MANCHESTER
1824
The University of Manchester

Deep Reinforcement Learning-Driven Automatic Controller Design for Swarm Robotics

A thesis submitted to the University of Manchester for the degree of
Doctor of Philosophy
in the Faculty of Science and Engineering

2023

Seongin Na
Department of Electrical and Electronic Engineering

Contents

Word count: 31739

List of figures

List of tables

List of publications

1. **S. Na**, M. Raoufi, A.E.Turgut, T. Krajnik and F. Arvin, 'Extended Artificial Pheromone System for Swarm Robotic Applications', Proceedings of the ALIFE 2019: ***The 2019 Conference on Artificial Life***, 608-615, 2019 (*Chapter 3*)

2. **S. Na**, Y. Qiu, A.E. Turgut, J. Ulrich, T. Krajnik, S. Yue, B. Lennox and F. Arvin,'Bio-inspired Artificial Pheromone System for Swarm Robotics Applications', ***Adaptive Behavior***, 29 (4), 395-415, 2021 (*Chapter 3*)

3. **S. Na**, H. Niu, B. Lennox and F. Arvin, 'Universal artificial pheromone framework with deep reinforcement learning for robotic systems', 2021 ***6th International Conference on Control and Robotics Engineering*** (ICCRE), 28-32, 2021 (*Chapter 4*)

4. **S. Na**, H. Niu, B. Lennox and F. Arvin, 'Bio-inspired collision avoidance in swarm systems via deep reinforcement learning', ***IEEE Transactions on Vehicular Technology***, 71 (3), 2511-2526, 2022 (*Chapter 4*)

5. **S. Na**, T. Rouček, J. Ulrich, J. Pikman, T. Krajník, B. Lennox and F. Arvin, 'Federated Reinforcement Learning for Collective Navigation of Robotic Swarms', ***IEEE Transactions on Cognitive and Developmental Systems***, 2023 (*Chapter 5*)

Terms and abbreviations

- **RL** - Reinforcement Learning
- **DL** - Deep Learning
- **MARL** - Multi-Agent Reinforcement Learning
- **ANN** - Artificial Neural Network
- **DRL** - Deep Reinforcement Learning
- **FL** - Federated Learning
- **ANOVA** - Analysis of Variance
- **DQN** - Deep Q-Network
- **DDPG** - Deep Deterministic Policy Gradient
- **PhERS** - Pheromone for Every Robot Swarm
- **PER** - Prioritised Experience Replay
- **HLER** - HighLighted Experience Replay
- **NH-ORCA** - Non-Holonomic Optimal Reciprocal Collision Avoidance
- **IoT** - The Internet of Things
- **IDDPG** - Independent Deep Deterministic Policy Gradient
- **SEDDPG** - Shared Experience Deep Deterministic Policy Gradient
- **SNDDPG** - Shared Network Deep Deterministic Policy Gradient
- **FLDDPG** - Federated Learning Deep Deterministic Policy Gradient
- **DARPA** - Defense Advanced Research Projects Agency

Abstract

The growing reliance on robots in modern society, especially in sectors involving laborious or hazardous tasks, has accentuated the limitations of single-robot systems in dynamic and complex environments. Swarm robotics emerges as a promising alternative, leveraging the collective capability and decentralised nature akin to social insects to address these challenges, thus paving the way for robust, adaptable, and resilient systems capable of handling real-world complexities autonomously. Consequently, the urgent need to design adept and adaptive controllers for swarm robotic systems has spurred rapid advancements in the domain of automatic controller design.

This thesis focuses on the pivotal role of automatic controller designs in advancing swarm robotic systems. Despite substantial advancements, the current approaches face a vital trade-off between performance efficacy and pragmatic design processes. This research emphasises the development and enhancement of Deep Reinforcement Learning (DRL)-driven automatic controller designs, which encapsulate the potential merits of both on-line and off-line processes, aiming to bridge the reality gap encountered in the transition from simulated to real-world environments.

To address the centralisation issues in the traditional Multi-Agent Reinforcement Learning (MARL) frameworks, which contradict the decentralised essence of swarm robotic systems, this thesis proposes a Federated Learning (FL)-based DRL training strategy. The main objective is to foster a decentralised approach in DRL-driven automatic controller design, potentially motivating the evolution of more proficient and adaptive swarm robotic systems.

Structured into three primary aims with corresponding objectives, the thesis endeavors to scrutinise the impacts of realistic factors on swarm robotic systems, advocate for DRL-driven automatic controller designs, and propose a novel FL-based DRL strategy to alleviate the centralisation issues inherent in conventional MARL approaches. Through a methodical investigation and critical analysis, this thesis aspires to encourage a new direction in swarm robotics research, steering closer to the realisation of systems proficiently navigating complex real-world scenarios.

Keywords: Swarm Robotics, Automatic Controller Design, Robot Learning, Deep Reinforcement Learning, Multi-Agent Reinforcement Learning, Federated Learning, Decentralisation.

Declaration of originality

I hereby confirm that no portion of the work referred to in the thesis 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 statement

i The author of this thesis (including any appendices and/or schedules to this thesis) owns certain copyright or related rights in it (the "Copyright") and s/he has given The University of Manchester certain rights to use such Copyright, including for administrative purposes.

ii Copies of this thesis, either in full or in extracts and whether in hard or electronic copy, may be made *only* in accordance with the Copyright, Designs and Patents Act 1988 (as amended) and regulations issued under it or, where appropriate, in accordance with licensing agreements which the University has from time to time. This page must form part of any such copies made.

iii The ownership of certain Copyright, patents, designs, trademarks and other intellectual property (the "Intellectual Property") and any reproductions of copyright works in the thesis, for example graphs and tables ("Reproductions"), which may be described in this thesis, may not be owned by the author and may be owned by third parties. Such Intellectual Property and Reproductions cannot and must not be made available for use without the prior written permission of the owner(s) of the relevant Intellectual Property and/or Reproductions.

iv Further information on the conditions under which disclosure, publication and commercialisation of this thesis, the Copyright and any Intellectual Property and/or Reproductions described in it may take place is available in the University IP Policy (see `http://documents.manchester.ac.uk/DocuInfo.aspx?DocID=24420`), in any relevant Thesis restriction declarations deposited in the University Library, The University Library's regulations (see `http://www.library.manchester.ac.uk/about/regulations/`) and in The University's policy on Presentation of Theses.

Acknowledgements

I am deeply grateful to the University of Manchester for supporting my studies with the President's Doctoral Scholar (PDS) Award. Receiving this prestigious award, bestowed upon only a select number of doctoral students, has been an immense honour. It has afforded me the academic freedom to explore diverse fields and engage in interdisciplinary research.

Heartfelt thanks are due to my supervisors, Farshad Arvin and Barry Lennox, for their unceasing support. Farshad, in particular, you have been an outstanding supervisor, guiding me since my undergraduate days. You have imparted not only research skills but also invaluable life lessons, much like a caring uncle. Your friendship and the joy of sharing life's stories and news have been a great source of comfort. I will always cherish our memories, including our spontaneous meetings at E1p in the Sackville Street Building, intense sessions of writing papers and sharing life stories over Zoom during the COVID-19 pandemic. Your unwavering support and readiness to assist at any time have been invaluable, and I anticipate that our relationship will continue to thrive beyond this acknowledgement.

I must also express my gratitude to Tomas Krjanik for his varied support from the outset of my research journey. His exceptional commitment, including flying from Prague to Manchester for in-person troubleshooting, has been remarkable. His invitations to Prague, where I could collaborate with his students, have greatly aided my research. Conversations with Tomas always expanded my perspective, and the barbecues at his home remain a cherished memory of my PhD journey. My thanks also to Jiří Ulrich, Tomáš Rouček, Zdeněk Rozsypálek, and others for being excellent colleagues and assisting with my research in robotics, localisation systems, and machine learning setups. Their collaboration at CTU has underscored the importance of teamwork. I am grateful to Jan Pikman for enhancing my research as part of an undergraduate project. Additionally, I appreciate Simon Baxter's invitation to the University of Melbourne, which significantly enriched my experience with field experiments.

I would like to express my gratitude to my mentors. Inmo Jang has provided invaluable advice on research, career paths, and personal life decisions. Erwin Lopez's warm and humorous welcome made me feel at home in the D9 lab and the new office. Hanlin Niu offered practical help and advice for my thesis's second chapter. Junyan Hu inspired me with his passion for research, and Fatemeh Rekabi Bana imparted extensive knowledge in control theories and research methodologies.

My peers have also played a significant role in my journey. I am thankful to Afreen Islam and Yu-Hsiang Su for their companionship and support. Kefan Wu, Yifeng He, Zheyu Liu, Shiyi Wang, Wenxing Liu, Xueliang Cheng, and Kanzhong Yao have been excellent lab mates, sharing many

long hours together. Hangyeol Go has been a constant friend both in and out of the lab. Thomas Johnson, Andy West, Anthony Zhengyi Jiang, Paul Baniqued, and Daniel Johnson have made me feel part of the larger robotics group.

Lastly, I extend my gratitude to my family and friends. My parents, Sangsoo Na and Gwangim Jeong, have supported me wholeheartedly in every aspect. Their boundless love has been instrumental in the completion of this thesis. My brother, Minseong Na, has always inspired me to be a better person. The Manchester Yedam Church community has significantly contributed to my fulfilling life in Manchester, making this thesis possible. Finally, I am profoundly thankful to God, whose presence as an embodiment of love has guided me towards love in all aspects of my life.

Chapter 1

Introduction

1.1 Swarm Robotics

Robots have become an integral part of modern society, notable for performing tasks autonomously thanks to their programmable nature and the capabilities to sense and act accordingly. They are especially prominent in sectors demanding laborious tasks or operations in environments that are dangerous for humans. This includes roles in manufacturing and rescue missions, amongst others.

However, deploying a single robot has its set of limitations, especially in large-scale, complex, dynamic and hazardous environments. The chances of successfully completing a mission in such conditions diminish significantly when relying on just one robot. Enhancing a single robot's hardware and software could be a solution, but it involves considerable costs and time.

An alternate and potentially more effective strategy is to employ a group of robots, an approach known as Swarm Robotics [1]. This approach not only allows for coverage of a larger area but also offers redundancy with the increased number of robots. This ensures the mission's continuity even if a part of the group fails and adaptivity to diverse and unpredictable requirements of real-world missions. This collective capability, featuring social insects in nature, fosters collaboration that exceeds the potential of individual entities. For instance, a group of robots can carry out collaborative transportation tasks, adapting to the specific needs of the environment and task, which resembles the self-organising behaviour of army ants that construct artefacts with their bodies to adapt to the environments [2].

However, designing such a system is more complex than creating a single-robot system. It requires decentralisation, where robots function independently, communicating only with nearby robots or interacting directly with the environment. This decentralisation creates a resilient and adaptable system, with robots capable of undertaking similar tasks and adjusting the group size as needed, enhancing the system's resilience and adaptability.

Creating an effective swarm robotic system, though promising, presents several challenges. A pivotal aspect is developing a competent control system, an aspect that directly influences individual and collective behaviours. Recognising the potential of swarm robotics in real-world applications, developing this control system has emerged as a crucial research area [3], [4].

While controlling a single robot has been researched extensively over the years, ensuring collective behaviour without escalating complexity remains a significant challenge [5]. Much of the existing research in swarm robotics has been confined to controlled environments, not equipping real-world complexities. The next step in this field involves creating a system where robots can work harmoniously without human intervention, showcasing resilience and adaptability in dynamic and potentially hazardous environments.

1.2 Automatic Controller Design

With the realisation of swarm robotic systems that exclude the need of human designers, automatic control design methods have stood as a vital area of focus in the advancement of swarm robotics. Automatic controller design is a study that seeks to devise control systems for swarm robots with minimal human intervention. This facet of research has been highlighted as a pivotal component for the successful deployment of swarm robotic systems in real-world missions [3], [5]. Despite substantial advancements, the quest for a universally effective methodology remains, with current approaches grappling with a balance between performance efficacy and pragmatic design processes [6], [7].

To circumnavigate the challenges inherent in ”on-line” designs, which involve real-time controller development during missions and risk prolonged design periods and potential physical damage, an ”off-line” approach has been considered. This entails the preliminary design of controllers within simulated environments, removing the need of physical robots involved in the design process. However, it increases the degree of reality gap issues, a phenomenon where controllers fail to transition successfully from simulated to real-world environments.

At the forefront of this evolving landscape is robot learning-based automatic controller design, a paradigm that integrates the merits of both on-line and off-line design processes to bridge the reality gap. Reinforcement Learning (RL) and its more sophisticated counterpart, Deep Reinforcement Learning (DRL), which employ Artificial Neural Networks (ANNs) as function approximators, stand as potent techniques in this realm. These methods are equipped to facilitate seamless transitions between simulated training environments and actual operational contexts. Nevertheless, the deployment of robot learning methodologies, particularly DRL-driven automatic controller design, has been somewhat limited, a scenario that signals the necessity for intensified research and development in this domain [7].

However, the widespread adoption of DRL-driven automatic controller design has encountered resistance, primarily due to its incompatibility with the decentralised framework characteristic of swarm robotics systems. The prevalent Multi-Agent Reinforcement Learning (MARL) framework necessitates a degree of centralisation, a requirement that contradicts the inherent decentralised structure of swarm robotic systems. Consequently, this has hindered the progression and effectiveness of DRL applications within this context. As such, fostering a decentralised approach in DRL-driven automatic controller design emerges as a vital topic, a move that promises to catalyse the

development of more proficient and adaptive swarm robotic systems capable of tackling complex real-world challenges.

1.3 Aims, Objectives and Overall Structure

This thesis focuses on the further development and enhancement of DRL-driven automatic controller designs, a critical component in advancing the functionality of swarm robotic systems. The primary contribution is the advancement of DRL-driven automatic controller designs alongside addressing the centralisation issues noted in the MARL-based robot learning for automatic controller design.

The structure of this thesis is delineated into three distinct aims, each complemented by specific objectives that are meticulously crafted to fulfil the respective aims. These aims and objectives are detailed as follows:

- **Aim 1**: To identify the need for automatic controller design for swarm robotic systems in a setting close to real-world environments.
  - **Objective 1**: To investigate the impact of real-world conditions on the collective behaviour of swarm robotic systems.
- **Aim 2**: To develop a DRL-driven automatic controller design, proving its viability and showcasing its advantage over traditional methods.
  - **Objective 2**: To formulate a DRL-driven automatic controller design for swarm robotic systems utilising realistic scenarios in a simulated environment.
- **Aim 3**: To devise a strategy for DRL-driven automatic controller design to alleviate the centralisation issue in the MARL framework.
  - **Objective 3**: To propose and evaluate a Federated Learning (FL)-based DRL training strategy to reduce centralisation while improving performance efficacy.

This thesis begins with a comprehensive literature review, followed by three main chapters, each dedicated to addressing the respective aims and accompanying objectives outlined above.

- In Chapter 2, a detailed literature review is presented to give readers a foundational understanding of both DRL-driven automatic controller design and the potential of an FL-based DRL-training strategy for swarm robotic automatic controller design.

- In Chapter 3, a systematic exploration of the impacts of simulated real-world conditions on the collective behaviour of swarm robotic systems is undertaken. Capitalising artificial pheromone communication that mimics communication of nature swarms that lead higher adaptivity, the effect of real-world conditions on a complex swarm robotic system is investigated. This investigation highlighted the imperative need for automatic controller designs capable of operating in real-world conditions within the field of swarm robotics.

- Chapter 4 advocates for the DRL-driven automatic controller design, which is currently a promising yet underexplored method within the domain of robot learning. This approach is introduced and applied to facilitate basic low-level behaviours in swarm robotic systems with the realistic scenario of autonomous vehicles, formulated as a swarm system, showcasing its effectiveness and highlighting existing issue of centralisation due to the intrinsic nature of the MARL framework, where the issue hinders its establishment as a universal methodology for automatic controller designs.

- Chapter 5 addresses the issue of centralisation stressed in Chapter 4, intrinsic to the MARL method. Here, a Federated Learning (FL)-based DRL training strategy is presented as a viable alternative to MARL. This chapter explores the potential of the proposed FL-based DRL strategy to counteract the centralisation issues inherent in MARL, aiding the progression of DRL-driven automatic controller designs into a universal methodology, aligning with the grand challenges identified in swarm robotics [4].

Chapter 2

Literature Review

In this chapter, a comprehensive literature review is provided to equip appropriate background information to understand the context of this thesis and relevant technical details. This chapter begins with a broad overview of swarm robotics, followed by an in-depth discussion on artificial pheromone communication, highlighting the specific system adopted in this thesis. Subsequently, the topic transitions to the realm of automatic controller design, outlining prevailing methods and emphasising the imperative need for advancing DRL-driven designs. Detailed technical insights into DRL are provided, with a focus on the specific algorithm underpinning this thesis. To address the centralisation challenges of DRL, the concept of Federated Learning (FL) is introduced, underscoring its efficacy in reducing centralisation while highlighting its added benefits observed in other domains.

2.1 Swarm Robotics

Swarm robotics is a field of study of how to make a group of individual robots collaborate or collectively solve a common task without any forms of centralised communication [1], [8]. The early motivation and inspiration for studying swarm robotics comes from the prosperity of social animals utilising collective behaviours such as ants, bees and birds [3]. In recent decades, researchers have identified that such complex collective behaviours performed by social animals emerge from individuals who lack of global representation or knowledge of the task [9]. Furthermore, it has been observed that no leader that guides the rest of the group exists in the collective behaviours of social insects. Although they can only utilise limited local observations and interactions for large-scale complex tasks, they can perform collective task very effectively with high redundancy resulted from the large number of individuals. The high redundancy in social animals promoted that they perform complex behaviours and maintain a large group in nature. Adopting the mechanism of social animals featuring high redundancy, researchers have devised and developed swarm robotics to design a robotic system effective for complex environments and tasks where and what nature swarms thrive [10].

High redundancy in swarm robotics creates three unique characteristics of swarm robotics: robustness, scalability and flexibility. In swarm robotics, robustness describes the system's tolerance to failures and its ability to maintain safety; it is often used synonymously with fault-tolerance.

In swarm robotic system, high degree of robustness is achieved by massive redundancy of quasi-homogeneous agents and the avoidance of single point of failures that divides a swarm into sub-groups. It leads the swarm robotic system to be tolerant from failure even when the proportion of individuals in the swarm fail. Scalability is achieved as quasi-homogeneous individuals require only local communication without the access to the global information to be involved in a swarm. This property allows the entire swarm to maintain its functionality without resetting the way the existing members interact regardless the number of members increases or decreases. Flexibility (also often called adaptivity) in a swarm is also achieved by quasi-homogeneous individuals who are not assigned a specific role in a swarm. Since there is no specialisation of individuals in a swarm, robots can dynamically allocate themselves to different tasks to match the requirements of specific environment and operating conditions. The three advantages of swarm robotics are crucial where the environment causes failures of individuals, requires diverse sizes of the swarm, and changes dynamically. Thus, swarm robotic systems are highly promising for the domains where the task and operating condition is too hazardous for humans or single robot to perform task without failure, or the environment is large-scale and complex and dynamic. The examples of such task or environments include eradicating oil leaks in the vast ocean, extraterrestrial planetary exploration, and large-scale surveillance.

Since the emergence of swarm robotics, various research directions have been explored. These encompass the collective behavior exhibited by swarm robotic systems, the development of robotic platforms for swarm robotics research, the methodologies for individual robot controller design, communication strategies, and additional interdisciplinary studies that facilitate the operation of swarm robotic systems in real-world environments. Together, these directions constitute the most prominent areas of focus in the field [4], [11].

2.2 Artificial Pheromone Communication System for Swarm Robotics

In Chapters 3 and 4 of this thesis, biologically inspired pheromone-based communication is incorporated as an important part for swarm robotic systems to perform complex collective behaviour, though it is not a central focus of investigation in the research studies. This section illuminates the inherent characteristics of biological pheromones, illustrating their successful manifestations in nature, which in turn underscores their potential efficacy as a communication tool within swarm robotic systems. Subsequently, a range of artificial pheromone systems is presented, highlighting various experimental demonstrations and the gradual advancements made in this nascent yet promising field, holding significant potential for future applications.

Drawing closer to the specific framework of this thesis, COS-Φ, an artificial pheromone system that serves as an essential part of the research studies conducted in this thesis, is detailed. This part distinguishes COS-Φ against other existing artificial pheromone systems, facilitating delineation of its features and functionalities.

2.2.1 Pheromone in Biology

Pheromone is defined as chemical substance secreted to the outside of an individual and detected by conspecifics triggering them to conduct stereotyped behaviour and/or have developmental changes [12]. Pheromones are essential factors in animal communication and trigger physiological change in a wide range of behavioural and ecological contexts in nature [13]–[17]. Pheromones play important roles as communication means for diverse taxonomic groups from yeast and insects to mammals [18].

Pheromonal communications were found in vertebrate animals [19]. For example, *2-methylbut-2-enal*, which is contained in rabbit milk, improves nipple-search behaviour of rabbit pups [20]. Several studies have found that the physiological and psychological effects elicited by chemosignals arise also in humans [21]–[24]. For example, it is found that female reproductory state is affected by male axillary extract which mediates the hormonal change in female brain [23]. These results showed that pheromonal communication is widely used across diverse taxonomic groups including human beings.

Although a broad spectrum of animals use pheromonal communication as their communication mechanism, social insects utilise pheromone most effectively [25]. Pheromone enables a whole group of social insects to communicate effectively as an externalised and spatialised shared memory although the individuals have limited memory and sensory-motor capability. Also, agents only need local sensing ability to detect pheromone rather than global sensing. Moreover, a collective behaviour in a group of social insects can be optimised appropriately deploying pheromones [26], [27]. For example, the Argentine ant *Iridomyrmex humilis* finds the shortest path from its colony to the food source using pheromone and its feedback mechanisms [25], [27].

Monomorium pharaonis, called Pharaoh's ants, which are usually found in human habitats, utilise multiple pheromones that are vital for food-foraging behaviour in dynamic and competitive environments [28]. They create pheromone trails to fetch to and return from the food source using three types of trail pheromones: non-volatile attractive pheromone, volatile attractive pheromone and repellent pheromone. The three different kinds of pheromones work respectively as long-term memory of the trail, attraction leading to the currently rewarding trail and a stop-sign. The sophisticated use of the three types of pheromones enables the ant colony to create optimal pheromone trails from the nest to the food source in dynamically changing environments. *Bombus hortorum*, known as bumblebee, leaves chemical cue on flowers which allows detection and avoidance of recently depleted flowers [29]. Similarly, the use of chemical cue enhances the efficiency of foraging behaviours of bee colonies since it prevents meaningless visits of depleted flowers. Majority of social insects utilise queen pheromones. This type of pheromones characterise queen and other reproductive individuals and are extremely important to maintain the whole colony [30], [31]. For example, a queen-pheromone of *Lasius niger*, known as the black garden ant, regulates worker sterility so that the reproduction behaviour of the colony is controlled by the queen [32]. As another example, if the queen of a colony fails by any reasons including viruses and pesticides, the secretion

of queen pheromone in the colony decreases, and as a result the colony might collapse [33]. The above examples clearly show the efficacy of pheromone as a communication strategy for social insects.

2.2.2 Pheromone in Swarm Robotics

The efficacy of pheromone in social animals is applicable in swarm robotic systems as both of them consists of a group of individuals without a central leader, explained more specifically with the following reasons. First of all, pheromone communication requires only local information for individual robots that can mitigate the needs of functionality to obtain global information, e.g. precise localisation in global coordinate system, long-range communication [34]. Second, regardless of the complexity and scale of the task and environments, the individuals only need to have a relatively lower memory capacity since deposited pheromone embeds information in the environment not requiring individual robots to store the information [35]. Third, its potential to optimise the performance of a group task via the use of a combination of multiple types of pheromone and feedback mechanisms is a highly desirable feature for robot swarms to encrypt complex sets of information with different strengths and lengths of expiration [26]. Finally, pheromone allows for robot swarms to have fully decentralised communication [36].

The potential of pheromone communication has been realised in the field of swarm robotics; thus, research studies to capitalise on pheromone communication in swarm robotic systems have emerged. The seminal work that implemented pheromone communication in robot swarm was introduced in 1999 [37]. This work employed *Cinnamomum camphora*, known as Camphor, as the medium of artificial pheromone for trail following behaviours for the robotic systems which embodied odour releasing and sensing functionality. The results of this work introduced the feasibility of using pheromones in robotic systems as a communication medium. From the emergence of this seminal work, research studies have investigated the feasibility of various means to actualise artificial pheromones including chemical substances, RFIC tags and light-emitting devices. The research studies introduced in the following paragraphs show the feasibility and characteristics of the medium used as an artificial pheromone. These works are categorised based on the type of artificial pheromone employed, rather than a chronological or developmental progression. It is important to note that there is no temporal connection between these methods; they have emerged in parallel to actualise similar ideas. Their development was not driven by a sequential improvement or by overcoming the limitations of one method in favour of another. Rather, these different approaches were implemented independently, each characterised by its unique features and attributes.

The seminal work introduced in this study employs chemical substances as a medium for artificial pheromones, specifically utilising ethanol as an artificial pheromone in swarm robotic systems as demonstrated in the two early studies [38], [39]. These cited works illustrate the feasibility of creating a fully autonomous robotic system capable of collective behaviours, such as trail-following, through pheromone communication. Although chemical-based artificial pheromones bear a strong

resemblance to those found in nature, there are essential differences that render their utilisation in engineered systems less effective.

Firstly, the fundamental disparities between biological and engineered systems mean that robots require specialised types of sensors and actuators not commonly used in conventional robotic systems. The rarity of these components can hinder development in comparison to systems employing standard sensors and actuators. Secondly, unlike natural swarms that operate without external guidance, robot swarms are designed to perform human-directed tasks and thus require a certain level of control. The chemical nature of the substances makes them highly susceptible to environmental influences, complicating the precise execution of desired behaviours. Lastly, the utilisation of chemical substances as artificial pheromones presents potential risks to human operators, given that they may be adversely affected by them. This issue adds a layer of complexity and further emphasises the inherent challenges in aligning chemical substance-based artificial pheromones with the efficacy of their natural counterparts.

Drawing on the principle of pheromones, which leave information in the environment, RFID systems have been employed as a means of pheromone communication in swarm robotics [34], [40]. In these studies, RFID tags were strategically distributed across the operational floor space. As robots passed over these tags, they transmitted data that was then stored. The subsequent behaviour of the robotic swarm was guided by this previously transmitted data, functioning in a manner analogous to natural pheromones. A research study by Alfeo et al. [41] illustrates the practical applicability of RFID tags as a medium of pheromone communication in real-world scenarios, specifically for autonomous waste management in urban environments. The research validated the effectiveness of RFID-based pheromone communication, showing that this approach outperformed traditional centralised control methods for waste management vehicles. However, while RFID-based swarm robotic systems have shown promise, their deployment in real-world scenarios presents challenges. Specifically, these systems require environments to be appropriately equipped with RFID tags, which are necessary items that may limit their use in potential operating settings for swarm robotics.

Virtualisation offers a novel approach to actualising pheromone-based communication in swarm robotic systems through the use of virtually generated pheromones. This method enables individual robots to access and interact with virtual pheromones. Recent studies have explored virtualisation for the implementation of artificial pheromones in swarm robotic systems. For example, the study [42] introduced a mechanism for path selection by a foraging robot swarm using ”virtual ants.” In this model, robots locally transmitted messages, referred to as virtual ants, which deposited virtual pheromones, guiding the path selection.

Other research has expanded this concept by creating virtual maps to mark deposited pheromones, making them accessible to all robots. The augmented reality for Kilobots (ARK) system, introduced by the study utilising Kilobots [43], provides a striking example. It establishes a virtual environment where Kilobots, designed for large-scale robotics experiments, can transmit and sense information in real-time. Through ARK, robots are equipped with virtual sensors and actuators, granting them access to shared virtual environments where they can release and detect virtual pheromones.

Subsequent works, such as the study that leveraged this system to investigate quality-based foraging [44].

Similar to the ARK system, Kilogrid [45] was proposed as a virtualisation system that primarily uses bidirectional infrared communication between Kilobots and grids mounted under the arena. This system also enables the creation of virtual pheromones, accessible to both the robots and a remote PC in real-time. Virtualisation simplifies the use of various pheromone types, allowing for easy modification in sophisticated experimental settings.

Despite these advancements, virtualisation systems like ARK and Kilogrid do have their limitations. Notably, both systems suffer from low communication frequency, leading to significant delays in communication. This constraint must be considered when harnessing these methods in the rapidly evolving field of swarm robotics.

RFID systems and virtualisation each present distinct advantages for pheromone communication, with the former facilitating the manipulation of pheromone information and the latter providing flexibility. Light, as a medium, combines the benefits of both RFID and virtualisation. Garnier et al. [9] pioneered the use of projected light from the video projector mounted above the operating arena as an artificial pheromone, demonstrating its advantages. From a design standpoint, light sensors and emitters are more accessible and deployable compared to the technologies required for RFID and virtualisation systems. Additionally, the control and modification of light emitted from engineered devices offer flexibility in implementing various types of pheromone and their spatio-temporal development, a key aspect in mimicking natural pheromone-based communication.

COSΦ represents a light-based artificial pheromone system designed for swarm robotics [46]. It functions through the computation of a 2D matrix that depicts the distribution of pheromone within a rectangular field. The system deploys a displayable screen as an interactive arena, enabling robots to engage with the emulated pheromone patterns. Additionally, COSΦ facilitates the deposition of pheromones by employing a tracking system to identify the position, orientation, and ID of the robots. A notable advantage of COSΦ is its capability to provide high-resolution emulation of pheromones, as the resolution is determined by the display device itself. This allows for a more nuanced influence on robot behaviour, beyond what low-resolution methods can achieve. Furthermore, the conversion of artificial pheromones from the computed 2D matrix simplifies the manipulation of various pheromone types with distinct properties and facilitates simulation at different time speeds. Such features render COSΦ suitable for both simulating and studying artificial pheromone-based swarm robotic systems. It also offers a platform for experimental testing of prospective pheromone-based swarm robotic systems, which, although not yet capitalised for the real-world applications due to technical limitations, appear promising for future development with the advent of supporting technologies.

Subsequent research has expanded upon the COS Φ system to explore various collective robotic behaviours. Sun et al. [47] adapted the system to simulate multiple types of pheromones through the use of multi-color hues, enabling the investigation of food-foraging and aggregation behaviours

via both attractive and repellent pheromones.

2.3 Automatic Controller Design in Swarm Robotics

Swarm robotic system controller design plays an important role in the manifestation of intricate collective behaviours, transforming preliminary system specifications into actionable robotic behaviours. A significant challenge in swarm robotics is not just ensuring individual robot behaviour, but also realising emergent properties from collective interactions. Deriving a desired global collective behaviour from local behaviours and vice versa is challenging due to the non-linear dynamics of the swarm robotic system performing collective behaviours [48]. Therefore, controller design is an important topic of research in swarm robotics requiring developing diverse effective design methods [49]. In this section, overview of swarm robotic controller design is introduced, highlighting automatic controller design as a topic of interest. the taxonomy of swarm robotic controller design is illustrated in Figure 2.1. For a more detailed classification of automatic controller design methods, the design phase is also marked with different edge colours. For easy reference, the corresponding section number for each method introduced in this section is marked and the taxonomical location of the main topic of this thesis is coloured red.

While there is an absence of standardised methodologies in controller design, the prevailing paradigms can be primarily bifurcated into manual and automatic design methods [3]. The manual controller design is the most basic form of controller design, predominantly used in many swarm robotics research studies [10], [49]. Manual controller design includes any type of methodologies that require human designers to design the microscopic behaviour of a robot swarm to lead a desired macroscopic behaviours. Typical methodologies adopted as manual controller design are designing microscopic behaviours based on finite-state machines or behaviour trees using biologically identified behaviours and modelling approaches [49]. Automatic controller design encompasses any approach that excludes the intervention of the human designer during the design process after initial specification phase [3]. Available methods for automatic controller design are introduced in the following sections in depth.

The salient difference between manual and automatic controller design is the degree of human intervention during the iterative design phase. Manual design, despite its efficacy in elementary scenarios, tends to be impotent against the escalating complexities inherent in real-world swarm robotic system implementations. Contrarily, automatic design augments efficiency and scalability by automating behavioural adjustments, thereby addressing challenges presented by complex and unpredictable real-world scenarios. The pivotal shift towards automatic controller design entails converting the design process into an optimisation problem, where an objective function is optimised for the best performance.

The semi-automatic design blends elements of both manual and automatic controller design. Early research studies of automatic controller design adopted semi-automatic design [50]–[53]. In this method, a human designer remains involved, periodically adjusting the controller based on obser-

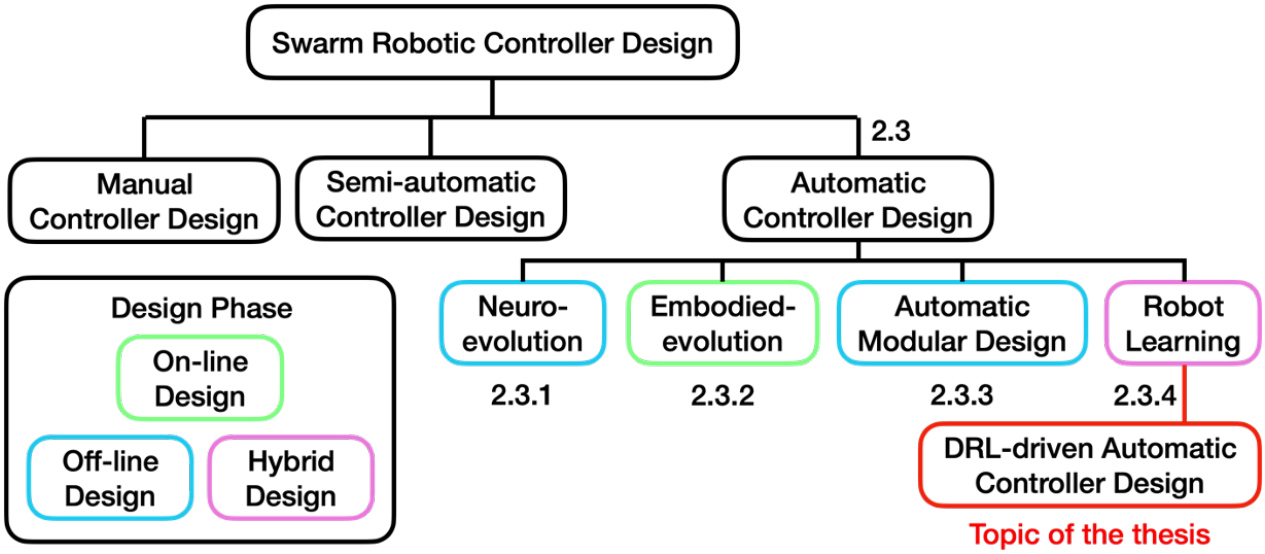

Figure 2.1. An overview of swarm robotic controller design taxonomy

vations. For example, they may tweak parameters to boost exploration when necessary. Though semi-automatic design provides a glimpse into the potential of full automation, it shares manual design's pitfalls: the controller's quality is contingent upon the designer's skill and direction [7]. Consequently, reproducibility may vary with the intricacy of the controller design [54].

Fully automatic controller design eliminates human involvement post the initial mission specification [5]. Essentially, it ensures total automation throughout the design process [55]. There is no general methodology for automatic controller design, but several design methods have been developed with different pros and cons. Automatic controller designs can be bifurcated into on-line and off-line approaches based on when the design process is executed [7]. In the on-line design method, the design process is executed while the swarm performs the mission in the target environment; in contrast, in off-line design, the design process is executed before the swarm operates the mission in the target environment.

While on-line methods utilise real-world data, they can suffer from lengthy design process, potential robot harm, and a narrow emphasis on local behaviors, neglecting overall system efficacy due to the robot's limited local data access. Off-line methods, conversely, leverage simulated environments for controller design prior to real-world deployment, ameliorating many of the on-line method's limitations. However, the persistent challenge remains the reality gap—the divergence between simulated and real-world environments. The repercussions of this gap range from necessitating minor modifications to rendering controllers ineffectual.

Carefully considering the challenges inherent in each design phase is crucial when developing automatic controller design methods, as it determines the effectiveness of the designed controller of a robot swarm deployed for the mission in the target environment. Birattari et al. [5] envisages futuristic real-world applications of swarm robotic systems, exemplified by garden management business scenarios. Such applications underscore the flexibility offered by off-line methods, where multiple pre-designed controllers can be readied in anticipation of varied tasks, effectively navigat-

ing operational time constraints.

In the following sections, four methods for automatic controller design are introduced sequentially: Neuro-evolution, embodied evolution, automatic modular design and robot learning. The characteristics, pros and cons and representative research studies are described for each method with discussion, altogether justifying the importance of developing DRL-driven automatic controller design, which is a subcategory of robot learning, under an automatic controller design perspective.

2.3.1 Neuro-evolution

The prevailing method in swarm robotics for automatic controller design is the evolutionary swarm robotics approach [56]. This method integrates the concepts of evolutionary robotics [57] into swarm robotic systems. In evolutionary robotics, robot controllers are optimized using evolutionary algorithms based on a mission-specific objective function, often termed as the fitness function. The objective function evaluates the performance of the control instance based on the pre-defined evaluation metrics. Within a pool of diverse options of control instances, poorly performing controllers are discarded; conversely, best performing controllers are selected for the next iteration until it reaches the defined termination condition. During the iteration, control instances can vary based on recombination and mutation rules to explore wide parameter spaces for obtaining better controller instances. This process resembles evolutionary process in nature, that is why it is named evolutionary algorithm.

Werner and Dyer [58], [59] introduced the seminal studies of collective behaviours using evolutionary robotics techniques. In these works, they observed the successful emergence of communication strategies necessary for the defined mating and herding tasks. Reynolds, who developed the famous boids [60], which are the virtual creatures widely used for studying flocking, utilised evolutionary robotics for the virtual creatures to avoid static obstacles and a manually programmed predator [61]. Not only the virtual agents that do not have physicality, but also evolutionary robotics approach succeeded in designing swarm robotic tasks with diverse simulated robots, e.g. Khepera [62], ALICE [63].

Although the research studies using virtual agents or simulated robots showed the feasibility of evolutionary robotics techniques to develop useful collective behaviours, only a few case studies have demonstrated the usefulness of evolutionary robotics in swarm robotic scenarios. A seminal work using real robots to show the feasibility and effectiveness was presented by Quinn et al. [52]. With a group of three simulated and physical robots that rely on noisy data from infrared sensors, navigation in formation was successfully performed. Similarly, Nelson et al. [64] used evolutionary robotics algorithm to develop robotic swarms playing the “capture the flag” game. In the setting of two robotic teams playing the game against each other, good competitive strategies were developed in simulated environment and deployed in the robot teams, demonstrating playing games in the target environment.

One most famous research project that evolutionary swarm robotics approach developed by large

volume of research studies is the SWARM-BOTS project [2], [65]. The aim of the project is to study innovative approaches to the design and implementation of self-organising and self-assembling artifacts. In this project, several collective behaviours have been designed using evolutionary robotic techniques, e.g. collective alignment, hole avoidance and self-organising synchronisation. The controllers were designed in the simulated environments and deployed both in the simulated and physical experimental setup. For the implementation of the designed controllers in the physical robots, a mobile robotic platform with a gripper, called *s-bot* [65], was deployed. Through this seminal project, evolutionary swarm robotics approach demonstrated its potential to design complex collective behaviours working in the real-world environment.

In the SWARM-BOTS project, artificial neural network (ANN) was adopted as a control instance to be optimised using the evolutionary robotics techniques. The case where artificial neural network is used as the control instance of evolutionary robotics techniques is called *neuro-evolution*. ANNs are non-linear function approximators that are aimed to approximate any types of functions expressing only with the sum of the product of first-order functions. Depending on the design of ANNs, various types of controllers can be designed. For this reason, ANNs are popular for the evolutionary robotics techniques and the cases where evolutionary robotics techniques use ANNs are further categorised as neuro-evolution.

More recently, neuro-evolution approach has been introduced to more diverse scenario and environments to develop demonstrating its adaptivity for the real-world missions, which cannot be found in the manual controller design methods. Duarte et al. [6] developed a controller for a swarm of unmanned aquatic surface vehicles for homing, dispersion, clustering and area inspection tasks, that are sub-tasks for the marine environmental monitoring system. Using NEAT [66] as an algorithm to implement neuro-evolution based controller design, controllers were designed in the specially designed simulated environment. The best three controller were selected after evolutionary process, and deployed for the real robot experiments. The performance of the selected controllers for the real environmental monitoring tasks was demonstrated at a similar level of those in the simulated environment, showing that neuro-evolution approach is effective for the highly realistic applications whereas only simulated or highly controlled environments were used for validation of neuro-evolution approaches previously.

Hasselmann et al. [67] presented a comprehensive comparative analysis on diverse available techniques to implement off-line neuro-evolution approach, including NEAT [66], Covariance Matrix Adaptation Evolutionary Strategy (CMA-ES) [68], Exponential Natural Evolution Strategies (xNES) [69] and EvoStick [70]. Taking diverse behavioural scenarios that are widely adopted for the missions of robot swarms, requirements for neuro-evolution approaches to design mission-specific behaviours were investigated.

Although successful application scenarios have been demonstrated using neuro-evolution approaches, following with further detailed comparative analyses, neuro-evolution approaches have encountered two major challenges: fitness engineering and the reality-gap [7].

Fitness engineering refers to the process and method of computing fitness score of the family of evolutionary algorithm methods. In other words, fitness engineering is about how to generate an appropriate objective function that drives the evolutionary process. Two specific challenges are well-known as a result of incorrectly designed objective function: bootstrapping and deception [6]. Bootstrapping is a phenomenon where the objective function cannot provide effective selection pressure when the instance is in low-performance region of the search space. Absence of evacuation from low-performance region causes the controller instance falls in the suboptimal performance. The issue of deception causes when the case of local optima is not considered. When deception exists, the design process converges to the easily reachable local optima rather than the global optimum that is intended to be found. It is commonly known that *a priori* knowledge of designers can help the objective functions mitigate bootstrapping and deception issues, which the process is called feature engineering. However, the necessity of *a priori* knowledge counteracts against the purpose of automatic controller design, to minimise human expertise and intervention during the design process.

Reality-gap means the discrepancy between the simulated environments where the controller is designed and the target environment where the designed controller is deployed in the physical robot in the target environment, occurring performance drop. Reality-gap is unavoidable for off-line design method as it requires the simulated environment to design the controller for the sake of effective and safe design. The catastrophic effect of reality-gap has been observed even in the case of sophisticated off-line neuro-evolution method. The discovery of catastrophic effect of reality-gap has raised the investigation of off-line design methods that demonstrated good performance in the simulated environments or highly controlled laboratory environments [5], [67]. The substantial effect of reality-gap demonstrated that off-line design methods without mitigation strategies would not be useful for the missions for a robot swarm in the target environment.

2.3.2 Embodied Evolution

The most straightforward method to overcome the reality-gap problem is to avoid using an alternative environment (i.e. simulated environment) and to design the controller directly in the target environment. To mitigate the issue of reality-gap of off-line neuro-evolution approaches, the counterpart design method which is conducted on-line has been developed, called embodied evolution [7], [71]–[73]. In embodied evolution, individual robots execute a decentralised and asynchronous evolutionary algorithm whereas a centralised neuro-evolution algorithm is run in an off-line method. During the design process, the robots periodically select new control instances from their own pool and exchange the designed controllers with neighbour robots. The aim of embodied evolution is also to obtain the single best version of the controller instances, typically composed with ANNs.

Thanks to the on-line and decentralised nature of the design process, embodied evolution counteracts the limitation of off-line design methods. Since the design process is executed in the target environment directly, no reality-gap exists to be concerned. Furthermore, the design process can be

accelerated as the decentralised and asynchronous evolutionary algorithm is executed on individual robots in parallel. By exchanging control instances among individuals, the best control instance can be found faster than the centralised on-line algorithm is utilised. Additionally, the nature of decentralisation and asynchronicity enables the swarm scalable. The scalability that embodied evolution can offer is highly desirable for a robot swarm as this property allows a robot swarm to be adaptive to diverse scenarios as well as the swarm becomes fault-tolerant. Despite the promising aspect of embodied evolution, the number of research studies is low that use real robots than those studied off-line neuro-evolution approaches due to the intricacy of using real robots. In the seminal study where the term embodied evolution is coined, a swarm of mobile robots performed phototaxis tasks outperforming manually designed controllers [71], [72]. Using embodied evolution for more practical tasks, Prieto et al. [74] designed a swarm of e-puck robots to perform collective cleaning tasks. To leverage the existing evolutionary algorithm that demonstrated effectiveness in the case of off-line neuro-evolutionary approaches, Silva et al. [75] proposed an on-line and decentralised variant of NEAT [66], called odNEAT, and evidenced that it approximated the performance of rtNEAT [76], the improved version of NEAT, and outperformed existing on-line version of neuro-evolution methods in the three tasks: aggregation, integrated navigation and obstacle avoidance, and phototaxis. Notably, odNEAT presented higher fault-tolerance than the centralised counterpart, displaying greater generalisation. More recently, Cambier et al. [77] proposed an evolutionary language model for the effective exchange of information to tune the parameters of a probabilistic controller performing aggregation tasks. The recent case study of embodied evolution for designing practical foraging task [78], where the concept of embodied evolution is framed as a form of robot learning coined as social learning, highlights the benefits of embodied evolution approaches accompanied with effective communication strategies to overcome the issue of reality-gap.

Though the existing research studies have demonstrated that embodied evolution can provide a more adaptive and fault-tolerant design process than the off-line and centralised counterpart neuro-evolution methods, major challenges of on-line methods remain [7], [55]. First, the requirement of embodied evolution burdens the robotic platform. Individual robots must carry software that executes the design process as well as the control software. Typical robotic platforms used in swarm robotic research embody limited hardware; therefore, additional requirements on memory and computation to the control instances could not be viable. Second, the on-line design process may take a relatively long time. The lengthy design process may consume power more than the batteries can provide. Also, when the mission requires timely execution, a long design process must be avoided, e.g. rescue missions. Third, the on-line design process might cause permanent damage to physical robots. Since the purpose of the design process is to find the best controller by deriving an optimal solution in the given search space, robots under the execution of embodied evolution must explore the controller instances that may cause physical damage. Finally, the individual robot may not effectively assess the performance of the chosen genome run on itself. It is challenging to evaluate the chosen genome for the individual in terms of global performance as robots utilise local perception only.

Existing research studies in embodied evolution have not devised solutions for the first three

challenges, which are caused by the intrinsic nature of the mechanism of embodied evolution. On the other hand, the challenge of having no global knowledge of performance assessment of individual robots has been addressed with three solutions: open-ended evolution, decomposition and simulation-based assessment. Open-ended evolution is an evolutionary algorithm that has no explicit objective function but assesses the controller instance based on its ability to reproduce [74], [79], [80]. In other words, it works under the principle of "Survival of the fittest". In open-ended evolution, implicit selection pressure is exerted based on the survivability of genomes of controller instances than explicit selection pressure with the fitness function. Over time, the genomes with the highest survivability will remain last. Open-ended evolution does not aim to generate desired behaviour, but it encourages the complex behaviours to emerge based on survivability.

Decomposition of the global objective function into rewards for the actions of individual robots [75] is another solution to mitigate the lack of performance assessment by individuals at a global level. Despite its plausibility, it is particularly difficult for the tasks where cooperation is required or delayed reward is given to individual robots. In these tasks, the contribution of the action of individual robots is not clear. Therefore, optimisation upon the decomposed objective function may not lead the optimal global behaviour of a robot swarm. This challenge is also highlighted in the field of robot learning with a different name: credit assignment.

Simulation-based assessment alleviates the credit assignment problem of decomposition of the global objective function. It is a relatively new method that utilises simulation to evaluate the quality of individual genomes. Jones et al. [81] proposed a behaviour-tree based on-line evolution algorithm that uses on-board simulator to assess the quality of the genome. Using a simple 2D physics-based simulator that is executed on the onboard computer of robots, the performance of the controller instance is assessed in a holistic view. Although it mitigates the credit assignment problem, it might encounter the issue of computational burden as it runs a simulator in the onboard computer where the control instance and design process are executed. Furthermore, another type of reality-gap between the assessment in the simulated environment and the reality can cause inaccuracy. For example, the actual behaviour of the neighbouring robots might differ from those in the simulated assessment of collective behaviours.

2.3.3 Automatic Modular Design

The issue of reality-gap has emerged from the off-line controller design methods. Online methods including embodied evolution attempt to minimise the reality-gap by implementing a design process in the target environment. Although the reality-gap may be alleviated through on-line methods, several inherent challenges including possible harm and time constraints that a robot swarm could encounter during the design process still remain.

Automatic modular design is a more recently highlighted design method that provides pre-designed modules utilising domain knowledge in the automatic design process to achieve better results [5], [7], [70], in contrast to neuro-evolutionary approaches that require no *a priori* domain knowledge.

Instead of using ANNs as a controller instance, automatic modular design methods utilise controllers constituted of multiple software modules that form a greater complex architecture. Typical examples of the software modules used for automatic modular design are finite state machines (FSMs) and behaviour trees (BTs). The software modules are pre-designed by human experts harnessing the domain knowledge.

Not only does integration of domain knowledge contribute to better controller performance, but its modular structure alleviates the issue of reality-gap of off-line design methods. The problem of reality-gap in off-line neuro-evolution approach is deemed as the issue of over-fitting, which means that the parameters of control instances are excessively specialised to perform the task in the setting where the controller is designed [5], [54], [67]. In the fields of statistics or machine learning, over-fitting refers to the state of modelling error when a function is too closely tailored to a limited set of data points. When the function that forms a model from the given set of data points is over-fitted, it generates output with a large error when a new data point is input. Likewise, from the point of view where the problem of reality-gap is regarded as an over-fitting issue, the controller designed with neuro-evolution is too optimised for the simulated environment to produce useful output in the target environment while a robot swarm is deployed for the mission. In other words, ANNs provide a large representational power which leads to counter-productive results when it is deployed in the target environment. By intentionally decreasing the representational power of the control instance, the issue of reality-gap can be mitigated. Automatic modular design methods reduce representational power by using a combination of software modules that define low-level behaviour of a robot instead of ANNs.

AutoMoDe [70] is a general framework of automatic controller design where a modular software architecture is generated by assembling predefined low-level modules. In AutoMoDe, room for choosing predefined modules and optimisation algorithms for assembling and fine-tuning is given to designers for further specification.

In the first original paper, the proof-of-concept reference model of AutoMoDe, called AutoMoDe-`vanilla` [70], is proposed adopting finite state machine as the modular architecture and F-race [82] as the optimisation algorithm. Six low-level behaviours and six conditions constitute the modular architecture. For more detail, please see … In comparison with the baseline off-line automatic design method called EvoStick [70], AutoMoDe was outperformed by EvoStick in the simulated environment, but surpassing the performance in the real environment. AutoMoDe is also compared with the two other methods in which human experts were involved: U-Human and C-Human. In U-Human, the human expert was left free to design controllers. Less flexibly, in C-Human, the human experts designed controller with the same modules given for AutoMoDe. In other words, the human experts played the role of optimisation in C-Human. AutoMoDe outperformed U-Human but was outperformed by C-Human. This comparison in results between AutoMoDe and two human designed controllers present that the modular structure of AutoMoDe provided benefit in designing controllers over manual controller design, which is the main point of automatic controller design, but the optimisation algorithm used in AutoMoDe was limited.

One variant of AutoMoDe-vanilla is called AutoMoDe-chocolate [83].
In AutoMoDe-chocolate, an improved version of F-race algorithm, called Iterative F-race [84], is used and showed the greater performance than the AutoMoDe-vanilla and C-Human algorithm. This result shows that the choice of optimisation algorithm is a major factor of the performance of AutoMoDe.

Another variant of AutoMoDe-vanilla, called AutoMoDe-maple [85], replaced finite state machine based architecture with behaviour tree. Comparative analysis on the performance of the swarm between AutoMoDe-vanilla and AutoMoDe-maple presented that there is no distinguishable difference in terms of overcoming reality-gap. This means that the mitigating effect from reality-gap of AutoMoDe series is obtained from the modular structure of the design, not a specific choice of modular architecture.

Apart from the introduced variants, several more variants have been proposed to improve AutoMoDe, for instance, by considering economic trade-offs [86], different modules for specific exploration tasks, different optimisation algorithm choices.

Although AutoMoDe families showed that the low representational power of automatic modular design mitigated reality-gap, several limitations exist inherently in the mechanism of automatic modular design. Unlike neuro-evolution approaches that do not necessitate pre-designed modules, automatic modular design requires the design cost of low-level modules for specific missions. Designing low-level modules not only requires substantial cost but also restricts the generalisation of the swarm to perform diverse missions. Since the low-level modules are specialised for performing the tasks aimed during the design process, the swarm is likely to perform the tasks out of consideration. Ironically, this shortcoming of automatic modular design is identical to the problem of manual controller design. To resolve this limitation, a general set of modular architecture must be proposed [7], [54].

2.3.4 Robot Learning

Robot learning presents a viable alternative to robot evolution methodologies for automatic controller design, as detailed in recent trends in robot learning and swarm studies [7]. Many robot learning techniques capitalise on Reinforcement Learning (RL) to fine-tune the robot controllers to enact desired behaviours. RL employs data acquired from interactions between an agent and target environments to optimise the controller [87]. The success rate is determined by a predefined objective function that utilises the gathered data. The designated optimisation function enhances the controller in such a way that the calculated score is maximised. For further insight into the working principles of RL, please refer to Section 2.4.

Pioneering applications of RL in swarm robotics were introduced by Mataric [88], [89]. In the initial study, four robots executed a foraging task in a noisy and dynamic environment utilising RL [88]. This approach encountered complexities in state space, prompting the introduction of state clustering methods to streamline the input space for the RL algorithm. However, it maintained

a centralised RL structure, clashing with the decentralised nature of swarm robotics. Subsequent research proposed communication strategies to facilitate decentralised RL, addressing critical issues in multi-agent RL environments [89].

Several notable instances demonstrate the application of RL in multi-robot systems with a limited number of agents. For example, Chen et al. put forward decentralised multi-robot collision avoidance algorithms using DRL for up to six robots, showcasing significant advantages, particularly as task complexity increased [90]. Meanwhile, Hu et al. developed a multi-robot navigation system capable of manoeuvring in a realistic domestic setting, significantly outperforming traditional control methods [91]. Huttenrauch et al. were the first to apply DRL within a swarm robotics context, introducing a method to address the fluctuating state space inherent in swarm robotics [92]. Despite the promising results in simulated environments, real-world validations remain scarce.

While the studies mentioned above undeniably highlight the advantages of DRL-based automatic controller design over traditional manual approaches, they also unveil several limitations. Typically, these works employ a Multi-Agent Reinforcement Learning (MARL) framework, with controllers undergoing centralised training before decentralised deployment. This centralised aspect contradicts the decentralised principles of swarm robotics, potentially making real-time updates unfeasible during actual swarm robotic missions. Moreover, the predominance of centralised training restricts online design adaptations, potentially hindering applications in more realistic settings due to the lack of real-time, decentralised training capabilities.

Both robot evolution and learning methods optimise controller instances based on a predefined objective or fitness function crafted by designers to achieve a specific collective behaviour. Robot learning, however, offers a more direct approach to optimising controllers using this function, whereas robot evolution primarily evaluates controllers to select the optimal instances. During the design process, robot evolution methods are not influenced by the objective function, allowing for a broader exploration of search space. In contrast, robot learning strives to maximise the score derived from the objective function, facilitating quicker optimisation but potentially limiting exploration to narrower spaces. Despite these benefits, the reliance on real-world data remains crucial, with the majority of current research utilising simulated environments, a practice that may exacerbate the reality gap.

Automated modular design techniques can potentially bridge this gap by employing modular structures to adjust the bias-variance trade-off more effectively. This adds more complexity and expert knowledge that counteract with the purpose of automatic controller design. On the other hand, robot learning methods could offer more direct and simple measures to manage bias-variance trade-off, such as managing Artificial Neural Network sizes or introducing early stopping during training.

Unlike the clear delineation between off-line and on-line design strategies in robot evolution methods, robot learning offers a more integrated approach to the design process. While initial designs can be developed off-line using simulated environments, data from real-world scenarios can be

integrated to enhance the generalisation of controller instances, thereby mitigating the reality gap. Furthermore, robot learning techniques can exclusively utilise real-world data for on-line design, potentially reducing design time and safeguarding robots from physical damage or battery depletion during missions [93], [94].

Applying transfer learning, a method that leverages pre-trained models for quicker adaptation to target tasks, can further reduce design time and the likelihood of robot damage during on-line design processes, helping to bridge the reality gap [95], [96]. Though integrating both off-line and on-line design methods within robot learning for swarm robotic systems remains an unresolved challenge, various applications in other robotic domains suggest that a hybrid approach could potentially enhance the design of swarm robotic systems, effectively reducing both reality gap issues and risks associated with exploration in target environments.

One critical aspect in the development of evolutionary swarm robotics is the challenge of creating suitable objective functions that guide individual agents to produce desired collective behaviours. This process is notably more complex within the realm of robot learning, where agents are not only assessed but optimised based on these functions. Strategies to simplify the design of these functions are being developed, including imitation learning, which uses example trajectories as a blueprint for controller instance optimisation [97], [98]. Learning from demonstration techniques, which analyse individual agent behaviours from supplied videos to optimise controllers, are also being explored [99]–[101]. Though these techniques provide avenues for developing effective solutions, they are contingent upon the availability of optimal behaviour examples, potentially limiting exploratory advancements. Nonetheless, promising results can still be achieved with simplistic reward designs and appropriate robot learning algorithms [102].

Despite the benefits demonstrated by robot learning methods, the current reliance on the centralised MARL framework restricts further progress, especially concerning on-line design processes. The development of decentralised implementation strategies is therefore critical to advancing robot learning methods, fostering real-world applications in complex and dynamic environments while minimising the reality gap.

In comparison to robot evolution, robot learning remains in its infancy, albeit with significant potential to emerge as a comprehensive methodology for automatic controller design, addressing major shortcomings of existing strategies. Given the limited number of studies that specifically apply robot learning to swarm robotic systems, further research is necessary to validate its efficacy as a universal approach for automatic controller design, encompassing the development of fully decentralised robot learning implementations.

2.4 Deep Reinforcement Learning

In the section 2.3.4, Deep Reinforcement Learning (DRL) emerges as a predominant strategy in the domain of robot learning methods, holding promise despite its infancy in the field of automatic

controller design. This section delves deeper, providing a comprehensive and technical illustration to facilitate a better understanding of the principles governing DRL-driven automatic controller designs, as a prerequisite for Chapters 4 and 5.

Reinforcement learning (RL) is an automated approach for finding the best solution to a task by maximising a numerical reward signal [87]. In RL, a subject that performs a task is defined as an agent. An agent interacts with the environment and obtains transition samples. Based on the current state information, an agent decides what action it should take and in response the environment returns a numerical reward signal and information regarding the next state. This process involves the transfer of information relating to the transition from one state to another, as well as action and reward data and hence is referred to as a transition sample. Using the transition samples, RL can be applied to find the best solution for an agent to maximise total cumulative reward, i.e. an optimal policy, where policy refers to the mapping between states and actions.

In RL, the problem is generally modelled as a Markov Decision Process (MDP) defined by a tuple (S, A, P, R, γ) [103], where:

- S: the state space, representing all possible states the environment can return to the agent.
- A: the action space, representing all possible actions the agent can take.
- P: the state transition probability, $P(s'|s, a)$, describing the probability of transitioning to state s' from state s by taking action a.
- R: the reward function, $R(s, a, s')$, indicating the expected reward for transitioning from state s to state s' by taking action a.
- γ: the discount factor, a value between 0 and 1, representing the importance rate of future rewards in the current state.

This tuple based on the MDP framework, called as transition sample, is collected and used for further computation to derive controller. Value function and policy of an agent can be computed using the transition samples.

The value function is central to RL. Value function evaluates how good a state or a state-action pair is in a given environment. There are two main types of value functions:

1. State-Value function ($V^\pi(s)$): It represents the expected cumulative reward of starting from state s and following policy π, mathematically defined as:

$$V^\pi(s) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t R_t \mid S_0 = s \right]$$

2. Action-Value function ($Q^{\pi}(s, a)$): It represents the expected cumulative reward of starting from state s, taking action a, and thereafter following policy π, mathematically defined as:

$$Q^{\pi}(s, a) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^{t} R_{t} \mid S_{0} = s, A_{0} = a\right]$$

A policy (π) is a strategy that the agent uses to determine its actions. It is mathematically represented as a mapping from states to actions. The goal in RL is to find an optimal policy π^{*} that maximises the expected cumulative reward.

The recursive nature of the value functions can be described by the Bellman equations. Bellman equations characterises the relationship between the value of a state and the value of its successor state. For the state-value function, it is given by:

$$V^{\pi}(s) = \sum_{a} \pi(a|s) \sum_{s', r} p(s', r|s, a) \left[r + \gamma V^{\pi}(s')\right]$$

For the action-value function, it is given by:

$$Q^{\pi}(s, a) = \sum_{s', r} p(s', r|s, a) \left[r + \gamma \sum_{a'} \pi(a'|s') Q^{\pi}(s', a')\right]$$

Since the goal of RL is to find the optimal policy that maximises total cumulative reward that an agent receives in an episode.

Several algorithms have been devised to find the optimal policy. The following three algorithms are the most basic algorithms to derive optimal policy [87]:

1. **Value Iteration**: A dynamic programming method that iteratively updates the value function until it converges to find the optimal value function and thereby the optimal policy.

2. **Policy Iteration**: Another dynamic programming method that alternatively updates the policy and the value function to find the optimal policy.

3. **Q-Learning**: A model-free method that iteratively learns the Q-value function, converging to the optimal Q-value function and thereby facilitating the discovery of the optimal policy [104].

RL has manifested its efficacy in several domains where optimal solutions are vital, encompassing areas such as robotics [105] and industrial process control [106]. Despite RL's demonstrated ability to derive optimal solutions, there exist limitations that pose significant challenges for its application.

A fundamental challenge in RL is the 'curse of dimensionality'. This refers to the exponential increase in computational demands as the state or action space within a problem expands, rendering

RL unsuitable for many real-world contexts. Given that real-world problems are characterised by an extensive number of states and actions, the task of computing such vast state and action spaces, along with the ensuing complexity, becomes extremely intricate.

In recent times, the emergence of Deep Learning (DL) and advancements in infrastructure facilitating high-speed computations have given rise to the rapid development of Deep Reinforcement Learning (DRL). DRL, which integrates Artificial Neural Networks (ANNs) to approximate value functions and policies in environments with high-dimensional state and action spaces, was first introduced in [107], showcasing its ability to play video games with high-dimensional inputs, surpassing human experts. This seminal work rapidly encouraged the application of DRL in applications and algorithmic developments across various sectors including robotics [108], [109], autonomous vehicles [110], healthcare [111], finance [112], and chip design [113].

Current DRL algorithms can be categorised into three distinct branches, depending on the functions that are approximated by the neural networks utilised to ascertain the optimal policy. The first method, known as the policy-based method, employs the neural network to approximate the policy function, which is optimised subsequently [114], [115]. The second strategy, deemed as the value-based method, cultivates a value network to derive an optimal policy [107]. Here, the value network delineates the merit of an agent being in a particular state, directing the policy to steer the agent towards states with the highest values. The third branch is the actor-critic method [116], which nurtures both policy and value networks, utilising them together to craft an optimal policy. These networks, referred to as actor and critic networks, undertake the roles of action execution and evaluation in respective states.

2.4.1 Deep Deterministic Policy Gradients

Deep Deterministic Policy Gradients (DDPG) [117] algorithm played a vital role as a foundational backbone of DRL-driven automatic controller design in Chapter 4 and 5 of this thesis. Here, the justification of the choice and technical detail of DDPG is elaborated.

A critical prerequisite for real-world deployment of swarm robotic systems is the adaptability to inherent complexity and unpredictability. This adaptability necessitates the development of an automatic controller tailored to the unique dynamics of swarm robotic systems. DRL serves this need by providing a data-driven approach to automatic controller design, capable of effectively adapting to unforeseen changes and uncertainties. The selection of a specific DRL algorithm requires a thorough analysis of the characteristics of the agent, the environment, and the intended task. Considerations must include the properties of the robots' input sensory systems, the desired output format, a careful balance between task exploration and exploitation, and the challenges associated with obtaining environmental samples.

DDPG, an actor-critic based DRL algorithm, emerges as a suitable solution for the previously delineated prerequisites of continuous state and action spaces, as well as the need for high sample efficiency. Specifically designed for continuous control tasks, DDPG has been proven to excel in

both simulated and real-world applications [91]. One of its key characteristics is the utilisation of an experience replay buffer, where transition samples are stored and subsequently reused. This feature enhances sample efficiency, a vital specification previously emphasised for swarm robotic systems.

Algorithm 1: DDPG algorithm

Randomly initialise critic neural network with normal linear layers, $Q(s, a|\theta^Q)$ and actor neural network layers $\pi(s|\theta^\pi)$ with parameters θ^Q and θ^π
Initialise target network Q' and π' with parameters $\theta^{Q'} \leftarrow \theta^Q$, $\theta^{\pi'} \leftarrow \theta^\pi$
Initialise replay buffer R
for *episode = 1, ..., M* **do**
    Initialise the states $s_t = s_1$
    for *t = 1, ..., T* **do**
        Run K actors and collect transition samples $D_t = (s_t, a_t, r_t, s_{t+1})$ into a replay buffer
        if *t = 0 mod* t_{train} **then**
            Sample N transitions (s_i, a_i, r_i, s_{i+1}) from a replay buffer with the sampling probability $P(i)$
            Set $y_i = r_i + \gamma Q'(s_i, \pi'(s_{i+1}|\theta^{\pi'})|\theta^{Q'})$
            Set weighted updates for networks: $\omega_i = (\frac{1}{B} \cdot \frac{1}{P(i)})^\beta$
            Update critic network by minimising the loss: $L_Q = \frac{1}{N}\omega \sum_i (y_i - Q(s_i, a_i|\theta^Q))^2$
            Update actor network using the sampled policy gradient: $\nabla_{\theta^\pi} J \approx \frac{1}{N}\omega \sum_i \nabla_a Q(s, a|\theta^Q)|_{s=s_i, a=\pi(s_i)}$
                $\cdot \nabla_{\theta^\pi} \pi(s_i|\theta^\pi)|_{s=s_i}$
        end
        if *t = 0 mod* t_{target} **then**
            Update the target networks: $\theta^{Q'} \leftarrow \theta^Q$,$\theta^{\pi'} \leftarrow \theta^\pi$
        end
    end
end

Moreover, the experience replay buffer's design within DDPG allows for the random selection of transition samples, effectively breaking temporal correlations that could otherwise hinder the training of the neural network. Since correlated data can lead to training failure, this decoupling of consecutive transition samples underscores the necessity of the experience replay buffer [107].

Building upon the foundational concept of experience replay, several research works have further refined this mechanism through diverse sampling strategies, such as hindsight experience replay [118], attentive experience replay [119], and prioritised experience replay [120]. These strategies are tailored to optimisation objectives; for instance, hindsight experience replay is specialised for multi-goal learning scenarios. Among them, Prioritized Experience Replay (PER) [120] stands out as a widely adopted strategy to enhance performance and reduce training time, measuring the importance of transitions through the magnitude of temporal difference error (TD-error) for an effective neural network model update. The probability of sampling transition in PER, *i* is defined as:

$$P(i) = \frac{p_i^\alpha}{\sum_k p_k^\alpha}, \tag{2.1}$$

where α is a probability constant. Higher priority leads to a higher probability of that sample being used during training. Higher sample efficiency and performance has been reported using PER, compared to uniform sampling strategies. Due to its sample efficiency, this sampling strategy is used with DDPG in several continuous control domains including robotics [91], [121]. Algorithm 1 describes the process of DDPG algorithm.

2.5 Federated Learning

Federated Learning (FL) is an advanced decentralised approach to machine learning that facilitates the training of models across multiple devices without necessitating the centralisation of data on a primary server [122]–[124]. This innovative method was principally developed to augment the privacy and security of individual agents, thereby substantially mitigating the risk of data breaches during the training phase. Initially, FL was conceptualised to refine machine learning models involved in the autofill functionalities of smartphone keyboards, with a focus on enhancing the performance of local agents without compromising personal data security at a global server level, whilst capitalising on diversity of multiple heterogenous trained model of local agents [123].

Within the structure of FL, diverse devices, denoted as nodes, independently compute modifications to the model at a local level. Subsequently, these modifications are transmitted to a central server where they are amalgamated to formulate an improved comprehensive global model.

The decentralised operating methodology of FL provides three major benefits: i) privacy preservation, ii) reduction in communication with the central server, iii) customisation of models. No exchange of local training data safeguards the privacy and security of local agent. Moreover, exclusion of data exchange reduce the volume of communication. Although the global model is obtained as a collective of locally trained models, locally trained models are specialised for the local data.

Domains where the aforementioned advantages are particularly helpful have adopted FL-based training strategies for machine learning applications. The two example domains are introduced below.

In the medical sector, Federated Learning (FL) presents substantial advantages, particularly in harnessing data for medical diagnoses without violating privacy norms [125]. Traditionally, local medical authorities have been restricted by the limited diversity and volume of patient data available for training diagnostic models. This is mainly due to the stringent policies that prevent the sharing of patient data beyond local confines. FL stands as a transformative solution to this challenge, empowering local medical authorities to enhance their diagnostic models by leveraging the locally trained models with diverse datasets held by other authorities, all while protecting data privacy and security.

Brisimi et al. [126] devised a binary classifier capable of predicting hospitalisations due to cardiac events by leveraging the extensive network of Electronic Heart Records data dispersed across various hospitals and individuals. This innovative approach not only addresses the inherent privacy and security concerns but also significantly reduces communication costs. Furthermore, it has been observed to hasten the convergence rate when compared to centralised benchmarks.

Warnat-Herresthal et al. [127] have successfully employed classifiers developed through FL-based strategies for the diagnosis of four prominent diseases: COVID-19, tuberculosis, leukaemia, and various lung pathologies. Utilising a comprehensive dataset composed of 16,400 blood transcriptomes

and 95,000 x-ray images sourced from multiple institutions, the study showcased that the proposed classifiers exhibited superior performance compared to those restricted to data from individual sites alone.

The Internet of Things (IoT) is a network where various physical devices and objects are linked together. These items are fitted with sensors and software that help them to share data and information with one another, and with other systems over the internet. Since these devices gather data locally and communicate with other devices, maintaining privacy and security becomes very important. In this scenario, IoT systems can greatly benefit from Federated Learning (FL), a method that can help improve the performance of IoT devices by using data more efficiently and securely. Recently, the use of FL in different IoT systems has grown significantly [128], [129].

Illustrating this, a research study by Yu et al. [130] developed a model for detecting objects using FL. This model used a collection of image data from various IoT devices, showing that the proposed method was more adept than the standard Single Shot MultiBox Detector approach.

Similarly, a project by Elbir et al [131]. devised traffic predictors by implementing machine learning models directly at the peripheral devices, such as vehicles, drawing upon data sets encompassing road layouts, traffic dynamics, and meteorological conditions. Their case studies, which involved the training of 3D object detection models and the selection of Millimeter-Wave Beams for establishing communication links, revealed that the FL-based traffic prediction models surpassed their counterparts that employed centralised training strategies.

FL is predominantly used for supervised learning applications. Including the two aforementioned example domains, most of the current FL applications have been adopted for regression and classification tasks. More recently, FL has also been applied for the domains where local agent performs sequential decision-making, e.g. control, and time-series prediction, and adaptivity is required in dynamic environments, where the dataset cannot be given before operation. For an effective training model that performs sequential decision-making well, DRL can be used. FL framework for DRL has been proposed for the collective of local agents performing sequential decision-making tasks. IoT is the typical application domain as FL-based DRL provides the capability for highly complex decision-making to orchestrate multiple resource-intensive devices [111], [132], [133].

With the proven performance in classification tasks and the potential that enhance decision-making capability when multiple agents are collaborating without privacy concerns, the nascent FL-based methods show great promise in robotics applications [134], [135]. Liu et al. [134] utilised FL-based DRL for designing the controller of a robot for navigation and collision avoidance tasks over diverse environments. The trained controller showed a greater performance than when trained by DRL without FL framework. However, the robot learns the controller sequentially, rather than multiple robots in separate environments learning controllers simultaneously harnessing FL. Yu et al. [135] proposed FL-based image classifiers used for collaborative robotic systems. While the end-to-end controller is learned utilising FL in [134], FL is used to learn image classifiers to detect obstacles for navigation and collision avoidance behaviour. Importantly, the results demonstrated that reality-gap

is reduced as the FL-based method enables continuous updates to the deployed models in the real mission, showing promises to mitigate reality-gap through the FL-based approaches.

As introduced in the existing literature, FL is an emerging framework where a system with multiple agents requires an aggregated model preserving privacy and security and reducing communications, yet leveraging collective capability that results in greater generalisation and faster speed of convergence than single agent is used.

Chapter 3

The Imperative Need of Automatic Controller Design for Swarm Robotic Systems in Real-World Conditions

This chapter unveils a research study underscoring the vital need for automatic control design in swarm robotic systems. The investigation delves deeply into the behaviours exhibited by swarm robots in various real-world factors that have not been fully explored in laboratory settings where past research has predominantly been situated. In this study, a mobile-robot-based swarm robotic system demonstrates aggregation behaviour through the use of artificial pheromones as a medium of communication. Building upon the foundations laid by previous research, which employed static artificial pheromone cues to analyse aggregation behaviours, this study incorporates positive feedback mechanisms and realistic environmental factors to create a more realistic representation of swarm aggregation behaviours. Specifically, environmental factors such as diffusion and wind effects have been integrated into the emulated artificial pheromone. A set of experiments of swarm robotic systems with varying population sizes was conducted to unravel the intricate effects of these factors on aggregation behaviour. The experimental findings indicate a substantial variations in aggregation behaviour when confronted with realistic factors, thereby advocating for the development of more adaptive control design methodologies. Such advancements are essential in facilitating the actualisation of desired swarm behaviours in real-world scenarios, highlighting a promising pathway towards more robust and adaptive swarm robotic systems.

3.1 Background

Capitalising pheromone communication that enable nature swarms thrive, swarm robotic systems that utilises artificial pheromone have been gradually developed. One of the past studies proposed COSΦ [46], a light-based artificial pheromone system that leverages low-cost components to build a physical platform to experiment with real robots easily. Given versatility of the system, it is easy to emulate artificial pheromone under diverse simulated scenarios with arbitrary properties.

Previously, a seminal study [46] explored foraging behaviours of a mobile robot swarm relying on pheromone communication [136]. The successful implementation of foraging behaviour via artificial pheromone communication leaves two directions of subsequent studies: (1) the exploration of diverse behaviours and (2) the replication of realistic factors to examine the viability of artificial pheromone-based swarm robotic systems in real-world environments. Through branching out in these research directions, it is anticipated that critical insights necessary to development of a swarm robotic system capitalising on artificial pheromones in real-world settings will be unveiled.

In this research study, aggregation behaviour, which is a basic self-organising pattern of swarm robotic system, is investigated. To replicate realistic environmental factors, diffusion and wind effects have been applied into the dynamics of artificial pheromone. For further advancement, positive feedback loop for strengthening aggregation behaviours has been added. A detailed comparative analysis has been undertaken to dissect the influence of each integrated real-world factor on the swarm behaviours. The experimental findings suggest that these factors not only induce significant variations in aggregation behaviours but their interplay further exacerbates the complexity of behavioural patterns. The highlighted influence of realistic factors underscores the imperative need for more adaptive controllers, capable of maintaining the desired behavioural attributes with changing environmental dynamics.

To summarise, this research study contains the following contributions and findings:

- Positive feedback (pheromone injection), diffusion and wind effects are added to the existing light-based artificial pheromone system [46]. The system is extended to emulate the phenomena where pheromones in the real world are exposed.
- The effect of positive feedback, diffusion and wind effects are investigated on aggregation behaviours of a swarm of mobile robots with different population sizes, concluding that the interplay of factors influences aggregation behaviours in a complex and unpredictable manner.
- Complex interplay of realistic factors necessitates more adaptive controllers for swarm robotic systems to achieve desired performance efficacy.

3.1.1 Technical Detail of COSΦ System

The COSΦ system simulates the several types of artificial pheromone and their interaction simultaneously. The resultant artificial pheromone is displayed on an LCD screen, which plays a role of ground arena for robots, and it triggers predefined responsive behaviours of the employed robots. The system displays the resultant pheromone and the rest part of the arena as a gray-scale image with the size of the screen. The degree of brightness of all pixels in a grey-scale image is represented as $\mathbf{I}$, which is a two dimensional matrix with the size of resolution of the LCD screen. It is determined by $\mathbf{\Phi}$, which is a two dimensional matrix representing the intensity of pheromone having same size with $\mathbf{I}$. Each element of $\mathbf{I}$, is equivalent to brightness of the corresponding pixel. The brightness of the image at position (x, y), $\mathbf{I}(x, y)$, is defined as:

$$\mathbf{I}(x,y) = \sum_{i=1}^{n} c_i \mathbf{\Phi_i}(x,y) \quad , \tag{3.1}$$

where $\mathbf{\Phi_i}(x, y)$ represents the intensity of the i^{th} pheromone at position (x, y) and c_i denotes the influence of the i^{th} pheromone on the screen. $\mathbf{I}(x, y)$ is determined by the summation of multiplication of $\mathbf{\Phi_i}(x, y)$ and c_i, describing that multiple n number of individual pheromone fields can be overlapped. As an example of how the model works, the combination of three different pheromones having different influence to the screen can be displayed on a single pixel after the calculation of $\mathbf{I}(x, y)$ by Equation 3.1.

Individual pheromone field is updated every time step in the system. To replicate spatio-temporal development of the pheromone in the real world, the spatio-temporal developement model is derived from the simplified version of Navier-Stokes equation, which characterises the model of fluid flow [137], [138]. The spatio-temporal development model of pheromone is given by

$$\begin{aligned} {\mathbf{\Phi}_i}^{k+1}(x,y) = &- \mathbf{u} \cdot {}^{grad}{\mathbf{\Phi}_i}^{k}(x,y) - \frac{\ln(2)}{e_{i\Phi}} {\mathbf{\Phi}_i}^{k}(x,y) \\ &+ \kappa_i \nabla {}^{grad}{\mathbf{\Phi}_i}^{k}(x,y) + \iota_i(x,y) \quad , \end{aligned} \tag{3.2}$$

where $\mathbf{\Phi_i^{k+1}(x, y)}$ is the intensity of i^{th} pheromone at the discrete time $k + 1$, $\mathbf{\Phi_i^k(x, y)}$ is the intensity of i^{th} pheromone at the discrete time k, ${}^{grad}{\mathbf{\Phi}_i}^{k}(x, y)$ is a two dimensional vector quantity that characterises the gradient of pheromone intensity at a given position (x, y), where it is mathematically defined by Equation 3.3, $\mathbf{u}$ represents the velocity vector which linearly shifts the pheromone on the arena, resembling the effect of wind, $e_{i\Phi}$ determines the evaporation rate of i^{th} pheromone which is characterised by half-life, κ_i is the diffusion constant of i^{th} pheromone, and $\iota_i(x, y)$ corresponds to newly injected pheromone at the position (x, y) on the screen.

$$\begin{aligned} {}^{grad}{\mathbf{\Phi}_i}^{k}(x,y) = &\frac{{\mathbf{\Phi}_i}^{k}(x+1,y) - {\mathbf{\Phi}_i}^{k}(x-1,y)}{2}\mathbf{i} + \\ &\frac{{\mathbf{\Phi}_i}^{k}(x,y+1) - {\mathbf{\Phi}_i}^{k}(x,y-1)}{2}\mathbf{j} \quad , \end{aligned} \tag{3.3}$$

Computing Equation 3.2 for every values of x, y allows to calculate new intensities of pheromones from their previous state, which, in turn, determines the new pixel values of the gray-scale image displayed on the screen.

The parameters shown in the right-hand side of Equation 3.2 can be divided into two categories: i) environmental effects and ii) pheromone injection. Environmental effects include evaporation rate, e_Φ, diffusion constant, κ, and velocity vector, **u**, where i is omitted to generalise. They influence in the pheromone released on the arena unconditionally and constantly while the system is running. Their effects are described in the next subsection. While the environmental effects have constant influences in the arena, the injection of pheromone, $\iota(x, y)$ affects the intensity of pheromone in the arena only when the pheromone is injected by the predefined conditions. Under the conditions, the pheromone is injected with a circular shape with a given intensity. The injection of pheromone, $\iota_i(x, y)$, is defined as:

$$\iota_i(x,y) = \begin{cases} s_\Phi, & \text{if } \sqrt{(x-x_r)^2 + (y-y_r)^2} \leq l_\Phi/2 \\ 0, & \text{otherwise} \end{cases} \tag{3.4}$$

where (x_r, y_r) respectively represent the x and y coordinates of a robot in the arena, s_Φ is the intensity of injected pheromone at a time and l_Φ is the diameter of injected pheromone. Within a circle with the diameter, l_Φ, where the centre of the circle is the position of the robot, (x_r, y_r), the pheromone is uniformly injected with s_Φ. Pheromone injection is used as a mean of positive feedback in the computational model of artificial pheromone.

3.2 Artificial Pheromone System

In this section, the extended part of COSΦ artificial pheromone communication system, and the auxiliary tracking system used to provide positional information of individual robots and the artificial pheromone cue are explained. For the COSΦ system, the mechanism of the system and the mathematical model to express the property of artificial pheromone are described with a detailed explanation of each environmental effect applied on spatio-temporal development of artificial pheromone. Following the description of the COSΦ system, the open-source localisation system for swarm robotic system, SwarmCon [139], and integration between the tracking system and the COSΦ system are described.

3.2.1 Environmental Effects on Pheromone

In the dynamics model of artificial pheromone described in Section 2.2.3, environmental effects, evaporation, diffusion and wind effect, are included, but not fully utilised. Here, each environmental effect is described and the practical implementation detail is elucidated.

The environmental effects contribute to the versatility of the system in emulating realistic conditions

that affect the distribution of the pheromones in the environment over time. More importantly, the implementation of diffusion and wind effect is crucial in this research study to investigate the swarm aggregation behaviour under realistic conditions. Here, the four phenomena included in the spatio-temporal development model are introduced and explained with its implementation detail in the system.

Evaporation:

Evaporation is the process by which the surface of a liquid turns into the gas phase. As volatile chemical substances, evaporation occurs in secreted pheromones. In several works on the kinetic properties of pheromones, the half-life of pheromones, which is the time required for pheromones to decay by half, is investigated as a metric [140]. Conforming this practice, the half-life of the pheromone is adopted, i.e. $e_{i\Phi}$ represents the half-life of the pheromone, see Equation 3.2.

Diffusion:

Diffusion is a movement of molecules from a region of higher concentration to a region of lower concentration. Implementing diffusion in the pheromone system is indispensable to be a realistic system since diffused pheromone from the source has a great impact on swarm behaviour [141]. In this work, diffusion is implemented using Gaussian blur instead of directly using the term, $\kappa_i \bigtriangledown^2 \Phi_i(x, y)$, which is a mathematical definition of diffusion. The advantage of Gaussian blur over the original definition is it can emulate faster diffusion with lower computational costs. The intensity of pheromone at the position (x, y) after application of the Gaussian blur is given by

$$\begin{aligned} \boldsymbol{\Phi_i^{k+1}}(x, y) &= \big(\boldsymbol{\omega} * \boldsymbol{\Phi_i^k}\big)(x, y) = \\ &\sum_{s=-a}^{a} \sum_{t=-b}^{b} \boldsymbol{\omega}(s, t) \boldsymbol{\Phi_i^k}(x - s, y - t), \end{aligned} \tag{3.5}$$

where $\Phi_i^{k+1}(x, y)$ is the intensity of i^{th} pheromone at the discrete time $k+1$, $\Phi_i^k(x, y)$ is the intensity of i^{th} pheromone at the discrete time k and ω is a two dimensional kernel matrix with the size of $(2a + 1) \times (2b + 1)$ defined as:

$$\boldsymbol{\omega}(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}, \quad w \in \mathbb{R}^{2a+1 \times 2b+1}, \tag{3.6}$$

where σ is a standard deviation of the Gaussian distribution of the kernel matrix. Equation 3.6 shows that the elements of ω are determined by the Gaussian distribution. Apart from the computational efficiency, the Gaussian blur allows more intuitive control of the diffusion rate and area. Adopting Gaussian blur into the pheromone system does not principally violate properties of diffusion for two reasons: i) The higher intensity of pheromone decreases and the lower intensity of pheromone increases after computation of Gaussian blur and ii) the total amount of pheromone is preserved after every computation.

Wind Effect:

In real and dynamic environments, the position of the released pheromone can be shifted. A natural cause of the shift in nature is *advection*, which is the flow of any fluid, e.g. air, that transfers pheromone from one position to another. As it is described as wind effect, the shift in the spatio-temporal model is named as wind effect. The movement of released pheromone in the same direction is modelled as $\mathbf{u} \cdot \nabla(\mathbf{x}, \mathbf{y})$. The two dimensional velocity vector is defined as:

$$\mathbf{u} \cdot \nabla_{\mathbf{i}}(x, y) = u_x \cdot \frac{\partial \mathbf{\Phi_i}(x, y)}{\partial x} + u_y \cdot \frac{\partial \mathbf{\Phi_i}(x, y)}{\partial y} \quad , \tag{3.7}$$

where u_x, u_y respectively represents the speed along x-axis and y-axis.

3.2.2 Tracking System

The modified version of fast and precise open-source localisation system [139], called SwarmCon, is used to track the position, orientation and ID of the robots, thereby precisely injecting pheromone at the position of the robots on the LCD screen in real-time. The system captures images using a digital camera mounted above the screen, searches for black-and-white roundel patterns, and converts their image coordinates into the real-world coordinates in the Cartesian plane. By attaching unique patterns on the individual robots, each robot can have unique ID. The localisation precision can achieve millimeters scales and hundreds of patterns can be tracked simultaneously in real-time [139].

By transforming the captured images into the coordinates of the tracked patterns in the defined area and sending the data to the pheromone system, the tracking system allows the pheromone system to release the pheromone at precise positions of the robots in real-time. Precise localisation of the patterns can be achieved through auto-calibration of the system that makes the pheromone system robust under any circumstances. e.g. external disturbance during experiments. The system defines the area of the arena in the coordinate system from the image by setting the four corner tags attached on the corners of the frame placed on the screen, and determine the coordinates of patterns inside of the defined area. Since the system requires less than 50 µs to calculate one robot position, the difference between the pheromone injection position and the actual robot position is determined by the time required to transfer the image via the USB interface and the delay caused by the graphic interface driver. Nevertheless, the difference between the expected pheromone positions and the actual positions is negligible.

3.3 Experimental Setup

With the expanded COSΦ system, experiments were designed to investigate the impact of positive feedback and environmental effects on collective behaviours in swarm robotic systems.

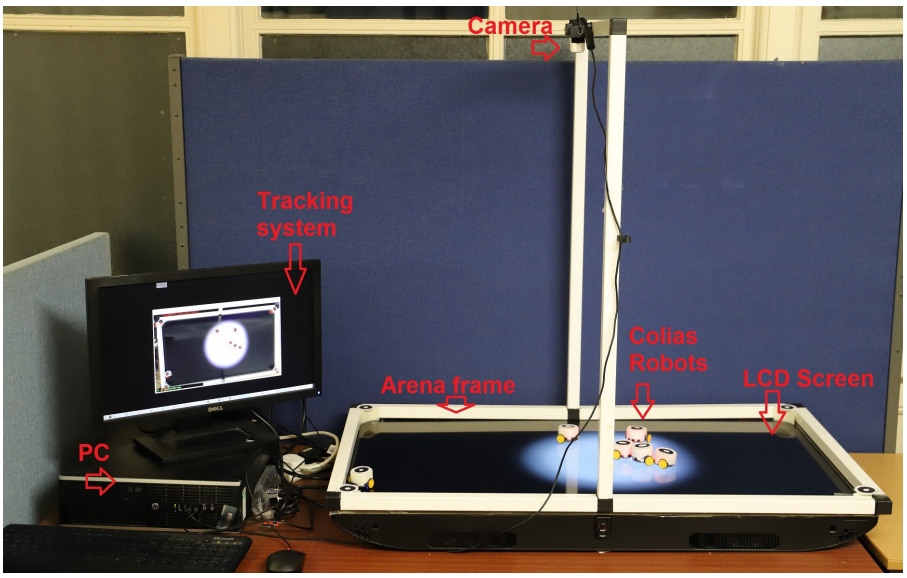

Figure 3.1. Experimental setup used for the pheromone system, including a PC for tracking robots and generating pheromone, a digital camera for tracking robots position, a horizontally placed 42" LCD screen, aluminium frame around the arena and Colias mobile robots.

In this section, the physical configuration of the COSΦ system and the robotic platform is explained. Following the explanation of hardware components used for the systems, the experimental design is described. Not only the description of the experimental design, the metrics to evaluate the performance of the collective behaviours and the statistical analysis method are also explained.

3.3.1 System Configuration

To implement the COSΦ system used for experiments, a physical arena built with the off-the-shelf components was used. In the physical arena, mobile robots were deployed as agents performing collective behaviours as a swarm. The specification of the physical arena and mobile robot is explained in detail.

Arena

In the physical artificial pheromone system, a high-definition (HD) 42" LCD screen with the size of 92×50 cm^2 is used as an arena on which the pheromones are displayed and robots operated. The light sensors of the robots face the screen, hence the sensors allow the robot to read the illuminance at the current position of the robot. On the top of the screen, the aluminium arena frame is set on the edge. The frame allows the robots to detect the boundary of the arena, therefore, they can turn in another direction when they are close to the boundary. The four corner tags are attached to the frame which allows the tracking system to define the arena. A low-cost digital camera is mounted on the frame above the centre of the arena. By receiving the images from the arena in real-time, the tracking system determines the robots' positions and status if the robots are randomly moving or waiting. Figure 3.1 shows the arena setup with a PC which controls the pheromone system, connected to the LCD screen.

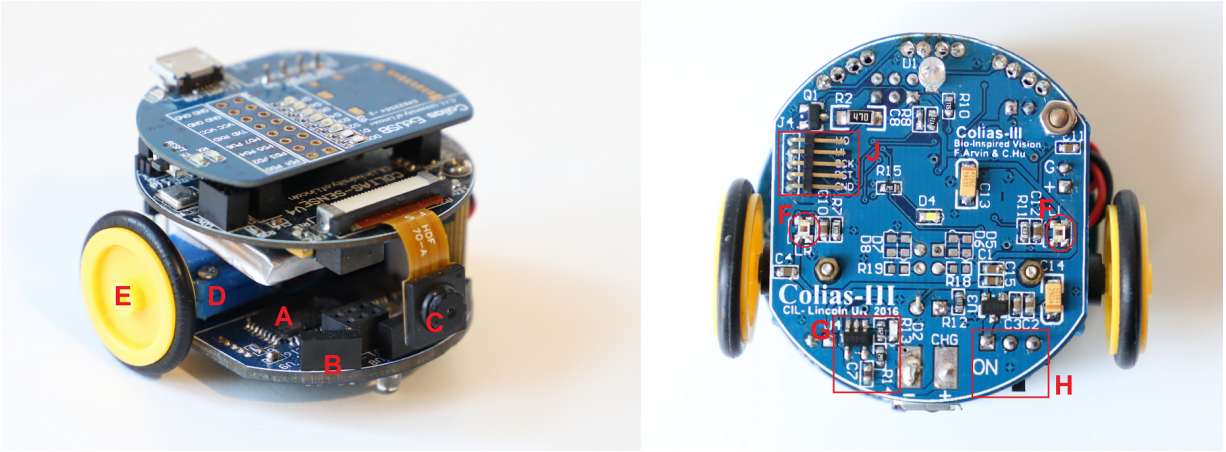

Figure 3.2. Colias micro-robot, a swarm robotic platform (left) frontal view of the robot and (right) bottom board of Colias with pheromone sensing ability. Different modules of Colias are: A) main processor, B) IR proximity sensors, C) digital camera, D) micro-motors with gearhead, E) 22 mm wheels, F) pheromone detectors (light intensity sensors), G) battery recharging unit, H) main switch, and J) ISP programming port.

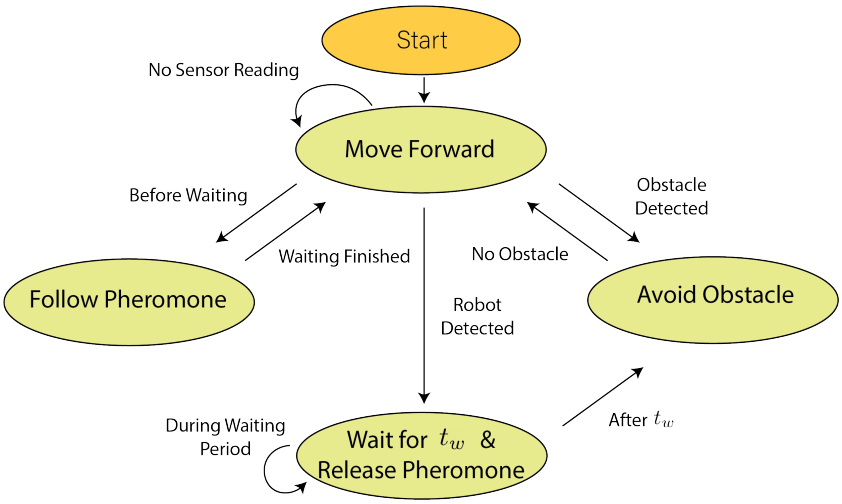

Figure 3.3. State machine of the implemented swarm scenario

Robotic Platform

Colias micro-robot [142] which was developed for swarm robotic applications, was used as a robotic platform. The front and bottom view of the robot is shown in Figure 3.2. It is a small robot with a diameter of 4 cm and has simple functionalities. The robot is a differential wheeled robot whose movement is determined by two micro DC gearhead motors directly connected to the wheel with a diameter of 2.2 cm. The speed of the robot in the forward motion is a maximum of 35 cm/s. The rotational speed of each motor is controlled by pulse-width modulation (PWM). Each motor is driven by the embedded H-bridge DC motor driver which draws an average current of 35$\pm$5 mA when there is no load and 150$\pm$20 mA at maximum in stall conditions. For its sensing, the robot has three infrared (IR) proximity sensors which include pairs of IR emitter and receiver in front of the robot. It is used to detect objects, obstacles or other robots, within a distance of approximately 2$\pm$0.5 cm. Additionally, the robot has two light (illuminance) sensors at the bottom next to the wheels. The light sensors are used to read light intensity on the ground where the robot is located, i.e. read the intensity of the pheromone in this research study. The robot's power consumption is

approximately 800 mW.

In this research study, the behaviour of the robot was determined for achieving pheromone-based aggregation adopted from one of the previous studies [143]. The state machine of the scenario is described in Figure 3.3. The robot begins to move forward after it is switched on. The rotational speed of the left and right wheels, N_l and N_r, are defined as:

$$\begin{aligned} N_l &= \frac{s_l - s_r}{\alpha} + \beta \ , \\ N_r &= \frac{s_r - s_l}{\alpha} + \beta \ , \end{aligned} \tag{3.8}$$

where α is the velocity sensitivity coefficient, β is the biasing speed and s_l and s_r are the sensor readings from the left and right light sensors. β is defined as:

$$\beta = 100 - \frac{s_r + s_l}{2} \ . \tag{3.9}$$

β is modelled in order for the robot to have a slower speed where the average intensity detected by both the sensors is higher so that the robot stops where the robot reaches the area with the considerably high intensity of pheromone. As depicted in Figure 3.3, the robot has two different states transited after it detects an object. If the object is an obstacle, it rotates in the opposite direction at which the obstacle is. The robot distinguishes whether the object is an obstacle or another robot by checking if the IR signal detected by sensors is emitted from other robots, not from ones embodied in the robot itself. If the object is another robot, it transits to the waiting state and starts to inject a pheromone at the position of the robot. The waiting time t_w is defined as:

$$t_w = t_{wmax} \frac{\overline{s}_{avg}}{\overline{s}_{avg} + 25} \quad , \tag{3.10}$$

where t_{wmax} is the maximum waiting time, 20 s, which happens at the highest pheromone intensity, $\overline{s}_{avg}$ is the averaged value of s_l and s_r. Depending on the illuminance on the bottom of the robot, the robot can wait between 0 to 20 s. While the robot is waiting, pheromone of a circular shape with a diameter of 2.5 cm is injected to the environment. The rate of pheromone injection is approximately 20% of the maximum intensity per second, which suggests that it takes approximately 5 s to reach the maximum intensity.

3.3.2 Experiments

Following the objective that is to investigate the impact of realistic conditions on collective behaviour of swarm robotic system, the experiments are systematically designed to ablate the effect of each variable in the extended COSΦ system, i.e. positive feedback and environmental effects. The set of experiments adopted the pheromone-based aggregation scenario as a specific collective behaviour of a swarm. To investigate the effect of the positive feedback (pheromone injection) and the environmental effects (diffusion and wind effect), the settings of positive feedback and

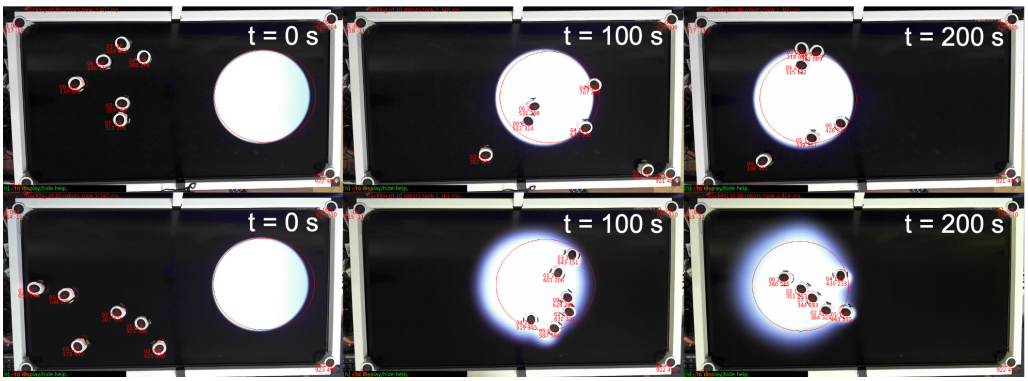

Figure 3.4. The collection of 6 randomly selected screenshots during experiments. The first row shows the screenshots from an experiment with no diffusion and fast cue speed without pheromone injection and the second row shows the screenshots from an experiment with medium diffusion and fast cue speed with pheromone injection taken at $t = 0$ s, $t = 100$ s, $t = 200$ s from left to right.

environmental effects are varied.

The experiments have four independent variables: i) pheromone injection, ii) diffusion, iii) wind effect and iv) population of a robot swarm. The experiments are organised into two main settings based on the presence of wind effect, referred to as static and dynamic cue configurations, describing whether wind effect exists or not. Each cue configuration has three different settings of diffusion coefficients to investigate the interplay of diffusion and wind effect on the aggregation behaviour of the swarm robotic system. For each setting of cue configuration and diffusion coefficient, the presence of pheromone injection called without Φ and with Φ respectively in the following sections, is varied. Finally, every combination of wind effect, diffusion and pheromone injection is experimented with two population sizes, 4 and 6.

In every set of experiments, the identical initial condition was provided for the pheromone-based aggregation scenario. A circular artificial pheromone cue with a diameter of 25 cm was generated at the beginning of the experiments at the position (x_c, y_c) = (70, 25) cm of the arena, which exists in the right half of the screen, where the coordinates (0, 0) refers to the bottom left corner of the arena. The markers indicating the position and orientation of robots were randomly generated within the left half of the screen. Before the experiment began, the robots were placed on the markers, and when the experiments started, the markers disappeared, the robots then started to move and the main circular cue was generated on the specified position simultaneously. The duration of an experiment is $T = 300$ s. Note that, in the experiments in the configuration without Φ, the pheromone was not injected while the robot was waiting after colliding with another robot. This is to investigate the impact of pheromone injection on the collective behaviour of the swarm. The screenshots from experiments with different configuration are shown in Figure 3.4. Table 3.1 shows the parameters and their values used in the experiments.

Static cue configuration

In static cue configuration, the position of the cue (x_c, y_c) remained the same during the experiments. In other words, the cue did not move ($u = 0$ cm/s). In this configuration, three different

diffusion coefficients, $\kappa \in \{0\%, 50\%, 75\%\}/T$ with two swarm sizes of $N \in \{4, 6\}$ robot were investigated. To show the effect of diffusion intuitively, how much pheromone at the centre of the cue is diffused over the duration of the experiment, T, is given as a constant κ. Each diffusion coefficient $\kappa \in \{0\%, 50\%, 75\%\}/T$, described in Equation 3.2 is equivalent to $\sigma \in \{0, 6, 20\}$, $a, b = 7$ for the Gaussian blur kernel matrix, which is characterised in Equation 3.5 and 3.6. To help the readers to understand the impact of each parameter value easily, the three different diffusion coefficients, $\kappa \in \{0\%, 50\%, 75\%\}/T$, are named *"No Diffusion", "Medium Diffusion"* and *"Fast Diffusion"* respectively. For each different diffusion setting in this configuration, 5 independent runs of experiment for T were carried out.

Dynamic cue configuration

In dynamic cue configuration, the position of the cue (x_c, y_c) was moved horizontally with a constant speed. Moreover, the injected pheromone during the experiments also moved at the same speed as the cue. In this configuration, two different cue speeds were applied: i) the centre of the cue moves with speed of $u = 0.1$ cm/s on the arena and the centre of the cue moves with speed of $u = 0.2$ cm/s. The two cue speed is named as *"Medium Cue Speed"* and *"Fast Cue Speed"* respectively to increase readability for readers. Similar to the static cue configuration, 5 independent runs of experiments with three different diffusion, $\kappa \in \{0\%, 50\%, 75\%\} \ / \ T$, and two swarm sizes, $N \in \{4, 6\}$, for duration of T were conducted.

Metrics

To evaluate the aggregating behaviour of the swarm, two parameters were defined: i) size of aggregate, n_a, and ii) cohesiveness, d_{coh}. The size of the aggregate determines how many robots are aggregated, which is equivalent to the number of robots waiting on the circular cue. The robots that are waiting on the outside of the cue because of the injected pheromone are not counted. Cohesiveness determines the quality of the aggregation behaviour of the swarm. Cohesiveness is the reciprocal of the average value of the distances of the robots from the centre of the cue. Higher cohesiveness means the robots stay closer to each other. The cohesiveness is defined as:

$$d_{coh} = \frac{1}{\frac{1}{N}\sum_{i=1}^{N} \|(x_i, y_i), (x_c, y_c)\|} \quad , \tag{3.11}$$

where (x_i, y_i) is the Cartesian coordinates of the i^{th} robot of N robots.

Statistical Analysis

To statistically analyse the observed results from experiments, Analysis of Variance (ANOVA) test was conducted. ANOVA test is used to analyse the difference among groups caused by differences in factors [144]. F-statistic (F) in ANOVA indicates how the factor makes a difference between the

Table 3.1. List of parameters and their values.

Parameter	Description	Value / range	Setting
N	Population	$\{4, 6\}$ robots	
T	Duration of experiments	300 s	
u	Cue speed	$\{0, 0.1, 0.2\}$ cm/s	*No, Medium, Fast Cue Speed*
κ	Diffusion coefficient	$\{0\%, 50\%, 75\%\}/T$	
textitNo, Medium, Fast Diffusion			
e_Φ	Pheromone half-life	1000 s	
t	Time	$\{0 - 300\}$ s	
r_c	Radius of the cue	12.5 cm	
t_{wmax}	Maximum waiting time of robot	20 s	
t_w	Waiting time of robot	$\{0 - 20\}$ s	

means of the samples of different groups. If F-statistic is high, the factor is regarded as a significant factor for sample means. Typically, $F > 1$ is considered a high F-statistic. Additionally, p-value, which is the smallest significance level at which the hypothesis that the factor does not significantly impact on samples is rejected. When $p < 0.05$, it is regarded that the impact of the factor is significant. In thiss set of experiments, multiple independent variables, population, diffusion, cue speed and time, exist for two dependent variables, size of aggregate and cohesiveness. To analyse of each independent variable among multiple independent variables, Factorial ANOVA was adopted.

3.4 Results

The experimental results were depicted in line plots showing the aggregation performance, both size of aggregate and cohesiveness. In the line plots, a line represents the median of the observed data from 5 repetitions and the shaded region surrounding the line represents the inter-quartile range of the data.

3.4.1 Static Cue Configuration

Here, the variance of the aggregation performance of robots is depicted with different diffusion rates in static cue configuration. In each plot, the observed data from experiments with *no diffusion, medium diffusion* and *Fast Diffusion* are represented as red, blue and green lines, respectively. Figures 3.9, 3.10, 3.11 and 3.12 show the size of aggregate and cohesiveness with $N \in \{4, 6\}$ robots.

Diffusion

First, the impact of diffusion on swarm behaviour is investigated. Figures 3.5 and 3.7 shows the size of aggregate with three different diffusion rates (*No Diffusion, Medium Diffusion* and *Fast Diffusion)* in two different population sizes $N \in \{4, 6\}$ both without Φ and with Φ. In both figures, a decrease in the size of the aggregate was observed when *Medium Diffusion* and *Fast Diffusion* was applied

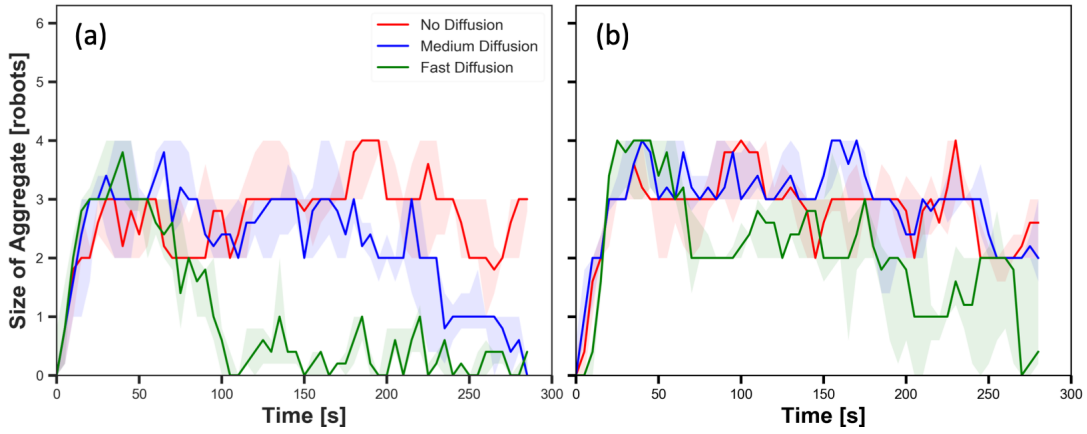

Figure 3.5. The size of aggregate in experiments with static cue configuration, different diffusion rates (*No Diffusion, Medium Diffusion* and *Fast Diffusion* and (a) without and (b) with pheromone Φ injection in $N = 4$ robots

.

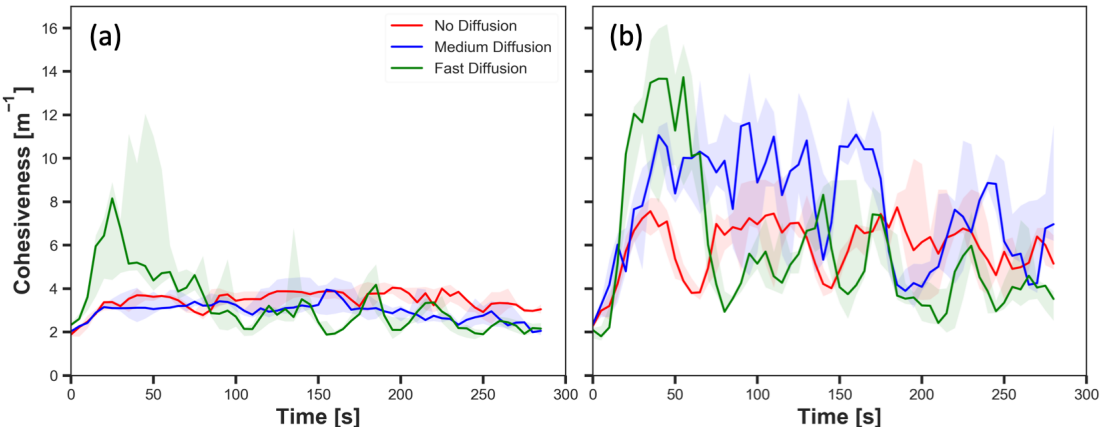

Figure 3.6. The cohesiveness in experiments with static cue configuration, different diffusion rates (*No Diffusion, Medium Diffusion* and *Fast Diffusion*) and (a) without and (b) with pheromone Φ injection in $N = 4$ robots.

without Φ. In Figure 3.5(a), the size of aggregate with *Medium Diffusion* and *Fast Diffusion* began to decrease after a certain time ($t = 70$ s) while the size of aggregate with *No Diffusion* stably stayed in the range from 2 to 4 throughout the experiment. Moreover, the decrease in the size of aggregate with *Fast Diffusion* was more rapid than the decrease with *Medium Diffusion*. In Figure 3.5(a), the size of aggregate with *Fast Diffusion* decreased at about $t = 50$ s and it reached zero at $t = 105$ s whereas the size of aggregate with *Medium Diffusion* started to decrease clearly at $t = 225$ s. These phenomena are also shown in Figure 3.7 with $N = 6$ robots. The size of aggregate with *Medium Diffusion* and *Fast Diffusion* decayed after a certain time whereas the size of aggregate with *No Diffusion* stayed in a range from 3 to 5 quite stably.*Fast Diffusion* caused a radical decrease in the size of aggregate from $t = 30$ s while *Medium Diffusion* led less steep decrease than *Fast Diffusion*.

Different diffusion rates led to different cohesiveness in the robots. In Figure 3.6(a), cohesiveness with *Fast Diffusion* before $t = 50$ s was considerably higher than with the two other diffusion rates. A similar phenomenon was observed in Figure 3.8(a), where cohesiveness with *Fast Diffusion* was the highest among three different diffusion parameters before $t = 30$ s.

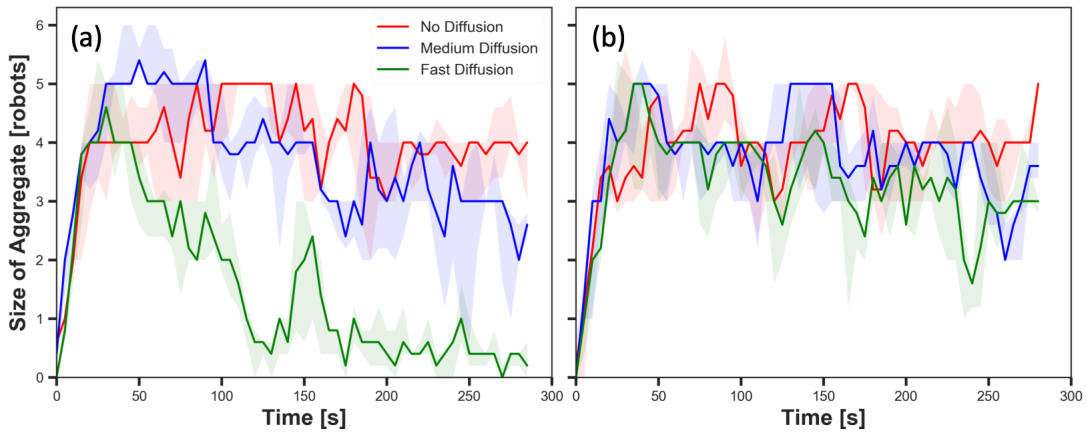

Figure 3.7. The size of aggregate in experiments with static cue configuration, different diffusion rates (*No Diffusion, Medium Diffusion* and *Fast Diffusion*) and (a) without and (b) with pheromone Φ injection in $N = 6$ robots.

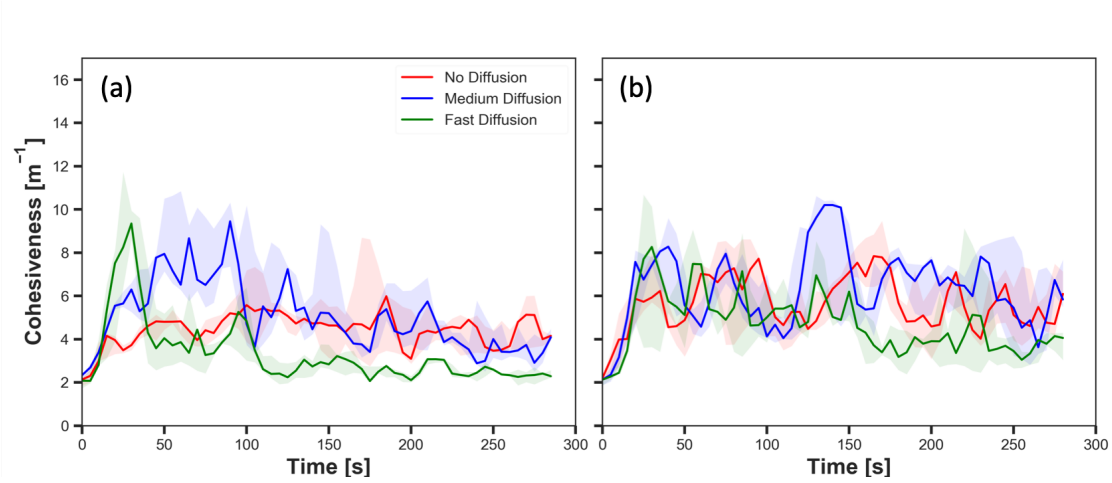

Figure 3.8. The cohesiveness in experiments with static cue configuration, different diffusion rates (*No Diffusion, Medium Diffusion* and *Fast Diffusion*) and (a) without and (b) with pheromone Φ injection in $N = 6$ robots.

As it was expected, diffusion led to a gradual decay in pheromone intensity in the system, therefore, robots decreased their speed when the cue was diffused than when there was no diffusion. The degree of decay in intensity was higher in the outer parts of the cue than in the inner parts. When the robots reached the edge of the diffused cue, they kept moving forward while they stayed on the edge of the cue that was not diffused. As a result, the robots approached closer to the centre of the cue in experiments with diffusion than in experiments without diffusion. Therefore, diffusion led to high cohesiveness of the swarm while the intensity of the pheromone was sufficient for robots to stay on the cue.

Pheromone

Comparing Figures 3.5(a) and 3.5(b), it is shown that robots aggregated in higher probability on the cue throughout the experiments regardless of diffusion rates with Φ than without Φ. Moreover, the decrease in size of aggregate in experiments with $N = 4$ robots in the case of *Fast Diffusion* with Φ was considerably delayed. This difference caused by pheromone injection was also observed in

experiments with $N = 6$ robots. In Figure 3.7, it is shown that the decay in the size of aggregate with *Medium Diffusion* and *Fast Diffusion* was slower with Φ than without Φ. The impact of diffusion that increases the cohesiveness of robots was amplified by pheromone injection. In Figure 3.6, it is observed that the range of cohesiveness was remarkably higher with Φ than without Φ. For the cases without being subjected by the amount of diffusion – *No Diffusion* and *Medium Diffusion*– the cohesiveness was above 6 with Φ while most of the time it ranged from 2 to 4 without Φ. The increase in cohesiveness with Φ in $N = 6$ is also shown in Figure 3.8, although it seems not as influential as in $N = 4$.

The results suggested that the pheromone injection offsets the impact of diffusion on the size of aggregate. Since the intensity of the cue increases when a pheromone is injected, the robots are likely to stay on the cue. Hence, the size of the aggregate is higher with Φ than without Φ. Due to the higher size of the aggregate, the cohesiveness also increases. In experiments with diffusion, robots are more likely to inject pheromone while they are waiting close to the centre, therefore, the cohesiveness is higher.

Statistical Analysis

To statistically analyse the results, a fully nested ANOVA test with factors of population, diffusion and time was carried out to find the significance of the factors. Also, the most effective factor on the size of aggregate and cohesiveness in both configurations: i) with Φ and ii) without Φ was determined. Tables 3.2 and 3.3 show the results of ANOVA tests on the size of aggregate and the cohesiveness with different cue configurations respectively. For the size of aggregate, diffusion and time were the significant factors while the population had no effects on the size of aggregate in both configurations ($p > 0.05$). Diffusion had the most significant influence in both configurations. The significance of time suggests that the size of aggregate was time-variant. In accordance with the results of ANOVA test on the size of aggregate, diffusion and time were the significant factors in the cohesiveness. The most influential factor for the cohesiveness is also diffusion in both configurations (F = 12.66 for with Φ and F = 30.58 for without Φ). Population size did not have a significant impact on the cohesiveness and time had a weaker impact than diffusion as identical to the results in the ANOVA test on the size of aggregate.

3.4.2 Dynamic Cue Configuration

In this section, the aggregation performance of the robots with three different diffusion rates *(No Diffusion, Medium Diffusion* and *Fast Diffusion)* and different cue speeds ($u \in \{0.1, 0.2\}$ cm/s) is presented in the dynamic cue configuration. The dynamic cue configuration with two different speeds: i) *Medium Cue Speed* and ii) *Fast Cue Speed*. Figures 3.9, 3.10, 3.11 and 3.12 show the size of aggregate and cohesiveness with $N \in \{4, 6\}$ robots with *Medium Cue Speed* respectively. The subsequent four figures, Figures 3.13, 3.14, 3.15 and 3.16 show the size of aggregate and cohesiveness with $N \in \{4, 6\}$ robots with *Fast Cue Speed* respectively.

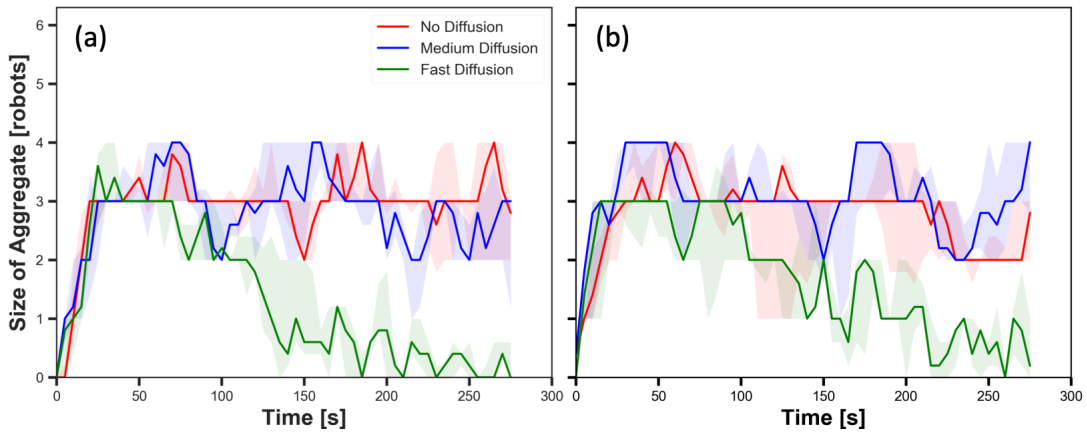

Figure 3.9. The size of aggregate in experiments with *Medium Cue Speed*, different diffusion rates (*No Diffusion, Medium Diffusion* and *Fast Diffusion*) and (a) without and (b) with pheromone Φ injection in $N = 4$ robots.

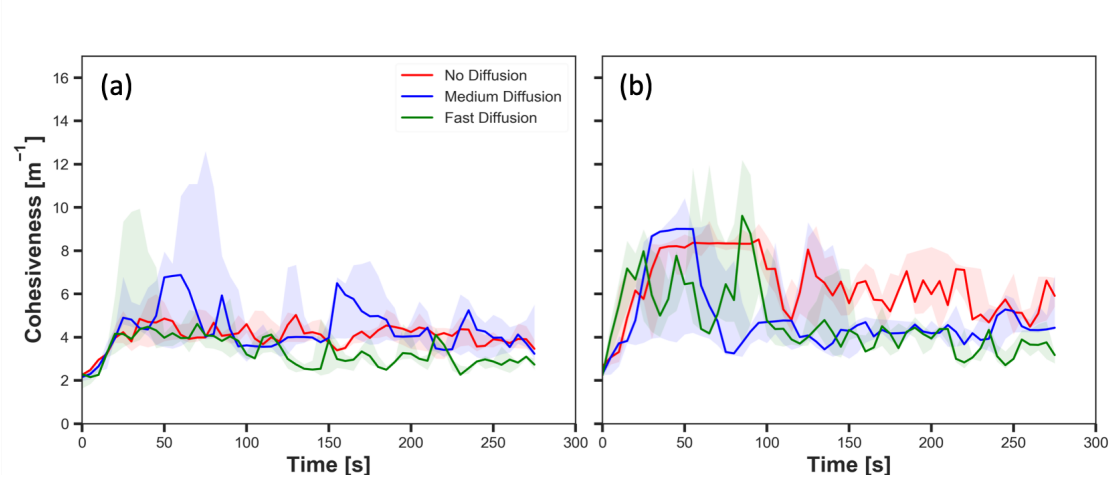

Figure 3.10. The cohesiveness in experiments with *Medium Cue Speed*, different diffusion rates (*No Diffusion, Medium Diffusion* and *Fast Diffusion*) and (a) without and (b) with pheromone Φ injection in $N = 4$ robots.

Diffusion:

The results shown in Figures 3.9(a), 3.11(a), 3.13(a) and 3.15(a) display the impact of diffusion on the size of aggregate in dynamic cue configuration with two population sizes ($N \in \{4, 6\}$ robot) and two cue speeds (Medium cue speed and *Fast Cue Speed*). The decrease in the size of aggregate appeared in *Fast Diffusion* in all the four figures regardless of the cue speed and population. Moreover, this effect only arose in *Fast Diffusion*. The size of aggregate with *Medium Diffusion* with different population size and cue speed which is displayed in Figures 3.9(a), 3.11(a), 3.13(a) and 3.15(a), did not have notable different trends compared to the size of aggregate with *No Diffusion* in both cue speed.

The results shown in Figures 3.10(a), 3.12(a), 3.14(a) and 3.16(a) demonstrated the impact of diffusion on the cohesiveness in dynamic cue configuration with two population ($N \in \{4, 6\}$ robots) and two cue speeds, Medium cue speed and *Fast Cue Speed*. In Figure 3.10(a), it is shown that with *Medium Cue Speed*, the cohesiveness with diffusion was higher than with *No Diffusion*. The cohesiveness with *Medium Diffusion* was higher than with *No Diffusion* throughout the experiment.

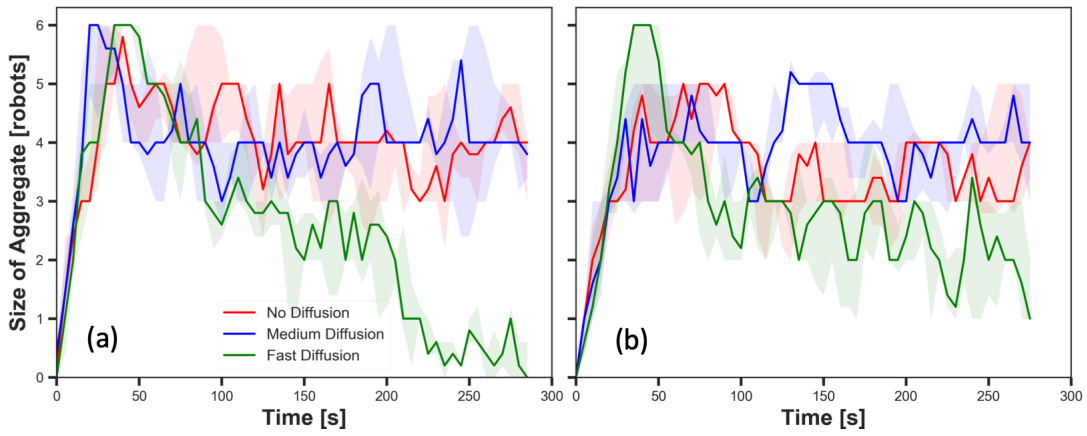

Figure 3.11. The size of aggregate in experiments with *Medium Cue Speed*, different diffusion rates (*No Diffusion, Medium Diffusion* and *Fast Diffusion*) and (a) without and (b) with pheromone Φ injection in $N = 6$ robots.

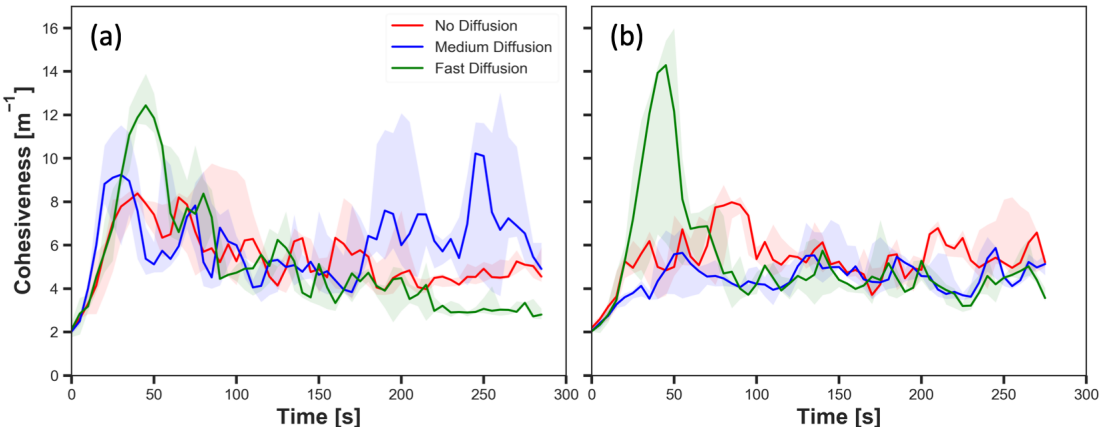

Figure 3.12. The cohesiveness in experiments with *Medium Cue Speed*, different diffusion rates (*No Diffusion, Medium Diffusion* and *Fast Diffusion*) and (a) without and (b) with pheromone Φ injection in $N = 6$ robots.

Although the median of the cohesiveness with *Fast Diffusion* was lower than with *No Diffusion* after $t = 80$ s, the inter-quartile range suggests that the cohesiveness with]*Fast Diffusion* was higher than with *No Diffusion* in two sets of experiments. Especially, it was high with *Fast Diffusion* when the cue still remained. Likewise, with population $N = 6$ robots, the cohesiveness was higher with diffusion than *No Diffusion* (see Figure 3.12(a)). When the cue was still not diffused over a certain amount, the cohesiveness with *Fast Diffusion* surpassed the other two diffusion rates. With *Medium Diffusion*, the cohesiveness was higher than the other two diffusion rates at the end of the experiments. The increase in the cohesiveness resulting from increasing diffusion rate does not clearly appear in experiments with *Fast Cue Speed*. In Figure 3.14, the difference in the cohesiveness between experiments with *No Diffusion* and *Medium Diffusion* and *Fast Diffusion* is not as clear as seen in medium-speed cases.

The results showed that the effect of diffusion led to decay in the size of aggregate and an increase in the cohesiveness as seen in static cue configuration, however, the degree of the effect was smaller in dynamic cue configuration than in static cue configuration. As the cue speed increased, the diffusion effect became less influential. It is because the moving cue dragged the robots in the same direction

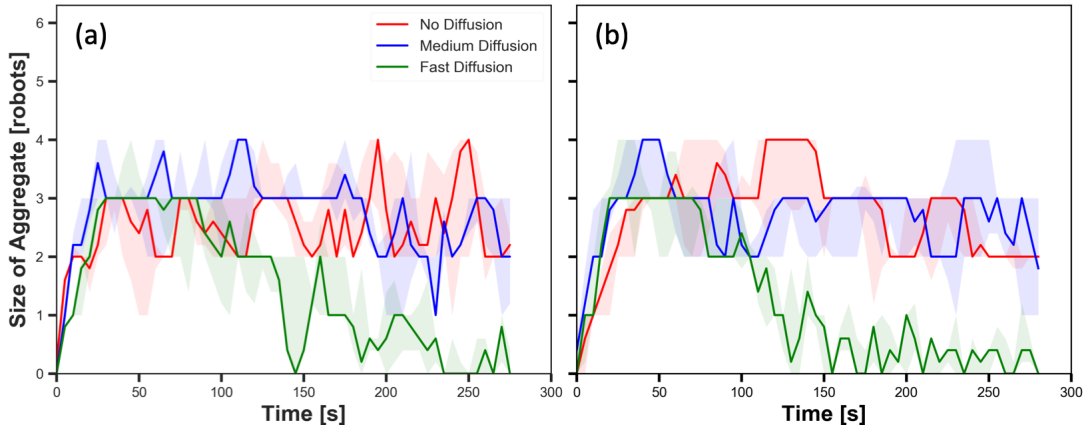

Figure 3.13. The size of aggregate in experiments with *Fast Cue Speed*, different diffusion rates (*No Diffusion, Medium Diffusion* and *Fast Diffusion*) and (a) without and (b) with pheromone Φ injection in $N = 4$ robots.

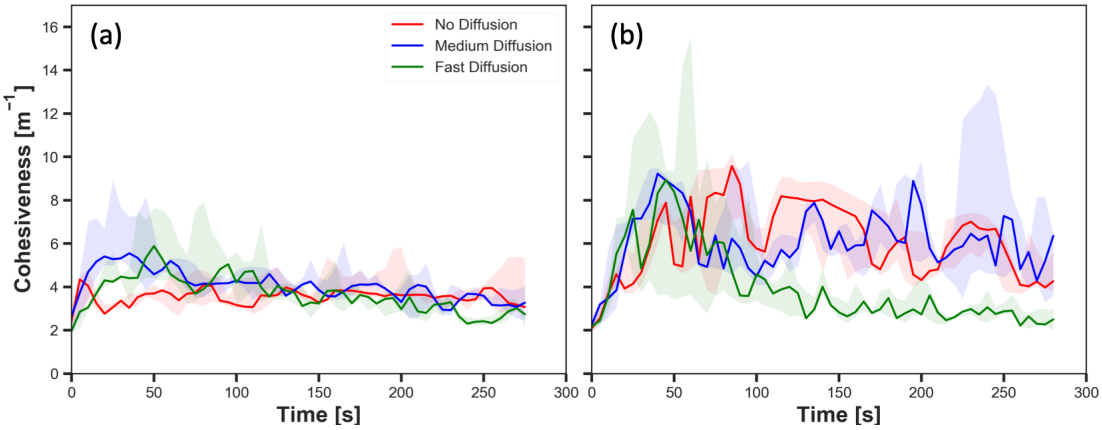

Figure 3.14. The cohesiveness in experiments with *Fast Cue Speed*, different diffusion rates (*No Diffusion, Medium Diffusion* and]*Fast Diffusion*) and (a) without and (b) with pheromone Φ injection in $N = 4$ robots.

as the cue and it hampered robots staying closer to the centre of the cue.

Pheromone

The impact of pheromone injection on the size of aggregate is displayed in Figures 3.9, 3.11, 3.13 and 3.15. In the three Figures 3.9, 3.11 and 3.15, the size of aggregate in experiments with *Fast Diffusion* slowly decreased later with Φ than without Φ. This effect of pheromone injection seemed greater in experiments with $N = 6$ robots than $N = 4$ robots. Figures 3.10, 3.12, 3.14 and 3.16 depict the difference in the cohesiveness in experiments with three diffusion rates *(No Diffusion, Medium Diffusion* and *Fast Diffusion)* between with Φ and without Φ. Pheromone injection had a greater impact on the cohesiveness with $N = 4$ than $N = 6$ regardless of cue speed. The impact of pheromone injection seemed stronger in experiments with $N = 4$ robots than $N = 6$ robots with smaller differences in different cue speeds.

The reason that the effect of pheromone injection that cancelled the effect of diffusion is greater in experiments with $N = 6$ robots than $N = 4$ robots is that the probability of collision rose as the

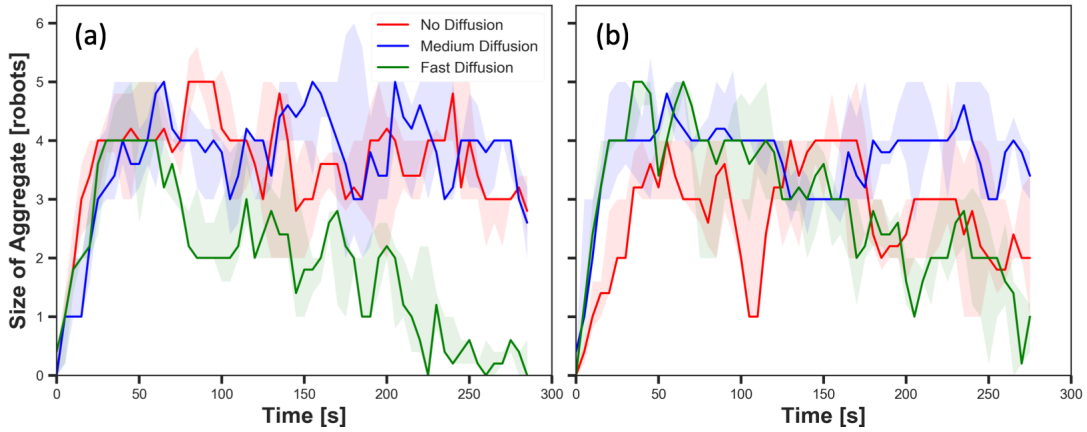

Figure 3.15. The size of aggregate in experiments with *Fast Cue Speed*, different diffusion rates (*No Diffusion, Medium Diffusion* and *Fast Diffusion*) and (a) without and (b) with pheromone Φ injection in $N = 6$ robots.

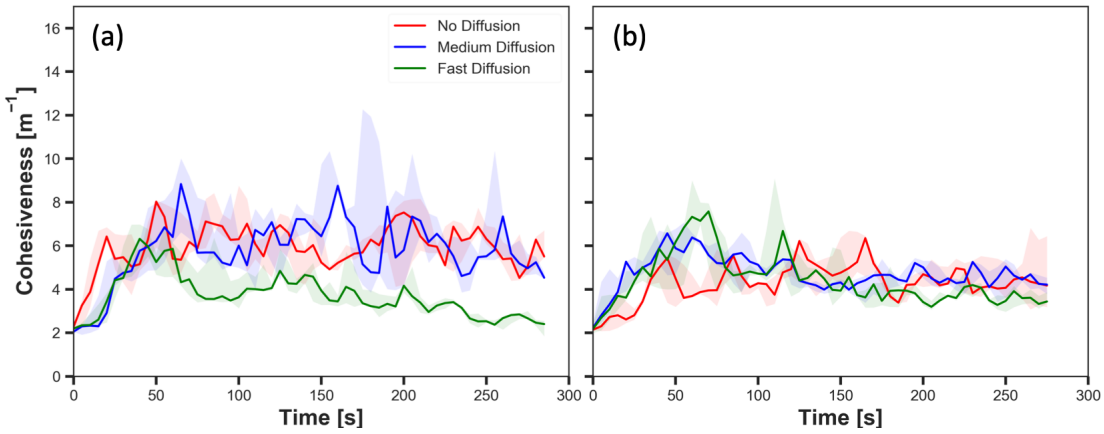

Figure 3.16. The cohesiveness in experiments with *Fast Cue Speed*, different diffusion rates (*No Diffusion, Medium Diffusion* and *Fast Diffusion*) and (a) without and (b) with pheromone Φ injection in $N = 6$ robots.

population increased. Similarly, the impact of pheromone injection on the cohesiveness in $N = 4$ can be explained that rather than they collided with each other outside of the cue, they collided with higher probability when they were close to the centre of the cue, therefore, the cohesiveness was higher with Φ. However, with $N = 6$ robots, the robots collided with each other more frequently in the outer part of the cue than with $N = 4$ robots, thus, there was no remarkable increase in the cohesiveness.

Cue speed

The impact of cue speed was also investigated by comparing Figures 3.9, 3.10, 3.11, and 3.12 with Figures 3.13, 3.14, 3.15 and 3.16 respectively. The first four figures show the size of aggregate and the cohesiveness in experiments with $N \in \{4, 6\}$ robots with *Medium Cue Speed* and the last four figures show the size of aggregate and the cohesiveness in experiments with $N \in \{4, 6\}$ robots with *Fast Cue Speed*. The diminished impact of pheromone injection on the size of aggregate is observed as caused by the increase in the cue speed. Comparing Figures 3.9(b) and 3.13(b), the decay of the

size of aggregate with *Fast Diffusion* occurred faster with *Fast Cue Speed* than with *Medium Cue Speed*. Similarly, the smaller impact of pheromone injection on the size of aggregate with *Fast Cue Speed* than *Medium Cue Speed* was observed in experiments with $N = 6$ robots comparing Figures 3.11 and 3.15. Whereas the pheromone injection with *Medium Cue Speed* seemed to have an impact only with *Fast Diffusion* (see Figure 3.10 and 3.12, the pheromone injection with *Fast Cue Speed* affected the size of aggregate with *No Diffusion* (see Figure 3.16. It is also seen the increase in the cue speed causes a decrease in the cohesiveness. This observation is not clearly shown in $N = 4$. Comparing Figures 3.10 and 3.14, the cohesiveness rather increased as the cue speed increased with Φ, whereas the cohesiveness slightly decreased without Φ for all diffusion rates. However, the cohesiveness considerably decreased in experiments with $N = 6$ robots (see Figures 3.12 and 3.16). The difference notably featured the cohesiveness with *Fast Diffusion*, the difference was not as remarkable in experiments with *No Diffusion* and *Medium Diffusion*.

The difference observed between the experiments with two different cue speeds was likely to occur because the cue moved away from the robots which were in the waiting phase more quickly with the faster cue speed. The position at which the pheromone was injected was always where the robot was waiting. Therefore, the position of injected pheromone and the cue moved away further from each other in experiments with *Fast Cue Speed* than *Medium Cue Speed*. As a result, the greater number of robots tended to stay on the injected pheromone outside the cue with *Fast Cue Speed*, thereby the decrease in the size of aggregate. The decrease in cohesiveness by increasing the cue speed was also caused by the same reason why the impact of pheromone injection decreased. Since the cue passed the robots too fast while they were waiting after the collision at the centre of the cue, the cohesiveness rapidly decreased especially when it was high. This observation also supports why there was no considerable difference of the cohesiveness between experiments with *Medium Cue Speed* and *Fast Cue Speed*, where in both cases of *No Diffusion* and *Medium Diffusion* was applied.

Statistical Analysis

The results from the experiments in the dynamic cue configuration with two different cue speeds were also analysed using ANOVA test (see Tables 3.2 and 3.3). The statistical analysis revealed that diffusion and time were two significant factors ($p < 0.05$) for both experiments with and without Φ. Diffusion was the most significant factor in both the size of aggregate and cohesiveness, both with and without Φ regardless of the cue speed. It is also shown that this system was time-variant ($p < 0.05$). Although both the size of aggregate and cohesiveness with two different cue speeds were influenced by diffusion the most, the impact of diffusion on cohesiveness with *Medium Cue Speed* is the lowest. F = 13.6 and 13.0 for experiments with and without Φ respectively, while F is greater than 30 in the test of size of aggregate with *Medium Cue Speed* and both the metrics with *Fast Cue Speed*.

Table 3.2. Results of ANOVA test for size of aggregate with different cue speeds, u.

Factor	With Φ		Without Φ	
	F	*p*	*F*	*p*
Static, $u = 0$ cm/s				
Population (N)	0.46	0.54	0.00	0.96
Diffusion (κ)	17.00	0.00	64.81	0.00
Time (t)	3.14	0.00	6.41	0.00
Medium, $u = 0.1$ cm/s				
Population (N)	0.20	0.68	0.01	0.95
Diffusion (κ)	39.98	0.00	43.05	0.00
Time (t)	2.76	0.00	4.64	0.00
Fast, $u = 0.2$ cm/s				
Population (N)	0.06	0.82	0.07	0.81
Diffusion (κ)	32.88	0.00	50.31	0.00
Time (t)	3.55	0.00	3.65	0.00

Table 3.3. Results of ANOVA test for cohesiveness with different cue speeds, u.

Factor	With Φ		Without Φ	
	F	*p*	*F*	*p*
Static, $u = 0$ cm/s				
Population (N)	0.42	0.55	1.37	0.31
Diffusion (κ)	12.66	0.00	30.58	0.00
Time (t)	3.23	0.00	3.28	0.00
Medium, $u = 0.1$ cm/s				
Population (N)	0.30	0.61	4.80	0.10
Diffusion (κ)	13.60	0.00	13.00	0.00
Time (t)	2.84	0.00	3.28	0.00
Fast, $u = 0.2$ cm/s				
Population (N)	0.14	0.73	2.22	0.21
Diffusion (κ)	31.90	0.00	29.56	0.00
Time (t)	3.30	0.00	3.38	0.00

3.5 Discussion

The observation made from the results suggests that positive feedback and environmental effects have an impact on the performance of collective behaviour of robots. The impact of positive feedback and the environmental effects of robot swarm across different population sizes is discussed in more detail.

3.5.1 Static Cue Configuration

In the static cue configuration, the trend of swarm behaviour did not noticeably vary between two population sizes ($N \in \{4, 6\}$ robot). Statistical analysis of the results showed that the population size did not affect the size of aggregate and cohesiveness. It means that the ratio of the size of aggregate given the population size did not differ by the population size. In other words, the aggregation performance increased linearly with the population size. This relationship between the population and aggregation performance efficacy was also reported in other research studies [143], [145]. While the decrease in aggregation performance efficacy using pheromone-based communication

with high-density robot swarms was reported in the studies [146], [147], this work did not showcase the decrease in the performance due to relatively small sizes of robot swarms. This means that more investigation with large sizes of swarms is required to investigate aggregation performance efficacy with realistic conditions.

In the set of experiments without Φ, the effect of diffusion on the swarm aggregation performance is investigated when there is no feedback provided. In the plots, the decrease in the size of the aggregate and increase in the cohesiveness with the increased diffusion rate was observed. It was revealed that diffusion had the most significant impact both on the size of aggregate and cohesiveness. This result is identically showcased to the impact of diffusion on the swarm behaviour of robots reported in the earlier study [143]. Such impact of diffusion indicates that the pheromone-based aggregation behaviour of the swarm robotic system is susceptible to the effect of diffusion, implying its susceptibility to achieve the desired behaviour [148].

In the set of experiments with Φ, the impact of pheromone injection as a positive feedback mechanism on the aggregation performance is reported with different diffusion rates. The results revealed that activation of the feedback mechanism had a substantial impact on the swarm behaviour. The effect of diffusion that reduces the size of aggregate was diminished by the feedback via pheromone injection. The statistical analysis confirmed that pheromone injection was one of the significant factors affecting the swarm aggregation performance (see Table 3.4). This result implies that the positive feedback mechanism is a great way to compensate for the environmental effect. Positive feedback can be used as a mitigating strategy for robotic swarms to overcome environmental effects when they are used in real-world conditions. On the other hand, it also means that the positive feedback adds more complexity to analyse and predict the behaviours of the swarm. A similar impact of the pheromone injection that increases the swarm behaviour was reported in research studies both in biology [141], [149] and robotics [143], [150].

3.5.2 Dynamic Cue Configuration

As identical in the static cue configuration, the population size did not have a significant impact on the swarm performance according to the statistical analysis (see Tables 3.2 and 3.3)

In the set of experiments without Φ, similar results in the experiments with the static cue configuration were observed for both two cue speeds. The increase in the diffusion rate led to a decrease both in the size of aggregate and the cohesiveness. Although the identical impact of diffusion appeared in both static and dynamic cue configurations, the degree of impact was differently observed. In the dynamic cue configuration, it is shown that the time taken for the robots to completely leave the cue was delayed than in the static cue configuration. This delay indicates the moving cue dragged the robots to the cue rather than randomly moving. Despite the mitigating effect of dynamic cues by wind effect, the statistical analysis confirmed that diffusion was the most influential factor of the swarm performance. Nevertheless, the impact of the interplay between dynamic cue and diffusion generates more complex variation in the aggregation behaviour. This counterbalancing effect of

wind and diffusion effects reinforces the unpredictability of aggregation behaviour of pheromone-based swarm robotic systems operating under realistic conditions.

The effect of pheromone injection in the dynamic cue configuration was investigated in the same manner used in the static cue configuration. The results demonstrated that the pheromone injection affected the swarm performance. Although the observed impact was identical to the static cue configuration, the observation in the dynamic cue configuration was not as clear as in the static cue configuration. In Figures 3.9 and 3.13, the trend of the size of aggregate in experiments with Φ was not clearly distinguished from experiments without Φ whereas the impact of pheromone injection weakening the diffusion effect was clearly shown in the other configurations.

Despite this vagueness of the impact of pheromone injection shown in the mentioned figures, the results in corresponding figures displaying cohesiveness still showed that the pheromone injection had an impact. As seen in Figures 3.10 and 3.14, the increased cohesiveness in experiments with Φ compared to without Φ is described. This result posited that pheromone injection allowed the robots to stay close to the cue rather than randomly roaming around. The impact of pheromone injection was significant on the swarm performance according to the statistical analysis. The robustness of the swarm using the feedback via pheromone injection against the environmental factors was similarly reported in the studies both in robotics and biology [150], [151]. Nonetheless, the relative subtlety of the effect of positive feedback on the aggregation behaviour with the dynamic cue configuration hinders achieving the targeted aggregation behaviour supported by positive feedback.

Figure 3.17 shows the impact of cue speed comprehensively. The results depict that the cue speed did not deteriorate the size of aggregate. Interestingly, with the medium cue speed, the size of aggregate reached the highest regardless of diffusion and pheromone injection. This observation suggests that the gradient made by moving pheromone with a moderate speed leads to the highest responsiveness on the pheromone cue for robots. This guidance effect by using the gradient in pheromone trails was reported in a study on bee communication [152]. As well as the observation of the maximised dragging effect with the medium cue speed, it is worth noting that the size of aggregate with the fast cue speed fluctuated shown in Figure 3.17(d). As mentioned in the previous results section, it is shown that the swarm aggregate was formed around the cue. Although the swarm was still forming aggregation under the higher intensity of cue movement, it deteriorates the quality of aggregation, supported by low cohesiveness shown in Figure 3.14 and 3.15. The statistical analysis showed the cue speed was significantly influential to the robot swarm. The intensity of the environmental effect determines the influence on the collective behaviour of swarm robotic systems. As the degree of environmental effect varies unpredictably in real-world conditions, the collective behaviour of a swarm robotic system is likely to be unstable under such conditions when no mitigating strategies exist for relieving environmental effects to obtain the desired behaviours.

Summarising the in-depth analysis of the impact of the variables, The substantial impact of the complex interplay of positive feedback and environmental factors necessitates more adaptive controllers for swarm robotic systems in real-world applications.

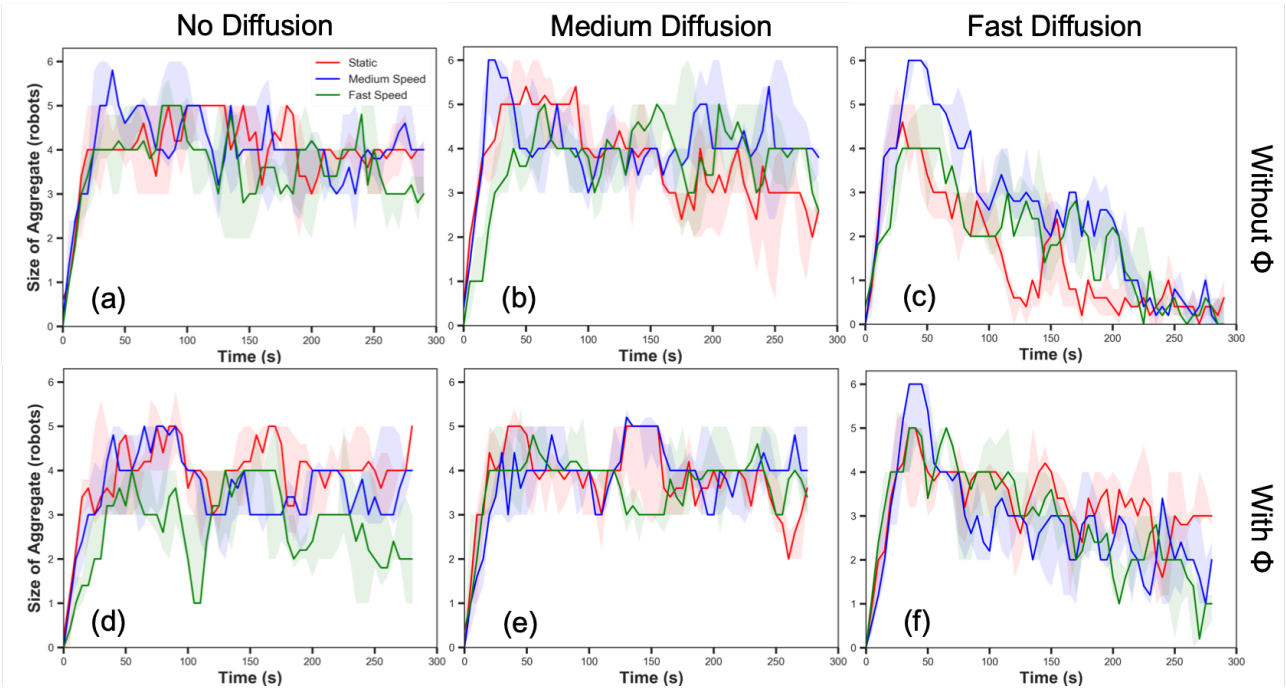

Figure 3.17. The size aggregate in experiments with different cue speeds ($u \in \{0, 0.1, 0.2\}$ cm/s), diffusion rates ($\kappa \in \{0, 50, 75\}\%/T$) and without and with pheromone (Φ) injection in $N = 6$ robots.

Table 3.4. Results of fully-nested ANOVA test.

Factor	Size of Aggregate		Cohesiveness	
	F	p	F	p
Population (N)	0.05	0.83	0.55	0.50
Diffusion (κ)	12.37	0.01	1.17	0.41
Pheromone (Φ)	3.27	0.02	5.48	0.00
Cue Speed (u)	5.55	0.00	13.32	0.00
Time (t)	3.91	0.00	3.22	0.00

3.6 Summary

In this chapter, **Aim 1** and the following **Objective 1** are achieved by investigating the aggregation behaviour of a pheromone-based swarm robotic system with the emulated realistic factors. Specifically, the effects of four independent variables: positive feedback (pheromone injection), environmental effects (wind and diffusion), and population were systematically investigated. The experimental results show that these variables significantly impact aggregation performance, leading to complex effects. The interplay between real-world factors creates a more intricate impact than individual factors alone. For example, medium diffusion and wind effect increase aggregate size, while their sole presence reduces it.

The significant influence of realistic conditions on swarm robotic systems highlights their potential to deviate unpredictably from desired behaviour in real-world scenarios. To address this, sophisticated controllers for individual robots must be designed to overcome and adapt to adverse effects while achieving desired behaviours. This chapter's findings spur the research in the next chapter, where automatic controller design is adopted to enhance adaptivity for pheromone-based swarm

robotic systems in diverse realistic conditions. Compared to the manual controller design, which is used to design aggregation behaviour in this chapter, automatic controller design is expected to be more adaptive and robust in a complex environment. In conclusion, this chapter successfully identifies the effect of realistic conditions, confirming the need for automatic controller design. Therefore, this chapter supports the next chapter, whose aim is to propose a DRL-driven automatic controller for swarm robotic scenarios.

Chapter 4

Deep Reinforcement Learning-Driven Automatic Controller Design

In Chapter *3*, the impact of realistic conditions on the collective behaviour of a pheromone-based swarm robotic system was examined. The findings revealed that these conditions significantly influence collective behaviour, making the system more susceptible to real-world operating environments. Consequently, the results underscore the need for a more sophisticated, adaptive controller that can respond to the complexities of actual conditions.

In this chapter, the Deep Reinforcement Learning (DRL)-driven automatic controller design is proposed as a promising type of adaptive controller. About the scenario, the focus shifts from the simple aggregation scenario in the previous chapter to the application of an artificial pheromone system for a realistic scenario involving swarm robotic systems, envisioned as a future transportation model. In this context, a multitude of autonomous vehicles on the road is treated as a swarm system. Pheromone-based communication is employed to manage this system, providing scalability and flexibility in alignment with the inherent characteristics of pheromone-based interactions. This approach facilitates essential tasks such as collision avoidance for individual autonomous vehicles.

Building on this scenario, this chapter introduces a research study that proposes a DRL-driven automatic controller design to enhance performance efficacy. The investigation delves into the effectiveness of the proposed model as complexity increases, aligning with the preceding chapter's emphasis on the importance of adaptive controller design in complex real-world conditions. Finally, this chapter validates the potential of DRL for automatic controller design in swarm robotic systems under realistic constraints. It concludes by outlining a new direction for mitigating the issue of centralisation, a further real-world constraint in promising operating environments.

4.1 Background

Although pheromone-based swarm robotic systems are in a nascent stage, many potential application domains exist in the real world. Traffic management for autonomous vehicles is one of the domains where pheromone-based swarm robotic systems can benefit.

With the emergence of machine learning and other new technologies, the development of autonomous vehicles has been drawing greater attention as a possible method of future transportation [153]. The wide deployment of autonomous vehicles is expected to bring invaluable benefits to society across diverse aspects including reduced traffic congestion, savings in fuel/battery usage and most importantly, increased safety. However, if autonomous vehicles are to receive widespread acceptance for use on roads then robust collision avoidance mechanisms in general operating environments are essential [154], [155].

Large numbers of autonomous vehicles operating together on a network of roads can be regarded as a swarm, where a swarm is a system that consists of a large number of agents, controlled in a decentralised manner using local communication between individual agents or agents and the environment, that work towards a common goal [156]. Swarms are frequently observed in nature where they perform complex tasks using large numbers of agents, all with limited capabilities, such as the colonies of ants, bees and termites. Since autonomous vehicles can be controlled independently and use local communication to achieve a common goal, e.g. minimising collisions, whilst optimising other driving factors, such as congestion and travel times, they can be regarded as a swarm system.

Like a wide range of practical domains including optimisation [157], vehicle routing [158] and robotics [45], [159]–[161], which adopted pheromone-based communication systems to address particular challenges, traffic management for autonomous vehicles as a swarm system, pheromone-based communication can thrive thanks to its decentralised, scalable and optimising features. Despite the potential of pheromone-based communication facilitating traffic management for autonomous vehicles, the issue of intricacy for controller design arises. Designing an individual controller for a fleet of autonomous vehicles is not a trivial task especially when indirect pheromone-based communication is widely used for coordination. As demonstrated in Chapter 3, pheromone-based swarm systems are particularly vulnerable in real-world environments, which makes designing controllers with manual controller design an almost insurmountable challenge.

The two widely recognised basic low-level behaviours of autonomous vehicles and similar fields are collision avoidance and waypoint navigation. To design effective collision avoidance including navigation, a wide range of techniques including control theory [162], potential fields [163], trajectory planning [164], [165], adaptive control [166], [167], velocity-based approaches [168]–[170] and biologically-inspired method [171] have been proposed.

More recently, DRL-based approaches have been proposed that utilise large amounts of data to derive optimal controllers particularly effective on complex and dynamic environments [90], [91], [172], [173]. The recent success of DRL-based approaches in diverse multi-agent domains

incentivises the use of DRL in automatic controller design for swarm systems. From a technical perspective, the use of DRL-based controller design can be regarded as effective for the traffic management scenario of autonomous vehicles with added complexity from pheromone-based communication. Also from the perspective of automatic controller design, the application of DRL for this swarm system gives an opportunity to investigate DRL-driven automatic controller design as a viable option for real-world scenario. In this research study, DRL-driven automatic controller design is proposed for traffic management scenarios for autonomous vehicles and investigated via comparative analysis with two conventional approaches in a classic multi-agent and swarm system perspective respectively.

This research study contains three major contributions as follows:

- A bio-inspired PhERS (Pheromone for Every Robot Swarm) framework is proposed for a collision avoidance scheme in swarm systems. The framework is validated using three traffic scenarios with increasing complexity in a realistic simulated environment. Through experiments, the controllers using this framework showed greater performance and flexibility than the traditional centralised control method.

- A DRL-driven automatic controller design is proposed for pheromone-based swarm systems. The DRL-driven controller is designed to provide collision avoidance and navigation in pheromone-based swarm systems. Comparisons show that the proposed controller performs better than the traditional centralised controller, NH-ORCA [169], and the manually-tuned controller.

- A novel bio-inspired sampling strategy for experience replay buffer, Highlight Experience Replay (HLER), is proposed and integrated with the Deep Deterministic Policy Gradient algorithm (DDPG). The obtained results from this work showed the proposed HLER sampling strategy outperformed the Prioritized Experience Replay (PER) [120] sampling strategy in terms of training speed reducing to an average 27% of training time with PER in three experimental stages. The training speed of HLER is increased by integrating the Gaussian noise on parameters of a neural network (NN) to incentivise exploration for finding a better policy rapidly [174].

4.2 Artificial Pheromone Framework

To implement pheromone-based swarm system in the simulated environments, PhERS (Pheromone for Every Robot Swarm) framework is proposed, in an expandable way for the future use in diverse environments and applications.

Figure 4.1 illustrates the general overview of the proposed framework. Within the framework, virtual pheromones are managed inside each of the virtual grids, which act as fields of pheromones having different characteristics. The framework is able to incorporate a varying number of virtual grids and hence a variety of virtual pheromones can be defined, which helps to simulate complex interplay of pheromones and induced behaviours. For example, if repellent and attractive

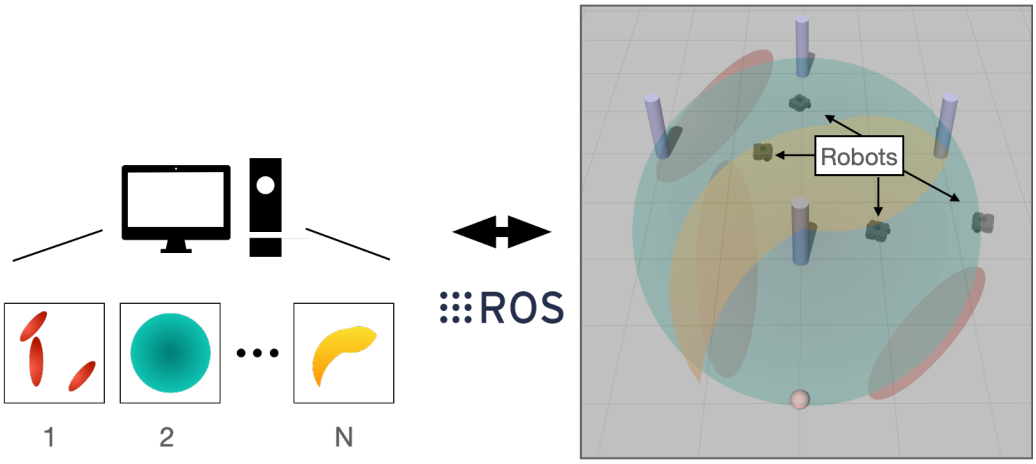

Figure 4.1. An overview of PhERS framework. Multiple pheromone grids (1,2,...,N) are computed in the main PhERS controller and integrated with the environment, which retrieves and updates pheromone intensity values requested as a ROS message.

pheromones are utilised simultaneously, then the complex trail traffic management systems of ants can be replicated.

Figure 4.2 illustrates the proposed architecture of the PhERS framework. It consists of three parts: i) main PhERS controller, ii) data storage and iii) communication network. The main PhERS controller orchestrates all the pheromone grids (Phero-Grid) in the system. When the main PhERS controller is initialised, it creates the specified number of Phero-Grids, containing the individual characteristics requested. After initialisation, the controller updates the Phero-Grids at each time-step, based on the predefined spatio-temporal development model of each pheromone. The data storage is used to store the data contained within the Phero-Grids, which represents the intensity of each of the pheromones, in the main PhERS controller at the specific time-step. The stored Phero-Grid data can be retrieved when requested. The communication network is used to manage communication between the main PhERS controller and the agents. When the agents read pheromone values, the positional information of the agents is sent to the main PhERS controller via the communication network using Robotic Operating System (ROS) messages, which is a developed for robotic application orchestrates communication between multiple robotic devices effectively. The main PhERS controller then retrieves the requested pheromone data from the agent through the communication network. Likewise, the communication network sends the received pheromone intensity at a given position from the robot to the main PhERS controller so that the released pheromone data is applied to the Phero-Grids.

This framework is implemented to create virtual pheromones on a robotic simulated environment (Gazebo). With the current technological limitations, it is impossible for real road infrastructure, not having required physical devices and technologies and social understandings. For future deployments on autonomous vehicles, cloud computing can be a promising method to implement PhERS framework. The example of remote computing server implementing pheromone system is shown in several works using virtualisation to implement pheromone for swarm robotic systems [43], [45]. The virtual pheromone grids that are applied to the environment could be generated and

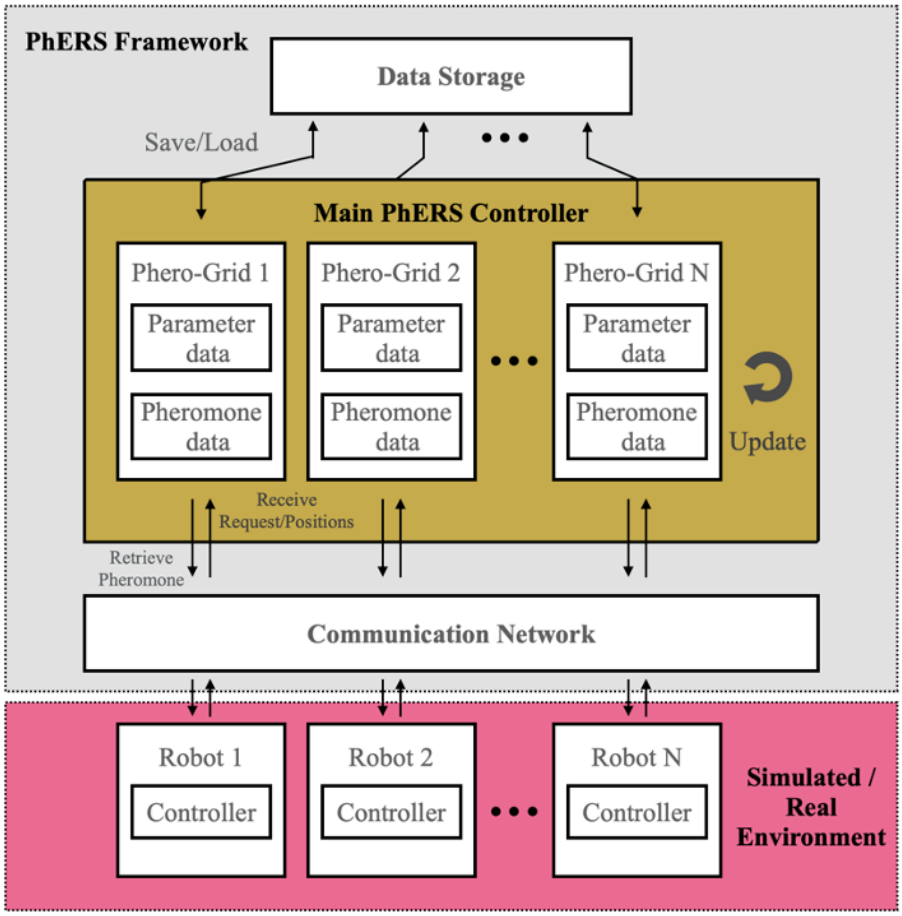

Figure 4.2. Architecture of PhERS framework. The framework consists of (i) data storage, (ii) main controller and (iii) communication network. Phero-Grid represents an entity that stores parameter settings and pheromone data.

updated in the remote cloud system using real-time communication with the vehicles. Using future generation communications, such as 5G, and high-performance cloud servers, the application of virtual pheromones to a large number of fast moving autonomous vehicles, with minimal delays is feasible [175], [176]. Moreover, the virtual pheromones can be generated and managed by the traffic infrastructure. The development of sensing technology and vehicle-to-infrastructure communication systems enabled to propose smart roads [177]. With the smart road technologies, the virtual pheromones can be implemented and managed by local smart traffic infrastructures in a decentralised manner rather than using a centralised server such as a cloud computing system.

There are several advantages for using PhERS for collision avoidance compared with other more traditional collision avoidance methods. For the traditional collision avoidance using centralised control, our pheromone-based communication method is more flexible to any increase in environmental complexity. The PhERS framework provides decentralised control and so can manage more effectively when the number of agents is large and/or varying, i.e. high scalability. Moreover, for the traditional collision avoidance using decentralised control, and using different sensory devices, e.g. LiDAR and camera, the PhERS-based collision avoidance system has the benefit that it can represent future collision hazards that cannot be detected using the sensory devices used when applying the traditional methods. For example, by marking the trail of a vehicle using a slow volatile pheromone, the vehicle can ensure there is a safe distance from other following vehicles.

Although the PhERS framework is able to replace the traditional collision avoidance methods using individual sensory devices such as cameras and LiDARs, it is more beneficial to be used as a higher level system while the traditional collision avoidance is used for lower level. The higher level system can be defined as traffic level system while the traditional individual level system can be classified as vehicle level system. When they are used together, but as different level of layers of collision avoidance, vehicle level can cover more immediate collision avoidance behaviours that require minimal delay at a vehicle level and the PhERS-based system can be used to cover more complex situations at a traffic level. By synthesising the two levels, the vehicles can prevent collisions when the traditional sensory devices are damaged or malfunctioning.

The virtual pheromones in the PhERS framework use a predefined spatio-temporal update model inherited from the original research study [46]. Identical to the artificial pheromone system in Chapter 3, the PhERS framework utilises the same spatio-temporal update model.

4.3 Deep Reinforcement Learning Algorithm

The DRL-driven automatic controller design capitalises DDPG as a backbone algorithm. Standing on DDPG, additional proposed techniques improved the performance efficacy for the use as a method for automatic controller design. Technical detail of DDPG algorithm is described in Section 2.4.1. The proposed strategies to improve the basic DDPG algorithm are described below.

The DDPG algorithm employs an experience replay buffer to store and retrieve transition samples, making the design of the buffer critical to sampling efficiency. This efficiency is a fundamental consideration in algorithm design. In the context of this research study, a novel sampling strategy, referred to as HighLight Experience Replay (HLER), is introduced to optimise sampling efficiency from the experience buffer.

HLER prioritises transition samples with a higher absolute value of rewards, considering them the ”highlights” of an episode. This strategy draws inspiration from the human brain’s ability to more effectively recall highlight events than neutral ones [178]. For instance, studies have demonstrated that images evoking negative emotions are retrieved more effectively than those evoking neutral emotions [179], and that the retrieval of positive contextual information leads to increased brain activity in comparison to neutral information [180].

The anticipated benefit of utilising HLER is an enhanced training of the neural network on samples associated with high absolute rewards. While Prioritized Experience Replay (PER) emphasises the selection of samples with high Temporal Difference error (TD-error) for rapid neural network optimisation, HLER centers on samples with high absolute rewards to expedite learning of significant events. In the specific scenario of this study, concerning collision avoidance for autonomous vehicles, the hypothesis is that HLER may prove more efficient than the PER strategy. The rationale behind this preference is that a collision, being a pivotal event, can be regarded as a highlight, thus aligning more closely with the principles underpinning HLER.

The proposed HLER strategy assigns a priority to each transition sample in a replay buffer, collected from K actors. The priority of the i^{th} sample, p_i is assigned as follows:

$$p_i = \begin{cases} k_{HL} \cdot p_{base}, & |r_t| \geq r_{HL} \\ p_{base}, & \text{otherwise} \end{cases}, \tag{4.1}$$

where p_{base} is the default priority, k_{HL} is a coefficient that is applied to samples that are considered highlights, and r_{HL} is a reward value to determine whether the sample is considered a highlight or not.

When p_i is specified to be equal to $k_{HL} \cdot p_{base}$, the priority of the previous l number of samples, $p_{i-l+1:i}$, are also assigned values of $k_{HL} \cdot p_{base}$. This defines the previous l samples as the moments contributing to the highlight sample. The sampling probability of the i^{th} sample in HLER sampling strategy follows (2.1) as PER.

Algorithm 2: DDPG algorithm with HLER sampling strategy and noisy network.

Randomly initialise critic neural network with normal linear layers, $Q(s, a|\theta^Q)$ and actor neural network with noisy linear layers $\pi(s|\theta^\pi)$ with parameters θ^Q and θ^π
Initialise target network Q' and π' with parameters $\theta^{Q'} \leftarrow \theta^Q$, $\theta^{\pi'} \leftarrow \theta^\pi$
Initialise replay buffer R
for *episode = 1, ..., M* **do**
    Initialise the states $s_t = s_1$
    for *t = 1, ..., T* **do**
        Run K actors and collect transition samples $D_t = (s_t, a_t, r_t, s_{t+1})$ into a replay buffer
        if $|r_t| \geq r_{HL}$ **then**
            Assign priority $k_{HL} \cdot p_{base}$ for $D_{t-l:t}$
        else
            Assign priority p_{base} for D_t
        end
        if *t = 0 mod* t_{train} **then**
            Sample N transitions (s_i, a_i, r_i, s_{i+1}) from a replay buffer with the sampling probability $P(i)$
            Set $y_i = r_i + \gamma Q'(s_i, \pi'(s_{i+1}|\theta^{\pi'})|\theta^{Q'})$
            Set weighted updates for networks: $\omega_i = (\frac{1}{B} \cdot \frac{1}{P(i)})^\beta$
            Update critic network by minimising the loss: $L_Q = \frac{1}{N}\omega \sum_i (y_i - Q(s_i, a_i|\theta^Q))^2$
            Update actor network using the sampled policy gradient: $\nabla_{\theta^\pi} J \approx \frac{1}{N}\omega \sum_i \nabla_a Q(s, a|\theta^Q)|_{s=s_i, a=\pi(s_i)}$
                $\cdot \nabla_{\theta^\pi} \pi(s_i|\theta^\pi)|_{s=s_i}$
        end
        if *t = 0 mod* t_{target} **then**
            Update the target networks: $\theta^{Q'} \leftarrow \theta^Q$,$\theta^{\pi'} \leftarrow \theta^\pi$
        end
    end
end

In contrast to PER, the HLER sampling strategy is posited to be more adept at selecting vital transition data encapsulating behaviors the neural network must assimilate, such as collision avoidance. In scenarios where critical and neutral behaviors are distinctly separable, HLER has the potential to hasten training and result in an elevated cumulative reward throughout an episode, signifying an efficient progression from an initial to a terminal state. Due to its underlying difference of HLER to PER, it is expected to learn a specific behaviour required to equip faster despite the possibility of sub-optimal performance in non-highlighted behaviours. Since HLER sampling strategy is

heuristically designed inspired by biological phenomena and the performance can vary depending on the target behaviour selection, parameter and reward shaping, it is difficult to formally analyse its performance, but requires empirical comparative examination.

Alongside the imperative of high sampling efficiency, a strategy to enhance robustness against unpredictability is equally crucial. The DDPG algorithm, producing deterministic output, may be vulnerable to external perturbations and the non-stationary nature of tasks and environments unless countermeasures are in place. To address this susceptibility and augment exploration capability, the "noisy network" approach was adopted, wherein noise is added to the neural network parameters [174]. The essence of noisy networks lies in the integration of stochastic features by introducing Gaussian noise into the weight and bias values of each linear layer within the neural network. Thus, a linear layer in a neural network with p inputs and q outputs can be formulated as follows:

$$y = wx + b, \tag{4.2}$$

where $x \in \mathbb{R}^p$ is the layer output, $w \in \mathbb{R}^{p \times q}$ represents the weight matrix and the bias is denoted by $b \in \mathbb{R}^q$. However, the linear layer of a noisy neural network can be characterised as follows:

$$y := (\mu^w + \sigma^w \odot \epsilon^w)x + \mu^b + \sigma^b \odot \epsilon^b, \tag{4.3}$$

where $\mu^w \in \mathbb{R}^{q \times p}, \mu^b \in \mathbb{R}^q, \sigma^w \in \mathbb{R}^{q \times p}, \sigma^b \in \mathbb{R}^q$ are the learnable parameters, ϵ^w and ϵ^b are the adaptive Gaussian noise values that are added to the weight and bias of the layer and $\odot$ is an element-wise multiplication operator. A noisy linear layer was applied to the actor network, to help improve the exploration ability of the agent. Unlike ϵ-greedy [87], another popular exploration method, the noisy network approach generates state-dependent noise, which helps reduce undesirable exploration in the states that do not require exploration (e.g. states where the next state causes a large penalty). Moreover, the noisy network approach can be used for different DRL problems as the parameter noise are adapted over training, while the fixed Gaussian parameter space noise approach [181] requires manual fine-tuning of the noise parameters. For these reasons, noisy network approach was chosen to improve robustness against noises and further provide exploration ability when collecting transition samples.

The DDPG algorithm with the HLER sampling strategy technique and noisy network is represented in Algorithm 2. When the algorithm is executed, it initialises the actor and critic neural networks, the target network and the replay buffer. The algorithm iterates for M training episodes and in each episode, a total of T steps of transition samples are collected from the K number of actors. Sample priorities are assigned using the HLER sampling strategy. In every t_{train} time step, the N number of transition samples are selected based on P_i and TD-target and y_i is assigned. In line 16, ω determines the degree of the parameters of the neural network are updated, where β is a constant value and B is the total batch size. In every t_{target} in every episode, the target actor and critic networks are updated.

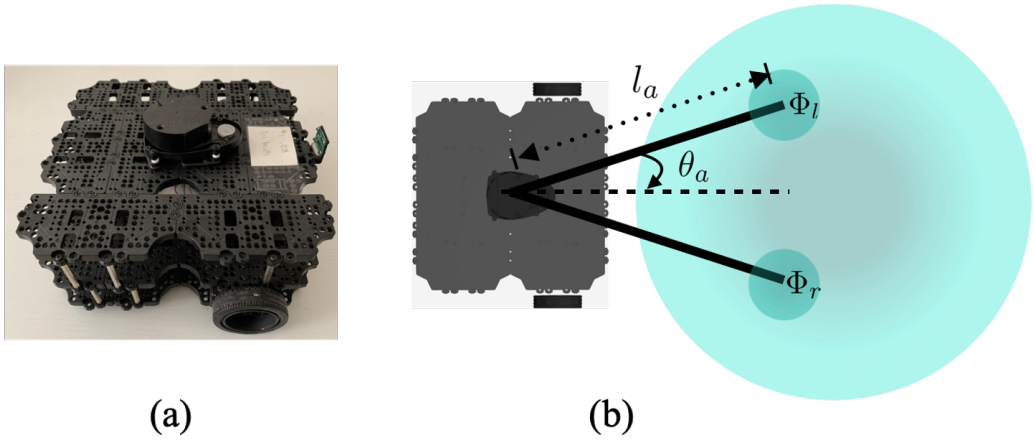

Figure 4.3. The photo of Turtlebot3 Waffle Pi mobile robot, (a), and an illustration of the two virtual antennae are attached on Turtlebot3 Waffle Pi in the Gazebo simulated environment, (b). l_a is the length of antennae, α is the tilt angle, and Φ_l, Φ_r are the pheromone intensities measured by the left and right antennae respectively.

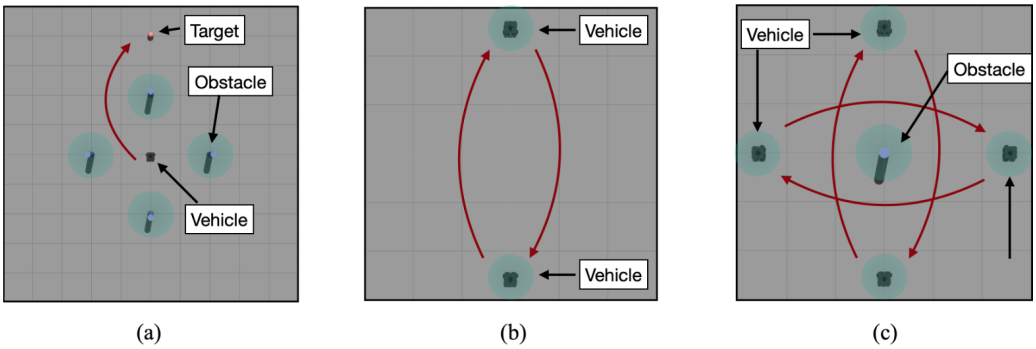

Figure 4.4. Three experimental stages. (a), (b) and (c) illustrate Stage 1, 2 and 3 respectively. The cyan coloured shaded circles around the robots and obstacles illustrate the area where pheromones are injected and the red arrows show the arbitrary trajectory of the robots on a mission.

4.4 Experiments

The experiments conducted in this research study aim to accomplish the **Objective 2**. A comparative analysis is carried out to examine the performance efficacy of the proposed DRL-driven automatic controller design. The following sections describe i) the specific experimental scenario, ii) the traditional multi-agent collision avoidance method, iii) the conventional manual controller design, iv) the specification of the DRL-driven automatic controller design, and v) the metrics used to evaluate the performance of the controllers. Through the experiments, the suitability of the PhERS framework is also validated.

4.4.1 Experimental Scenario

Three stages of experimental scenarios were designed to emulate real-world navigation and collision avoidance situations for autonomous vehicles, with each stage increasing in complexity. These experiments were conducted within the Gazebo simulation environment, leveraging its realistic physics engine. The simulated differential-driven mobile robots, referred to as *Turtlebot3*, were deployed to model the autonomous vehicles. The *Turtlebot3* is a general-purpose wheeled robot, widely utilised

for educational, research, and product prototyping purposes, as depicted in Fig. 4.3 (a). Fig. 4.4 illustrates the experimental stages and each stage is explained as follows.

Stage 1– Single Robot in a Static Scenario

In the first experimental scenario, a situation involving one robot and four static cylindrical obstacles was simulated, reflecting a real-world condition where an autonomous vehicle must navigate around stationary obstructions in traffic. During initialisation, the robot was positioned at the origin (0,0) in the 2D Cartesian coordinate system of the arena, with obstacles placed at the coordinates $[(2,0),(-2,0),(0,2),(0,-2)]$ meters, respectively. The navigation system's goal was to guide the robot to a random position 4 meters away from the origin. This stage is illustrated in Fig. 4.4 (a).

Stage 2– Multi-robot in a Dynamic Scenario

In the second scenario, two robots were deployed. Tasked with navigating to each other's initial positions, they had to avoid colliding with one another. This stage simulates a situation where two autonomous vehicles are driving towards each other, necessitating collision avoidance. During the initialisation phase, the robots were placed 5 meters apart. Fig. 4.4 (b) depicts this stage.

Stage 3– Multi-robot in a Complex Dynamic Scenario

In the third scenario, four robots and one static obstacle were deployed. This stage simulates a situation where four autonomous vehicles approach a roundabout without fixed traffic rules. Similar to Stage 2, the robots' objective is to navigate to the positions of the robot they initially face, avoiding collisions with both other robots and the static obstacle. During the initialisation phase, the robots were positioned at coordinates $[(2.5,0),(-2.5,0),(0,2.5),(0,-2.5)]$ meters, respectively, in the 2D Cartesian arena, with the static obstacle centered at (0,0). Fig. 4.4 (c) illustrates this stage.

For collision avoidance, two distinct types of artificial pheromones were developed: i) a non-volatile pheromone, utilised for static obstacle avoidance, and ii) a highly volatile pheromone, designed for dynamic obstacle avoidance. This dual-pheromone concept draws inspiration from natural alarm pheromones that elicit aggressive or avoidant responses in organisms, such as ants [182], [183].

In the experimental setup, non-volatile pheromones surround static obstacles during the initialisation phase. Unlike the short-lived volatile pheromones employed for dynamic obstacles, non-volatile pheromones are presumed to communicate with static obstacles, and a longer-lasting variant is employed. This non-volatile type is released in a circular shape with a radius of 0.5 m, exhibiting no evaporation or diffusion (i.e., $e_\Phi = 0$ s and $\kappa = 0$). Conversely, the volatile pheromone is dispersed with a 0.3 m radius, accompanied by a rapid evaporation half-life of $e_\Phi = 0.5$ s.

The selection of the radius and evaporation rate was empirically determined to ensure the effective utilisation of each pheromone type in collision avoidance. Particularly, the choice of e_Φ for the

volatile pheromone was inspired by previous work where highly volatile repellent pheromones were employed to avert wall collisions during navigation [46].

Each experimental stage was conducted 100 times across three distinct control design setups: 1) a traditional controller design for multi-agent collision avoidance; 2) a manual controller design with pheromone communication, and 3) a DRL-driven automatic controller design also accompanied by pheromone communication. While the traditional and manual controller designs relied on predefined settings, the DRL-driven automatic controller underwent a separate training phase for each stage of experiments. Upon completion of training, the fully trained controller was then employed in the subsequent evaluation of performance. A comparative analysis was performed during this training process, assessing the proposed strategy against two established baselines. This comparison revealed the relative effectiveness and distinct capabilities of the proposed strategy.

4.4.2 Traditional Multi-agent Collision Avoidance

Non-Holonomic Optimal Reciprocal Collision Avoidance (NH-ORCA) is a centralised method for multi-robot collision avoidance and navigation [169], [170]. Unlike its predecessor, ORCA [168], NH-ORCA integrates the non-holonomic characteristics of mobile robots to calculate the optimal velocity required for collision-free motion. By considering the position and velocity of each agent, the algorithm defines the necessary velocities for the subsequent sample time, thus ensuring a collision-free trajectory. As the Turtlebot3, a non-holonomic wheeled mobile robot, was selected as the robotic platform for this study, NH-ORCA provides a particularly effective means of collision avoidance for the given experimental setup. Its ability to guarantee collision-free navigation when the positions and velocities of the agents within a swarm are known makes NH-ORCA a fitting choice for the baseline algorithm in the comparative analysis with the proposed pheromone-based collision avoidance strategy. Further insights into the NH-ORCA algorithm can be found in the original paper [170].

4.4.3 Manual Controller Design

The manual controller design was proposed as the baseline to compare the performance of DRL-driven automatic control design and was designed to perform navigational tasks whilst avoiding collisions. A flowchart for the manual controller design is provided in Fig. 4.5. During operation, the robot assesses pheromone intensity via the tips of its virtual antennae, as depicted in Figure 4.3. These virtual antennae, affixed to the robot for the purpose of sensing virtual pheromones, extend 0.45 meters (i.e., $l_a = 0.45\,\text{m}$) on both left and right sides and are each tilted by θ_a and $-\theta_a$ respectively from the robot's heading vector. In the context of real vehicles, equivalent virtual antennae could be realised either within each vehicle's cyber-physical space or through a centralised cloud system that manages all virtual pheromone information. Should physical materials be utilised to implement the pheromone, corresponding physical antennae might be affixed to the vehicles' front sides.

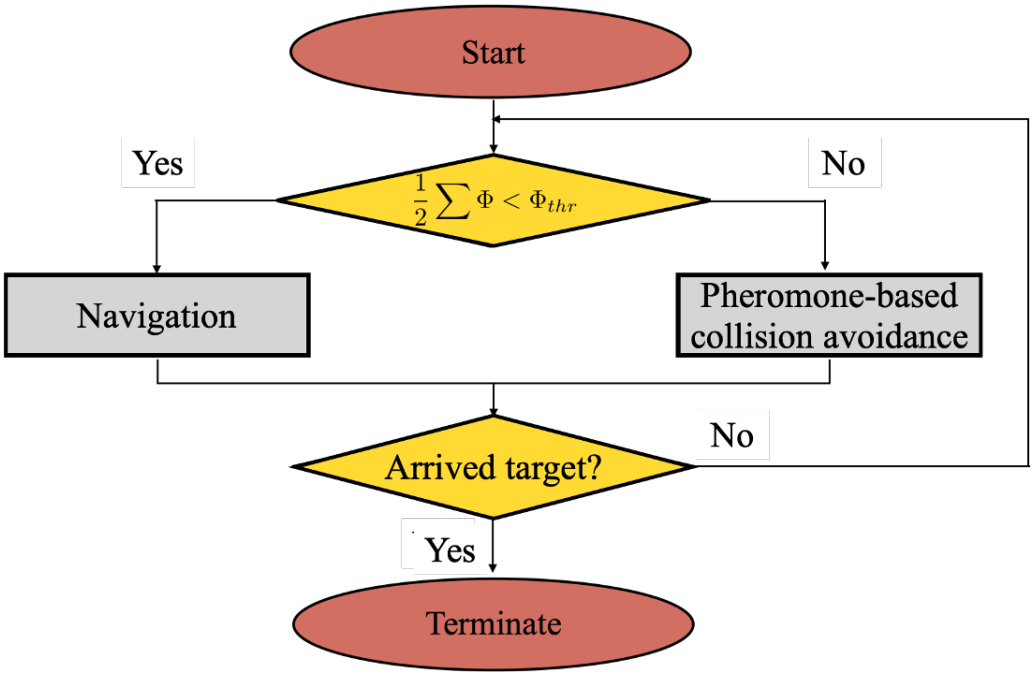

Figure 4.5. A flowchart of the manual controller design.

The pheromone intensities detected by the left and right antennae are denoted as Φ_l and Φ_r, respectively. If the sum of these intensities, $\Phi_l + \Phi_r$, exceeds a predetermined threshold, Φ_{thr}, the robot's motors are regulated to avoid a collision. Conversely, if the combined intensity falls below this threshold, the robot's motors are set to navigate toward the target, irrespective of the pheromone intensity sensed.

Inspiration for the design of this pheromone-based collision avoidance approach was drawn from the earlier studied deployed manual controller design [46], where pheromone-based foraging and aggregation techniques were applied to a swarm of mobile robots using dual artificial pheromone inputs. The collision avoidance model employed in this work was adapted from the controllers used in those studies and is defined as follows:

$$\begin{aligned} \beta &= \beta_{const} - \frac{\Phi_l + \Phi_r}{2} \\ w_l &= \beta + \alpha(\Phi_l - \Phi_r), \quad w_r = \beta + \alpha(\Phi_r - \Phi_l) \ , \end{aligned} \tag{4.4}$$

where β is a bias velocity, β_{const} is a constant bias velocity, Φ_l, Φ_r represent the pheromone intensity readings from the left and right antennae respectively, α is a sensitivity gain applied to the difference in pheromone intensities measured by the two antennae, and w_l, w_r denote the left and right wheel velocities of the mobile robot, respectively. Optimal values of β_{const} and α, yielding the best performance, were identified through preliminary experiments using a diverse range of parameters prior to the main experiments.

During these preliminary trials, parameters were assigned within the bounds of $0.6 \leq \beta_{\text{const}}, \alpha \leq 1.4$ in increments of 0.1. The controller was then assessed over 20 repetitions for each stage of experimentation, and the parameter set demonstrating the strongest performance was selected for

the main evaluation. Results from these preliminary tests are available in Appendix A. Parameter combinations for each stage yielding the highest performance metrics were deemed the optimal sets for those stages. The range for these parameters was empirically chosen to encompass values enabling successful collision avoidance behaviours with navigation behaviours. Parameters outside this range caused the vehicle to exhibit unrealistic backward movement or fail in avoiding collisions.

The robot's navigation behaviour is delineated in Equation (4.5). When the pheromone intensity detected by both antennae fell below the threshold value Φ_{thr}, the robot executed its navigation tasks.

$$\begin{aligned} &\phi = \text{atan2}\left(\frac{y_{targ} - y}{x_{targ} - x}\right), \quad \psi = \phi - \theta, \\ &v = v_{const}, \\ &\omega = \min\left(1, \max\left(-1, \omega_{coef}\, \text{atan2}\left(\frac{\sin(\psi)}{\cos(\psi)}\right)\right)\right), \end{aligned} \tag{4.5}$$

where $(x_{\text{targ}}, y_{\text{targ}})$ denote the 2D Cartesian coordinates of the target, θ represents the angular displacement between the robot's local coordinate frame and the global frame, v_{const} is a predetermined linear velocity, and ω_{coef} is a coefficient for angular velocity. By employing this equation, the robot ascertains (v, ω), the linear and angular velocities, respectively. For the experiments, $v_{\text{const}} = 0.5\,\text{m/s}$, and $\omega_{\text{coef}} = 10$, values that were empirically determined to minimise fluctuations. Table 5.1 compiles the parameters and corresponding values utilised in the experimentation.

4.4.4 DRL-driven automatic controller design

The details of the DRL-driven automatic controller design are elaborated, encompassing four vital design specifications: *i*) observation space, *ii*) action space, *iii*) reward design, and *iv*) actor and critic neural network structure. Utilising these design specifications, the DRL-driven automatic controller design was implemented for comparative analysis of the proposed strategy in the training phase and evaluative phase: *1*) PER sampling strategy, *2*) HLER sampling strategy, and *3*) HLER sampling strategy complemented with a noisy network (HLER+noisy).

Observation Space

During the experiments, observations were made of the pheromone intensity and gradient at the tips of the virtual antennae, resulting in a total of four measurements. These included the current pheromone intensities, Φ_l, Φ_r, and the differences in pheromone intensities between the current and previous time steps, $\Delta\Phi_l, \Delta\Phi_r$. This approach was inspired by insects' response to pheromone gradients in nature through chemotactic behaviour [184]. The absolute values of pheromone intensity were normalised within the range $[0 - 1]$ to enhance the effectiveness of training in DRL. For navigation, four additional observation inputs were employed: the polar coordinates of the distance to the navigation goal (d, θ), and the linear and angular velocities of the robot at the previous time

step. Providing the previous linear and angular velocities as inputs enabled the robot to learn the relationship between the velocities across two consecutive time steps. Consequently, the DRL controller takes eight observational inputs, each within a continuous range, i.e., $o_t \in (8 \times 1)$.

Action Space

The DRL-driven automatic controller design must produce two outputs to maneuver the non-holonomic mobile robot: translational velocity, v, and rotational velocity, w, denoted as $a = [v, w]$. These velocities are constrained within specific ranges, with $v \in (0, 1)$ m/s and $w \in \left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$ rad/s, to adhere to the robot's motion limitations.

Reward Design

One critical element in the design of the DRL-driven automatic controller design is the construction of the reward functions, as selecting appropriate values can significantly influence the controller's ability to manifest desirable behaviours. In the experiments conducted, the reward functions were meticulously crafted to facilitate the learning of three specific tasks: i) pheromone-based collision avoidance, ii) navigation, and iii) trajectory smoothing. These functions, encompassed within five distinct types of rewards, are integral to the effective learning of collision avoidance and navigation behaviour by the DRL-driven controller. The precise formulation of these reward functions is detailed in Equation (5.2).

$$
\begin{aligned}
& r = r_c + r_g + r_p + r_v + r_w \\
& r_g = \begin{cases} R_g, & \text{if arrived goal} \\ 0, & \text{otherwise} \end{cases}, \quad r_c = \begin{cases} R_c, & \text{if collision} \\ 0, & \text{otherwise} \end{cases} \\
& r_d = \begin{cases} ad, & d > 0 \\ -ad, & \text{otherwise} \end{cases}, \quad r_v = \begin{cases} R_v, & \text{if } v < v_{min} \\ 0, & \text{otherwise} \end{cases}, \\
& r_w = \begin{cases} R_w, & \text{if} |w| > w_{max} \\ 0, & \text{otherwise} \end{cases}.
\end{aligned}
\tag{4.6}
$$

The total reward, r, was computed as an aggregation of various rewards and penalties, each reflecting a specific aspect of the robotic behaviour. This includes the achievement of the goal, r_g, collision incidents, r_c, progress towards the goal, denoted by r_d (calculated as the product of the step length, d, and a tunable factor, a, empirically set to 4.0), and penalties associated with linear velocity, r_v, and angular velocity, r_w. The detailed values for these parameters, tailored to the experiments, are delineated in Table 5.1. It should be noted that these values can be adjusted to suit different experimental setups and task requirements.

Table 4.1. Parameters and values for DRL training configurations

Parameters	Values	Parameters	Values
v_{const}	0.5	a	4.0
ω_{coef}	10	v_{min}	0.2 m/s
β_{const}	0.6 - 1.4	ω_{max}	0.8 rad/s
α	0.6 - 1.4	R_v	-1
R_g	100	R_ω	-1
R_c	-100	p_{base}	0.5
k_{HL}	2.0	r_{HL}	20
t_{out}	60 (s)	N	512
T	256	γ	0.90

Neural Network Architecture

The architecture of the neural network used for the experiments is shown in Fig. 4.6. The actor network comprised input and output layers, representing observations and actions, respectively. It included three fully connected layers, each containing 512 neurons, with rectified linear activation function (ReLU) nonlinearities positioned between the input and output layers. The linear and angular velocities were constrained using sigmoid and tanh activation functions, resulting in limits of $0 \leq v_t \leq 1$ and $-1 \leq \omega_t \leq 1$, respectively.

Similarly, the critic network was structured with input and output layers corresponding to observations and state-action values. It contained three fully connected layers with the same number of parameters as the actor network. However, a distinct feature of the critic network was the concatenation of actions after the first layer, which were then introduced into the second layer.

Additional Training Details

For other parameters of DDPG algorithm, batch size, N, of 512, episode length, T, of 256, discount factor, γ, of 0.90 are used.

4.4.5 Metrics

The performance of the controllers in the experiments was evaluated using three distinct metrics:

- *Success Rate* (ρ_s): Defined as the percentage of successful runs, where no collisions occurred, and the completion time remained below the threshold, t_{out}.
- *Completion Time* (t_{comp}): Represents the duration required for all robots to reach their targets without collision and within the time constraint of t_{out}.
- *Trajectory Efficiency* (η_t): Calculated as the average ratio between the Euclidean distance connecting the start and end points of the robots, and the actual distance traveled by the robots during each successful run.

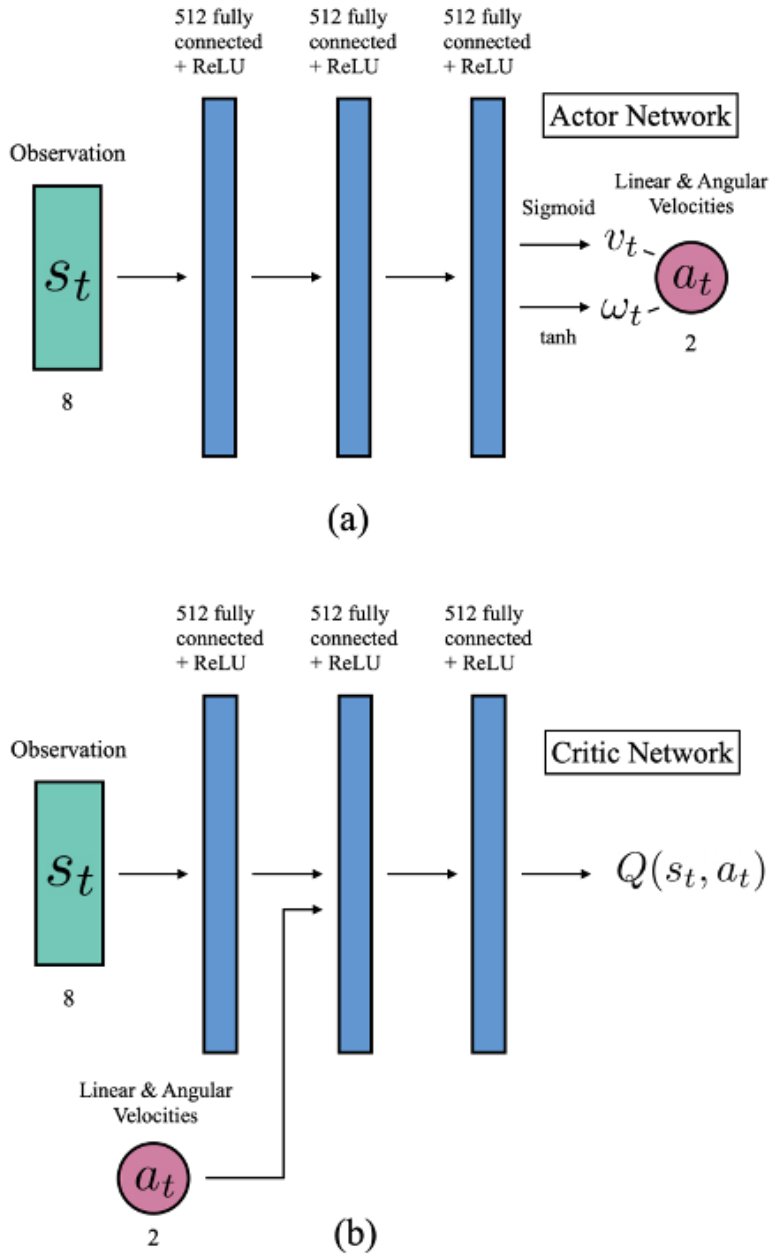

Figure 4.6. Neural network architecture for the DRL-driven controller. (a) and (b) represent the network architectures for the actor and critic networks respectively.

4.5 Results & Discussion

The experimental results were categorized into subsections to detail the performance of the traditional centralised NH-ORCA controller, manual controller, and DRL-driven automatic controller design. Included within the reported results is a comparative analysis of three different sampling strategies, along with an examination of the robustness against noise in the DRL-driven automatic controller design. Figure 4.7 illustrates the experimental results across all stages. Subsequently, a comparison was made between the traditional centralised NH-ORCA controller and both manual and DRL-driven automatic controller designs employing virtual pheromone. A further discussion of these results follows.

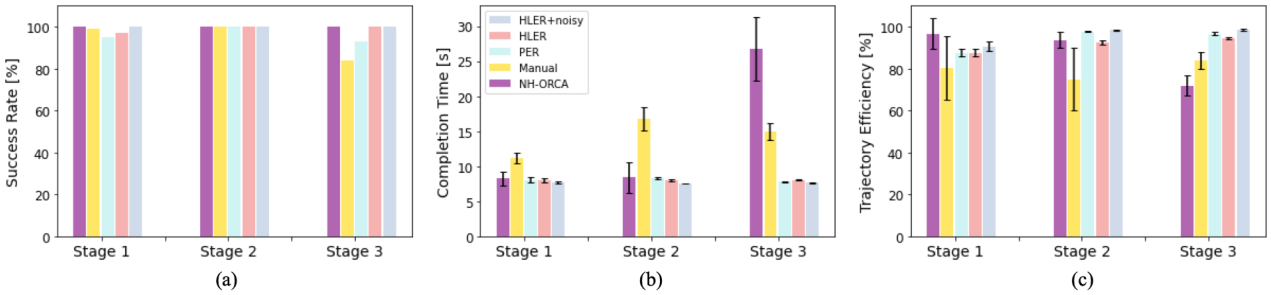

Figure 4.7. The bar plots show the experimental results with the traditional controller with NH-ORCA, the manual controller design (HT), and DRL-driven automatic controller design with three different training strategies, PER, HLER and HLER with noisy network in three experimental stages. (a), (b), and (c) illustrate the success rate, completion time and trajectory efficiency in Stage 1, 2 and 3 respectively. The error bars represent standard deviation of each metric.

4.5.1 Traditional Centralised Controller

Figure 4.7 presents the experimental results for the NH-ORCA controller across three stages, including the corresponding three metrics. The numerical values for these metrics can be found in Table 4.2. As indicated by the results, the NH-ORCA controller achieved a success rate of 100% in all three stages, underscoring the high stability of the centralised controller when provided with position and velocity information. However, this stable success rate contrasts sharply with the notable increase in completion time and decline in trajectory efficiency observed in Stage 3. This discrepancy highlights that while the traditional centralised controller ensures stability, it may lack effectiveness in complex environments for finding an optimal strategy. In essence, the centralised controller demonstrated limited adaptivity as complexity increased from the swarm robotic perspective.

4.5.2 Manual Controller Design

The preliminary experiments facilitated the selection of the best parameter sets for each stage, with further insights on the parameter choices detailed in Appendix A. Figure 4.7 displays the experimental results for the three stages, and the numerical values are provided in Table 4.2. The results highlight a significant limitation of the manual controller design in complex environments. While achieving success rates of 99% and 100% in Stages 1 and 2 respectively, the success rate decreased to 84% in Stage 3. This pronounced drop underscores that the performance of the manual controller design diminishes with increasing task and environmental complexity.

4.5.3 DRL-driven automatic controller design

Here, the experimental results for the DRL-driven automatic controller design utilising three distinct DRL algorithms (PER, HLER, HLER+noisy) are delineated. Initially, the training performance across the three stages is compared among the algorithms, highlighting the advantages of the proposed DRL method over baseline approaches. Subsequently, a detailed comparison of the experimental results for each stage is provided for the three algorithms.

Table 4.2. Experimental results of the controllers in the three stages.

Stage	Controller	ρ_s	t_{comp}	η_t
Stage 1	NH-ORCA	100%	8.25 ± 0.95	0.9666 ± 0.0738
	Manual	99%	11.16 ± 0.75	0.8036 ± 0.1527
	PER	95%	8.04 ± 0.36	0.8765 ± 0.0196
	HLER	97%	8.06 ± 0.30	0.8762 ± 0.0186
	HLER+noisy	100%	7.68 ± 0.17	0.9068 ± 0.0220
Stage 2	NH-ORCA	100%	8.39 ± 2.20	0.9367 ± 0.0381
	Manual	100%	16.74 ± 1.68	0.7486 ± 0.1485
	PER	100%	8.34 ± 0.15	0.9762 ± 0.0030
	HLER	100%	7.96 ± 0.11	0.9234 ± 0.0097
	HLER+noisy	100%	7.54 ± 0.06	0.9837 ± 0.0021
Stage 3	NH-ORCA	100%	26.81 ± 4.54	0.7187 ± 0.0477
	Manual	84%	14.95 ± 1.16	0.8394 ± 0.0387
	PER	93%	7.81 ± 0.08	0.9672 ± 0.0065
	HLER	100%	8.01 ± 0.07	0.9440 ± 0.0048
	HLER+noisy	100%	7.63 ± 0.06	0.9850 ± 0.0032

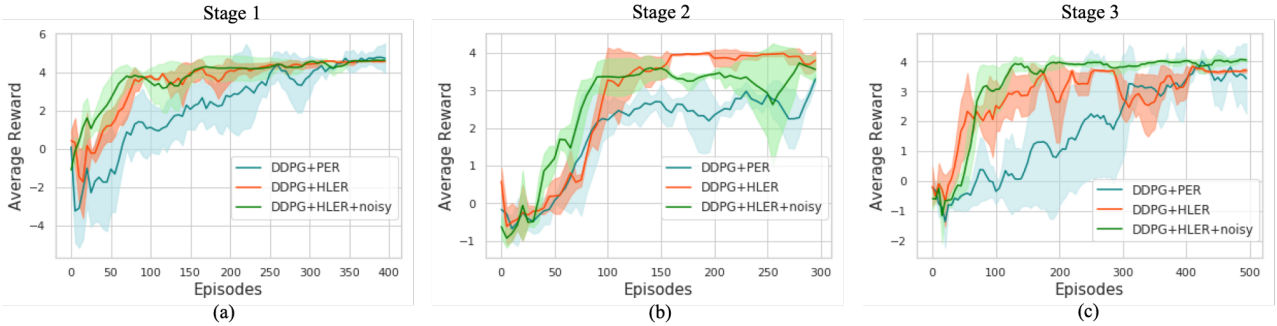

Figure 4.8. Average reward per episode during training with PER, HLER and HLER with noisy network. (a), (b) and (c) illustrate the change in average reward over training in Stage 1, 2 and 3 respectively. The bold line and shading represent the mean and standard deviation of the reward over 3 different runs with different random seeds respectively.

Figure 4.8 illustrates the average reward per episode during the training phase for each stage, while Table 4.2 and Figure 4.7 present the results obtained using the DRL algorithm with the proposed enhancement, as well as the other two baseline methods.

Training Efficiency

It was consistently observed that the HLER sampling strategy accelerated training speed in comparison to the PER sampling strategy. Additionally, the incorporation of a noisy network further enhanced the training speed. Quantitatively, the HLER with a noisy network required only 25%, 27%, and 30% of the training time needed by the PER sampling strategy to reach convergence in Stage 1, Stage 2, and Stage 3, respectively, with an average reduction of 27% across all stages. These results were consistent with expectations, as HLER was specifically designed for faster training through prioritized sampling of highlighted transitions, and the implementation of a noisy network facilitated better exploration. Together, these features led to more efficient searches for optimal policies across all experimental stages. Regarding sampling complexities, the proposed HLER with a noisy network method required 10.4M, 11.8M, and 17.0M training steps to reach convergence for Stage 1, Stage 2 and Stage 3 respectively.

Performance Evaluation

In Stage 1, the DRL-driven automatic controller designs with both PER and HLER sampling strategies showed similar performances across the three metrics. However, integrating a noisy network with HLER led to enhanced performance in all aspects.

In Stage 2, a parallel trend was observed where PER and HLER exhibited no significant differences. Although trajectory efficiency was higher with PER, HLER yielded a faster completion time. This outcome was attributed to the fact that the DRL-driven automatic controller design trained with HLER produced slightly longer trajectories to further ensure collision avoidance. Controllers that incorporated noisy networks surpassed the others in all metrics.

Stage 3 presented a more complex picture. The controller utilising the PER sampling strategy outperformed its HLER counterpart in terms of completion time and trajectory efficiency, albeit at a lower success rate. This is explained by the fact that PER minimised completion time at the expense of success rate. Conversely, the use of HLER ensured a 100% success rate, as this strategy prioritizes collision avoidance. Once again, controllers employing the noisy network approach demonstrated superior performance, consistent with results from the previous stages. This further supports the notion that the noisy network enhances the agent's exploration, optimisming both safety and performance.

4.5.4 Comparison between the traditional centralised controller and decentralised pheromone-based controllers

Here, we present a comparison among the traditional centralised NH-ORCA controller, manual, and DRL-driven automatic controller designs with a pheromone-based collision avoidance strategy. The objective is to highlight the differences between centralised and decentralised (pheromone-based) approaches.

For the manual controller design with pheromone-based communication compared to the NH-ORCA controller:

- In Stage 3, it demonstrated a lower success rate.
- In Stages 1 and 2, the NH-ORCA controller exhibited higher performance in terms of completion time and trajectory efficiency.
- In Stage 3, the manual controller design surpassed the NH-ORCA controller in completion time and trajectory efficiency.

These findings imply that the decentralised, manual controller design approach leads to more effective behaviours in complex environments, despite a lower success rate.

For the DRL-driven automatic controller design with pheromone-based communication compared to the NH-ORCA controller:

- It achieved equally high success rates across all three experimental stages.
- It greatly outperformed the NH-ORCA controller in completion time and trajectory efficiency, especially in Stage 3.

This indicates that the DRL-driven approach offers a similar degree of stability to the centralised controller but adds greater flexibility and effectiveness in complex environments.

In conclusion, while the centralised controller ensures stability across diverse scenarios, its performance diminishes with increasing complexity. Conversely, the decentralised, pheromone-based approach offers higher flexibility, with its effectiveness dependent on the specific controller used. A more detailed comparison between manual and DRL-driven automatic controller designs utilising pheromone-based communication is presented in the subsequent section.

4.5.5 Comparison between the manual and DRL-driven automatic controller design

Here, the list of more in-depth comparison between the manual and DRL-driven automatic controller designs utilised with the pheromone-based approach. Specifically, the controller trained with the proposed DRL algorithm (HLER+noisy) was used, as it exhibited superior performance among the DRL algorithms employed in the experiments.

- **Success Rate:** The manual controller design achieved 99% and 100% in Stage 1 and 2 but only 84% in Stage 3. This decline in Stage 3 illustrates the limitations of the manual design as environmental complexity increases. Conversely, the DRL-driven automatic design maintained a consistent success rate of 100% across all stages, demonstrating resilience to varying complexities.
- **Completion Time:** The DRL-driven automatic design considerably outpaced the manual design in all stages. In autonomous vehicle applications, this is vital as it reduces travel time, significantly enhancing user satisfaction.
- **Trajectory Efficiency:** Again, the DRL-driven automatic design outshone the manual design in all stages. This attribute is especially critical in real-world scenarios, where lower trajectory efficiency may lead to congestion, inefficient movement, and a higher risk of collisions.
- **Robustness:** The DRL-driven automatic design also demonstrated superior robustness, with lower standard deviations for both completion time and trajectory efficiency. This suggests that the manual design's performance may vary significantly with minor changes in initial conditions, making it less desirable in real-world environments.

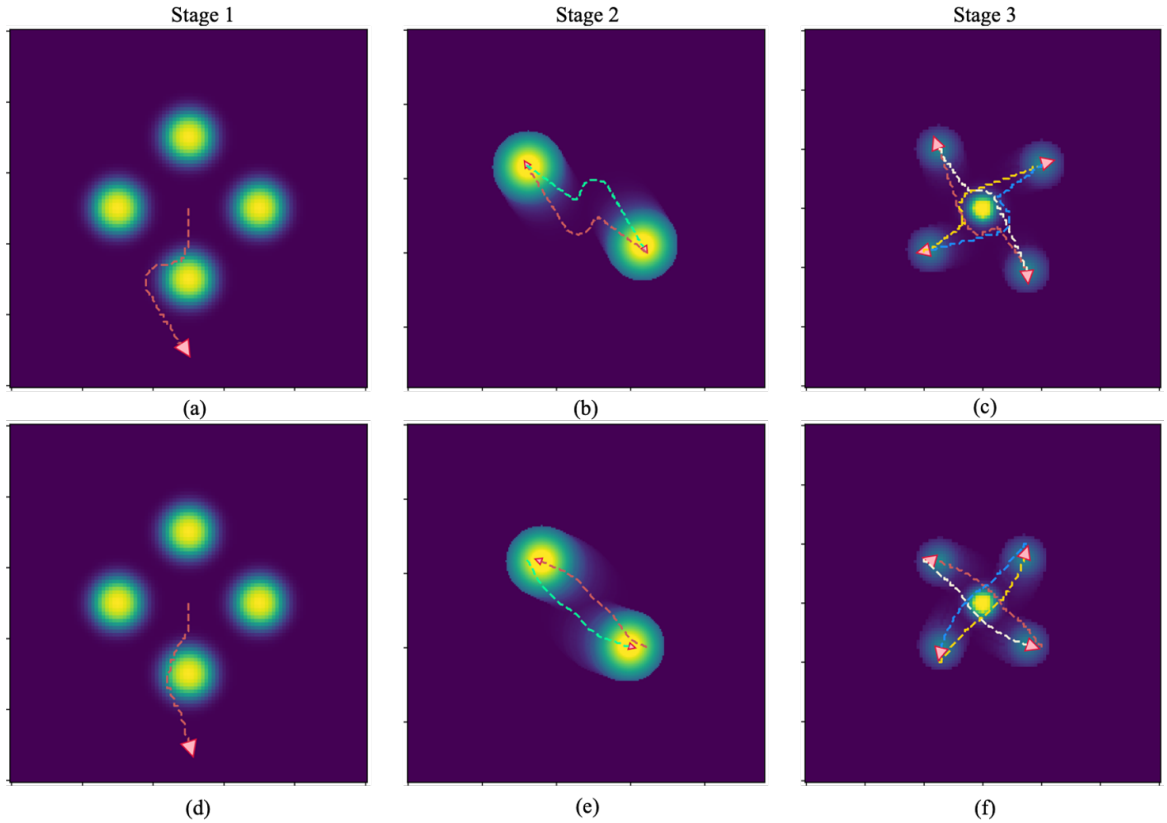

Figure 4.9. Trajectories of the robots in the three experimental stages. Subfigures (a), (b), and (c) depict the trajectories using the manual controller design in Stages 1, 2, and 3, respectively. Subfigures (d), (e), and (f) illustrate the trajectories using the DRL-driven automatic controller design (employing DDPG, HLER, and a noisy network) in the corresponding stages. Triangles indicate the final position and orientation of the robots. The intensity of the yellow shading on the map represents the concentration of pheromones, with a higher intensity indicating a greater concentration.

- **Practical Benefits:** The automation of the optimisation process in the DRL-driven design contrasts with the laborious tuning required for the manual design. This advantage is particularly valuable in swarm systems, where manual tuning may be highly challenging due to real-world complexity [3].

In conclusion, the DRL-driven automatic controller design for pheromone-based collision avoidance excelled over the manual design in aspects including stability, flexibility, robustness, and ease of tuning. Coupled with the comparison to the centralised controller, the DRL-driven design assures high flexibility from the pheromone-based approach and robust stability and performance through DRL optimisation.

4.5.6 Discussion

Trajectory Analysis

To elucidate the effectiveness of the pheromone-based collision avoidance strategy, the trajectories of the robots employing two distinct controllers (manual controller design and DRL-driven automatic controller design with DDPG + HLER + noisy network) were meticulously analysed across all three experimental stages. Figure 4.9 depicts these trajectories, highlighting key differences in

navigational behaviour.

The manual controller design manifested a heightened sensitivity in its repulsive response to obstacles. Upon nearing an obstacle, the controller would abruptly alter the robot's course, resulting in rapid evasion. This pattern was consistently observed across all stages, demonstrating a tendency towards overreaction.

Conversely, the DRL-driven automatic controller design exhibited a more tempered and smooth response. The absence of sudden directional changes enabled the robots to navigate effectively while actively evading collisions. This well-balanced approach underscores the capacity of the DRL-driven design to harmonise conflicting requirements, optimising overall performance in alignment with the predefined objectives. In practical applications, such fluid movement is advantageous, likely reducing collisions and enhancing user satisfaction.

Furthermore, the investigations revealed striking similarities between the trajectories produced by the DRL-driven automatic controller in Stage 3 (Figure 4.9 (f)) and those documented in other studies employing laser proximity sensors for collision avoidance and navigation [108]. This resemblance suggests that the implementation of virtual pheromones can achieve behaviours akin to those yielded by more conventional methods, thereby validating the innovative approach undertaken in this research.

Pheromone Characteristics

Though the experimental results revealed the superior performance of the DRL-driven automatic controller design over the traditional manual controller design, the optimality of this design remains undetermined. As demonstrated in the study using artificial pheromones for urban waste management [41], the selection of appropriate parameters for artificial pheromones significantly influences the performance of a swarm robotic system. However, this study did not include exhaustive testing to identify optimal parameters for the virtual pheromone, such as evaporation half-life, diffusion rate, shape, or the gradient of the injected pheromone. This limitation represents an area for future exploration, particularly concerning the selection of optimal parameters in more complex environments.

Assumptions and Constraints in DRL-driven Automatic Controller Design

One unrealistic assumption in training DRL-driven automatic controller design for swarm robotic systems is the existence of a central computing unit. This unit is assumed to conduct training shared by all robots without any communication constraints with them, as simulated robots were deployed sharing the computing unit with the environment. This assumption is problematic and must be mitigated, as it contradicts the decentralisation principles of the swarm robotic systems. Consequently, it leads to an overestimation of the systems' robustness, scalability, and flexibility, and an underestimation of the potential instability and limitations of global communication between

the robots and the central computing unit. These factors are particularly significant in environments where swarm robotic systems are likely to be deployed. Nevertheless, this research study affirms the potential of DRL-driven automatic controller design, and for the purposes of training and evaluation, this realistic constraint is not included, omitting further considerations of implementation detail.

4.6 Summary

This chapter investigates the development of a DRL-driven automatic controller design for swarm robotic systems, with a particular focus on autonomous vehicles engaged in collision avoidance at a traffic level. By extending the concept of pheromone communication—a promising approach to swarm robotic communication—into a more complex and realistic context, it enhances individual collision avoidance for efficient vehicle coordination.

The application of swarm robotic systems to real-world scenarios calls for a controller design that can handle complexity, non-stationarity, and unpredictability, as detailed in Chapter 2. The proposed DRL-driven automatic controller, employing the proposed HLER sampling strategy and exploration strategy with noisy network, was tested in a collision avoidance context, revealing heightened adaptivity to task and environmental complexity. Comparative analyses underscored its performance advantage over traditional centralised methods and manual controllers utilising pheromone communication, especially with escalating complexity that shows better adaptivity, showcasing the achievement of **Aim 2 & Objective 2**.

One critical aspect of this study is the assumption of the presence of a central server for controller training and stable connections between the server and individual robots. This assumption, as earlier delineated, contradicts the decentralisation principles inherent to swarm robotic systems. It leads to an overestimation of robustness, scalability, and flexibility, and underestimates potential instability and limitations, particularly in environments where swarm robotic systems are likely to be deployed.

In conclusion, this chapter emphasises the need for strategies to mitigate reliance on a central server, a vital step towards making DRL-driven controller designs practical and effective for real-world swarm robotic applications. The chapter affirms the potential of DRL-driven automatic controller design, but also underscores the necessity for a realistic and thorough examination of constraints, ensuring alignment with the inherent principles of decentralisation in swarm robotic systems. To address the issue of centralisation, Federated Learning (FL) is adopted for training DRL algorithms used for automatic controller design for swarm robotic systems. This chapter successfully showcases DRL-driven automatic controller design and highlights the need for decentralisation to thrive as a viable automatic controller design method for swarm robotic systems, which is addressed in the next chapter.

Chapter 5

Towards Decentralised Deep Reinforcement Learning-Driven Automatic Controller Design

Chapter 4 delineated the promise of DRL-driven automatic controller design within the swarm robotics framework, establishing its merit through a comparative analysis against manual controller design and traditional multi-agent robotic control approaches. This analysis vividly showcased the superior performance of the DRL-driven system in governing basic robotic behaviours and its greater adaptivity in navigating escalating complexities within tasks and environments. However, it is imperative to acknowledge that these benefits are predominantly derived from the MARL framework. Therefore, to fully actualise and validate the potent capabilities of DRL-driven automatic controller design, especially in the context of real-world swarm robotic expeditions, mitigating the existing constraints imposed by the MARL framework on swarm robotic applications becomes a vital endeavour.

In this Chapter, FL has been incorporated into DRL-driven automatic controller design for swarm robotics as a remedy for centralisation issue. Echoing the investigations of Chapter 4, navigation and collision avoidance behaviour have been adopted as foundational low-level elements in fostering collective actions across various swarm robotic missions. This chapter conducts an in-depth analysis of FL's influence on the efficiency of DRL-driven automatic controller design and its potential to reduce reliance on central computing units. This demonstration underscores the advantages of the FL framework not only diminishing the degree of centralisation, but also the capability to reduce the reality-gap, by offering a higher level of generalisation compared to both MARL and single-agent RL methodologies. This chapter concludes with the promise of FL-based DRL training strategy for DRL-driven automatic controller design to pave the way to be a general methodology for automatic controller design.

5.1 Background

Utilising the FL framework to implement the DRL algorithm for multi-agent systems has found increasing adoption in many multi-agent applications. This strategy mitigate the limitation of the MARL framework, which inherently lacks the ability to facilitate decentralisation. The use of the FL framework not only facilitates decentralisation but also harnesses collective intelligence, thereby enhancing the overall efficiency of the system [123]. This advantage of FL framework training DRL algorithm for multi-agent systems in several domains matches with the need for the robot learning methods for automatic controller design, by offering a robust solution to the centralisation dilemma, the curse of dimensionality, and credit assignment issues that challenge the MARL framework.

In this research study, FL-based DRL training strategy is proposed and investigated on the basic swarm robotics scenario, actualising the speculation that FL can provide a promising alternative to MARL-based DRL-driven automatic controller design. Choosing the basic navigation and collision avoidance as the behaviours to be examined, training curve, number of communication and performance efficacy of the trained controllers were examined to. Through real-robot experiment, the transferrability of the trained controller to investigate effect of the reality-gap is shown. The novelty of this research study is two-fold. From the perspective of the swarm robotics community, this research study is the first study that addresses the centralisation issue in robot learning methods in automatic controller design. From the perspective of general robotics, this research study introduces the application of FL into multi-robot systems accompanying experiments deploying real-robotic platforms.

The major contributions and findings of this research study are listed as follows:

- FL-based DRL training strategy, FLDDPG, is proposed applying FL framework to the existing DDPG algorithm with enhanced model parameter update measures. It is first application of FL into swarm robotic scenario.

- During the training phase, FLDDPG exhibited a substantial reduction in communication with the central computing unit coupled with a low incidence of failure cases, thereby demonstrating partial decentralisation coupled with heightened stability.

- In the evaluative phase, FLDDPG manifested superior performance efficacy in comparison to baseline measures during simulation experiments. This superiority was further accentuated in real-robot experiments, illustrating the formidable potential of FLDDPG in narrowing the reality-gap.

In summation, this research shows the capability of FL-based DRL training strategy as a partially decentralised approach whilst not reducing performance efficacy. Furthermore, it demonstrated the ability to reduce reality-gap thanks to its inherent generalisation effects.

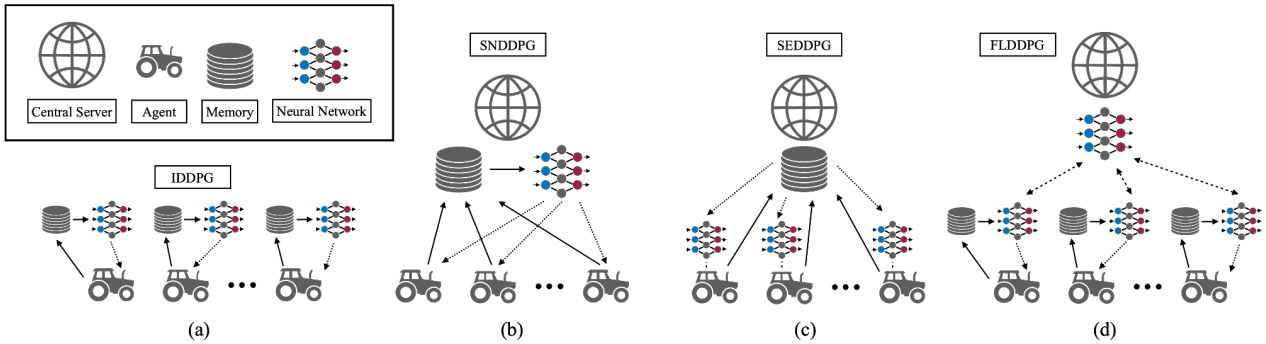

Figure 5.1. Deep Reinforcement Learning training strategies for swarm robotic systems. (a) Individual DDPG (IDDPG), (b) Shared Network DDPG (SNDDPG), (c) Shared Experience DDPG (SEDDPG) and (d) the proposed Federated Learning DDPG (FLDDPG).

5.2 Methodology

This section introduces the proposed federated learning (FL)-based DRL training strategy for swarm robotic systems. Capitalising the DDPG algorithm as a common backbone, which is described in Section 2.4.1., three traditional DRL training strategies that can be utilised as automatic controller design are introduced. Two traditional strategies, SNDDPG and SEDDPG, adopt MARL framework as consistent communication between agents and the central server is necessary, i.e. fully centralised communication. Finally, the proposed FL-based DRL training strategy for automatic controller design is illustrated. Figure 5.1 illustrates how the three traditional training strategies and our proposed strategy are utilised for swarm robotic systems in this study. It is worth noting that every strategy utilises a standard type of MDP, not a Partially Observable Markov Decision Process (POMDP), which assumes that states are partially observable. In other words, it is assumed that agents have full access to state information without uncertainty.

5.2.1 Traditional DRL Training Strategies

Here three traditional DRL training strategies using DDPG algorithms as backbone are introduced: i) Independent DDPG, ii) Shared Network DDPG and iii) Shared Experience DDPG.

Independent DDPG

Independent DDPG (IDDPG) training strategy is the most naïve application of DDPG to train a robot swarm. Fig. 5.1 (a) illustrates IDDPG training strategy. IDDPG deploys individual neural networks and local memory for each robot in a swarm. In other words, there is no communication between the robots and the server for training. The advantage of IDDPG is that it does not use a central server to share data samples or neural network parameters. However, this training strategy does not allow to utilise the collective nature of swarm robotic systems for learning.

Shared Experience DDPG

Shared Experience DDPG (SEDDPG) training strategy is one of the state-of-the-art training strategy for multi-agent systems [185]. Fig. 5.1 (c) illustrates SEDDPG training strategy. In this training strategy, the agents have individual neural networks and a shared memory. In the original paper, it is stated that the advantage of SEDDPG is that by sharing it can encourage exploration, thereby faster convergence and better performance. Despite the improvement in the training speed and performance, the robots still share the collected data with the central server, which requires centralisation of the system.

Shared Network DDPG

Shared Network DDPG (SNDDPG) is a training strategy that uses a shared neural network and a shared memory. This is a typical implementation of DDPG in a centralised multi-agent scenario. Fig. 5.1 (b) illustrates SNDDPG training strategy. In each time step, the robots transfer the transition sample data to the shared memory located in the central server. In every training period, the network update is performed by the central server and the model is distributed to the individual agents. The advantage of this method is that the central server can be trained using the data samples collected by different agents in different environments. Compared to IDDPG, it encourages the networks to learn from more diverse data, leading to a more generalised controller. However, it requires frequent communication between each robot and the central server. The more frequent the communication between robots and the central server is required, the more vulnerable the training is as the communication is unstable.

5.2.2 FL-based DRL Training Strategy

Using the concept of FL and DDPG algorithm, a FL-based DRL training strategy, FLDDPG, is proposed. For more details about FL, please see Section 2.5. The process of FLDDPG is illustrated in Fig. 5.1 (d), showing only the neural network weights are shared in the central training server, without sharing the locally collected data. This feature can largely reduce the number of communications between the robots and the server as collected data sharing takes a large part in DRL settings as in SNDDPG and SEDDPG.

Algorithm 3 (d) describes the algorithm of FLDDPG. At the beginning of the algorithm, actor and critic networks and local memories are initialised with the number of robots. The data collection and neural network parameter update are performed individually for each robot. After initialisation, the transition samples are collected from the robots for T time steps and stored in the replay buffer in every episode. Every t_{train} times in every episode, the actor and critic networks are updated by the following process. First, l number of transition samples are randomly selected from the local memories. Second, the target, y_i, is calculated for each transition sample. Third, using the target, the temporal-difference loss, L_Q, is calculated and the critic networks are updated in a way minimising

Algorithm 3: DDPG with FL (FLDDPG)

1 Randomly initialise N critic neural networks, $Q_{1,...,N}(s, a|\theta^{Q}_{1,...,N})$ and N actor neural networks $\pi_{1,...,N}(s|\theta^{\pi}_{1,...,N})$ with weights $\theta^{Q}_{1,...,N}$ and $\theta^{\pi}_{1,...,N}$
2 Initialise N target networks $Q'_{1,...,N}$ and $\pi'_{1,...,N}$ with weights $\theta^{Q'}_{1,...,N} \leftarrow \theta^{Q}_{1,...,N}$, $\theta^{\pi'}_{1,...,N} \leftarrow \theta^{\pi}_{1,...,N}$
3 Initialise replay buffers $R_{1,...,N}$
4 **for** *episode = 1, ..., M* **do**
5 Initialise the states $s_t = s_1$
6 **for** *t = 1, ..., T* **do**
7 Run N actors and collect transition samples $D_t = (s_t, a_t, r_t, s_{t+1})$ into $R_{1,...,N}$
8 **if** *t = 0 mod* t_{train} **then**
9 Sample l transitions (s_i, a_i, r_i, s_{i+1}) from local replay buffer memories, $R_{1,...,N}$
10 Set $y_i = r_i + \gamma Q'(s_i, \pi'(s_{i+1}|\theta^{\pi'})|\theta^{Q'})$
11 Update critic networks by minimizing the loss: $L_Q = \frac{1}{N}\sum_i (y_i - Q(s_i, a_i|\theta^{Q}))^2$
12 Update actor networks using the sampled policy gradient:
13 $\nabla_{\theta^{\pi}} J \approx \frac{1}{N}\sum_i \nabla_a Q(s, a|\theta^{Q})|_{s=s_i, a=\pi(s_i)}$
14 $\cdot \nabla_{\theta^{\pi}} \pi(s_i|\theta^{\pi})|_{s=s_i}$
15 **end**
16 **if** *t = 0 mod* t_{target} **then**
17 Update the target networks:
18 $\theta^{Q'} \leftarrow \theta^{Q}$, $\theta^{\pi'} \leftarrow \theta^{\pi}$
19 **end**
20 **end**
21 **if** *episode = 0 mod* T_{wa} **then**
22 Perform *Soft Weight Update* for actor and critic networks for all robots
23 $\theta_{wa} = \frac{1}{N}\sum_{k=1}^{N} \theta_k$
24 **for** *i = 1, ... , N* **do**
25 $\theta_i = \tau\theta_i + (1 - \tau)\theta_{wa}$
26 **end**
27 **end**
28 **end**

the loss. Then, the actor networks are updated using the sampled policy gradient. Additionally, for every t_{target} time step, the newest actor and critic networks are assigned to the target networks. In Algorithm 3, lines 19 to 24, the process of neural network weight averaging and update in the central server are described. In every weight averaging and update period, T_{wa}, the local neural network weights are updated with the averaged weights. For a more efficient weight update, a new method called *Soft Weight Update* is proposed. In the seminal work of FL [123], the hard weight averaging called `FedAvg` was performed to average the local neural network parameters. The hard weight averaging is described in (5.1).

$$\begin{aligned} \theta_{wa} &= \frac{1}{N}\sum_{k=1}^{N} \theta_k \\ \theta_{1,...,N} &= \theta_{wa}, \end{aligned} \tag{5.1}$$

where θ_{wa} denotes the averaged weights of neural network, N is the number of robots, and $\theta_{1,...N}$ are neural network weights for N robots. With the hard weight update method, the averaged weights are directly assigned to the local neural network weights. The problem with the hard weight update method is that whenever the local weights are updated with the averaged weights, adverse changes in the neural network can occur, decreasing the efficiency of individual controllers in their corre-

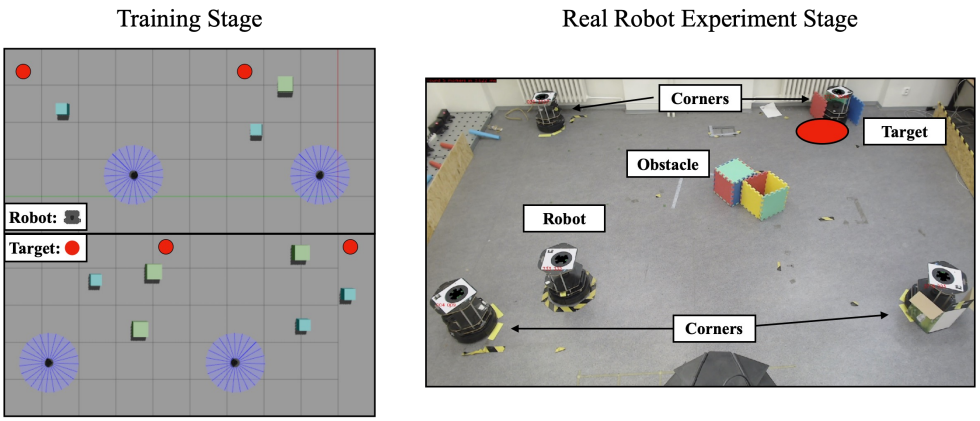

Figure 5.2. Experimental arenas for (a) training and (b) for real-robot experiments. In (a), four robots learn navigation to the target and collision avoidance independently under different environmental configurations. In (b), a robot performs navigation to the target and collision avoidance in a different environment than the training environment.

sponding environments and tasks after the update. To prevent such adverse changes in the neural network update after the weight update, a soft weight update method is proposed. The concept of soft weight update is that the local neural network weights are fractionally updated with the averaged weights. In the line 23 of Algorithm 3, the neural network weights are updated with the sum of $\tau\theta$ and $(1-\tau)\theta_{wa}$, where τ represents an update constant within the range $[0, 1]$. When τ is close to zero, the local neural network weights are completely replaced with the averaged ones. In contrast, when τ is close to 1, it becomes analogous to IDDPG.

5.3 Experiments

Experiments were designed to evaluate the performance of four different training strategies, IDDPG, SEDDPG, SNDDPG and FLDDPG, under the limited communication bandwidth scenario. To evaluate the training strategies with the swarm robotic scenario, a collective learning scenario of navigation and collision avoidance for swarm robotic systems was adopted. Fig. 5.2 illustrates the collective learning simulation environment and its real-robot evaluation environment. In Fig. 5.2 (a), the robots learn navigation and collision avoidance using the four training strategies. Although the robots do not communicate each other for collective behaviours, the robots help other robots to obtain a better controller utilising diversity of collected samples and trained models for larger number of agents than a single robot. Therefore, the collective learning scenario can be regarded as a swarm robotic scenario.

After training models, evaluation of the trained models were performed in the same environment in Fig. 5.2 (a). To evaluate the robustness and generalisation ability of the trained models with the four strategies, the real-robot platform was deployed in the environment illustrated in Fig. 5.2 (a). The trained robot used in the simulated environment is the Turtlebot3 Burger robot, which is a differential drive mobile robot with laser sensors to detect obstacles. The real-robot platform used for the real robot experiment is described in the later section. In the real-robot experiment, only a

single robot was used to assess clearly the performance of a collectively learned model from the swarm robotic systems rather than deploying multiple robots, which can add a higher degree of complexity affecting the evaluation of the trained model.

To add the constraint of limited communication bandwidth, the total data volume to be exchanged was calculated, a product of communication bandwidth and total communication time. The total data volume was calculated from the total data transferred during the training with FLDDPG. For each transfer of neural network parameters, it takes 0.55 MB one-way and 1.1 MB for the complete cycle of weight update. Since the weight update period, T_{wa} = 1, 120 weight update occurs for one training instance. Therefore, the total transferred data volume for training with FLDDPG is 132 MB.

With this total data volume, the update period for SEDDPG and SNDDPG were chosen to distribute update events over the training evenly. Both SEDDPG and SNDDPG include the data transfer of experience replay buffer, which size is 2.4 MB for one-way and 4.8 MB for the full cycle of transfer. For SEDDPG, the update period of 5 was chosen over the total 120 episodes as the total transferred data volume is 115.2 MB, while when the update period is four it is 144 MB, which is greater than the upper limit. For SNDDPG, a total 2.95 MB is transferred per update. Therefore, the update period of 3 was chosen over the total 120 episodes, resulting total transferred data volume of 118 MB. Unlike the three algorithms, the limitation of the total transferred data volume does not affect training process of IDDPG. The training of models using four strategies applied to these update period settings.

5.3.1 DRL Implementation

Here the DRL implementation details for four training strategies are described. The four important design specifications are introduced as: i) observation space, ii) action space, iii) reward design and iv) actor and critic neural network structure.

Observation Space

During the training, the robot collected observation to learn navigation and collision avoidance. For navigation, the distance between the target and the current position of the robot in polar coordinates (d, θ_d) were collected. For collision avoidance, 24 sensor readings from the laser rangefinder were collected. After collecting the 24 laser sensor readings, the readings are normalised in the range between [0, 1], and the normalised readings are inversed so that when the obstacle is close to the robot, the normalised sensor reading is close to 1 for a more effective neural network training. While the range of the laser sensor is up to 3.5 m, only the value below 0.8 m was used to learn more effective collision avoidance algorithm.

Action Space

There are two action values in our setting: i) translational velocity, v, and ii) rotational velocity, ω, i.e. $a = [v, \omega]$. The range of velocities were limited to $v \in (0, 0.25)$ m/s and $w \in (-\frac{\pi}{2}, \frac{\pi}{2})$ rad/s in translation and rotation respectively to reflect the motion constraints and safety for the robot.

Reward Design

There were three types of reward functions that were applied in the experiments. The reward functions were designed to enable the learning of i) navigation, ii) collision avoidance. Equation (5.2) describes the reward functions used in the experiments.

$$
\begin{aligned}
r &= r_g + r_p + r_c + r_a \\
r_g &= \begin{cases} R_g, & \text{if arrived goal} \\ 0, & \text{otherwise} \end{cases} \\
r_p &= \begin{cases} ad, & d > 0 \\ -ad, & \text{otherwise} \end{cases} \\
r_c &= \begin{cases} R_c, & \text{if collision} \\ 0, & \text{otherwise} \end{cases} \\
r_a &= \begin{cases} -e^{max(s_{laser})*\lambda}, & \text{any}(s_{laser}) > 0 \\ 0, & \text{otherwise} \end{cases}
\end{aligned}
\tag{5.2}
$$

The total reward, r, was the sum of a series of rewards, r_g, r_p, r_c and r_a. In r_g, the goal arrival reward, R_g, is given when the robot arrived goal. In r_p, ad or $-ad$ is given depending on whether the robot is approaching to or moving away from the goal (specified to be equal to a step length, d, multiplied by a tunable factor, a, which was set empirically to 4.0), In r_c, the collision penalty, R_c is given when the robot collides to the obstacle. In r_a, the approaching penalty is given when any of the laser sensor value, s_{laser}, is higher than zero, i.e., the laser finder detects obstacles. λ is a parameter to set the intensity of approaching penalty. The parameter values used in the experiments are provided in Table 5.1.

Neural Network Architecture

In DDPG, each robot requires two sets of neural networks: i) actor network and ii) critic network. The actor network consisted of 3 fully connected layers with 512, 512 parameters followed by a rectified linear activation function (ReLU) nonlinearities between the input and output layers. Outputs were connected to sigmoid and tanh activation functions to limit the range to ($0 \leq v \leq 1$

Table 5.1. Parameters and values for DRL training strategies

Parameters	Values
Total number of episodes, M	120
Training steps per episode, T	1024
Total number of robots, N	4
Discount factor, γ	0.99
Soft weight update factor, τ	0.5
Goal reward, R_g	100.0
Collision penalty, R_c	-100.0
Progress reward factor, a	4.0
Approaching penalty parameter, λ	$\log 2$

and $-1 \leq \omega \leq 1$). The range of v and w is further processed to limit the velocity applying the motion constraints of the robot as described with action space. The critic network, consisted of input and output layers, which were observations and state-action values, and 2 fully connected layers with the same number of parameters as the actor network. Unlike the actor network, after the first layer, actions were concatenated and fed into the second layer.

5.3.2 Metrics

Training Performance Metrics

To evaluate the training performance of each training strategy, we measured three training performance metrics are below.

- Average reward, r_{avg}, is the averaged value of the re- ward obtained for one episode over four agents in the experiment.
- Catastrophic interference, N_{ci}, is the number of the events when the average reward changes by more than 50% of the the range between the maximum and minimum values of average reward over the training.
- Failed agent, N_{fa}, represents the average number of failed agents during one training instance. The training agents are regarded as failed agents when the difference between the average reward at the beginning and the end during training is within 1, and the difference between the maximum and minimum value is under 1.5.

Evaluation Performance Metrics

To assess the performance of trained models with IDDPG, SNDDPG, SEDDPG and FLDDPG in the evaluation stage, we defined two performance metrics: i) mission success rate and ii) mission completion time.

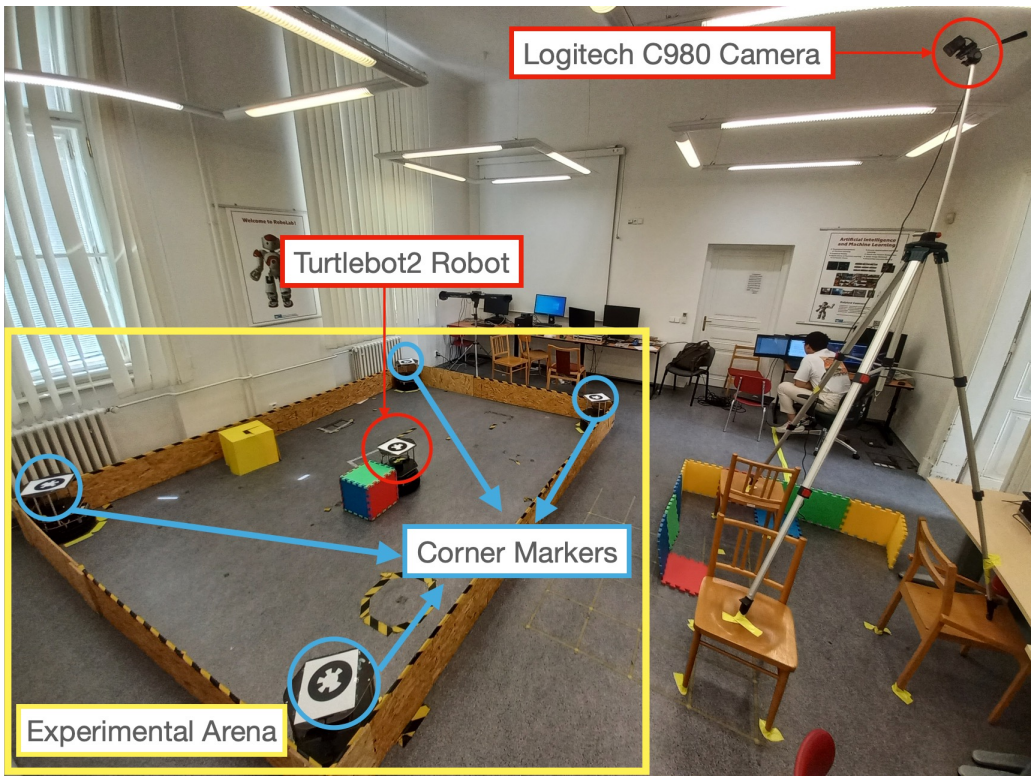

Figure 5.3. A snapshot of the experimental arena. The turtlebot2 mobile robot performs navigation and collision avoidance in the rectangular arena. The overhead camera is used for real-time localisation of the robot in the two-dimensional arena coordinate frame, which is determined by the four corner markers.

- Success rate, ρ_s, is the rate of the successful episode from all the episodes without collision within one episode.

- Completion time, t_{comp}, represents the average time taken for all the robots to reach the targets without collision and within the time limit.

Success rate shows the robustness of the controller and completion time represents the optimisation performance of each training strategy. In the simulated evaluation experiment, the success rate and completion time were calculated by averaging performance of the four agents in the environment over 20 runs with four trained models for each agent during the training stage.

5.3.3 Real-Robot Experiment

Experiments including real-robots were performed in $4 \times 3\ m$ arena with one robot. The arena had several obstacles similar to the ones in the simulation, such as boxes. The camera is mounted outside the arena, and the main PC is connected to the camera for real-time localisation. The snapshot of the experimental arena is shown in Fig. 5.3.

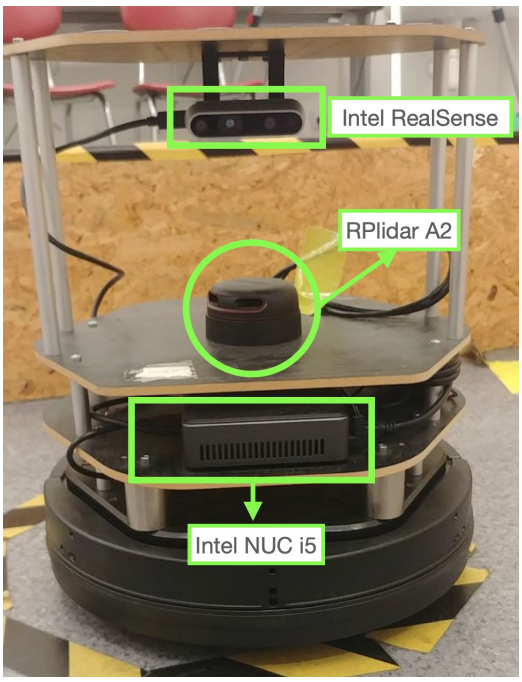

Figure 5.4. Single Turtlebot2 from top to bottom with a camera, RPlidar, Intel NUC and base platform.

Hardware Specification

Turtlebot2 mobile robot was used as a robotic platform, depicted in Fig. 5.4. The Turtlebot2 is a tower-shape differential drive mobile robot with a diameter of 38 cm and height of 60 cm. The robots use a differential drive with a coaster wheel and can turn on the spot with a maximum rotational speed of 3 rad/s and translational speed of 0.65 m/s. Robots were equipped with an RPlidar A2 which is planar LiDAR mounted parallel to the ground with a 10 m maximum range (covering the entire arena) with 720 beams covering 360°at a rate of 10 Hz. The minimum range is 0.2 m, within the robot's footprint. Robots are also equipped with Intel RealSense camera. However, the camera is not used in the experiments. For computation, the robot is mounted with Intel NUC i5 of 8th generation equipped with Ubuntu 20.04 and ROS Noetic.

System Specification

Master PC with AMD Threadripper 3960X, Nvidia RTX 3090 and 64GB RAM was used as a centralised hub to which all the robots are connected via Wi-Fi. The master PC was used for inter-robot communication and training neural network models. Additionally, this PC provides a position for all the robots running relevant ROS nodes for the tracking system.

Tracking System

To provide each robot with its position and orientation in the arena, the Logitech C980 camera was used with 1080p @30 Hz resolution to detect and localise fiducial markers placed on top of the robot. The WhyCode fiducial markers [186], [187] were used for the external localisation of

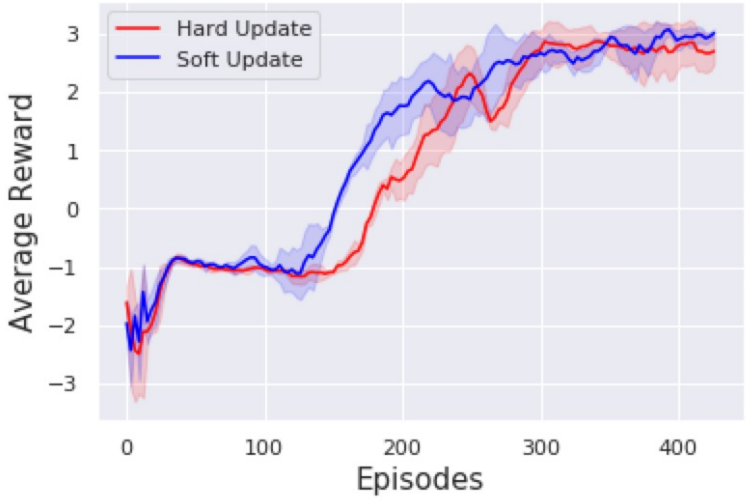

Figure 5.5. Average reward per episode of two different weight averaging methods: (red line) hard update and (blue line) soft update.

the robots. The WhyCode is a low-cost vision-based localisation system capable of real-time pose estimation of extensive number of black-and-white circular fiducial markers. Unlike its predecessor, WhyCon [188], WhyCode is capable of unique identification and full 6 degree-of-freedom (DOF) pose estimation with high precision using only an off-the-shelf web camera. All this was computed and then transferred to each robot from the master PC via Wi-Fi. The 6 DOF estimated information was used as state information for the trained neural network to infer desirable actions for the task.

5.4 Results & Discussion

5.4.1 Results of FLDDPG Design Experiments

In this section, the results from experiments to evaluate the two weight averaging methods: i) hard update and ii) soft update were reported and discussed. This experiment uses the same experimental scenario of collision avoidance and navigation. However, it is conducted with a smaller number of time steps per episode and a different reward design than the main experiment in a fast manner to evaluate the performance of hard update and soft update only. Fig. 5.5 shows the results from the experiments testing the training performance of FLDDPG with hard update and soft update.

In Fig. 5.5, it is illustrated that the average reward started to increase earlier and converged faster with the soft update method than the hard update method. It is found that the performance of each robot temporarily decreased after the weight update with the averaged model using the hard update method. Since the averaged model generalises individual local models, its performance in each local environment varies. The result shows that the soft update method prevented the adverse conversion from the local model to the averaged model, resulting in faster training by 18% reducing training time compared to the hard weight update.

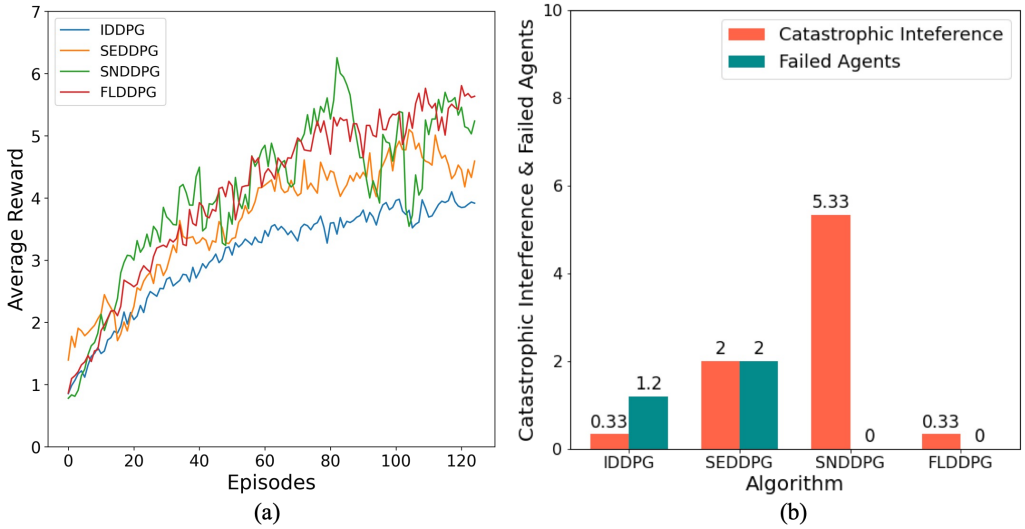

Figure 5.6. The training results of four different algorithms: IDDPG, SNDDPG, SEDDPG and FLDDPG. (a) Average reward per episode over training. (b) Average number of catastrophic interference and failed agents per training.

5.4.2 Training Performance of the Four Strategies

In this section, the outcomes of experiments evaluating four different training strategies for teaching robots navigation and collision avoidance are presented. The experiments were conducted in an environment depicted in Figure 5.2, with each strategy operating under limitations on total data transfer. The results are divided into two main figures: Figure 5.6 (a) focuses on average rewards for each strategy, whereas Figure 5.6 (b) discusses the average occurrences of catastrophic interferences and failed agents.

Average Rewards

Figure 5.6 (a) reveals that FLDDPG yields the highest average rewards by the end of training, followed by SNDDPG, SEDDPG, and lastly, IDDPG. This ranking suggests that FLDDPG offers the best overall performance. An interesting observation is the larger amplitude of reward fluctuations for SNDDPG compared to other strategies, an issue that will be quantitatively elaborated later.

Catastrophic Interferences and Failed Agents

Figure 5.6 (b) shows the occurrences of catastrophic interferences and failed agents per training instance. IDDPG and FLDDPG exhibit the lowest rates of catastrophic interference, averaging 0.33 per training instance. In contrast, SNDDPG experiences the highest rate at 5.33, which helps to explain its aforementioned reward fluctuations. SEDDPG has a moderate rate of 2.

- For IDDPG, the low rate of catastrophic interferences can be attributed to its independent training approach, where agents do not influence each other. However, this also results in a poorer average reward and a higher rate of agent failures.

- SEDDPG suffers from an elevated rate of catastrophic interferences due to the long intervals between updates of the experience replay buffer, disrupting individual learning.
- The high level of catastrophic interference for SNDDPG is due to poor data collection between updates, leading to a vicious cycle of non-improvement.
- FLDDPG and SNDDPG benefit from shared experiences and network updates, thus avoiding agent failures and maintaining a more robust average reward.

In summary, these findings provide valuable insights into the merits and limitations of each training strategy, laying the groundwork for future research in this area.

5.4.3 Simulation Evaluation Results

In this section, the trained models using four different strategies (IDDPG, SEDDPG, SNDDPG, and FLDDPG) are evaluated within the same simulated environment in which they were trained. Each model was run 20 times, with outcomes averaged over these runs and four agents.

Success Rates and Completion Time

Figure 5.7 presents the evaluation results, showcasing success rates in Figure 5.7 (a) and average completion times in Figure 5.7 (b). SEDDPG shows the poorest performance with a 5% success rate, likely owing to the high number of catastrophic interferences and failed agents experienced during training, as evidenced in Figure 5.6. IDDPG and SNDDPG recorded success rates of 11% and 10%, respectively. In contrast, FLDDPG far outperforms the others with a success rate of 26%.

- **SEDDPG:** The low success rate can be attributed to the high occurrences of catastrophic interference and failed agents during the training phase.
- **IDDPG and SNDDPG:** While IDDPG had more failed agents and SNDDPG had more catastrophic interferences, both ended up with similar, low success rates.
- **FLDDPG:** The high success rate is most likely due to the significantly low number of catastrophic interferences (0.33) and the complete absence of failed agents during training.

Comparative Performance

FLDDPG not only had the highest success rate but also the shortest average completion time, approximately 60% that of its nearest competitor, SEDDPG. These results manifest the robustness and generalisation capabilities of FLDDPG, marking a 2.36-fold improvement over the second-best method (IDDPG) in terms of success rate.

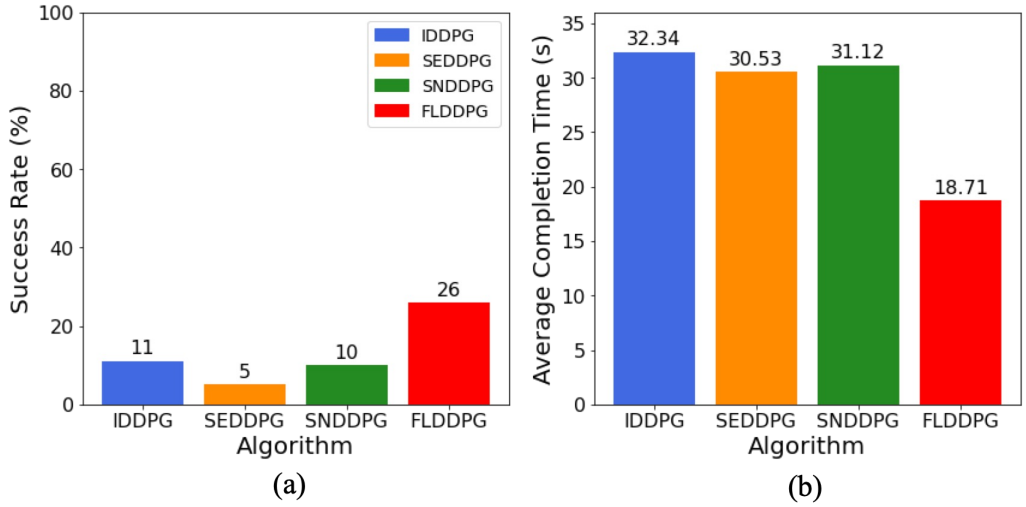

Figure 5.7. The evaluation experiment results with the four different DRL training strategies. (a) and (b) show success rate and average completion time of 4 agents over 20 runs of the four algorithms respectively.

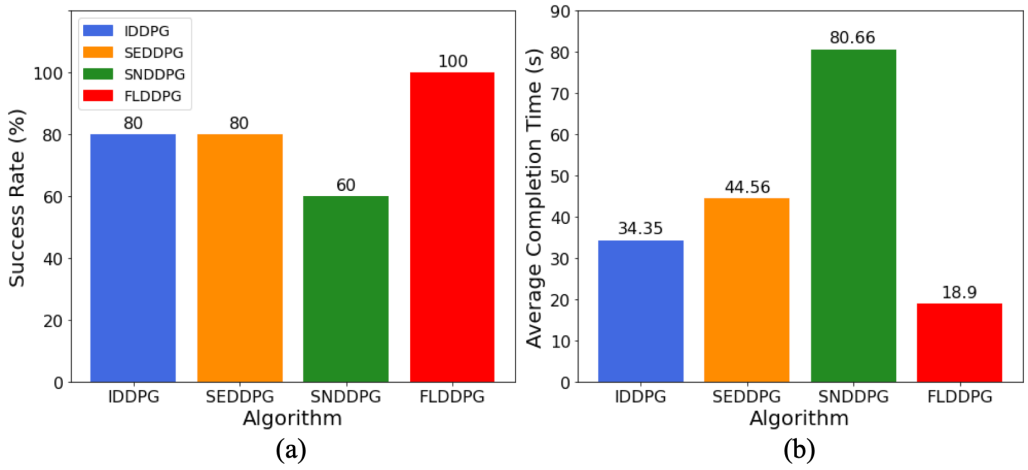

Figure 5.8. The real robot experiment results with the four different DRL training strategies. (a) and (b) show success rate and average completion time over 5 runs of the four algorithms respectively using the most successful model.

In conclusion, the evaluation clearly demonstrates the superiority of FLDDPG in both success rate and average completion time, reinforcing its robustness and adaptability compared to other strategies.

5.4.4 Real-Robot Experiment Results

For the real-robot experiments, the most successfully trained model was selected from each strategy, adhering to identical communication period constraints. Figures 5.8 (a) and (b) depict the respective success rates and average completion times for these strategies.

- **Success Rate:** FLDDPG showcased the highest success rate among all the strategies.
- **Average Completion Time:** In addition to its leading success rate, FLDDPG also recorded the shortest average completion time.

The results from real-robot experiments corroborate the findings from the simulated evaluation experiments, affirming the robustness and generalisation capabilities of FLDDPG. Specifically, the FLDDPG model performed exceptionally well when transferred to a real-robot platform, both in terms of success rate and completion time.

5.4.5 Discussion

The findings, collected from three distinct points–i) training performance, ii) simulation-based evaluation, and iii) real-robot evaluation–underline several pivotal contributions.

Contributions of FLDDPG

- **Prevention of Individual Training Failures:** The FLDDPG strategy notably reduced the failure rate of individual training in collective learning scenarios. This is achieved by periodically aggregating individual neural networks, showing marked improvement over independent learning scenarios, such as IDDPG.
- **Robustness and Generalisation:** FLDDPG demonstrates superior robustness and generalisation abilities, even under communication bandwidth limitations. These qualities are evident both in simulation evaluations and real-robot settings.

Progressing Towards Decentralisation with FLDDPG

FLDDPG presents a marked reduction in centralisation compared to traditional MARL-based strategies. By eliminating the need to exchange training data, FLDDPG significantly lowers the volume of data needed for model training within the same timeframe. This reduction enhances the stability of communications between agents and the central server, especially in scenarios with limited bandwidth where traditional centralised strategies might struggle.

While FLDDPG substantially reduces the drawbacks associated with centralised MARL-based training strategies, it does not eliminate centralisation completely due to the reliance on a central server.

However, FLDDPG introduces the potential for achieving full decentralisation by enabling parameter sharing and collective learning through peer-to-peer communication, bypassing the need for a central server. Although full decentralisation has not yet been realised, FLDDPG is a pivotal step towards this goal, signalling progress in the field. Therefore, it can be stated that FLDDPG is located one step behind a fully decentralised method.

Implications for Real-world Scenarios

The experimental results point to the potential applicability of FLDDPG in autonomous swarm robotic systems operating in bandwidth-restricted, uncertain environments. Instances of such environments include underground mines and tunnels, commonly encountered in inspection or search-and-rescue missions. Limited communication bandwidth is indeed one of the most formidable challenges reported in the Defense Advanced Research Projects Agency (DARPA) subterranean challenge [189], [190], as well as in underwater robotic swarms [191].

Hence, the adoption of FLDDPG in analogous settings could significantly enhance the robustness and efficacy of deployed multi- and swarm-robotic systems over the conventional approach of MARL framework.

5.5 Summary

In this chapter, the introduction of FLDDPG, a novel Federated Learning-based DRL training strategy, addresses the centralisation issues associated with MARL frameworks. Experimental findings demonstrate that FLDDPG not only outperforms existing baseline methods but also does so while substantially reducing communication overhead. These outcomes suggest that FLDDPG offers a compelling advantage for autonomous swarm robotic systems operating in communication-limited environments such as underwater or underground spaces.

From the perspective of automatic controller design in swarm robotic systems, FLDDPG represents a pioneering approach, effectively alleviating the issue of centralisation in MARL framework, achieving **Aim 3 & Objective 3** of the thesis. Future work is envisioned to further adapt and expand FL-based DRL training strategies, targeting heterogeneous swarm robotic systems in natural and dynamic environments [190], [192]–[194]. In addition, the development of a fully decentralised DRL training strategy, facilitated by peer-to-peer communication, stands as a crucial next step and a significant challenge for DRL-driven automatic controller design.

Chapter 6

Conclusion

The thesis contributes to enhance and streamline the swarm robotics domain, focusing on the development and advancement of DRL-driven automatic controller designs, a critical component in advancing the functionality of swarm robotic systems for real-world scenarios. This conclusion synthesises the key findings and insights acquired through the consecutive chapters, highlighting the manner in which the predefined aims and objectives were successfully met. After ensuring the fulfilment of aims and objectives, key open challenges that are not addressed in this thesis are introduced, delimiting the realm of this thesis and calling other researchers to join and explore to advance swarm robotic systems. Finally, the meaning of this thesis is highlighted in a broader real-life context as a concluding message.

Aim 1 sought to investigate the influence of real-world conditions on the collective behaviour of swarm robotic systems, a preliminary scrutiny to foster systems that are more adaptive in response to real-world dynamics. To achieve **Objective 1**, the research study incorporated these real-world conditions into a pheromone-based swarm robotic system, scrutinising the effects on aggregation behaviour of a swarm of mobile robots. **Chapter 3** demonstrated that the emulated realistic effects have a substantial influence on the aggregation behaviours of pheromone-based swarm robotic systems. The complexity and unpredictability of real-world effects elucidated the limitations of manual controller designs, accentuating the imperative need for automatic controller designs capable of adeptly adapting to real-world conditions, thereby aligning perfectly with the target set by **Objective 1**.

In addressing **Aim 2**, the focus shifted to the development of a DRL-driven automatic controller design that illustrated adaptivity and superiority over traditional methods. **Objective 2** aimed at crafting DRL-driven automatic controller design on a swarm robotic scenario that harnesses pheromone-based communication and evaluating its efficiency. **Chapter 4** conducted a comparative analysis, demonstrating the proposed DRL-driven automatic controller that outclasses traditional centralised and manual controller designs by adeptly addressing complexity and eliminating the necessity for human experts in the design process, thus successfully achieving the **Objective 2**.

Aim 3 steered the research towards devising a strategy to alleviate the centralisation issues found in traditional MARL approaches. **Objective 3** mapped out a strategy to utilise a FL-based DRL training strategy, which was further evaluated through real-robot experiments. The findings in **Chapter 5** were seminal, illustrating a significant reduction in the degree of centralisation, notably

surpassing the performance efficacy of baseline methods, and effectively mitigating the reality-gap issue prevalent in automatic controller designs. These findings stressed the potential of the FL-based DRL strategy to make a breakthrough in automatic controller design for swarm robotic domain, thereby fulfilling **Objective 3** and showcasing a promising pathway towards the resolution of grand challenges in automatic controller design in swarm robotics.

There are several key open challenges remain to be addressed to further sculpt the pathway in this domain, encouraging other researchers to join and explore. Here below, six open challenges with suggestion for future research direction are introduced.

- **Expansion to Address Collective Behaviours:** The current DRL-driven automatic controller design predominantly caters to low-level behaviours. There is an imperative need to conceptualise and implement a broader spectrum of swarm robotic behaviours through this approach, to underscore its potential in nurturing emergent behaviours. Enhanced demonstrations, illustrating a diverse type of collective behaviours, could substantially support the significance of DRL-driven automatic controller design in mitigating design challenges under realistic scenarios. For example, if DRL-driven automatic controller design is applied into designing aggregation behaviour for swarm robotic systems, more adaptive aggregation behaviours can be obtained in a way that being more robust to the effect of realistic conditions than manual controller design utilised in Chapter 3. Moreover, more complex types of collective behaviour can be designed using DRL-driven automatic controller design. For instance, long-distance foraging, main strategy for maintaining a colony of ants [28] can be designed to be utilised as a strategy of collective transportation of swarm systems. As mentioned in Chapter 4, the need for strategy to coordinate large number of autonomous vehicles in traffic is rapidly rising. Therefore, investigating designing long-distance foraging using DRL-driven automatic controller design could be a meaningful step forward. Likewise, diverse types of collective behaviours, e.g. collective transport [157], flocking [60] can be designed and investigated its feasibility with DRL-driven automatic controller design for a wide range of real-world applications.

- **Comprehensive Comparative Analysis of Automatic Controller Design:** A critical necessity exists for a thorough analysis that contrasts DRL-driven automatic controller designs with other automatic controller design approaches including robot evolutionary swarm robotics and automatic modular design. Initiating a direct comparison between these methodologies could foster a deeper understanding of each technique's characteristics, potentially unveiling opportunities to develop a hybrid approach that amalgamates the strengths of both while mitigating their respective drawbacks. Clear understanding on methodologies of automatic controller design helps researchers to adopt an appropriate method for developing target application more easily. Furthermore, the potential development of hybrid design that can serve as a general design methodology for automatic controller design. Lacking of general design methodology is highlighted as a major limitation of automatic controller design method [7]. With a general design methodology, automatic controller design, swarm robotics in a broader context can be more rapidly advanced as the effort of researchers can converge into one direction. Conducting a comprehensive comparative analysis of automatic

controller design is a necessary step forward towards the overall development for swarm robotic systems under realistic environments.

- **Showcasing Online Design Processes:** The potential of DRL-driven automatic controller design in alleviating the existing issues with current online methods must be effectively showcased. This encompasses addressing notable challenges such as prolonged exploration times and the potential for physical damage to robots during the online design process. While this is theoretically plausible, there is a lack of empirical evidence to substantiate this claim. Researchers have been investigating the potential of evolutionary robotics method with its sub-field called Embodied Evolution [55], demonstrating successful stream of research studies implementing collective behaviours while a robot swarm operates in its mission. Likewise, DRL-driven automatic controller design methods can be investigated in the scenarios where the robots design their controller on-line. In contrast to the methods for embodied evolution, DRL-driven automatic controller design can leverage several techniques that facilitate on-line design processes to avoid violation of constraints of the mission, e.g. maximum duration of mission completion or safety constraints. By capitalising auxiliary techniques, such as transfer learning [96], on-line design process can significantly reduce exploration space, reducing further design time and possibilities taking dangerous actions while guaranteeing minimum level of performance.

- **Full Decentralisation of FL-based DRL Training Strategies:** Despite the advancements, the current FL-based DRL training strategies have not entirely eliminated the issue of centralisation. The proposal and experimentation with entirely decentralised methodologies in the swarm robotic context are of paramount importance, where the plausibility is already showcased in other domains including medical diagnosis [127]. The advent of such approaches could encourage a wider application of DRL-driven automatic controller design, facilitating online and decentralised design processes, which are quintessential for real-world applications. However, compared to the existing decentralised application in other domains, designing DRL-driven automatic controller requires more sophisticated requirements. For example, in DRL, states and actions of an agent sequentially impact. It means that unsuccessful parameter sharing that brings a negative impact could deteriorate controller performance in a global scale like chain reaction. Therefore, systematic investigation with basic design of full decentralisation starting from the current FL-based DRL training strategy, proposed in Chapter 5, is recommended. With the basic design of fully decentralised DRL training strategies, further development can be done easily with the advancement by addressing other key open challenges, which are listed in this section.

- **Advanced Methods for Parameter Sharing:** In Chapter 5, the basic type of parameter sharing method, weight averaging [123] with soft average method, has been adopted and utilised, yet providing performance advantage compared to the traditional MARL methods. Since the research study introduced in Chapter 5 is the nascent application of FL into a swarm robotic context, the most basic form of parameter sharing method is used. This means there is a large room for advancement of the FL-based DRL training strategies by devising diverse parameter sharing methods. For example, the advanced parameter sharing methods, such as FedProx [195] and SCAFFOLD [196]

used in other domains can be used to improve FL-based DRL training strategies. Not only applying existing advanced algorithms for FL in other domains, but also developing specific parameter sharing methods for swarm robotic scenarios could be another interesting research direction. For example, game theory [197], [198] could be used as an effective method to determine how parameter sharing is conducted among multiple individuals in a group to obtain a maximum global benefit while keeping individual benefit at a desired level. Moreover, leveraging graph neural networks (GNNs) to determine parameter sharing rules could be a promising research field as GNNs can be utilised to model a complex graph structure between agents, useful for designing parameter sharing rules under complex scenarios.

- **Deployment for Heterogeneous Swarm Robotic Systems:** One of the advantages of FL-based DRL training strategy is that it is more generalisable than the traditional MARL based DRL training strategies. In Chapter 5, having a high generalisation capability is a great advantage to overcome reality-gap issue, which is one of the two biggest challenges in automatic controller design. This generalisation capability can be also highlighted when heterogeneous robots are deployed in one group. Heterogeneous robots require different state, action, kinematics and dynamics profile, which discourage conventional MARL-based DRL training strategies. Still, desigining a generalisable controller design among heterogeneous robotic systems [199] is dramatically intricate challenges, even more understudied with DRL. Nevertheless, it is worth to investigate the potential of FL-based DRL training strategies to obtain generalisable controllers for heterogeneous swarm robotic systems. Together with advanced parameter sharing methods, developing methodologies to deal with heterogeneity of robots is crucial research direction as heterogeneous swarm robotic systems will be used for real-world applications by a high chance [189].

In retrospect, this thesis not only achieved its predefined aims and objectives but also significantly contributed to the pavement of new research directions for advancement of automatic controller design, which deemed as a vital topic for future swarm robotic systems. The three main chapters methodically justified the need for a decentralised approach in DRL-driven automatic controller design and showcased its prototype approach using FL. This heralds a promising future, moving towards the development of more proficient and adaptive swarm robotic systems capable of tackling complex real-world challenges, a critical step forward in the ever-evolving landscape of swarm robotics. The findings highlight the potential avenues for further research, albeit remaining open challenges, fostering a deeper understanding and facilitating the creation of more proficient and versatile swarm robotic systems equipped with automatic controller design in the forthcoming future. By addressing the open challenges, crucial developments will be made towards designing swarm robotic systems and infrastructures, practically solving unprecedented diverse physical challenges in the society, such as extraterrestrial exploration and rescue missions in the extreme environments. Since it is still an early stage for the field of swarm robotics, more for its sub-field automatic controller design, exciting journey is waiting for researchers to contribute developing a new type of robotic systems leveraging collective principles shown in thriving social animals in nature, addressing the most intricate challenges what the current solutions are not capable of.

References

[1] E. Şahin, "Swarm robotics: From sources of inspiration to domains of application," in *Swarm Robotics*, E. Şahin and W. M. Spears, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 10–20 (cited on pp. 17, 21).

[2] "Evolving self-organizing behaviors for a swarm-bot," *Autonomous Robots*, vol. 17, no. 2-3, pp. 223–245, 2004. DOI: 10.1023/B:AURO.0000033973.24945.f3 (cited on pp. 17, 30).

[3] M. Brambilla, E. Ferrante, M. Birattari, and M. Dorigo, "Swarm robotics: A review from the swarm engineering perspective," *Swarm Intelligence*, vol. 7, no. 1, pp. 1–41, 2013 (cited on pp. 17, 18, 21, 27, 90).

[4] M. Dorigo, G. Theraulaz, and V. Trianni, "Reflections on the future of swarm robotics," *Science robotics*, vol. 5, no. 49, Dec. 2020. DOI: 10.1126/scirobotics.abe4385 (cited on pp. 17, 20, 22).

[5] M. Birattari, A. Ligot, D. Bozhinoski, *et al.*, "Automatic off-line design of robot swarms: A manifesto," *Frontiers in Robotics and AI*, vol. 6, p. 59, 2019, ISSN: 2296-9144. DOI: 10.3389/frobt.2019.00059 (cited on pp. 18, 28, 31, 33, 34).

[6] M. Duarte, J. Gomes, V. Costa, *et al.*, "Application of swarm robotics systems to marine environmental monitoring," in *OCEANS 2016-Shanghai*, 2016, pp. 1–8 (cited on pp. 18, 30, 31).

[7] J. Kuckling, "Recent trends in robot learning and evolution for swarm robotics," *Frontiers in Robotics and AI*, vol. 10, 2023, ISSN: 2296-9144. DOI: 10.3389/frobt.2023.1134841. [Online]. Available: https://www.frontiersin.org/articles/10.3389/frobt.2023.1134841 (cited on pp. 18, 28, 30–33, 35, 111).

[8] H. Hamann, *Swarm Robotics: A Formal Approach*. Jan. 2018, ISBN: 978-3-319-74526-8. DOI: 10.1007/978-3-319-74528-2 (cited on p. 21).

[9] S. Garnier, F. Tache, M. Combe, A. Grimal, and G. Theraulaz, "Alice in pheromone land: An experimental setup for the study of ant-like robots," in *2007 IEEE Swarm Intelligence Symposium*, Apr. 2007, pp. 37–44 (cited on pp. 21, 26).

[10] E. Şahin, S. Girgin, L. Bayindir, and A. E. Turgut, "Swarm robotics," in *Swarm Intelligence: Introduction and Applications*, C. Blum and D. Merkle, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 87–100 (cited on pp. 21, 27).

[11] M. Dorigo, G. Theraulaz, and V. Trianni, "Swarm robotics: Past, present, and future [point of view]," *Proceedings of the IEEE*, vol. 109, no. 7, pp. 1152–1165, 2021. DOI: 10.1109/JPROC.2021.3072740 (cited on p. 22).

[12] P. Karlson and M. Lüscher, "'pheromones': A new term for a class of biologically active substances," *Nature*, vol. 183, pp. 55–56, 1959 (cited on p. 23).

[13] D. Baracchi, J. M. Devaud, P. D'Ettorre, and M. Giurfa, "Pheromones modulate reward responsiveness and non-associative learning in honey bees," *Scientific Reports*, vol. 7, no. 1, pp. 1–9, 2017 (cited on p. 23).

[14] C. Hostachy, P. Couzi, G. Portemer, *et al.*, "Exposure to Conspecific and Heterospecific Sex-Pheromones Modulates Gustatory Habituation in the Moth Agrotis ipsilon," *Frontiers in Physiology*, vol. 10, no. December, pp. 1–8, 2019 (cited on p. 23).

[15] J. Chalissery, A. Renyard, R. Gries, D. Hoefele, S. Alamsetti, and G. Gries, "Ants sense, and follow, trail pheromones of ant community members," *Insects*, vol. 10, p. 383, Nov. 2019 (cited on p. 23).

[16] O. O. Okosun, A. A. Yusuf, R. M. Crewe, and C. W. Pirk, "Tergal gland components of reproductively dominant honey bee workers have both primer and releaser effects on subordinate workers," *Apidologie*, vol. 50, no. 2, pp. 173–182, 2019 (cited on p. 23).

[17] K. Tateishi, Y. Nishimura, M. Sakuma, F. Yokohari, and H. Watanabe, "Sensory neurons that respond to sex and aggregation pheromones in the nymphal cockroach," *Scientific Reports*, vol. 10, no. 1, p. 1995, 2020 (cited on p. 23).

[18] S. Fields, "Phermone response in yeast," *Trends in Biochemical Sciences*, vol. 15, no. 7, pp. 270–273, 1990 (cited on p. 23).

[19] P. Brennan and F. Zufall, "Pheromonal communication in vertebrates," *Nature*, vol. 444, pp. 308–315, 2006 (cited on p. 23).

[20] B. Schaal, G. Coureaud, D. Langlois, and et al, "Chemical and behavioural characterization of the rabbit mammary pheromone," *Nature*, vol. 424, pp. 68–72, 2003 (cited on p. 23).

[21] S. Mutic, Y. F. Brünner, R. Rodriguez-Raecke, M. Wiesmann, and J. Freiherr, "Chemosensory danger detection in the human brain: Body odor communicating aggression modulates limbic system activation," *Neuropsychologia*, vol. 99, no. September 2016, pp. 187–198, 2017 (cited on p. 23).

[22] K. Stern and M. McClintock, "Regulation of ovulation by human pheromones," *Nature*, vol. 392, pp. 177–179, 1998 (cited on p. 23).

[23] G. Preti, C. J. Wysocki, K. T. Barnhart, S. J. Sondheimer, and J. J. Leyden, "Male Axillary Extracts Contain Pheromones that Affect Pulsatile Secretion of Luteinizing Hormone and Mood in Women Recipients1," *Biology of Reproduction*, vol. 68, no. 6, pp. 2107–2113, 2003 (cited on p. 23).

[24] C. Wedekind and S. Furi, "Body odour preferences in men and women: Do they aim for specific mhc combinations or simply heterozygosity," *Proceedings: Biological Sciences*, vol. 264, no. 1387, pp. 1471–1479, 1997 (cited on p. 23).

[25] B. Hölldobler and E. O. Wilson, *The Ants*. Belknap Press of Harvard University Press, 1990 (cited on p. 23).

[26] A. Denny, J. Wright, and B. Grief, "Foraging efficiency in the wood ant, formica rufa: Is time of the essence in trail following?" *Animal Behaviour*, vol. 62, pp. 139–146, 2001 (cited on pp. 23, 24).

[27] S. Goss, S. Aron, J.-L. Deneubourg, and J. Pasteels, "Self-organized shortcuts in the argentine ant," *Naturwissenschaften*, vol. 76, pp. 579–581, Dec. 1989 (cited on p. 23).

[28] D. E. Jackson and F. L. Ratnieks, "Communication in ants," *Current Biology*, vol. 16, no. 15, R570–R574, 2006 (cited on pp. 23, 111).

[29] T. Eltz, "Tracing pollinator footprints on natural flowers," *J Chem Ecol*, vol. 32, pp. 907–915, 2006 (cited on p. 23).

[30] L. Holman, "Queen pheromones and reproductive division of labor: a meta-analysis," *Behavioral Ecology*, vol. 29, no. 6, pp. 1199–1209, Apr. 2018 (cited on p. 23).

[31] S. A. Princen, A. Van Oystaeyen, C. Petit, J. S. van Zweden, and T. Wenseleers, "Cross-activity of honeybee queen mandibular pheromone in bumblebees provides evidence for sensory exploitation," *Behavioral Ecology*, pp. 1–8, 2019 (cited on p. 23).

[32] L. Holman, C. G. Jørgensen, J. Nielsen, and P. D'Ettorre, "Identification of an ant queen pheromone regulating worker sterility," *Proceedings of the Royal Society B: Biological Sciences*, vol. 277, no. 1701, pp. 3793–3800, 2010 (cited on p. 23).

[33] N. Steinhauer, K. Kulhanek, K. Antúnez, *et al.*, "Drivers of colony losses," *Current Opinion in Insect Science*, vol. 26, pp. 142–148, 2018 (cited on p. 24).

[34] A. A. Khaliq and A. Saffiotti, "Stigmergy at work: Planning and navigation for a service robot on an rfid floor," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, May 2015, pp. 1085–1092 (cited on pp. 24, 25).

[35] D. Jackson, S. Martin, M. Holcombe, and F. Ratnieks, "Longevity and detection of persistent foraging trails in pharaoh's ants, monomorium pharaonis (l.)," *Animal Behaviour*, vol. 71, pp. 351–359, Feb. 2006 (cited on p. 24).

[36] D. E. Jackson, M. Bicak, and M. Holcombe, "Decentralized communication, trail connectivity and emergent benefits of ant pheromone trail networks," *Memetic Computing*, vol. 3, no. 1, pp. 25–32, 2011 (cited on p. 24).

[37] R. A. Russell, "Ant trails - an example for robots to follow?" In *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)*, vol. 4, May 1999, 2698–2703 vol.4 (cited on p. 24).

[38] A. Purnamadjaja and R. Russell, "Bi-directional pheromone communication between robots," *Robotica*, vol. 28, pp. 69–79, Jan. 2010 (cited on p. 24).

[39] R. Fujisawa, S. Dobata, K. Sugawara, and F. Matsuno, "Designing pheromone communication in swarm robotics: Group foraging behavior mediated by chemical substance," *Swarm Intelligence*, vol. 8, no. 3, pp. 227–246, 2014 (cited on p. 24).

[40] Herianto and D. Kurabayashi, "Realization of an artificial pheromone system in random data carriers using rfid tags for autonomous navigation," in *2009 IEEE International Conference on Robotics and Automation*, May 2009, pp. 2288–2293 (cited on p. 25).

[41] A. L. Alfeo, E. C. Ferrer, Y. L. Carrillo, *et al.*, "Urban swarms: A new approach for autonomous waste management," *IEEE International Conference on Robotics and Automation*, pp. 4233–4240, 2019 (cited on pp. 25, 91).

[42] A. Campo, A. Gutiérrez, S. Nouyan, *et al.*, "Artificial pheromone for path selection by a foraging swarm of robots," *Biological cybernetics*, vol. 103, pp. 339–52, Nov. 2010 (cited on p. 25).

[43] A. Reina, A. Cope, E. Nikolaidis, J. Marshall, and C. Sabo, "Ark: Augmented reality for kilobots," *IEEE Robotics and Automation Letters*, vol. PP, pp. 1–1, May 2017 (cited on pp. 25, 73).

[44] A. Font Llenas, M. S. Talamali, X. Xu, J. A. Marshall, and A. Reina, "Quality-sensitive foraging by a robot swarm through virtual pheromone trails," in *International Conference on Swarm Intelligence*, Springer, 2018, pp. 135–149 (cited on p. 26).

[45] G. Valentini, A. Antoun, M. Trabattoni, *et al.*, "Kilogrid: A novel experimental environment for the kilobot robot," *Swarm Intelligence*, vol. 12, no. 3, pp. 245–266, Sep. 2018 (cited on pp. 26, 71, 73).

[46] F. Arvin, T. Krajník, A. E. Turgut, and S. Yue, "COSΦ: artificial pheromone system for robotic swarms research," in *IEEE/RSJ international conference on intelligent robots and systems (IROS)*, 2015, pp. 407–412 (cited on pp. 26, 46, 75, 80, 81).

[47] X. Sun, T. Liu, C. Hu, Q. Fu, and S. Yue, "Colcos φ: A multiple pheromone communication system for swarm robotics and social insects research," in *IEEE 4th International Conference on Advanced Robotics and Mechatronics (ICARM)*, Jul. 2019, pp. 59–66 (cited on p. 26).

[48] M. K. Heinrich, M. Wahby, M. Dorigo, H. Hamann, A. Cangelosi, and M. Asada, "Swarm robotics," *Cognitive Robotics*, 2022 (cited on p. 27).

[49] H. Hamann, *Swarm Robotics: A Formal Approach*. Springer International Publishing, 2018, ISBN: 9783319745282 (cited on p. 27).

[50] G. Baldassarre, V. Trianni, M. Bonani, F. Mondada, M. Dorigo, and S. Nolfi, "Self-organized coordinated motion in groups of physically connected robots," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 37, no. 1, pp. 224–239, 2007. DOI: 10.1109/TSMCB.2006.881299 (cited on p. 27).

[51] A. L. Christensen and M. Dorigo, "Evolving an integrated phototaxis and hole-avoidance behavior for a swarm-bot," in *Artificial Life X: Proceedings of the Tenth International Conference on the Simulation and Synthesis of Living Systems. Cambridge: MIT Press. A Bradford Book*, 2006, pp. 248–254 (cited on p. 27).

[52] R. I. Damper, M. Quinn, L. Smith, G. Mayley, and P. Husbands, "Evolving controllers for a homogeneous system of physical robots: Structured cooperation with minimal sensors," *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, vol. 361, no. 1811, pp. 2321–2343, 2003. DOI: 10.1098/rsta.2003.1258. [Online]. Available: https://royalsocietypublishing.org/doi/abs/10.1098/rsta.2003.1258 (cited on pp. 27, 29).

[53] A. F. T. Winfield and M. D. Erbas, "On embodied memetic evolution and the emergence of behavioural traditions in robots," *Memetic Computing*, vol. 3, no. 4, pp. 261–270, Dec. 2011, ISSN: 1865-9292. DOI: 10.1007/s12293-011-0063-x. [Online]. Available: https://doi.org/10.1007/s12293-011-0063-x (cited on p. 27).

[54] M. Birattari, A. Ligot, and G. Francesca, "Automode: A modular approach to the automatic off-line design and fine-tuning of control software for robot swarms," in *Automated Design of Machine Learning and Search Algorithms*, N. Pillay and R. Qu, Eds. Cham: Springer International Publishing, 2021, pp. 73–90, ISBN: 978-3-030-72069-8. DOI: 10.1007/978-3-030-72069-8_5. [Online]. Available: https://doi.org/10.1007/978-3-030-72069-8_5 (cited on pp. 28, 34, 35).

[55] N. Bredeche, E. Haasdijk, and A. Prieto, "Embodied evolution in collective robotics: A review," vol. 5, no. FEB, 2018 (cited on pp. 28, 32, 112).

[56] V. Trianni, *Evolutionary Swarm Robotics: Evolving Self-Organising Behaviours in Groups of Autonomous Robots* (Studies in Computational Intelligence). Springer Berlin Heidelberg, 2008, ISBN: 9783540776123 (cited on p. 29).

[57] S. Nolfi and D. Floreano, *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines* (A Bradford book). MIT Press, 2000 (cited on p. 29).

[58] G. M. Werner, "Evolution of communication in artificial organisms," *Artificial Life*, 1991 (cited on p. 29).

[59] G. M. Werner and M. G. Dyer, "Evolution of herding behavior in artificial animals," *from animals to animats*, vol. 2, pp. 393–399, 1993 (cited on p. 29).

[60] C. W. Reynolds, "Flocks, herds and schools: A distributed behavioral model," in *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, 1987, pp. 25–34 (cited on pp. 29, 111).

[61] C. W. Reynolds, "An evolved, vision-based behavioral model of coordinated group motion," *From animals to animats*, vol. 2, pp. 384–392, 1993 (cited on p. 29).

[62] M. Quinn, "A comparison of approaches to the evolution of homogeneous multi-robot teams," in *Proceedings of the 2001 Congress on evolutionary computation (IEEE Cat. No. 01TH8546)*, IEEE, vol. 1, 2001, pp. 128–135 (cited on p. 29).

[63] G. Caprari, T. Estier, and R. Siegwart, "Fascination of down scaling-alice the sugar cube robot," in *IEEE International Conference on Robotics and Automation (ICRA 2000): Workshop on Mobile Micro-Robots*, 2000 (cited on p. 29).

[64] A. L. Nelson, E. Grant, and T. C. Henderson, "Evolution of neural controllers for competitive game playing with teams of mobile robots," *Robotics and Autonomous Systems*, vol. 46, no. 3, pp. 135–150, 2004 (cited on p. 29).

[65] F. Mondada, G. C. Pettinaro, A. Guignard, *et al.*, "Swarm-Bot: A New Distributed Robotic Concept," *Autonomous Robots*, vol. 17, no. 2, pp. 193–221, 2004 (cited on p. 30).

[66] K. O. Stanley and R. Miikkulainen, "Evolving neural networks through augmenting topologies," *Evolutionary computation*, vol. 10, no. 2, pp. 99–127, 2002 (cited on pp. 30, 32).

[67] K. Hasselmann, A. Ligot, J. Ruddick, and M. Birattari, "Empirical assessment and comparison of neuro-evolutionary methods for the automatic off-line design of robot swarms," *Nature communications*, vol. 12, no. 1, p. 4345, 2021 (cited on pp. 30, 31, 34).

[68] N. Hansen and A. Ostermeier, "Completely derandomized self-adaptation in evolution strategies," *Evolutionary computation*, vol. 9, no. 2, pp. 159–195, 2001 (cited on p. 30).

[69] T. Glasmachers, T. Schaul, S. Yi, D. Wierstra, and J. Schmidhuber, "Exponential natural evolution strategies," in *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, 2010, pp. 393–400 (cited on p. 30).

[70] G. Francesca, M. Brambilla, A. Brutschy, V. Trianni, and M. Birattari, "Automode: A novel approach to the automatic design of control software for robot swarms," *Swarm Intelligence*, vol. 8, pp. 89–112, 2014 (cited on pp. 30, 33, 34).

[71] S. G. Ficici, R. A. Watson, and J. B. Pollack, "Embodied evolution: A response to challenges in evolutionary robotics," in *Proceedings of the eighth European workshop on learning robots*, Citeseer, 1999, pp. 14–22 (cited on pp. 31, 32).

[72] R. A. Watson, S. G. Ficici, and J. B. Pollack, "Embodied evolution: Distributing an evolutionary algorithm in a population of robots," *Robotics and autonomous systems*, vol. 39, no. 1, pp. 1–18, 2002 (cited on pp. 31, 32).

[73] G. Francesca and M. Birattari, "Automatic design of robot swarms: Achievements and challenges," *Frontiers Robotics AI*, vol. 3, no. MAY, pp. 1–9, 2016, ISSN: 22969144. DOI: 10.3389/frobt.2016.00029 (cited on p. 31).

[74] A. Prieto, J. A. Becerra, F. Bellas, and R. J. Duro, "Open-ended evolution as a means to self-organize heterogeneous multi-robot systems in real time," *Robotics and Autonomous Systems*, vol. 58, no. 12, pp. 1282–1291, 2010 (cited on pp. 32, 33).

[75] F. Silva, P. Urbano, L. Correia, and A. L. Christensen, "Odneat: An algorithm for decentralised online evolution of robotic controllers," *Evolutionary Computation*, vol. 23, no. 3, pp. 421–449, 2015 (cited on pp. 32, 33).

[76] K. O. Stanley, B. D. Bryant, and R. Miikkulainen, "Real-time neuroevolution in the nero video game," *IEEE Transactions on Evolutionary Computation*, pp. 653–668, 2005. [Online]. Available: http://nn.cs.utexas.edu/?stanley:ieeetec05 (cited on p. 32).

[77] N. Cambier, D. Albani, V. Frémont, V. Trianni, and E. Ferrante, "Cultural evolution of probabilistic aggregation in synthetic swarms," *Applied Soft Computing*, vol. 113, p. 108 010, 2021 (cited on p. 32).

[78] N. Bredeche and N. Fontbonne, "Social learning in swarm robotics," *Philosophical Transactions of the Royal Society B*, vol. 377, no. 1843, p. 20 200 309, 2022 (cited on p. 32).

[79] R. Bianco and S. Nolfi, "Toward open-ended evolutionary robotics: Evolving elementary robotic units able to self-assemble and self-reproduce," *Connection Science*, vol. 16, no. 4, pp. 227–248, 2004 (cited on p. 33).

[80] N. Bredeche, J.-M. Montanier, W. Liu, and A. F. Winfield, "Environment-driven distributed evolutionary adaptation in a population of autonomous robotic agents," *Mathematical and Computer Modelling of Dynamical Systems*, vol. 18, no. 1, pp. 101–129, 2012 (cited on p. 33).

[81] S. Jones, A. F. Winfield, S. Hauert, and M. Studley, "Onboard evolution of understandable swarm behaviors," *Advanced Intelligent Systems*, vol. 1, no. 6, p. 1 900 031, 2019 (cited on p. 33).

[82] M. Birattari, T. Stützle, L. Paquete, K. Varrentrapp, *et al.*, "A racing algorithm for configuring metaheuristics.," in *Gecco*, Citeseer, vol. 2, 2002 (cited on p. 34).

[83] G. Francesca, M. Brambilla, A. Brutschy, *et al.*, "Automode-chocolate: Automatic design of control software for robot swarms," *Swarm Intelligence*, vol. 9, pp. 125–152, 2015 (cited on p. 35).

[84] P. Balaprakash, M. Birattari, and T. Stützle, "Improvement strategies for the f-race algorithm: Sampling design and iterative refinement," in *Hybrid Metaheuristics: 4th International Workshop, HM 2007, Dortmund, Germany, October 8-9, 2007. Proceedings 4*, Springer, 2007, pp. 108–122 (cited on p. 35).

[85] J. Kuckling, A. Ligot, D. Bozhinoski, and M. Birattari, "Behavior trees as a control architecture in the automatic modular design of robot swarms," in *Swarm Intelligence: 11th International Conference, ANTS 2018, Rome, Italy, October 29–31, 2018, Proceedings 11*, Springer, 2018, pp. 30–43 (cited on p. 35).

[86] M. Salman, A. Ligot, and M. Birattari, "Concurrent design of control software and configuration of hardware for robot swarms under economic constraints," *PeerJ Computer Science*, vol. 5, e221, 2019 (cited on p. 35).

[87] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, Second. The MIT Press, 2018 (cited on pp. 35, 38, 39, 77).

[88] M. J. Matarić, "Reinforcement learning in the multi-robot domain," *Robot colonies*, pp. 73–83, 1997 (cited on p. 35).

[89] M. J. Mataric, "Using communication to reduce locality in distributed multiagent learning," *Journal of experimental & theoretical artificial intelligence*, vol. 10, no. 3, pp. 357–369, 1998 (cited on pp. 35, 36).

[90] Y. F. Chen, M. Liu, M. Everett, and J. P. How, "Decentralized non-communicating multi-agent collision avoidance with deep reinforcement learning," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 285–292, 2017 (cited on pp. 36, 71).

[91] J. Hu, H. Niu, J. Carrasco, B. Lennox, and F. Arvin, "Voronoi-based multi-robot autonomous exploration in unknown environments via deep reinforcement learning," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 12, pp. 14 413–14 423, 2020 (cited on pp. 36, 41, 71).

[92] M. Hüttenrauch, A. Oic, and G. Neumann, "Deep reinforcement learning for swarm systems," *Journal of Machine Learning Research*, vol. 20, pp. 1–31, 2019 (cited on p. 36).

[93] R. Agarwal, D. Schuurmans, and M. Norouzi, "An optimistic perspective on offline reinforcement learning," in *International Conference on Machine Learning*, PMLR, 2020, pp. 104–114 (cited on p. 37).

[94] S. Levine, A. Kumar, G. Tucker, and J. Fu, "Offline reinforcement learning: Tutorial, review, and perspectives on open problems," *arXiv preprint arXiv:2005.01643*, 2020 (cited on p. 37).

[95] F. Zhuang, Z. Qi, K. Duan, *et al.*, "A comprehensive survey on transfer learning," *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, 2020 (cited on p. 37).

[96] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2009 (cited on pp. 37, 112).

[97] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, and J. Peters. Now Publishers, Inc., 2018 (cited on p. 37).

[98] W. Li, M. Gauci, and R. Groß, "Turing learning: A metric-free approach to inferring behavior and its application to swarms," *Swarm Intelligence*, vol. 10, pp. 211–243, 2016 (cited on p. 37).

[99] S. Schaal, "Learning from demonstration," *Advances in neural information processing systems*, vol. 9, 1996 (cited on p. 37).

[100] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and autonomous systems*, vol. 57, no. 5, pp. 469–483, 2009 (cited on p. 37).

[101] K. Alharthi, Z. S. Abdallah, and S. Hauert, "Understandable controller extraction from video observations of swarms," in *Swarm Intelligence: 13th International Conference, ANTS 2022*, Springer, 2022, pp. 41–53. DOI: https://doi.org/10.1007/978-3-031-20176-9_4 (cited on p. 37).

[102] D. Silver, S. Singh, D. Precup, and R. S. Sutton, "Reward is enough," *Artificial Intelligence*, vol. 299, p. 103 535, 2021, ISSN: 0004-3702. DOI: https://doi.org/10.1016/j.artint.2021.103535. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0004370221000862 (cited on p. 37).

[103] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, 1994 (cited on p. 38).

[104] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, no. 3-4, pp. 279–292, 1992 (cited on p. 39).

[105] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1238–1274, 2013 (cited on p. 39).

[106] J. Abadie, A. Bobillier, D. Fournier, N. Gabas, A. Le Goff, and B. Ould Bouamama, "Reinforcement learning applied to linear quadratic regulation," *Proceedings of the European Symposium on Computer Aided Process Engineering*, vol. 1, 1994 (cited on p. 39).

[107] V. Mnih, K. Kavukcuoglu, D. Silver, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015 (cited on pp. 40, 41).

[108] T. Fan, P. Long, W. Liu, and J. Pan, "Distributed multi-robot collision avoidance via deep reinforcement learning for navigation in complex scenarios," *International Journal of Robotics Research*, vol. 39, no. 7, pp. 856–892, 2020, ISSN: 17413176 (cited on pp. 40, 91).

[109] S. Gu, E. Holly, T. Lillicrap, and S. Levine, "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates," in *2017 IEEE international conference on robotics and automation (ICRA)*, IEEE, 2017, pp. 3389–3396 (cited on p. 40).

[110] D. Isele, R. Rahimi, A. Cosgun, K. Subramanian, and K. Fujimura, "Navigating occluded intersections with autonomous vehicles using deep reinforcement learning," in *2018 IEEE international conference on robotics and automation (ICRA)*, IEEE, 2018, pp. 2034–2039 (cited on p. 40).

[111] S. Yu, X. Chen, Z. Zhou, X. Gong, and D. Wu, "When deep reinforcement learning meets federated learning: Intelligent multitimescale resource management for multiaccess edge computing in 5g ultradense network," *IEEE Internet of Things Journal*, vol. 8, no. 4, pp. 2238–2251, 2020 (cited on pp. 40, 43).

[112] Y. Deng, F. Bao, Y. Kong, Z. Ren, and Q. Dai, "Deep direct reinforcement learning for financial signal representation and trading," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 3, pp. 653–664, 2017 (cited on p. 40).

[113] B. Khailany, "Accelerating chip design with machine learning," in *Proceedings of the 2020 ACM/IEEE Workshop on Machine Learning for CAD*, 2020, pp. 33–33 (cited on p. 40).

[114] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *Proceedings of the 32nd International Conference on Machine Learning*, F. Bach and D. Blei, Eds., ser. Proceedings of Machine Learning Research, vol. 37, Lille, France: PMLR, Jul. 2015, pp. 1889–1897 (cited on p. 40).

[115] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv:1707.06347[cs.LG]*, 2017 (cited on p. 40).

[116] V. Mnih, A. P. Badia, M. Mirza, *et al.*, "Asynchronous methods for deep reinforcement learning," in *Proceedings of The 33rd International Conference on Machine Learning*, M. F. Balcan and K. Q. Weinberger, Eds., ser. Proceedings of Machine Learning Research, vol. 48, New York, New York, USA: PMLR, 20–22 Jun 2016, pp. 1928–1937 (cited on p. 40).

[117] T. P. Lillicrap, J. J. Hunt, A. Pritzel, *et al.*, "Continuous control with deep reinforcement learning," *arXiv:1509.02971[cs.LG]*, 2015 (cited on p. 40).

[118] M. Andrychowicz, F. Wolski, A. Ray, *et al.*, "Hindsight experience replay," *arXiv:1707.01495[cs.LG]*, 2018 (cited on p. 41).

[119] P. Sun, W. Zhou, and H. Li, "Attentive experience replay," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, pp. 5900–5907, Apr. 2020 (cited on p. 41).

[120] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," *arXiv:1511.05952[cs.LG]*, 2016 (cited on pp. 41, 72).

[121] M. Vecerik, T. Hester, J. Scholz, *et al.*, "Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards," *arXiv:1707.08817[cs.AI]*, 2018 (cited on p. 41).

[122] J. Konečný, H. B. McMahan, F. X. Yu, P. Richtarik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," in *NIPS Workshop on Private Multi-Party Machine Learning*, 2016. [Online]. Available: `https://arxiv.org/abs/1610.05492` (cited on p. 42).

[123] H. Brendan McMahan, E. Moore, D. Ramage, S. Hampson, and B. Agüera y Arcas, "Communication-efficient learning of deep networks from decentralized data," *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS*, vol. 54, 2017 (cited on pp. 42, 94, 97, 112).

[124] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 2, pp. 1–19, 2019 (cited on p. 42).

[125] M. J. Sheller, B. Edwards, G. A. Reina, *et al.*, "Federated learning in medicine: Facilitating multi-institutional collaborations without sharing patient data," *Scientific reports*, vol. 10, no. 1, p. 12 598, 2020 (cited on p. 42).

[126] T. S. Brisimi, R. Chen, T. Mela, A. Olshevsky, I. C. Paschalidis, and W. Shi, "Federated learning of predictive models from federated electronic health records," *International journal of medical informatics*, vol. 112, pp. 59–67, 2018 (cited on p. 42).

[127] S. Warnat-Herresthal, H. Schultze, K. L. Shastry, *et al.*, "Swarm learning for decentralized and confidential clinical machine learning," *Nature*, vol. 594, no. 7862, pp. 265–270, Jun. 2021, ISSN: 1476-4687. DOI: 10.1038/s41586-021-03583-3. [Online]. Available: https://doi.org/10.1038/s41586-021-03583-3 (cited on pp. 42, 112).

[128] D. C. Nguyen, M. Ding, P. N. Pathirana, A. Seneviratne, J. Li, and H. V. Poor, "Federated learning for internet of things: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 3, pp. 1622–1658, 2021 (cited on p. 43).

[129] A. Imteaj, U. Thakker, S. Wang, J. Li, and M. H. Amini, "A survey on federated learning for resource-constrained iot devices," *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 1–24, 2021 (cited on p. 43).

[130] P. Yu and Y. Liu, "Federated object detection: Optimizing object detection model with federated learning," in *Proceedings of the 3rd International Conference on Vision, Image and Signal Processing*, 2019, pp. 1–6 (cited on p. 43).

[131] A. M. Elbir, B. Soner, S. Çöleri, D. Gündüz, and M. Bennis, "Federated learning in vehicular networks," in *2022 IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*, IEEE, 2022, pp. 72–77 (cited on p. 43).

[132] X. Wang, C. Wang, X. Li, V. C. M. Leung, and T. Taleb, "Federated deep reinforcement learning for internet of things with decentralized cooperative edge caching," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 9441–9455, 2020 (cited on p. 43).

[133] P. Zhang, C. Wang, C. Jiang, and Z. Han, "Deep reinforcement learning assisted federated learning algorithm for data management of iiot," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 12, pp. 8475–8484, 2021 (cited on p. 43).

[134] B. Liu, L. Wang, and M. Liu, "Lifelong federated reinforcement learning: A learning architecture for navigation in cloud robotic systems," *IEEE Robotics and Automation Letters*, vol. 4, pp. 4555–4562, 2019, ISSN: 2377-3774 (cited on p. 43).

[135] X. Yu, J. P. Queralta, and T. Westerlund, "Towards lifelong federated learning in autonomous mobile robots with continuous sim-to-real transfer," *Procedia Computer Science*, vol. 210, pp. 86–93, 2022 (cited on p. 43).

[136] S. Camazine, J.-L. Deneubourg, N. R. Franks, J. Sneyd, E. Bonabeau, and G. Theraula, *Self-organization in biological systems*. Princeton university press, 2003 (cited on p. 46).

[137] J. Stam, "Stable fluids," in *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, USA: ACM Press/Addison-Wesley Publishing Co., 1999, pp. 121–128, ISBN: 0201485605 (cited on p. 47).

[138] J. H. Ferziger, M. Perić, and R. L. Street, *Computational Methods for Fluid Dynamics*, 4th ed. Springer International Publishing, 2020 (cited on p. 47).

[139] T. Krajník, M. Nitsche, J. Faigl, *et al.*, "A practical multirobot localization system," *Journal of Intelligent & Robotic Systems*, vol. 76, no. 3-4, pp. 539–562, 2014 (cited on pp. 48, 50).

[140] R. G. Vogt, L. M. Riddiford, and G. D. Prestwich, "Kinetic properties of a sex pheromone-degrading enzyme: The sensillar esterase of antheraea polyphemus," *Proceedings of the National Academy of Sciences*, vol. 82, no. 24, pp. 8827–8831, 1985 (cited on p. 49).

[141] T. Wyatt, *Pheromones and Animal Behaviour: Communication by Smell and Taste*. Cambridge University Press, 2003 (cited on pp. 49, 66).

[142] F. Arvin, J. Murray, C. Zhang, and S. Yue, "Colias: An autonomous micro robot for swarm robotic applications," *International Journal of Advanced Robotic Systems*, vol. 11, no. 7, p. 113, 2014 (cited on p. 52).

[143] F. Arvin, A. E. Turgut, T. Krajník, *et al.*, "ΦClust: Pheromone-Based Aggregation for Robotic Swarms," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 4288–4294 (cited on pp. 53, 65, 66).

[144] R. Scheaffer, M. Mulekar, and J. McClave, *Probability and Statistics for Engineers*. Cengage Learning, 2010 (cited on p. 55).

[145] F. Arvin, A. E. Turgut, T. Krajník, and S. Yue, "Investigation of cue-based aggregation in static and dynamic environments with a mobile robot swarm," *Adaptive Behavior*, vol. 24, no. 2, pp. 102–118, 2016 (cited on p. 65).

[146] E. R. Hunt, S. Jones, and S. Hauert, "Testing the limits of pheromone stigmergy in high density robot swarms," *Royal Society Open Science*, vol. 6, p. 190 225, 2019 (cited on p. 66).

[147] H. Hamann, "Towards swarm calculus: urn models of collective decisions and universal properties of swarm performance," *Swarm Intelligence*, vol. 7, no. 2, pp. 145–172, 2013 (cited on p. 66).

[148] D. Payton, M. Daily, R. Estowski, M. Howard, and C. Lee, "Pheromone robotics," *Autonomous Robots*, vol. 11, pp. 319–324, Nov. 2001 (cited on p. 66).

[149] D. J. Sumpter and M. Beekman, "From nonlinearity to optimality: Pheromone trail foraging by ants," *Animal Behaviour*, vol. 66, no. 2, pp. 273–280, 2003 (cited on p. 66).

[150] W. Liu, A. F. T. Winfield, J. Sa, J. Chen, and L. Dou, "Towards energy optimization: Emergent task allocation in a swarm of foraging robots," *Adaptive Behavior*, vol. 15, no. 3, pp. 289–305, 2007 (cited on pp. 66, 67).

[151] R. L. Stewart and R. A. Russell, "A distributed feedback mechanism to regulate wall construction by a robotic swarm," *Adaptive Behavior*, vol. 14, no. 1, pp. 21–51, 2006 (cited on p. 67).

[152] J. C. Nieh, F. A. Contrera, R. R. Yoon, L. S. Barreto, and V. L. Imperatriz-Fonseca, "Polarized short odor-trail recruitment communication by a stingless bee, Trigona spinipes," *Behavioral Ecology and Sociobiology*, vol. 56, no. 5, pp. 435–448, 2004 (cited on p. 67).

[153] P. Mallozzi, P. Pelliccione, A. Knauss, C. Berger, and N. Mohammadiha, "Autonomous Vehicles: State of the Art, Future Trends, and Challenges," in *Automotive Systems and Software Engineering: State of the Art and Future Trends*, Y. Dajsuren and M. van den Brand, Eds. Cham: Springer International Publishing, 2019, pp. 347–367, ISBN: 978-3-030-12157-0. DOI: 10.1007/978-3-030-12157-0_16 (cited on p. 71).

[154] J. Wang, J. Liu, and N. Kato, "Networking and communications in autonomous driving: A survey," *IEEE Communications Surveys Tutorials*, vol. 21, no. 2, pp. 1243–1274, 2019. DOI: 10.1109/COMST.2018.2888904 (cited on p. 71).

[155] Y. Fu, C. Li, F. R. Yu, T. H. Luan, and Y. Zhang, "A survey of driving safety with sensing, vehicular communications, and artificial intelligence-based collision avoidance," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–22, 2021. DOI: 10.1109/TITS.2021.3083927 (cited on p. 71).

[156] M. Schranz, G. A. Di Caro, T. Schmickl, *et al.*, "Swarm intelligence and cyber-physical systems: Concepts, challenges and future trends," *Swarm and Evolutionary Computation*, vol. 60, p. 100 762, 2021, ISSN: 2210-6502 (cited on p. 71).

[157] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization," *IEEE computational intelligence magazine*, vol. 1, no. 4, pp. 28–39, 2006 (cited on pp. 71, 111).

[158] K. L. Soon, J. M.-Y. Lim, and R. Parthiban, "Coordinated traffic light control in cooperative green vehicle routing for pheromone-based multi-agent systems," *Applied Soft Computing*, vol. 81, p. 105 486, 2019, ISSN: 1568-4946. DOI: https://doi.org/10.1016/j.asoc.2019.105486 (cited on p. 71).

[159] A. Campo, Á. Gutiérrez, S. Nouyan, *et al.*, "Artificial pheromone for path selection by a foraging swarm of robots," *Biological Cybernetics*, vol. 103, no. 5, pp. 339–352, 2010, ISSN: 1432-0770. DOI: 10.1007/s00422-010-0402-x (cited on p. 71).

[160] M. Salman, D. Garzón Ramos, K. Hasselmann, and M. Birattari, "Phormica: Photochromic Pheromone Release and Detection System for Stigmergic Coordination in Robot Swarms," *Frontiers in Robotics and AI*, vol. 7, no. December, p. 195, 2020 (cited on p. 71).

[161] S. Na, Y. Qiu, A. E. Turgut, *et al.*, "Bio-inspired artificial pheromone system for swarm robotics applications," *Adaptive Behavior*, vol. 4, no. 29, pp. 395–415, 2020 (cited on p. 71).

[162] J. Ji, A. Khajepour, W. W. Melek, and Y. Huang, "Path planning and tracking for vehicle collision avoidance based on model predictive control with multiconstraints," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 2, pp. 952–964, 2017. DOI: 10.1109/TVT.2016.2555853 (cited on p. 71).

[163] L. Tang, S. Dian, G. Gu, K. Zhou, S. Wang, and X. Feng, "A novel potential field method for obstacle avoidance and path planning of mobile robot," in *2010 3rd International Conference on Computer Science and Information Technology*, vol. 9, 2010, pp. 633–637. DOI: 10.1109/ICCSIT.2010.5565069 (cited on p. 71).

[164] A. Nakamura, Y. C. Liu, and B. G. Kim, "Short-Term Multi-Vehicle Trajectory Planning for Collision Avoidance," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 9, pp. 9253–9264, 2020, ISSN: 19399359. DOI: 10.1109/TVT.2020.3004752 (cited on p. 71).

[165] W. Liu, H. Niu, M. N. Mahyuddin, G. Herrmann, and J. Carrasco, "A model-free deep reinforcement learning approach for robotic manipulators path planning," in *2021 21st International Conference on Control, Automation and Systems (ICCAS)*, IEEE, 2021, pp. 512–517 (cited on p. 71).

[166] S. Shin, D. Ahn, Y. Baek, and H. Lee, "Adaptive aeb control strategy for collision avoidance including rear vehicles*," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, 2019, pp. 2872–2878. DOI: 10.1109/ITSC.2019.8916988 (cited on p. 71).

[167] J. Hu, P. Bhowmick, F. Arvin, A. Lanzon, and B. Lennox, "Cooperative control of heterogeneous connected vehicle platoons: An adaptive leader-following approach," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 977–984, 2020. DOI: 10.1109/LRA.2020.2966412 (cited on p. 71).

[168] J. van den Berg, S. J. Guy, M. Lin, and D. Manocha, "Reciprocal n-body collision avoidance," in *Robotics Research*, C. Pradalier, R. Siegwart, and G. Hirzinger, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 3–19 (cited on pp. 71, 80).

[169] J. Alonso-Mora, A. Breitenmoser, M. Rufli, P. Beardsley, and R. Siegwart, "Optimal reciprocal collision avoidance for multiple non-holonomic robots," in *Distributed Autonomous Robotic Systems: The 10th International Symposium*, Springer Berlin Heidelberg, 2013, pp. 203–216. DOI: 10.1007/978-3-642-32723-0_15 (cited on pp. 71, 72, 80).

[170] J. Alonso-Mora, P. Beardsley, and R. Siegwart, "Cooperative collision avoidance for nonholonomic robots," *IEEE Transactions on Robotics*, vol. 34, no. 2, pp. 404–420, 2018. DOI: 10.1109/TRO.2018.2793890 (cited on pp. 71, 80).

[171] C. Hu, F. Arvin, C. Xiong, and S. Yue, "Bio-inspired embedded vision system for autonomous micro-robots: The lgmd case," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 9, no. 3, pp. 241–254, 2017. DOI: 10.1109/TCDS.2016.2574624 (cited on p. 71).

[172] Y. Yuan, R. Tasik, S. S. Adhatarao, Y. Yuan, Z. Liu, and X. Fu, "Race: Reinforced cooperative autonomous vehicle collision avoidance," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 9, pp. 9279–9291, 2020. DOI: 10.1109/TVT.2020.2974133 (cited on p. 71).

[173] H. Niu, Z. Ji, F. Arvin, B. Lennox, H. Yin, and J. Carrasco, "Accelerated sim-to-real deep reinforcement learning: Learning collision avoidance from human player," in *2021 IEEE/SICE International Symposium on System Integration (SII)*, IEEE, 2021, pp. 144–149 (cited on p. 71).

[174] M. Fortunato, M. G. Azar, B. Piot, *et al.*, "Noisy networks for exploration," *arXiv:1706.10295[cs.LG]*, 2019 (cited on pp. 72, 77).

[175] K. Kuru, "Planning the Future of Smart Cities with Swarms of Fully Autonomous Unmanned Aerial Vehicles Using a Novel Framework," *IEEE Access*, vol. 9, pp. 6571–6595, 2021. DOI: 10.1109/ACCESS.2020.3049094 (cited on p. 74).

[176] J. Ren, G. Yu, Y. He, and G. Y. Li, "Collaborative cloud and edge computing for latency minimization," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 5, pp. 5031–5044, 2019. DOI: 10.1109/TVT.2019.2904244 (cited on p. 74).

[177] Y. Huang, L. Wang, Y. Hou, W. Zhang, and Y. Zhang, "A prototype iot based wireless sensor network for traffic information monitoring," *International journal of pavement research and technology*, vol. 11, no. 2, pp. 146–152, 2018 (cited on p. 74).

[178] F. Dolcos, K. S. LaBar, and R. Cabeza, "Remembering one year later: Role of the amygdala and the medial temporal lobe memory system in retrieving emotional memories," *Proceedings of the National Academy of Sciences*, vol. 102, no. 7, pp. 2626–2631, 2005 (cited on p. 75).

[179] J. A. Bisby, A. J. Horner, L. D. Hørlyck, and N. Burgess, "Opposing effects of negative emotion on amygdalar and hippocampal memory for items and associations," *Social cognitive and affective neuroscience*, vol. 11, no. 6, pp. 981–990, Jun. 2016, ISSN: 1749-5024. DOI: 10.1093/scan/nsw028 (cited on p. 75).

[180] S. Erk, S. Martin, and H. Walter, "Emotional context during encoding of neutral items modulates brain activation not only during encoding but also during recognition," *NeuroImage*, vol. 26, no. 3, pp. 829–838, 2005, ISSN: 1053-8119. DOI: https://doi.org/10.1016/j.neuroimage.2005.02.045 (cited on p. 75).

[181] M. Plappert, R. Houthooft, P. Dhariwal, *et al.*, "Parameter space noise for exploration," in *International Conference on Learning Representations*, 2018 (cited on p. 77).

[182] B. Hölldobler, E. O. Wilson, *et al.*, *The superorganism: the beauty, elegance, and strangeness of insect societies*. WW Norton & Company, 2009 (cited on p. 79).

[183] F. Fossum, J.-M. Montanier, and P. C. Haddow, "Repellent pheromones for effective swarm robot search in unknown environments," in *2014 IEEE Symposium on Swarm Intelligence*, 2014, pp. 1–8. DOI: 10.1109/SIS.2014.7011802 (cited on p. 79).

[184] E. Bonabeau, G. Theraulaz, J. L. Deneubourg, *et al.*, "A model for the emergence of pillars, walls and royal chambers in termite nests," *Philosophical Transactions of the Royal Society B: Biological Sciences*, vol. 353, no. 1375, pp. 1561–1576, 1998, ISSN: 09628436. DOI: 10.1098/rstb.1998.0310 (cited on p. 82).

[185] F. Christianos, L. Schäfer, and S. Albrecht, "Shared experience actor-critic for multi-agent reinforcement learning," in *Advances in Neural Information Processing Systems*, vol. 33, Curran Associates, Inc., 2020, pp. 10 707–10 717 (cited on p. 96).

[186] J. Ulrich, A. Alsayed, F. Arvin, and T. Krajník, "Towards fast fiducial marker with full 6 dof pose estimation," in *Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing*, 2022, pp. 723–730 (cited on p. 103).

[187] P. Lightbody, T. Krajník, and M. Hanheide, "An efficient visual fiducial localisation system," *ACM SIGAPP Applied Computing Review*, vol. 17, no. 3, pp. 28–37, 2017 (cited on p. 103).

[188] T. Krajník, M. Nitsche, J. Faigl, *et al.*, "A practical multirobot localization system," *Journal of Intelligent & Robotic Systems*, vol. 76, no. 3, 2014 (cited on p. 104).

[189] T. Rouček, M. Pecka, P. Čížek, *et al.*, "Darpa subterranean challenge: Multi-robotic exploration of underground environments," in *Modelling and Simulation for Autonomous Systems*, J. Mazal, A. Fagiolini, and P. Vasik, Eds., Cham: Springer International Publishing, 2020, pp. 274–290 (cited on pp. 109, 113).

[190] T. Rouček, M. Pecka, P. Čížek, *et al.*, "System for multi-robotic exploration of underground environments CTU-CRAS-NORLAB in the DARPA Subterranean Challenge," *Field Robotics*, 2021. [Online]. Available: https://doi.org/10.55417/fr.2022055 (cited on p. 109).

[191] J. Connor, B. Champion, and M. A. Joordens, "Current algorithms, communication methods and designs for underwater swarm robotics: A review," *IEEE Sensors Journal*, vol. 21, no. 1, pp. 153–169, 2021. DOI: 10.1109/JSEN.2020.3013265 (cited on p. 109).

[192] M. Stefanec, D. N. Hofstadler, T. Krajník, *et al.*, "A minimally invasive approach towards "ecosystem hacking" with honeybees," *Frontiers in Robotics and AI*, vol. 9, 2022 (cited on p. 109).

[193] A. Rudenko, T. P. Kucner, C. S. Swaminathan, R. T. Chadalavada, K. O. Arras, and A. J. Lilienthal, "Thör: Human-robot navigation data collection and accurate motion trajectories dataset," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 676–682, 2020. DOI: 10.1109/LRA.2020.2965416 (cited on p. 109).

[194] M. Duarte, V. Costa, J. Gomes, *et al.*, "Evolution of collective behaviors for a real swarm of aquatic surface robots," *PLoS ONE*, vol. 11, no. 3, pp. 1–25, 2016. DOI: 10.1371/journal.pone.0151834 (cited on p. 109).

[195] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of Machine learning and systems*, vol. 2, pp. 429–450, 2020 (cited on p. 112).

[196] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "Scaffold: Stochastic controlled averaging for federated learning," in *International conference on machine learning*, PMLR, 2020, pp. 5132–5143 (cited on p. 112).

[197] I. Jang, H.-S. Shin, and A. Tsourdos, "Anonymous hedonic game for task allocation in a large-scale multiple agent system," *IEEE Transactions on Robotics*, vol. 34, no. 6, pp. 1534–1548, 2018. DOI: 10.1109/TRO.2018.2858292 (cited on p. 113).

[198] D. Fudenberg and J. Tirole, *Game theory*. MIT press, 1991 (cited on p. 113).

[199] M. J. Schuster, M. G. Müller, S. G. Brunner, *et al.*, "The arches space-analogue demonstration mission: Towards heterogeneous teams of autonomous robots for collaborative scientific sampling in planetary exploration," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5315–5322, 2020. DOI: 10.1109/LRA.2020.3007468 (cited on p. 113).

Appendices

Appendix A

Chapter 3

A.1 Preliminary experiments for parameter choice of maunal controller design

As mentioned in Chapter 3, preliminary experiments were conducted to find suitable parameter values for each experimental stage. Each parameter set of β_{const} and α in the range of $(0.6 \leq \beta_{const} \leq 1.4, 0.6 \leq \alpha \leq 1.4)$ with the step size of 0.1 was tested 20 times for each experimental stage. The results with three performance metrics in each stage are illustrated in Fig. A.1, A.2 and A.3.

In Stage 1, the experimental results demonstrate a successful collision avoidance with a low β_{const} value ($\beta_{const} \leq 0.9$). Conversely, higher values of β_{const} and lower values of α lead to a drop in the success rate due to an increased likelihood of collision. This is attributed to the higher β_{const}, which accelerates the forward movement of the robot. Interestingly, a higher β_{const} is associated with a reduced completion time, while the effect of α depends on whether β_{const} is above or below 1.0. Trajectory efficiency generally increases with lower β_{const} and α, but a trade-off between the two parameters may deviate from this trend. The chosen optimal parameter set, taking into account success rate, completion time, and trajectory efficiency, is $\beta_{const} = 0.9$, $\alpha = 0.7$.

In Stage 2, the success rate is influenced primarily by β_{const}, with nearly no success for $\beta_{const} > 0.9$. Completion time generally decreases with lower α, although outliers exist that defy this trend. Trajectory efficiency shows a similar trend to completion time, and the optimal parameter set for Stage 2 is $\beta_{const} = 0.7$, $\alpha = 0.8$.

Stage 3 illustrates a more complex picture, with no clear general rule linking success rate to parameter values, reflecting the higher complexity of this stage. Three parameter sets yielded the highest success rates, with a slight trend towards lower completion time with higher α for $\beta_{const} \in (0.6, 0.7)$. Only a general trend that higher α led to higher trajectory efficiency was found. The chosen parameter set for Stage 3, $\beta_{const} = 0.7$, $\alpha = 1.3$, resulted in the fastest completion time, which was the second priority criterion.

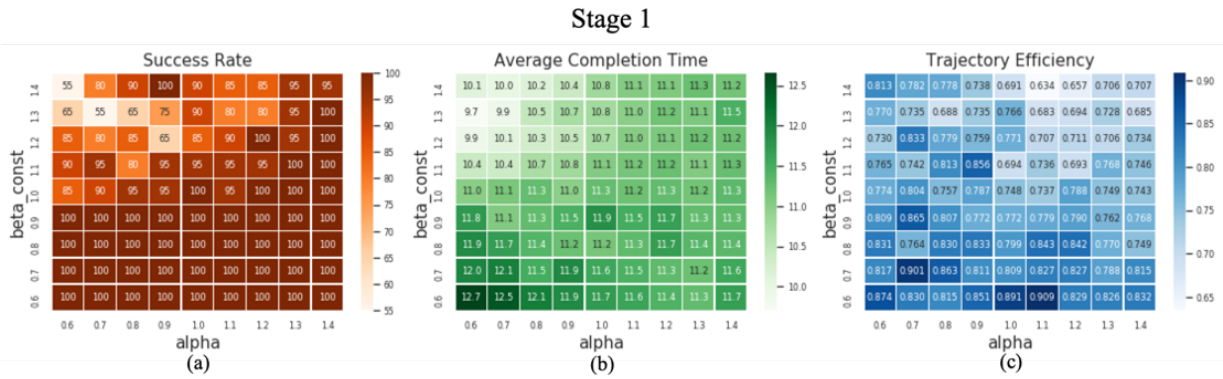

Figure A.1. Evaluation results of the manual controller design with different parameter sets in Stage 1. (a), (b), (c) illustrates success rate, average completion time and trajectory efficiency respectively.

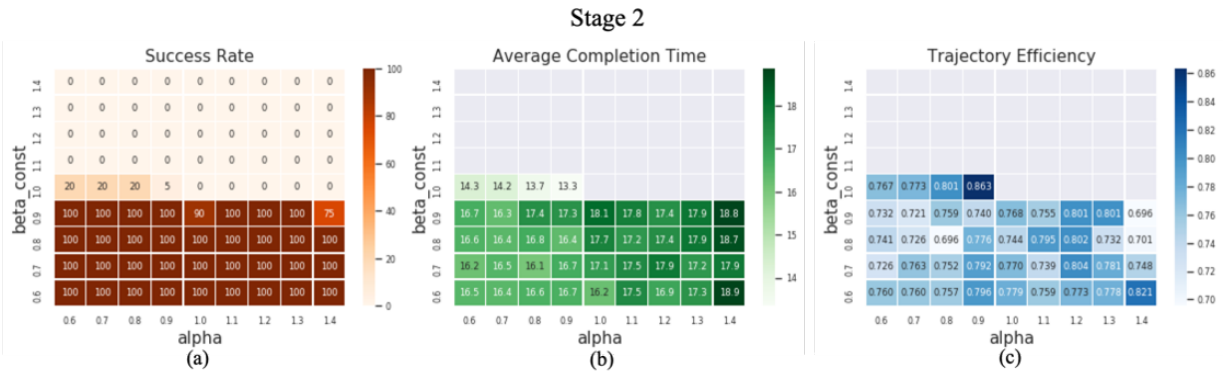

Figure A.2. Evaluation results of the manual controller design with different parameter sets in Stage 2. (a), (b), (c) illustrates success rate, average completion time and trajectory efficiency respectively.

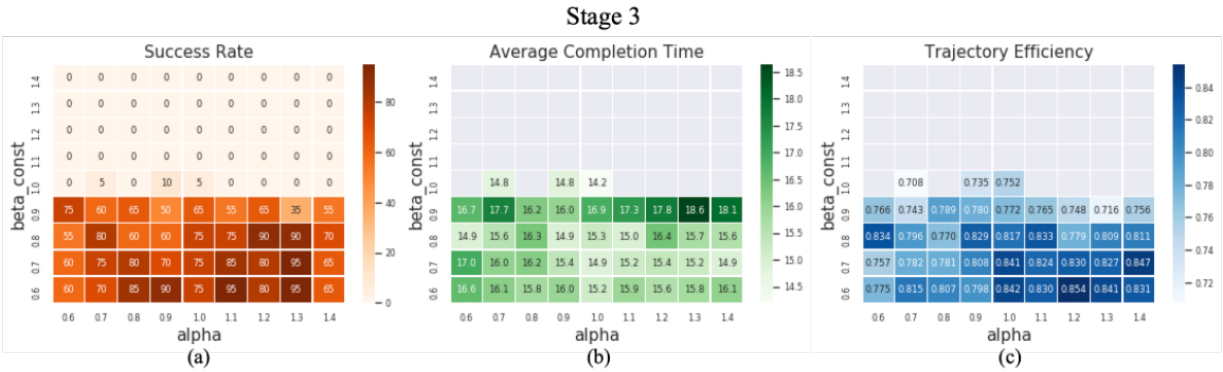

Figure A.3. Evaluation results of the manual controller design with different parameter sets in Stage 3. (a), (b), (c) illustrates success rate, average completion time and trajectory efficiency respectively.