm's structure to provide more flexibility for fine-tuning the swarm's control and communication strategies.

7.3 Introduction of Self-Healing Mechanisms

The research shows robot failures such as those hindering mobility while leaving other systems operational, can drastically disrupt the efficiency of the swarm [source]. To combat these vulnerabilities, it is crucial to integrate robust and effective *self-healing algorithms* into the swarm's behaviour.

Strong self-healing algorithms are essential for survival in distributed foraging swarms, where pheromone trails are the primary means of communication. Studies show that these kinds of swarms are susceptible to disturbances, particularly when it comes to partial robot malfunctions that impair movement without affecting other functions. [r1] The failures caused can be responsible for decreasing the speed and in most cases it can lead to total breakdown of the functioning of the swarm. Therefore, failure detection techniques are integrated to lower the impact which includes tracking the strength of pheromone trails, examining the robots which have stopped working and ultimately form a heartbeat signal among the swarm members. Once the immune system's granuloma formation process is successful, healing mechanisms can be modeled which are inspired by granuloma. This method would permit robots to resolve conflicts locally and enable them to detach themselves. For example, an object found by a defective robot could alert neighbours to finish the task assigned while the ones in close proximity may enhance pheromone deposition to restore impaired pathways. The swarm will begin exploratory activities if a crucial path is lost so that pheromone connections are restored. It is important for researchers to carefully check the balance between the possible communication cost of improved self-healing against the potential benefits of increased response efficiency which occurs from additional communications. [1]

7.4 Optimized Pheromone Strategy

The proposed multi-pheromone algorithm has the potential of significantly boosting the efficiency and adaptability of foraging robot swarms. The algorithm drives inspiration from complex communication networks of social insects such as ants because they use a variety of pheromone types for recruiting, task distribution, and trail construction[26].

Instead of relying solely on single "food found" pheromones, the algorithm would be inspired from complex communication networks of social insects such as ants. Alternatively, the robots would use distinct pheromones to indicate "high-yield foraging area", "low-yield foraging area", and "obstacle detected". This specialization would allow the swarm to quickly prioritize the most productive zones, keep away from distractions, and intelligently navigate obstacles.

To optimizes exploration, a hierarchical approach would be used. With the integration of broad, long-lasting pheromones used for making wide areas of interest and shorter-lived micro-pheromones for precise navigation trails within these zones. Moreover, it's imperative to use pheromones with varying rates of degradation. Backtracking would be discouraged by a temporary " recently explored" pheromone, resulting in find various foraging grounds, although the permanent pheromones may find important locations such as tracks back the base or food caches.

This multi-pheromone strategy offers major benefits to a decentralized foraging swarm. It allows for better task prioritisation, more efficient deployment of swarm members across wide areas, and a dynamic infrastructure that adjusts smoothly to changing discoveries and difficulties in the foraging environment.

7.5 Implementing Second Strategy

– TO BE COMPLETED The second strategy couldn't not be implemented which i want to see its effects on the results. The strategy states that if there are two robots foraging, one robot starts finding food or comes closer to food, so it turns its trail into the trail of "food found trail" . if the second robot comes across this trail (like kind of intersects this) then the robot should infer that the food is found there so no point going there, thus marking its trail as "still searching". By this strategy we will be able to explore in limited time without wasting energy as in two robots going for the same thing.

7.6 Summary

Due to temporal constraints and basic limits, the initial implementations of these mechanisms might not be responsible for every case possible. With all of these enhancements, it is expected that the foraging swarm will be substantially more efficient, adaptable and resilient in real-world-environments.

Bibliography

[1] D. Albani, J. IJsselmuiden, R. Haken, and V. Trianni. Monitoring and mapping with robot swarms for agricultural applications. In *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6. IEEE, 2017.

[2] M. Bakhshipour, M. J. Ghadi, and F. Namdari. Swarm robotics search & rescue: A novel artificial intelligence-inspired optimization approach. *Applied Soft Computing*, 57:708–726, 2017.

[3] G. Beni and J. Wang. Swarm intelligence in cellular robotic systems. In *Robots and biological systems: towards a new bionics?*, pages 703–712. Springer, 1993.

[4] E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press, 10 1999.

[5] M. Brambilla, E. Ferrante, M. Birattari, and M. Dorigo. Swarm robotics: a review from the swarm engineering perspective. *Swarm Intelligence*, 7:1–41, 2013.

[6] Crabbe. Mobile robot kinematics. `https://www.usna.edu/Users/cs/crabbe/SI475/current/mob-kin/mobkin.pdf`.

[7] Cyberbotics Ltd. *GCTronic' e-puck*. Cyberbotics.

[8] Cyberbotics Ltd. Webots: Robot simulator.

[9] Cyberbotics Ltd. *Webots User Guide R2023b: Using ROS*. Cyberbotics.

[10] G. Di Caro and M. Dorigo. *Ant colony optimization and its application to adaptive routing in telecommunication networks*. PhD thesis, PhD thesis, Faculté des Sciences Appliquées, Université Libre de Bruxelles ..., 2004.

[11] M. Dorigo and G. Di Caro. Ant colony optimization: a new meta-heuristic. In *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*, volume 2, pages 1470–1477 Vol. 2, 1999.

[12] M. Dorigo, G. Theraulaz, and V. Trianni. Swarm robotics: Past, present, and future [point of view]. *Proceedings of the IEEE*, 109(7):1152–1165, 2021.

[13] M. Duarte, J. Gomes, V. Costa, T. Rodrigues, F. Silva, V. Lobo, M. M. Marques, S. M. Oliveira, and A. L. Christensen. Application of swarm robotics systems to marine environmental monitoring. In *OCEANS 2016-Shanghai*, pages 1–8. IEEE, 2016.

[14] R. Fujisawa, H. Imamura, and F. Matsuno. Cooperative transportation by swarm robots using pheromone communication. In *Distributed Autonomous Robotic Systems: The 10th International Symposium*, pages 559–570. Springer, 2013.

[15] S. Garnier, J. Gautrais, M. Asadpour, C. Jost, and G. Theraulaz. Self-organized aggregation triggers collective decision making in a group of cockroach-like robots. *Adaptive Behavior*, 17(2):109–133, 2009.

[16] J. Guerrero and G. Oliver. Multi-robot coalition formation in real-time scenarios. *Robotics and Autonomous Systems*, 60(10):1295–1307, 2012.

[17] B. Hölldobler and E. O. Wilson. *The ants*. Harvard University Press, 1990.

[18] W.-C. Hong. Chaotic particle swarm optimization algorithm in a support vector regression electric load forecasting model. *Energy Conversion and Management*, 50(1):105–117, 2009.

[19] T. Inkaya, S. Kayalıgil, and N. E. Özdemirel. Swarm intelligence-based clustering algorithms: A survey. In *Unsupervised learning algorithms*, pages 303–341. Springer, 2016.

[20] A. Juliani, V.-P. Berges, E. Teng, A. Cohen, J. Harper, C. Elion, C. Goy, Y. Gao, H. Henry, M. Mattar, et al. Unity: A general platform for intelligent agents. *arXiv preprint arXiv:1809.02627*, 2018.

[21] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Proceedings of ICNN'95 - International Conference on Neural Networks*, volume 4, pages 1942–1948 vol.4, 1995.

[22] M. J. Krieger and J.-B. Billeter. The call of duty: Self-organised task allocation in a population of up to twelve mobile robots. *Robotics and Autonomous Systems*, 30(1):65–84, 2000.

[23] M. J. Krieger, J.-B. Billeter, and L. Keller. Ant-like task allocation and recruitment in cooperative robots. *Nature*, 406(6799):992–995, 2000.

[24] T. H. Labella, M. Dorigo, and J.-L. Deneubourg. Division of labor in a group of robots inspired by ants' foraging behavior. *ACM Trans. Auton. Adapt. Syst.*, 1(1):4–25, sep 2006.

[25] T. Liu, X. Sun, C. Hu, Q. Fu, and S. Yue. A multiple pheromone communication system for swarm intelligence. *IEEE Access*, 9:148721–148737, 2021.

[26] T. Liu, X. Sun, C. Hu, Q. Fu, and S. Yue. A multiple pheromone communication system for swarm intelligence. *IEEE Access*, 9:148721–148737, 2021.

[27] W. Liu, A. F. Winfield, and J. Sa. Modelling swarm robotic systems: A case study in collective foraging. *Towards autonomous robotic systems*, pages 25–32, 2007.

[28] S. K. Malu, J. Majumdar, et al. Kinematics, localization and control of differential drive mobile robot. *Global Journal of Research In Engineering*, 14(1):1–9, 2014.

[29] M. Mavrovouniotis and S. Yang. Ant colony optimization algorithms with immigrants schemes for the dynamic travelling salesman problem. In *Evolutionary Computation for Dynamic Optimization Problems*, pages 317–341. Springer, 2013.

[30] S. Na, Y. Qiu, A. E. Turgut, J. Ulrich, T. Krajník, S. Yue, B. Lennox, and F. Arvin. Bio-inspired artificial pheromone system for swarm robotics applications. *Adaptive Behavior*, 29(4):395–415, 2021.

[31] Open Source Robotics Foundation. Gazebo.

[32] Open Source Robotics Foundation. ROS.org.

[33] D. Payton, M. Daily, R. Estowski, M. Howard, and C. Lee. Pheromone robotics. *Autonomous Robots*, 11:319–324, 2001.

[34] M. Reyes-Sierra, C. C. Coello, et al. Multi-objective particle swarm optimizers: A survey of the state-of-the-art. *International journal of computational intelligence research*, 2(3):287–308, 2006.

[35] E. Şahin. Swarm robotics: From sources of inspiration to domains of application. In *International workshop on swarm robotics*, pages 10–20. Springer, 2004.

[36] T. D. Seeley. *The wisdom of the hive: the social physiology of honey bee colonies*. Harvard University Press, 2009.

[37] O. Simonin, T. Huraux, and F. Charpillet. Interactive surface for bio-inspired robotics, re-examining foraging models. In *2011 IEEE 23rd International Conference on Tools with Artificial Intelligence*, pages 361–368. IEEE, 2011.

[38] G. Theraulaz and E. Bonabeau. A brief history of stigmergy. *Artificial life*, 5(2):97–116, 1999.

[39] B. Webb. Swarm intelligence: From natural to artificial systems. *Connection Science*, 14(2):163–164, 2002.

[40] U. Wilensky and W. Rand. *An introduction to agent-based modeling: modeling natural, social, and engineered complex systems with NetLogo*. Mit Press, 2015.

[41] A. F. Winfield. Foraging robots. *Encyclopedia of complexity and systems science*, 6:3682–3700, 2009.

[42] Y.-m. Xia, J.-l. Chen, and X.-w. Meng. On the dynamic ant colony algorithm optimization based on multi-pheromones. In *Seventh IEEE/ACIS International Conference on Computer and Information Science (icis 2008)*, pages 630–635, 2008.

[43] U. Zangina, S. Buyamin, M. S. Z. Abidin, M. S. Azimi, and H. Hasan. Non-linear pid controller for trajectory tracking of a differential drive mobile robot. *Journal of Mechanical Engineering Research and Developments*, 43(7):255–270, 2020.