Multi-Criteria Study of Collective Swarm Decision Making in Large Decision Spaces

DISSERTATION

zur Erlangung des akademischen Grades

Doktoringenieur (Dr.-Ing.)

angenommen durch die Fakultät für Informatik
der Otto-von-Guericke-Universität Magdeburg

von M.Sc. Qihao Shan

geb. am 06.02.1995 in Nanjing

Gutachterinnen/Gutachter

Prof. Dr.-Ing. habil. Sanaz Mostaghim
Prof. Dr.-Ing. Heiko Hamann
Prof. Dr. Jonathan Lawry

Magdeburg, den 16.01.2025

Ehrenerklärung

I hereby certify that I have prepared this thesis without the unauthorized assistance of third parties and without the use of resources other than those indicated; external and my own sources used are identified as such. In particular, I have not used the help of a commercial doctoral advisor. Third parties have neither directly nor indirectly received monetary benefits from me for work related to the content of the submitted dissertation.

In particular, I did not knowingly:
- invent any results or conceal contradictory results,
- intentionally misuse statistical procedures to interpret data in an unjustified manner,
- plagiarize other people's results or publications,
- distort the results of other research.

I am aware that violations of copyright law may give rise to injunctive relief and claims for damages by the author as well as criminal prosecution by the law enforcing authorities. The work has not yet been submitted as a dissertation in the same or a similar form either in Germany or abroad and has not yet been published as a whole.

Magdeburg 14.10.2024

Qihao Shan

Abstract

It is a promising approach when designing distributed embodied intelligent systems to employ decentralized self-organized control paradigms, where the collective behavior of the system is controlled via local peer-to-peer interactions instead of a unified leader. A crucial building block of coherent self-organized systems is the ability to produce global decisions via local interactions between the agents and with the environment. This is a challenging design problem when constructing artificial swarm intelligence systems, as the agents need to consider information gained from both environmental exploration and inter-agent communication. Inspirations for the design of collective decision-making strategies in artificial systems can be drawn from the behaviors of natural intelligent swarms such as insect colonies, fish schools, and bird flocks. However, due to the characteristics of the biological agents and the analog environments they dwell in, natural collective decision-making strategies are specialized in decision-making scenarios with small decision spaces and simple quality distributions, making them ill-suited for many artificial applications.

This thesis seeks to expand on the capabilities of artificial collective decision-making strategies into scenarios with large decision spaces and complex quality distributions. Emphasis is placed on multi-option collective consensus problems (best-of-n problems), and the discrete collective estimation scenario is proposed to complement the binary and multi-feature collective perception scenario found in previous literature. Two novel collective consensus strategies that enable the agents to individually consider multiple options in parallel are proposed: one uses directly the Bayesian likelihood of the options being optimal, the other uses their rankings to represent the relative preferences by the agents. Their performances are shown to outperform bio-inspired opinion-based strategies in a multi-criteria framework that considers the speed versus accuracy trade-off in decision-making performance.

Subsequently, the discrete collective estimation scenario is extended to the many-option case, in which the discrete options far outnumber the available agents. The expanded decision space is shown to have different impacts on the decision-making performances depending on the quality distribution. It also brings the discrete collective consensus problem close to having a continuous decision space and reveals the upper limit of the proposed multi-option consensus strategies.

As a larger decision space has been shown to negatively affect the performances of collective decision-making strategies, two independent decision-making scenarios are subsequently studied for ways of limiting the decision space size during the decision-making process: The many-option collective consensus problem is further investigated using the collective preference learning scenario. The effects of limiting the decision space by enforcing a strict ranking among the sites are examined. Finally, a continuous task allocation scenario is studied to demonstrate how dynamic decision space restriction can help the swarm deal with multi-modal quality functions and produce specialized behaviors.

Zusanmmenfassung

Bei der Entwicklung intelligenter Mehrkörpersysteme ist der Einsatz dezentraler selbstorganisierter Steuerungsparadigmen ein vielversprechender Ansatz, bei dem das kollektive Verhalten des Systems über lokale Peer-to-Peer-Interaktionen und nicht über einen einheitlichen Leiter gesteuert wird. Ein entscheidender Baustein kohärenter selbstorganisierter Systeme ist die Fähigkeit, globale Entscheidungen über lokale Interaktionen zwischen den Agenten und mit der Umgebung zu treffen. Dies ist ein anspruchsvolles Designproblem bei der Konstruktion künstlicher Schwarmintelligenzsysteme, da die Agenten Informationen berücksichtigen müssen, die sie sowohl aus der Erkundung der Umgebung als auch aus der Kommunikation zwischen den Agenten gewinnen. Inspiration für die Entwicklung kollektiver Entscheidungsstrategien in künstlichen Systemen kann aus dem Verhalten natürlicher intelligenter Schwärme wie Insektenkolonien, Fischschwärmen und Vogelschwärmen gezogen werden. Aufgrund der Eigenschaften der biologischen Agenten und der analogen Umgebungen, in denen sie leben, sind natürliche kollektive Entscheidungsstrategien jedoch auf Entscheidungsszenarien mit kleinen Entscheidungsräumen und einfachen Qualitätsverteilungen spezialisiert, was sie für viele künstliche Anwendungen ungeeignet macht.

Diese Dissertation versucht, die Fähigkeiten künstlicher kollektiver Entscheidungsfindungsstrategien auf Szenarien mit großen Entscheidungsräumen und komplexen Qualitätsverteilungen auszuweiten. Der Schwerpunkt liegt auf kollektiven Konsensproblemen mit mehreren Optionen (Best-of-n-Probleme), und das Szenario der diskreten kollektiven Schätzung wird als Ergänzung zu dem in der bisherigen Literatur beschriebenen Szenario der binären und multifunktionalen kollektiven Wahrnehmung vorgeschlagen. Es werden zwei neuartige kollektive Konsensstrategien vorgeschlagen, die es den Agenten ermöglichen, mehrere Optionen parallel einzeln zu prüfen: Eine verwendet direkt die Bayes'sche Wahrscheinlichkeit, dass die Optionen optimal sind, die andere verwendet ihre Rangfolge, um die relativen Präferenzen der Agenten darzustellen. Ihre Leistung übertrifft nachweislich die von der Biologie inspirierten meinungsbasierten Strategien in einem multikriteriellen Rahmen, der den Kompromiss zwischen Geschwindigkeit und Genauigkeit bei der Entscheidungsfindung berücksichtigt.

Anschließend wird das Szenario der diskreten kollektiven Schätzung auf den Fall mit vielen Optionen erweitert, in dem die diskreten Optionen die verfügbaren Agenten bei weitem übertreffen. Es zeigt sich, dass der erweiterte Entscheidungsraum je nach Qualitätsverteilung unterschiedliche Auswirkungen auf die Entscheidungsleistung hat. Es bringt das diskrete kollektive Konsensproblem auch in die Nähe eines kontinuierlichen Entscheidungsraums und zeigt die Obergrenze der vorgeschlagenen Konsensstrategien mit mehreren Optionen auf.

Da sich gezeigt hat, dass ein größerer Entscheidungsraum die Leistung kollektiver Entscheidungsstrategien negativ beeinflusst, werden anschließend zwei unabhängige Entscheidungsszenarien untersucht, um Möglichkeiten zur Begrenzung der Entscheidungsraumgröße während des Entscheidungsprozesses zu finden: Das Problem des kollektiven Konsenses mit vielen Optionen wird anhand des Szenarios des kollektiven Präferenzlernens weiter untersucht. Die Auswirkungen einer Begrenzung des Entscheidungsraums durch die Durchsetzung einer strikten Rangfolge der Standorte werden untersucht. Abschließend wird ein Szenario der kontinuierlichen Aufgabenzuweisung untersucht, um zu demonstrieren, wie eine dynamische Einschränkung des Entscheidungsraums dem Schwarm helfen kann, mit multimodalen Qualitätsfunktionen umzugehen und spezialisierte Verhaltensweisen hervorzubringen.

Contents

List of Figures

List of Tables

Chapter 1

Introduction & Motivation

In recent decades, advancements in microprocessors, portable batteries, sensors, and actuators have vastly expanded the applicability of robotic platforms in performing real-world tasks [SA14]. The specific characteristics of many tasks favor a robot swarm solution, in which many simplistic robots are deployed cooperatively to fulfill the intended objective [DTT21]. However, it remains a difficult problem to control the large number of robots efficiently. Such challenges are answered by naturally existing multi-body systems with self-organized behaviors. Natural intelligent swarms such as insect colonies, bird flocks, and fish schools display complex collective behaviors via only local peer-to-peer interactions and with no centralized leadership [Cam+20]. The prevalence of natural intelligent swarms has inspired many self-organized collective strategies that can be applied to artificial swarm intelligence systems [MJ11] and can bring multiple benefits, including robustness, scalability, and flexibility [Bra+13].

An important building block of complex self-organized collective behaviors is collective decision making. It is a distributed process where a group of agents cooperatively reaches a global decision without being directed by a single leader [BRM17]. The collective decision-making strategies adopted by natural intelligent swarms vary according to the characteristics of the problems they are aiming to solve, which range from the discrete best-of-n consensus problem faced by honey bees during house hunting [Rei+17], to the continuous consensus problem faced during collective motion [Rey87], and the task allocation problem when creating specialized behaviors in various social insects [Gor16]. Two key characteristics affecting the viability of collective decision-making strategies are the size of the decision space and the distribution of the decision quality. Due to the analog environments faced by natural intelligent swarms, the majority of previous literature studies collective decision-making problems with either binary or continuous decision space. The gap of natural multi-option decision-making scenarios is translated into a lack of attention paid to the multi-option collective decision-making behaviors in artificial intelligent swarms [VFD17; Ibr+24]. This thesis seeks to bridge this gap by extending the capability of artificial collective decision-making strategies into scenarios with large decision spaces, while analyzing the influence of decision space size and quality distribution on the performances of collective decision-making strategies with a multi-criteria framework.

1.1 Self-Organized Systems and Swarm Intelligence

It is observed in physics that only a fraction of the total energy in a system can be used to do work. The rest, characterized by irregular movements of the system's constituent particles, is referred to as "disordered" energy and is denoted as entropy. As stated in the second law of thermodynamics, the entropy of a closed system cannot decrease with time. In other words, a closed system will always irreversibly tend toward disorder [Pri78]. However, given specific rules of intermolecular interactions, some open physical systems can tend toward order from a state of disorder given minimal outside interference, such as the phenomenon of spontaneous magnetization in metals [Yan52]. This process of forming order from disorder is referred to as self-organization or spontaneous order. From the point of view of thermodynamics, living systems maintain negative entropy, and hence avoid decay into disorder, via homeostasis at the microscopic level [Sch44]. Rather, the microscopic disorder in the form of random genetic mutation is the driving force behind evolution and hence the improvements and growth of species [Nei13].

Self-organized processes play an especially prevalent role in multi-body living systems such as insect swarms or animal flocks [Cam+20], which are often formed by simplistic agents with limited individual capabilities, making a centralized command structure difficult to form. The lack of permanent inter-agent links also means that the collective system is highly susceptible to random divergences in agents' behaviors. However, via efficient swarm intelligence mechanisms, these naturally existing multi-body systems can display coherent collective behaviors that exceed the capabilities of their constituent agents. The collective level coherence is created via a similar negative entropy process that governs intermolecular and intercellular interactions in single-body living systems.

To this end, four key characteristics are needed for inter-agent interactions in intelligent swarms, namely positive feedback, negative feedback, randomness, and repetitive interactions [Bon+99]. Positive and negative feedback within a decentralized system describes the effects of the actions or decisions of an agent on those of other agents in the system. Positive feedback denotes an amplifying effect between the actions of individual agents. Thus, an agent's behavior will encourage the other agents to adopt similar behaviors, causing it to propagate on a global scale within the whole swarm. This is exemplified by the behavior of the European honey bee (*Apis mellifera*) when collectively selecting a new nest site [SB01]. A scout bee first independently surveys a potential site. It then returns to its nest and tries to recruit its peers to its choice by performing a waggle dance in front of them. The strength of recruitment is determined by the quality of the site surveyed, causing better sites to be more frequently visited by subsequent scouts. This amplifying effect results in a final consensus of the swarm for the new nest site.

On the other hand, negative feedback denotes a diminishing effect between the actions of individual agents. In this case, an agent's behavior will discourage the other agents to adopt a similar behavior or encourage them to adopt a counteracting behavior. For example, when bird flocks are traveling together and a subset of them forms a congestion region, the other birds will collectively avoid the congestion region, hence balancing the concentration of agents and preventing collision [Ant+09].

At the same time, randomness contributes to the exploratory behavior of the system into its potential actions and choices. It serves two functions in a self-organized system. Firstly, through random exploration, individual agents can potentially discover actions or choices that yield more reward. Secondly, the random perturbations can then be amplified by the entire system using the aforementioned positive feedback to improve the overall behavior of the system. Randomness is coupled with repetitive interactions to ensure the agents are well-mixed during their interactions, which ensure the integrity of the swarm.

At first glance, it can be seen that the aforementioned characteristics of local interactions are inherently counteracting. A high degree of positive feedback and randomness leads to an exploratory and unstable system, while strong negative feedback encourages stable behaviors. The difficulty in designing an artificial swarm intelligence system lies in setting the local rules of interactions that adjust intelligently the relative importance of the aforementioned processes and lead to the desired behavior on the global scale.

1.2 Collective Decision-Making Problem in the Context of Swarm Intelligence

Among the various intelligent behaviors that can be performed by swarm intelligence systems, decision making is a crucial component. Collective decision making refers to the process where a group of agents collectively reaches a decision, which, once made, cannot be attributed to any single agent [BRM17]. Such processes exist beyond the collective behavior of simple agents. Indeed, group decision making has been studied in the context of human collective behavior for decades [Bla48; SP13]. Recently, researchers have also started to pay more attention to the decision-making behavior in animal flocks [CR03; CL09] and insect swarms [Joh+02; Gor16].

The decision-making problem in an ideal scenario can be modeled as follows:

$$d^* = \text{argmax}_d(q(d)). \tag{1.1}$$

Here $q(d)$ is the quality distribution function that depends on the chosen decision d, where the value d^* gives the optimal quality. The goal of the decision-making process is to search for the optimal decision d^*, so that the decision-making agent reaps the maximum reward. When there are numerous decision-making agents, the above model is expanded to the following:

$$\vec{d}^* = \text{argmax}_{\vec{d}}\Big(\sum_{i=1..N_{agent}} w_i q_i(d_i)\Big). \tag{1.2}$$

Here, the goal is to maximize the collective quality of decisions for all agents $1..N_{agent}$, that individually face their own quality distribution function $q_i(d_i)$, and is scaled by the weight w_i denoting the relative importance of the agent i. In scenarios where q_i are completely or nearly completely uncorrelated with each other, the search can only be done individually by the agents, and thus the problem can be divided into N_{agent} individual decision-making problems. However, realistically the agents in an intelligent swarm operate in proximity with each other, thus their individual quality distribution functions $q_i(d_i)$ are correlated and influenced by one another. Therefore, the exchange of information across agents can greatly improve the decision-making process in terms of speed, accuracy, and reliability. The study of collective decision-making scenarios and strategies thus focuses on utilizing inter-agent communication as well as interaction with the environment to reach an optimal global decision for the whole swarm.

In this thesis, the agents of the swarm are assumed to be homogeneous, which means they are of equal importance and are interchangeable with each other in terms of functionality. Therefore, the weights w_i in the aforementioned model can be ignored. Thus, the model can be transformed into the following:

$$\vec{d}^* = \text{argmax}_{\vec{d}}\Big(\sum_{i=1..N_{agent}} [f(d_i) + g(d_{1..N_{agent}}) + h_i]\Big). \tag{1.3}$$

In this case, the quality function of the individual agents q_i is broken up into 3 components. $f(d_i)$ denotes the individual quality function component that is the same for all agents. Its output is only determined by the decision of the agent i i.e. d_i. $g(d_{1..N_{agent}})$ denotes the collective quality function component that depends on the decisions of all agents. Finally, the term h_i denotes the random perturbation to the quality function as a result of limitations in information transfer and variations among local subgroups in the swarm. This effect is assumed to be random from the perspectives of the entire swarm, thus causing the actual quality functions observed by individual agents to be inaccurate.

Depending on the types of global decisions reached, collective decision-making scenarios can be divided into two main categories: consensus-forming scenarios and task allocation scenarios [VFD17]. The former category includes consensus-forming scenarios with continuous decision spaces such as flocking [Eml52; Olf06], and those with discrete decision spaces (also referred to as best-of-n problems when applied to artificial agents) such as shortest-path selection [Sch+16] and optimal-site selection [PZ09; PZ11; Val+16b]. On the other hand, task allocation scenarios in the context of swarm intelligence relax the need for the agents to converge to a single decision. Instead, the focus of the agents is on searching for the optimum of the quality function [BF01; Dua+12; Gor16; CMG20].

Using the aforementioned model, consensus-forming problems can be expressed as an optimization process searching for the optimum of the quality function $f(d_i)$. However, due to the random perturbation h, the quality function $f(d_i)$ is not directly observable from the perspective of the agents. Therefore, collective consensus-forming strategies adopt an indirect approach expressed in a dual-objective optimization process as follows:

$$\text{Consensus forming: } \vec{d^*} = \{ {\text{argmax}_{\vec{d}}(\sum_{i=1..N_{agent}}[f(d_i) + h_i]) \atop \text{argmin}_{\vec{d}}(\sum_{i=1..N_{agent}}[|\bar{d} - d_i| + h_i'])}. \tag{1.4}$$

Here, the optimal decision array $\vec{d^*}$ is obtained by simultaneously searching for the optima in the two quality functions observed by the individual agents, as well as keeping the variation of opinions within the swarm to a minimum. The latter is formulated here as the sum of the absolute differences between every individual agent's decision and the centroid of all decisions in the swarm $\bar{d}$ (or medoid in the case of discrete non-ordinal decisions). The consensus-forming process is expected to converge to the true optimum of $f(d_i)$ because of the random nature of h_i and h_i', and their effects are canceled out when considering many instances of the quality function at the same time by all agents in the swarm.

In natural intelligent swarms, discrete collective consensus forming is a common behavior and has been extensively studied in previous literature as best-of-n problems [VFD17]. The prevalent consensus-forming approaches employ population-based search in the decision space for the decision with the optimal quality. This process is usually realized in naturally existing intelligent swarms by agents who pick more optimal options actively recruiting their peers via dissemination behaviors such as the waggle dances by bees during collective selection of potential nest locations [SB01], or the pheromone-laying behaviors of ants during pathfinding [Den+90]. When implemented on artificial agents, such population-based search approaches for discrete consensus-forming problems are referred to as opinion-based strategies [VFD17]. Similar approaches are found in continuous consensus scenarios, such as flocking [Olf06], where despite the continuous decision space, the quality function has a smooth unimodal shape and can be solved with gradient-based decentralized optimization approaches. A minority of previous studies have investigated consensus approaches where the artificial agents consider multiple decisions in parallel from a probabilistic perspective [LLW21; BM21].

On the other hand, this thesis expresses task allocation problems as a generalization of the aforementioned consensus-forming framework, with the previous objective to minimize the variation of decisions replaced by a search for the optimum of the more generalized collective quality function $g(d_{1..N_{agent}})$:

$$\text{Task allocation: } \vec{d}^* = \{ \begin{matrix} \text{argmax}_{\vec{d}}(\sum_{i=1..N_{agent}}[f(d_i) + h_i]) \\ \text{argmax}_{\vec{d}}(\sum_{i=1..N_{agent}}[g(d_{1..N_{agent}}) + h'_i]) \end{matrix} . \tag{1.5}$$

Because convergence to a single decision is not needed for the task allocation problem, the task allocation behaviors displayed by natural intelligent swarms are modeled differently from consensus forming behaviors. Classical models hold the hypothesis that the individual agents have an internal policy optimized through evolution, and specialized behaviors arise when they respond to different environmental stimuli [BF01; Dua+11; Dua+12]. Newer research favors an online search for the optimal strategy utilizing inter-agent interactions [Gor16; CMG20].

The level of difficulty of a collective decision-making problem in both categories is determined by a few factors, including the shapes of the raw objective functions $f(d_i)$ and $g(d_{1..N_{agent}})$, the decision space size, and the spatial disparities of the objective functions described by h_i and h'_i. Particularly, due to the characteristics of biological agents, collective strategies tackling scenarios with large discrete decision spaces and complex quality distributions are rarely found among natural intelligent swarms. This limits the applicability of artificial intelligent swarm solutions to real-world problems.

1.3 Trade-offs in Collective Decision-Making Processes

Due to the dual-objective nature of both consensus-forming and task-allocation problems shown before, the microscopic decision-making behaviors of individual agents need to keep track of both individual and collective information input. In the case of consensus-forming scenarios, the decision-making process from the point of view of agent i can be expressed as follows:

$$\text{Consensus-forming agent } i\text{: } d_i^* = \{ \begin{matrix} \text{argmax}_{d_i}(f(d_i) + h_i) \\ \text{argmin}_{d_i}(|\bar{d}_i - d_i| + h'_i) \end{matrix} . \tag{1.6}$$

As an individual agent only has access to information in its locality, it can only observe its own estimate of the mean decision $\bar{d}_i$. In the microscopic view of collective consensus-forming behaviors, the two objective functions can be interpreted as an interaction between two sources of information, one from the individual agent's own interactions with the environment, and the other from its peers in the vicinity. The relative importance of the two sources of information controls the level of macroscopic consensus enforcement in the swarm. When the individual agent's own environmental interactions (the first objective function in Equation 1.6) are given more weight, the swarm has a weaker enforcement of consensus, thus slowing down the convergence and potentially increasing the decision accuracy by mitigating more of the effects of h_i and h'_i via repetitive environmental observations. On the other hand, when the level of uniformity with its peers (the second objective function in Equation 1.6) is given more weight, the swarm has a stronger enforcement of consensus, leading to a faster decision and potentially reducing the decision accuracy by only allowing limited environmental observations. Thus, the relative weights of the two objective functions control the extent of speed versus accuracy trade-off in collective consensus-forming processes, which has been observed in natural collective consensus-forming behaviors [Fra+03; PS06], and is also demonstrated in artificial intelligent swarms under multiple decision-making strategies [VHD15; Ebe+20; Aus+22].

On the other hand, for distributed task-allocation scenarios, the main source of trade-off is the interaction between the individual objective function $f(d_i)$ and the collective objective function $g(d_1..N_{agent})$:

$$\text{Task-allocation agent } i\text{: } d_i^* = \{ \begin{matrix} \text{argmax}_{d_i}(f(d_i) + h_i) \\ \text{argmax}_{d_i}(g'_i(d_{1..N_{agent}}) + h'_i) \end{matrix} . \tag{1.7}$$

Here the collective objective function g loses the unimodal shape faced by agents in consensus forming scenarios, and therefore the agents are not restricted to converge to a single decision. It is further complicated by the lack of centralized control of the swarm, and the collective performance can only be estimated in terms of $g'_i(d_{1..N_{agent}})$ by the individual agents using information from its vicinity. Thus, there is a more prevalent trade-off in the results produced by optimization between individual and collective objectives. This is demonstrated in natural intelligent swarms, such as honeybees adjusting the ratio of workers dedicated to temperature regulation in the nest depending on social context [CB13].

The existence of individual and collective objective functions for both collective decision-making paradigms means that the optimization search runs on non-stationary and heterogeneous objective functions, and hence convergence is not guaranteed. In addition, different configurations on the relative importance of both objective functions lead to different decision-making performances when considering multiple criteria, and often cannot be compared with each other without assigning the relative importance of the involved criteria. In meta-heuristic optimization problems, the interaction of conflicting objective functions leads to the existence of multiple optimal solutions, which form a Pareto frontier [Cen77]. Similarly, when comparing the performances of collective decision-making strategies, the Pareto frontiers of their performances produced at different design parameter configurations need to be examined using multi-criteria analysis.

1.4 Research Objectives

This thesis aims to expand the applicability of swarm intelligence design paradigms by investigating the design and performances of collective decision-making strategies in scenarios with large decision spaces, with an emphasis on multi-option collective consensus-forming scenarios.

Research Objective 1: Investigate novel consensus-forming strategies utilizing parallel consideration of discrete options in best-of-n problems where $n > 2$ using a multi-criteria framework.

Research Question 1.1: Is a parallel computation of option qualities using Bayesian statistics viable compared to existing opinion-based strategies in a best-of-2 collective perception problem?
Research Question 1.2: How can the computed option qualities be propagated in the swarm using parallel encoding of option preferences?
Research Question 1.3: How can the speed versus accuracy trade-off be accounted for in the comparison of the considered consensus strategies?

The first research objective starts from the best-of-2 collective perception scenario, with the aim to increase the number of potential discrete options that the swarm can handle. As the number of discrete options increases beyond 2, existing opinion-based strategies cannot guarantee coverage of all potential options during the decision-making process. This makes it beneficial to enable the swarm agents to consider multiple options simultaneously in best-of-n problems. The proposed approaches also need to be compared with existing ones while considering the speed versus accuracy trade-off to obtain the complete picture of their performances. It is thus important to consider multiple performance criteria during the analysis without assuming their relative importance.

Research Objective 2: Gauge the viabilities of the proposed multi-option consensus strategies in large discrete decision spaces under fair comparison.

Research Question 2.1: How can the communication efficiency of the proposed parallel encoding approaches be gauged while considering the scaling of message size with the number of discrete options?
Research Question 2.2: As the size of the discrete decision space increases far beyond the number of agents, to what extent are the performances of the proposed parallel encoding approaches negatively affected by the scaling message size?
Research Question 2.3: How are the performances of the considered consensus strategies affected by the shape of the quality function in the decision space?

Parallel consideration of potential options causes the computation complexities carried by individual swarm agents to scale linearly with the size of the discrete decision space. Here, the focus is placed on determining the communication efficiency of the proposed multi-option consensus strategies. This lays the foundation to investigate their performances in much larger decision spaces. The increased decision space also allows investigation into the effects of different quality distributions on the decision-making performances.

Research Objective 3: Explore the effects of restricting the decision space sizes on the performances of collective decision-making strategies.

Research Question 3.1: How can actively restricting the discrete decision space size facilitate the collective preference learning task?
Research Question 3.2: Can an intelligent swarm effectively handle multimodal quality functions via dynamic decision space restrictions?

As the performances of collective decision-making strategies are negatively affected by larger decision space sizes, the last research objective explores the effects of restricting the large decision spaces during the collective decision-making process in two independent scenarios: one with large discrete decision spaces, the other with multi-modal quality functions in continuous decision spaces.

1.5 Thesis Outline

The outline of the thesis is as follows. Firstly, Chapter 2 introduces the state-of-the-art collective decision-making approaches for artificial intelligent swarms. The presented strategies are classified according to their intended scenario and origin. Besides bio-inspired collective decision-making strategies, techniques have been drawn from previous research on networked artificial agents such as sensor networks, as well as models for human social networks.

The overview of the following chapters is shown in Figure 1.1. Chapters 3 and 4 address the first research objective, aiming to propose and test multi-option consensus strategies that utilize parallel encoding of option preferences. Chapter 3 starts with RQ 1.1 and investigates whether Bayesian statistics-based parallel quality computation can be employed in the classical best-of-2 collective perception scenario. Here, the binary decision space consisting of the two features is further discretized into the exact feature fill ratio hypotheses, which can more accurately estimate the likelihood of a particular feature being in the majority. Afterward, Chapter 4 deals with RQ 1.2 by looking into how the parallel representations of option qualities can be used by the swarm agents for multi-option consensus forming in the discrete collective estimation scenario. To this end,

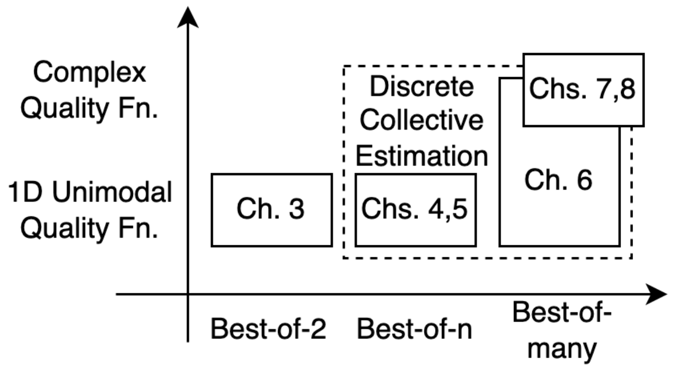


Figure 1.1: Overview of the decision-making scenarios investigated in each chapter

two decision-making strategies have been proposed: the first uses directly the computed Bayesian likelihoods of the options; the other uses the rankings of their likelihoods. RQ 1.3 is also tackled in Chapter 4 with the comparison of the proposed consensus strategies with existing ones using a multi-criteria framework, which considers the Pareto frontiers of each strategy's speed versus accuracy trade-off in its performance.

Chapters 5 and 6 focus on the second research objective and investigate the scalability of the proposed multi-option consensus strategies in large discrete decision spaces. Chapter 5 seeks to answer RQ 2.1 by adjusting the rates of information transfer under the different considered strategies to the same level in comparison. This allows the communication efficiency of the proposed parallel encoding approaches to be compared with existing opinion-based approaches when facing large discrete decision spaces. Chapter 6 addresses RQ 2.2 by further extending the discrete collective estimation scenario into the many-option case, where the number of discrete options far surpasses the number of available agents. The larger decision spaces also allow the quality functions to move beyond the simple 1-dimensional unimodal shape. RQ 2.3 is tackled here with the inclusion of multiple environmental features in the discrete collective estimation scenario, which create high dimensional decision spaces with more complex quality distributions.

Chapters 7 and 8 deal with the last research objective of exploring ways to effectively handle large decision spaces by restricting their sizes in collective decision-making problems. This is done in two independent scenarios. Chapter 7 answers RQ 3.1 by investigating the benefits and costs of restricting the decision space size via enforcing a strict ranking among the considered options by the agents in the collective preference learning scenario. Here, the constraint of having strict ranking makes the discrete decision spaces more sparse and novel mechanisms to update agents' opinions need to be proposed to enable their search for the optimum. Chapter 8 deals with RQ 3.2 and investigates how dynamic decision space restriction can improve the performances of embodied evolution in a task allocation scenario and generate specialized behaviors when facing multimodal quality functions.

Finally, Chapter 9 concludes the thesis and comments on possible future research directions based on the findings presented.

Chapter 2

State-of-the-Art of Collective Decision-Making Research

The existing approaches for achieving collective decision-making behavior in artificial intelligent swarms differ greatly from each other in different decision-making scenarios. In this chapter, state-of-the-art artificial collective decision-making approaches are summarized. The focus of this literature review is on consensus-forming techniques in swarm intelligence. The presented approaches are categorized according to their origin and the decision space sizes they are designed for. In addition, task allocation approaches in swarm intelligence are also introduced, with the emphasis placed on finding optimal specialized behaviors in large decision spaces.

2.1 Consensus Forming in Swarm Systems

An important collective behavior to ensure coherent performances in an artificial intelligent swarm is consensus forming. It represents the collective aggregation of opinions from the sparsely connected swarm agents into a consensus that best suits the information observed by all agents, who have only access to inaccurate or incomplete information individually. In this section, the existing approaches to distributed consensus forming are covered.

2.1.1 Bio-Inspired Consensus Forming in Discrete Decision Spaces

Among collective consensus-forming problems, best-of-n problems refer to discrete collective decision-making scenarios where the agents are tasked with collectively selecting a single option from n potential ones [VFD17]. An option is characterized by its quality and cost, which determines its optimality as the converged result and the difficulty of discovery by the agents, respectively. The goal of the agents in such scenarios is to converge onto the option with the highest associated quality.

The simplest agents that can handle such a decision-making task are assumed to be able to detect the quality of a single option at a control loop. This is the environmental cognitive capability possessed by individual agents in many natural intelligent swarms, such as ants during pathfinding behaviors [Gos+89] and honey bees during nest selection behaviors [SB01]. The consensus-forming approaches used by natural swarm agents give rise to opinion-based consensus strategies. Under such strategies, each agent possesses an explicitly chosen option at a given point in time. The agents modify their opinions collectively via exploration in the environment to verify the associated qualities of their chosen options. They then disseminate and advertise their opinions among their peers to encourage consensus in the swarm.

As many of the classical collective decision-making strategies have been inspired by the self-organizing behaviors of natural intelligent swarms, the prominent artificial decision-making scenarios have been constructed with close similarity to the natural ones. The two most common artificial decision-making scenarios involving mobile agents are shortest path finding [SC06; SC08; Mon+11; Gar+13; Sch+16] and optimal site selection [SMC06; Sch+09b; Sch+09a; PZ09; PZ11].

In shortest path-finding scenarios, the potential paths are only differentiated in terms of the time taken for the agents to traverse through them. Decision-making strategies proposed for shortest path-finding scenarios thus take advantage of the different option costs and allow the agents favoring shorter paths to have more opportunities to advertise their opinions to their peers in the nest area, eventually leading to convergence to the shortest path for all agents in the swarm [Mon+11; Sch+16].

In optimal site selection scenarios, the potential sites are differentiated in terms of both their quality and their discoverability from the nest area, referred to as option quality and option cost respectively. When the differences in both characteristics lead to the same optimal site, they are referred to as synergetic, and the same opinion-based collective decision-making strategies can be employed. Many decision-making strategies in this category carry over the design of scaling the dissemination frequencies of agents to their option's optimality from shortest path-finding scenarios, with the modification of the dissemination frequency being dependent on the option quality instead of the option cost. These strategies include Weighted Voter Model [VHD14] where the agent switches to a random opinion collected during dissemination, and Majority Rule [VHD15; Val+16a] where the agent switches to the majority opinion collected. In site selection scenarios where the differences in option quality and option cost are antagonistic, and lead to different optimal sites, the options with better qualities need to be differentiated in the decision-making process in a way that is independent of option cost. The common way to achieve this in opinion-based strategies without direct communication of option quality is to let robots committed to a particular decision abandon it spontaneously to prevent an early convergence to the option with low exploration cost, but also low associated quality [Cam+11;

Rei+].

Collective perception is a more recently proposed best-of-2 collective decision-making scenario [Val+16b]. The classical binary collective perception scenario consists of an arena filled with black and white colored tiles. A group of mobile robots roams the arena, aiming to determine the color occupying a greater proportion of the arena. It is distinguished from the aforementioned optimal site selection or shortest path finding in two important ways. Firstly, since the options of color black or white being in the majority are linked to global attributes of the environment, their associated qualities are not independently verifiable by the agents, who can only estimate their values via local environmental exploration. Secondly, there is a strong correlation between the qualities of the two options, as an increase in the likelihood of one color being in the majority decreases the likelihood of the other. In addition, as opposed to providing a distinction between the discoverability of the different options in site selection scenarios, collective perception scenarios can input different environmental evidence to individual agents via different feature patterns in the experimental environment [BM19]. This enables testing collective decision-making strategies in terms of their abilities to bring divergent agents in a swarm into consensus.

Earlier approaches to performing collective perception employed opinion-based decision-making strategies that were designed for site selection or path-finding scenarios, such as the Weighted Voter Model and Majority Rule [Val+16b; BM19]. Thus, these strategies carry over the design choice that separates the agents' behavior into alternating exploration and dissemination phases. During exploration phases, the agents perform random walks in the arena and sample the color of the arena floor beneath themselves to compute the frequency of observing the same color as the opinion they are currently holding, as an estimate of the fill ratio and thus option quality. On the other hand, during the dissemination phases, the agent broadcasts its own opinion to its peers to recruit them, with the strength of advertising corresponding to the computed quality. as well as the modulation of the lengths of dissemination phases as an indication of the current option quality. Since environmental exploration and opinion dissemination behaviors take place in the same physical area in the experimental environment, some later implementations of the collective perception problem allow the agents to perform these two tasks in parallel [Ebe+20; Aus+22].

When the agents in a swarm can only detect and consider the quality of a single option at a time, the number of options the swarm covers is constrained by the number of agents available. Therefore, if the swarm is facing a more complex environment with more discrete options, more agents need to be deployed; otherwise, the swarm is only able to process a smaller subset of all available options simultaneously. This effect is especially prevalent when there is a strong correlation between the qualities of options that are close to each other in the decision space, as the microscopic view on single options ignores inter-option correlations and leads to a waste of information.

2.1.2 Discrete Multi-Option Consensus Forming with Parallel Option Consideration

The constraint in scalability faced by bio-inspired consensus strategies in large decision spaces limits their applicability in more complex scenarios. This gives rise to belief fusion approaches to achieving distributed discrete consensus in artificial agents. In such approaches, individual agents attempt to compute the option qualities via direct environmental interactions and opinion pooling with their peers. For example, Bayesian belief fusion has been used for distributed consensus forming in sensor networks [Ala+04; Olf+06].

The same technique can be applied to swarm agents with random and sparse communications with each other. In best-of-n problems, consensus on the optimal option can be achieved by probabilistic inference on the likelihood of each option to be optimal based on past observations. This has been employed in site selection scenarios with circularly distributed sites, where the agents converge on the optimal site by performing pairwise comparisons of neighboring sites and opinion pooling of the computed likelihoods [LLW18a; LLW18b]. Here, the likelihoods associated with all options can be computed even if each agent is only able to observe a subset of them. The opinion pooling approach has also been shown to be able to handle noise in sensor readings. On the other hand, the effectiveness of probabilistic inference depends on the strength of correlation among the option qualities. Thus, the decision-making process when employing probabilistic representation of option qualities is negatively affected by the existence of local optima in quality distribution, as they complicate option quality correlation [LLW21].

In collective perception scenarios, due to the stronger correlation between the option qualities, probabilistic modeling of the quality function is more applicable. The earliest attempt used a Bayesian statistics-based quality computation for the likelihood of the two colors to be in the majority given the observations made [Ebe+20]. Probabilistic modeling makes it possible for the swarm agents to process multiple features in parallel in multi-feature collective perception [BM21] as opposed to tackling the features sequentially [EGN18].

Besides directly exchanging the exact qualities of discrete potential options, their relative optimality can also be expressed using their rankings in the order of decreasing favorability by the agents. Reaching a collective decision from a set of rankings can be done via centralized tallying as in real-life elections [San21] or via other rank aggregation approaches [Lin10]. On the other hand, decentralized iterative voting has been proposed as a model for opinion aggregation in social networks [Has+13; Bri+16]. Such models describe a distributed consensus process where an agent on a network iteratively holds localized elections among its immediate neighbors. The agent then modifies its own opinion based on their inputs. This process continues until the social network reaches an equilibrium, which can be a consensus of opinions under suitable parameter settings.

2.1.3 Consensus Forming in Continuous Decision Spaces

Modeling the distribution function of the option qualities becomes more important for collective consensus-forming scenarios in continuous decision spaces, which makes the enumeration of potential options impossible. A long-standing subject of study in this area is the production of coherent movement in mobile multi-agent systems. A common inspiration comes from the flocking behaviors in natural intelligent swarms [Olf06]. Consensus forming in terms of collective motion has been studied in multi-agent systems with different dynamics and communication topologies [AB22]. Studies on self-organized flocking further relax the assumption of the communication topology from a dynamic network spanning all agents into random and sparse communication between swarm robots [Tur+08; Fer+12]. A collective control strategy for such systems needs to ensure that the agents stay in proximity with each other, avoid collisions, and match the agents' velocities [Rey87]. Using the framework of collective decision making from the perspective of a single agent, this consensus problem can be interpreted to have a unimodal quality function in the decision space, where the more the agent deviates from the mean velocity of the flock, the lower the quality of its current velocity decision. A classical approach to reaching consensus in such problems is the Linear Consensus Protocol (LCP) [OM04]. This approach applies an acceleration that is linearly proportional to the deviation of an agent's velocity from the mean velocity of its neighbors.

The same approach can also be used in environmental cognition scenarios to reach a consensus regarding environmental features in a continuous decision space. The binary collective perception scenario can be extended with a continuous decision space into the collective estimation scenario, where the agents instead form a consensus regarding the exact concentration of a particular feature in the environment [SCD18; SCD20; RHR21; RRH23]. To this end, individual agents independently compute the concentration of environmental features they observe in their local environment. They then modify their estimations using a consensus protocol to bring them closer to the mean value computed by their neighbors. The unimodality of the quality function is similarly exploited as in collective motion problems.

2.2 Distributed Task Allocation in Swarm Systems

On the other hand, distributed task allocation problems relax the requirement of the swarm agents to reach the same decision at convergence, thus introducing multimodal quality functions from the perspective of the agents. The increased problem complexity makes a decentralized solution more challenging, and hence most established task allocation strategies either require a centralized communication hub or a densely connected communication network between the agents [KHE15; Jia16]. Such multi-agent approaches include coalition formation [SK98; VA07] and market-based approaches [Dia+06].

In contrast, natural swarm intelligence systems, such as insect colonies, are able to produce specialized behaviors among their agents with only local, random, and sparse communications. The underlying decision process is still being studied and not fully understood [CM09; LD19]. The classical approach to modeling the task allocation behaviors of natural intelligent swarms is the Response Threshold Model [BTD96]. It expresses the interactions between the environment and the agents during task allocation by having the agents respond to task-specific stimuli from the environment. If the stimulus exceeds the corresponding threshold, the agent has a high chance to engage in a task and vice versa. The different response thresholds of the agents are caused by evolution and the agents' prior experience [TN04; Wes+13]. This model gives rise to embodied evolutionary approaches to the creation of specialized behaviors in artificial multi-agent systems [MCB16]. The artificial distributed evolutionary process can produce specialized behaviors via reproductive isolation between agents.

A minority of existing studies investigate task allocation in natural intelligent swarms via interaction among the agents [GGT92; PGG96]. Recently, there has been a growing interest in such models [Gor16]. Notably, it has been identified in both empirical studies and simulations that the reward functions faced by the swarm agents produce specialization via an online optimization process [CB13; CMG20]. This model brings the task allocation problem closer to the existing literature in collective decision making and highlights the importance of quality distribution in the decision space. The problem of task allocation in intelligent swarms is further complicated by collective tasks whose rewards depend on the configuration of multiple agents, making mapping out the shape of the quality function from the perspective of a single agent difficult.

Chapter 3

Bayesian Statistics-Based Environmental Exploration and Application in the Best-of-2 Collective Perception Scenario

This chapter is based on [SM20].

This chapter starts from the best-of-2 collective perception scenario and introduces the Distributed Bayesian Hypothesis Testing approach to performing environmental classification on a decentralized platform. The proposed approach uses Bayesian statistical inference to compute the likelihood of different hypotheses regarding the environment based on collected observations. It is compared with the existing opinion-based collective consensus approaches in terms of performance and serves as the foundation for subsequent multi-option collective consensus strategies with parallel encoding of option preferences.

3.1 The Best-of-2 Collective Perception Scenario

The investigated collective perception scenario [Val+16b] is illustrated in Figure 3.1. The environment is filled with two features, represented by the colors black and white on the arena floor. To enable easy generation of experimental environments, the arena is discretized into tiles. Each tile can have one of the two features. The proportion of the arena surface filled by one of the features n is referred to as the fill ratio r_n. A group of mobile robots, marked in red in Figure 3.1, roam the arena. Their goal is to collectively determine which feature of the two covers the majority of area in the environment. Each robot has only simple reactive behavior and can only detect the presence of the features in its immediate vicinity. Therefore, they need to cooperate and pool the information gained from each individual to accurately decide on the majority feature in the global environment.

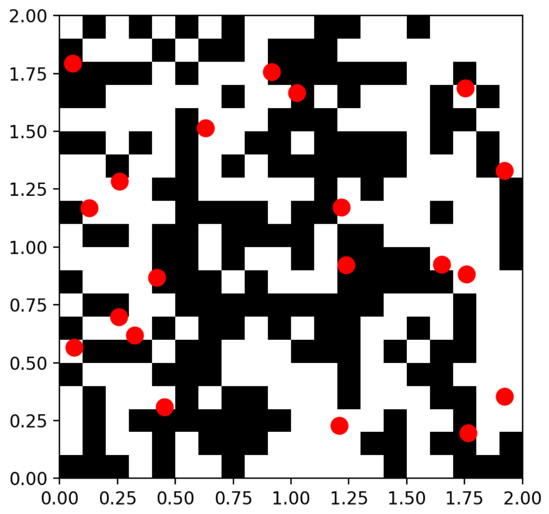


Figure 3.1: Example of the collective perception scenario; the arena is a $2m \times 2m$ square, covered in 400 square tiles shown in black and white; the red dots illustrate 20 mobile robots [SM22]

The collective perception scenario is a best-of-n collective decision-making problem with 2 options. The option qualities of the 2 features are their fill ratios. Since the fill ratios are global qualities and not directly measurable by the individual robots, they estimate the true fill ratios using the ratios of features they encounter during individual environmental exploration. This causes random variations between the information gained by different individuals and is the main obstacle in the collective decision-making process.

The performance of the robot collective is measured by the accuracy of the final decision and the time required to reach that decision. The final decision is more prone to inaccuracies and takes a longer time if the random variations between local information from the perspective of the individual robots overwhelm the difference in the true fill ratios of the two features. The relative significance of the random variations compared to the true fill ratio difference can be adjusted in two ways. Firstly, the closer the true fill ratio of the majority feature is to 0.5, the more difficult it becomes to distinguish them and reach the correct decision. Secondly, the variations in local information can be magnified by introducing clustered distribution of environmental features, which causes the robots to have different estimates of the global fill ratio [BM19].

3.2 Distributed Bayesian Hypothesis Testing

In this subsection, the proposed Distributed Bayesian Hypothesis Testing (DBHT) approach to collective perception is introduced. The mathematical background is presented first, and then the control mechanism of information exchange in the swarm is shown.

3.2.1 Low-Level Control Mechanisms for Environmental Perception

In order to thoroughly survey the environment, the robots are controlled by a low-level control program to perform a random walk routine in the experimental arena, as shown in Table 3.1. A robot alternates between two motion states A and B. During state A, the robot moves forward in a straight line; while during state B, the robot rotates at the same position in a random direction. The lengths of the two states are randomly distributed. The durations of state A are exponentially distributed with a mean of $40s$, while the durations of state B are uniformly distributed between $0s$ and $4.5s$. The robots are assumed to be able to detect obstacles $0.1m$ located in front of them. To avoid collisions, during state A, if an obstacle is detected, the robot enters state B. At the end of state B, if an obstacle is detected, the robot repeats state B with a new timer.

Table 3.1: Low-level control mechanism used to implement random walk [SM22]

Motion	Transition condition
A. Moving forward	Timer exp(40)s or obstacle detected
B. Turn in random direction	Timer $\mathcal{U}(0, 4.5)$s

3.2.2 Bayesian Inference Approach for Evidence Collection

In the proposed DBHT approach, the quality of one of the two feature options is computed as the probability of the fill ratio r being higher than 0.5. For generalization, one of the features is denoted as n out of $N = 2$ features:

$$q(n) = P(r_n > 0.5). \tag{3.1}$$

The environmental exploration is performed individually by the robots, that make repetitive observations and detect the features present in their current positions. The robots perform random walk routines in the arena to ensure a bias-free collection of observations. Given a series of observations made on the environment by an individual robot $\vec{ob} = [ob_1 ... ob_S]$, the probability of feature n being in the majority can be computed as the sum of the likelihood of the fill ratio being at every value between 0.5 and 1:

$$P(r_n > 0.5) = \int_{0.5}^{1} P(r_n = r'|\vec{ob})dr'. \tag{3.2}$$

If the space of possible fill ratios is discretized uniformly to an array of M fill ratio hypotheses $\vec{h}$, with h_m denoting the m^{th} discretized value, the integral can be approximated using the sum of likelihoods at h values greater than 0.5:

$$P(r_n > 0.5) \approx \sum_{m \forall h_m > 0.5} P(r_n = h_m|\vec{ob}). \tag{3.3}$$

This likelihood expression can be computed using Bayes' rule:

$$P(r_n = h_m|\vec{ob}) = \frac{P(\vec{ob}|r_n = h_m)P(r_n = h_m)}{P(\vec{ob})}. \tag{3.4}$$

Here $P(r_n = h_m)$ is the prior and $P(\vec{ob})$ is the marginal likelihood, both of which are assumed to be the same for all hypotheses. The chain rule can then be applied to $P(\vec{ob}|r_n = h)$ and obtain

$$P(\vec{ob}|r_n = h_m) = P(ob_1|r_n = h_m)P(ob_2|r_n = h_m, ob_1)...P(ob_S|r_n = h_m, ob_1, ..., ob_{\text{S-1}}). \tag{3.5}$$

Assuming the observations are all independent of each other, the dependencies on previous observations can be removed, and thus:

$$P(\vec{ob}|r_n = h_m) \approx P(ob_1|r_n = h_m)P(ob_2|r_n = h_m)...P(ob_S|r_n = h_m). \tag{3.6}$$

Assuming the robot is at a random position in the arena, the probability of making a particular observation is simply the fill ratio hypothesis h_m if the observed feature is the same as the feature n or $(1 - h_m)$ otherwise:

$$P(ob_s|r_n = h_m) = \{ \begin{matrix} \text{if } ob_s = n\text{: } h_m \\ \text{if } ob_s \neq n\text{: } (1 - h_m) \end{matrix}. \tag{3.7}$$

The likelihood array of the considered hypotheses $\vec{l}$ has a size of M and is computed by taking the elementwise products of the hypothesis values that correspond to the feature observed as follows:

$$\vec{l} = \begin{bmatrix} P(r_n = h_1|\vec{ob}) \\ P(r_n = h_2|\vec{ob}) \\ ... \\ P(r_n = h_M|\vec{ob}) \end{bmatrix} = \prod_{s=1..S} \{ \begin{bmatrix} h_1 & 1 - h_1 \\ h_2 & 1 - h_2 \\ ... & \\ h_M & 1 - h_M \end{bmatrix} \cdot \begin{bmatrix} ob_s = n \\ ob_s \neq n \end{bmatrix} \} = \prod_{s=1..S} (H \cdot \vec{o}_s). \tag{3.8}$$

Here H is the matrix of size $M \times 2$ containing all hypotheses and $\vec{o}_s$ is the vector describing the s^{th} observation and is $\begin{bmatrix}1\\0\end{bmatrix}$ when $ob_s = n$ and $\begin{bmatrix}0\\1\end{bmatrix}$ when $ob_s \neq n$. Every entry in $\vec{l^*}$ corresponds to a hypothesis in H. This operation can be easily decentralized, where each robot i computes its individual likelihood estimations, and the information is aggregated to form a collective result:

$$\vec{l^*} = \prod_i \vec{l}_{i,S_i} \text{ where } \vec{l}_{i,S_i} = \prod_{s_i=1..S_i} (H \cdot \vec{o}_{s_i}). \tag{3.9}$$

Here i iterates through the indices of the robots and s_i iterates through the observations made by robot i. In implementation, the likelihood arrays have to be normalized after each multiplication to avoid underflow. For the collective perception task, the swarm can decide on the majority feature n with enough certainty if the sum of likelihoods over the range $r_n > 0.5$ is over a threshold P_{th} after normalization.

Numerically, the decision space of possible fill ratios is discretized with $M = 10$ evenly space values. The hypothesis matrix for the feature represented by the color black in the arena is therefore as follows:

$$H = \begin{bmatrix} 0.05 & 0.95 \\ 0.15 & 0.85 \\ \cdots & \\ 0.95 & 0.05 \end{bmatrix}. \tag{3.10}$$

An observation is recorded as $\vec{o} = \begin{bmatrix}1\\0\end{bmatrix}$ when the observed color is black and as $\vec{o} = \begin{bmatrix}0\\1\end{bmatrix}$ when the observed color is white. The hypothesis likelihood array $\vec{l}$ is initialized with identical values in every entry, meaning that no information has been gathered, and the hypotheses cannot be distinguished in terms of likelihoods:

$$\vec{l}_{i,0} = \begin{bmatrix}0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1 & 0.1\end{bmatrix}^T. \tag{3.11}$$

During the process of environmental exploration, the likelihoods are updated iteratively after each new observation:

$$\vec{l}_{i,s_i} = \vec{l}_{i,s_i-1} \circ (H \cdot \vec{ob}_{s_i}). \tag{3.12}$$

3.2.3 Information Aggregation in Robot Swarm

Fusion of the individual opinions is done through a communication network with a tree topology. To construct such a topology, each robot's behavior is designed as a finite state machine with 4 states, as shown in Algorithm 1.

Algorithm 1 Distributed Bayesian Hypothesis Testing (DBHT) [SM20]

```
Input: Hypothesis matrix $H$ of size $M \times 2$; robot index $i \in 1..N_{robot}$, when robot is the leader $i = 1$
Output: Majority feature index $n^*$
1: Initialize $\vec{l_i}$ with respect to $M$; $State_i = 1$; $t = 0$; $G_i = \{\}$
2: Define $T_S$, $T_D$, $Dist_{max}$, $Neigh_{max}$
3: while $0.01 \leq \sum_{m=1..5} l^*_m < 0.99$ do
4:     if $State_i = 1$ then
5:         Perform random walk in the arena
6:         if $t\%T_S = 0$ then
7:             Collect observation and compute $\vec{o}$
8:             $\vec{l_i} = \vec{l_i} \circ (H \cdot \vec{o})$
9:         if $t\%T_D = 0$ and $i = 1$ then
10:            $State_i = 2$
11:    else if $State_i = 2$ then
12:        for $j = 1..N_{robot}$ do
13:            if $Dist_{i,j} < Dist_{max}$ & $State_j = 1$ & $i \neq j$ & $|G_i| < Neigh_{max}$ then
14:                $G_i = G_i \cup j$; $G_j = G_j \cup i$
15:                $State_j = 2$
16:    else if $State_i = 3$ then
17:        if $i = 1$ then
18:            if Messages from all neighbors received then
19:                $\vec{l^*} = \text{Normalize}(\vec{l_1} \circ \prod_{j \in G_1} Message_{j \to 1})$
20:                if $\sum_{m=1..5} l^*_m \geq 0.99$ then
21:                    $n^* = 1$
22:                else if $\sum_{m=1..5} l^*_m < 0.01$ then
23:                    $n^* = 2$
24:                $State_i = 4$
25:        else
26:            if Messages from all neighbors received then
27:                $Message_{i \to j} = \text{Normalize}(\vec{l_i} \circ \prod_{k \in G_i} Message_{k \to i})$
28:                $State_i = 4$
29:    else
30:        if $i = 1$ or $\exists j \in G_i : State_j = 1$ then
31:            $State_i = 1$; $G_i = \{\}$
```

One robot is designated as the leader, which is tasked with both exploring its immediate vicinity and collecting the opinions of other robots. The leader can be chosen by the user or elected in a self-organized way, e.g. as in [TVD18], before the decision-making process. In this chapter, the robot with index 1 is designated as the leader. All other robots are also assigned a unique index.

All robots start in state 1 (Algorithm 1 lines 4-10), where they perform a random walk routine and modify their own opinion by sampling the color of the ground beneath themselves. Sampling is done periodically with the interval T_S. An individual robot's opinion is computed iteratively, thus only the opinion after the last sampling needs to be stored, making the memory and computation complexity of the proposed approach $O(Neigh_{max}M)$, where $Neigh_{\text{max}}$ is the maximum number of neighbors a robot can have and M is the number of hypotheses. At every dissemination interval T_D, the leader will start a dissemination session. It switches to state 2 (Algorithm 1 lines 11-15), stops moving, and sends out signals to look for robots nearby. Only robots within communication distance that are in state 0 will respond, to ensure that the final communication topology has a tree structure. They will establish a connection with the leader and switch to state 2 too. They will also stop moving and send out signals to look for neighbors themselves, and the process continues.

After searching for neighbors, a robot will go into state 3 (Algorithm 1 lines 16-24) and pass the likelihood estimates in the network as messages. Message from robot n_1 to n_2 is an array of M numbers and is defined as follows:

$$Message_{i \to j} = \text{Normalize}(\vec{l_i} \circ \prod_{k \in G_i} Message_{k \to i}) \quad (i \neq 1). \tag{3.13}$$

Messages are normalized before being sent to avoid underflow. Once a robot sends its message, it switches to state 4 and rests. Message passing starts from the leave nodes and gradually converges towards the leader. The leader will compute the estimate of the whole swarm as follows:

$$\vec{l^*} = \text{Normalize}(\vec{l_1} \circ \prod_{j \in G_1} Message_{j \to 1}). \tag{3.14}$$

Once $\vec{l^*}$ is computed, the leader will switch back to state 1 and send out signals to its neighbors (Algorithm 1 lines 30-31). The other robots will switch to state 1 if one of their neighbors is in state 1. The process continues until all robots involved in the communication network are back in state 1. Communications thus stop and will start again at the next dissemination session. There is therefore no need for the robots to be in constant communication with each other. A new communication network will be constructed at every new dissemination session. An example of robot states during a dissemination session is shown in Figure 3.2.

The algorithm stops when one of the hypotheses has a normalized likelihood of at least 0.99. The leader can then report the result to the user or direct the swarm to perform other tasks dependent on the decision.

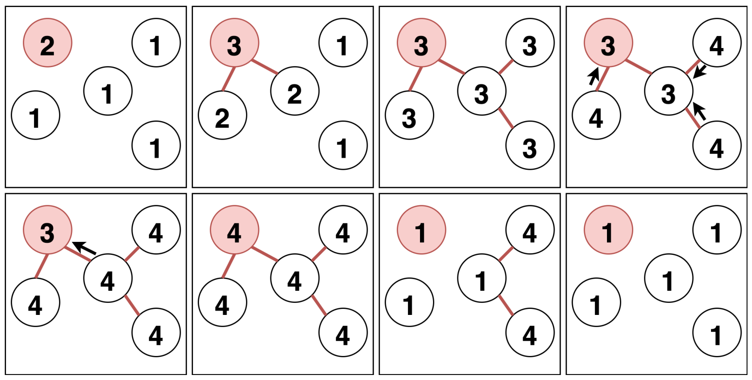


Figure 3.2: Illustration of states (numbers), communication links, and message passing of the robots during dissemination; the leader is marked in red. [SM20]

3.3 Baseline Opinion-Based Collective Perception Strategies

In order to gauge the viability of the proposed DBHT approach to the collective perception task, 3 existing opinion-based approaches to collective perception are employed as a baseline [Val+16b]: Direct Modulation of Voter-based Decisions (DMVD), Direct Modulation of Majority-based Decisions (DMMD) and Direct Comparison (DC).

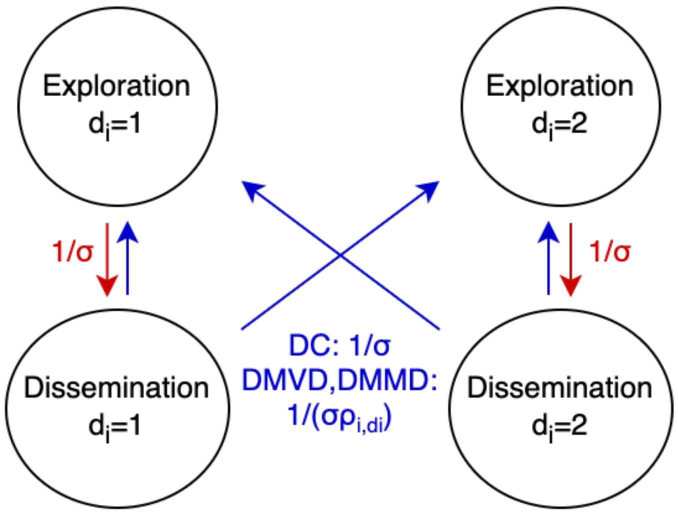


Figure 3.3: Probabilistic finite state machine showing the decision-making behavior of robot i under the considered opinion-based strategies in the best-of-2 problem

The robot's behaviors in all 3 opinion-based strategies are implemented as a probabilistic finite state machine, shown in Figure 3.3. Each robot i has an explicitly chosen decision d_i at a given point in time. It represents the feature that the robot thinks covers the majority of the environment. The robot's decision-making behavior alternates between the exploration and dissemination states. In the exploration states, the robot performs random walks and collects observations on the environment at every control loop. It records the feature present at its current location, and subsequently computes its own estimation of the quality of its current decision $\rho_i \in (0, 1]$ as the proportion of collected observations that correspond to its current decision d_i. The robot has a fixed probability of $1/\sigma$ per second during exploration states to switch to the dissemination states. This makes the lengths of the exploration states exponentially distributed, with the mean length being σ seconds.

During dissemination states, the robot continues to perform random walks to ensure mixture with its peers and propagation of information. The considered opinion-based strategies are distinguished from each other in their decision-making mechanisms in the dissemination states. Under DC strategy, the robot broadcasts its current chosen decision d_i and the computed associated quality ρ_{i,d_i}, while receiving the same information from its peers. The robot has the same transition probability of $1/\sigma$ per second to switch back to exploration states. During this transition, it computes its newly chosen decision as the one with the highest associated quality among its recorded decisions from its peers as well as its own decision.

On the other hand, under DMVD and DMMD strategies, the robot broadcasts only its chosen decision d_i to its peers and also only records the decisions of other robots. In order to ensure convergence to the decision with the optimal associated quality, the probability of transitioning from dissemination states to exploration states is modulated by the computed quality of the current associated quality ρ_{i,d_i}. Under DMVD strategy, the robot switches to the received decision of a random neighbor [VHD14], while under DMMD strategy, the robot conducts a local majority voting among its received decisions and switches to the winning decision [VHD15].

3.4 Experiments and Results

The experiments are conducted on 20 simulated e-puck robots [Mon+09] in a $2m \times 2m$ arena. The arena is divided into 400 square tiles filled with one of two colors representing the two features. The robots have a linear speed of $0.16m/s$ and a rotational speed of $0.75rad/s$. E-pucks can only perceive up to 5 neighbors simultaneously [Fra+12] and can communicate with up to 7 neighbors [Mon+09]. Therefore, for the proposed DBHT strategy, the maximum setting for the number of allowed neighbors $Neigh_{max}$ is 5. In addition, they cannot receive multiple messages simultaneously [SKG16], therefore setting $Neigh_{max}$ to 2 greatly reduces the probability of communication failure. Both parameter choices have been tested in subsequent experiments. For the baseline opinion-based strategies, the parameter σ is set to 10, following previous works on collective perception [Val+16b; SCD18].

3.4.1 Finding the Optimal Sampling and Dissemination Interval for the Proposed DBHT Strategy

The first set of experiments aims to determine the optimal sampling and dissemination interval, T_S and T_D, for the proposed DBHT strategy. It is expected that reducing the sampling interval could provide more samples per unit time. However, these samples will be collected closer to each other, and thus highly correlated. It then means that the independence assumption used in Equation 3.6 will not closely reflect reality. Therefore, the algorithm will produce less accurate results. On the other hand, increasing the dissemination interval means it takes longer on average for the leader to connect to a large number of robots. However, the robots can travel for a longer distance and collect more samples without violating the independence assumption during the time. Therefore, the result will be more accurate. To determine the optimal values for T_S and T_D, 100 tests are run with randomly distributed white tiles of proportions: 0.55, 0.65, 0.75, 0.85, 0.95. The average error and consensus time are compared across different T_S and T_D settings. Error is defined as the difference between the true proportion of white tiles and the computed proportion by the swarm.

Table 3.2: Average error using different sampling and dissemination intervals [SM20]

/10%	Sampling Interval T_S /s					
Dissem Interval T_D /s	0.1	0.2	0.5	1	2	5
1	0.3	0.216	0.072	0.032	0.026	0.036
2	0.278	0.204	0.064	0.038	0.022	0.028
5	0.208	0.124	0.044	0.028	0.018	0.024
10	0.17	0.128	0.052	0.01	0.018	0.024
20	0.094	0.076	0.026	0.026	0.016	0.006

Table 3.3: Average decision time using different sampling and dissemination intervals [SM20]

/s	Sampling Interval T_S /s					
Dissem Interval T_D /s	0.1	0.2	0.5	1	2	5
1	3.09	4.75	9.53	15.80	29.66	65.82
2	3.84	6.13	11.08	18.17	32.72	69.62
5	7.36	9.22	14.98	23.11	39.08	80.05
10	12.43	14.59	21.06	30.06	45.12	87.06
20	22.23	23.87	30.07	39.95	57.40	103.64

The results (Tables 3.2 and 3.3) show that as T_S increases from $0.1s$ to $1s$, there is a significant decrease in error. This is because the speed of the robots is $0.16m/s$, thus for $1s$, a robot would have traveled for $0.16m$. This is a bit bigger than the width ($0.1m$) and diagonal length ($0.141m$) of an individual tile, meaning that when collecting the next sample, the robot would have moved to the neighboring tile. Since the distribution of tiles is random, there is a very weak correlation between the colors of adjacent tiles, thus moving to the neighboring tile is enough to reduce the correlation to near zero. This is why when T_S is beyond $1s$, increasing it no longer reduces the error much but still increases the decision time. T_S thus has an optimal value of $1s$. This also agrees with the findings of Ebert et al. [Ebe+20], that collecting less correlated samples sparsely can improve decision-making accuracy. However, when T_S is $5s$, there is an increase in error compared to $2s$, since too high a T_S means too few samples would be collected, thus impeding accurate decision making. At the same time, increasing T_D provides moderate improvements in accuracy and in exchange, a moderate increase in decision time. A trade-off needs to be considered when applying the proposed approach to a real problem. Here, a T_D of $5s$ is chosen for later experiments.

3.4.2 Viability of the Proposed DBHT Approach Compared to the Baseline Collective Perception Approaches

To determine the effectiveness of the proposed DBHT strategy and compare it to other state-of-the-art consensus strategies, simulations are performed for all considered strategies at different decision difficulties, computed as the proportion of black and white features' fill ratios ($\rho_b^* = r_{black}/r_{white}$), as well as different patterns of feature distribution. For difficulty, the same test cases are used as in [SCD18; BM19], ρ_b^*=[0.52, 0.56, 0.61, 0.67, 0.72, 0.79, 0.85, 0.92]. The feature 1, represented by the color white, is always kept in the majority. The patterns are selected from the patterns used for matrix visualization and classified according to entropy (E_c) and Moran index (MI) as in [BM19]. E_c describes the densities of clusters in the pattern and MI describes the level of connectivity between clusters. For every ρ_b^* and pattern, the simulation is run 100 times. The performances of the considered strategies are measured by the exit probability, which is the probability that the swarm comes to the correct decision that the feature 1 is in the majority.

The test results are shown in Figure 3.4. The first row shows an example of each pattern tested. The second row shows the exit probability for every strategy considered. The shaded area indicates the standard deviation of the measurement of exit probability. It is computed theoretically by treating the exit probability as the mean of 100 Bernoulli trials. Thus the standard deviation is $\sqrt{p(1-p)/100}$, where p is the measured exit probability. The third row shows the mean decision time for every strategy considered. The shaded area indicates the standard deviation of all the samples.

It can be observed that the maximum neighbor limit only has a small impact on the performance of the DBHT strategy. Exit probability, when $Neigh_{max}$ is 2 and 5, is usually very close. Decision time is slightly longer when the limit is 2.

Random ($E_c \approx 0.5, MI \approx 0$) pattern is the most studied pattern in previous works. All considered algorithms have comparable performance when ρ_b^* is low, with DC having the lowest decision time. As ρ_b^* increases, DMVD and DMMD have a significant drop in accuracy and a rise in decision time. The accuracy of DBHT and DC is more resilient to

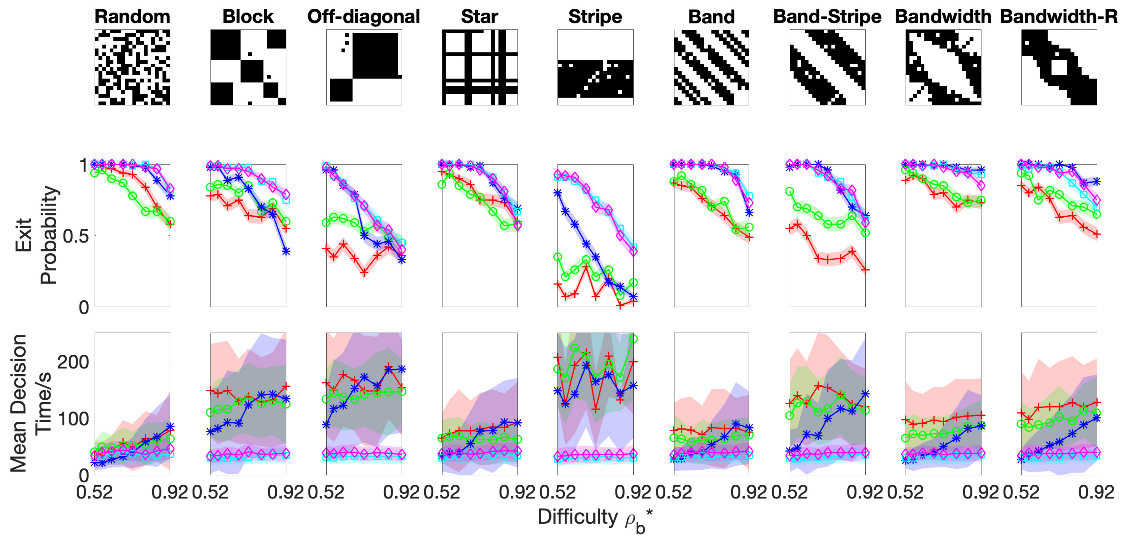


Figure 3.4: Exit probability and decision time for all strategies in 9 patterns over various difficulties; Red+:DMVD, Green○:DMMD, Blue∗:DC, Cyan□:DBHT $Neigh_{max} = 5$, Magenta◇ :DBHT $Neigh_{max} = 2$ [SM20]

the difficulty increase, however, DC also has a significant increase in decision time.

Star ($E_c \approx 0.8, MI \approx 0.4$) and **Band** ($E_c \approx 0.7, MI \approx 0.3$) pattern are observed to be more challenging than the random pattern for collective perception. DMVD and DMMD have lower accuracy and higher decision time compared to DC and DBHT. DC and DBHT are comparable in accuracy. They are also comparable in decision time when the difficulty is low, but DC's decision time increases significantly when the difficulty is high.

Bandwidth ($E_c \approx 0.9, MI \approx 0.6$) and **Bandwidth-R** ($E_c \approx 1, MI \approx 0.7$) see DBHT outperforming DMVD and DMMD in both accuracy and decision time. However, its accuracy is not as high as the DC strategy, especially at the highest tested difficulty of 0.92. In terms of decision time, DBHT and DC have similar performance at low difficulty, but the decision time of DC quickly rises as difficulty increases.

Block ($E_c \approx 0.9, MI \approx 0.8$), **Off-diagonal** ($E_c \approx 0.9, MI \approx 0.8$), **Stripe** ($E_c \approx 1, MI \approx 0.8$) and **Band-Stripe** ($E_c \approx 0.9, MI \approx 0.6$) are observed to be the most difficult collective perception scenarios, with highly clustered black tiles. In these scenarios, existing algorithms often become very inaccurate with long and volatile decision time. DBHT is able to outperform existing algorithms in terms of accuracy while maintaining a relatively constant decision time.

Overall, the DBHT strategy is able to produce higher perception accuracy compared to existing algorithms for scenarios with high task difficulty. In terms of decision time, DBHT can be slower than existing algorithms, especially DC, when the task is simple.

However, it is much faster than existing algorithms when the task is difficult both in terms of high ρ_b^* and clustered feature patterns, as the decision time of DBHT is largely independent of the 2 measurements of difficulty. This is because both DMVD and DMMD make use of modulation of positive feedback to enable decision making. Clustering of features forms local echo chambers among robots with similar opinions, making consensus forming within the swarm difficult. For DC, clustering of features can create robots with extreme quality estimation of its own opinion and thus rarely adopt its neighbors' opinion, therefore disrupting decision making of the swarm. In contrast, DBHT's operation is mostly unaffected by the clustering of features. Its opinion fusion is also able to produce a middle ground between robots with highly contrasting estimates. In addition, if collective perception is to be applied in the real world, ρ_b^* and feature patterns are usually unknown. Thus the volatile decision time of the three existing algorithms causes a halting problem. When the algorithm is running for a long time with no clear decision, there is a dilemma whether to keep the algorithm running for an even longer time or to recognize a failed run and restart the process, wasting past progress.

3.4.3 Estimation Accuracy and Effects of Limiting Maximum Neighbors

The performance of the DBHT strategy can also be measured by the difference between the most likely hypothesis computed by the swarm and the correct hypothesis that is closest to the true P_W. Among the test cases, ρ_b^* values of $[0.52, 0.56, 0.61]$ are classified to hypothesis $P_W = 0.65$ and the rest to $P_W = 0.55$. The average errors for all tested scenarios and different $Neigh_{max}$ are shown in the top row in Figure 3.5.

Across all patterns, the error usually spikes at ρ_b^* value of 0.61 or 0.67. These ρ_b^* are in the middle of 2 classes which can cause some error during classification. In addition, the 2 plotted curves are very close to each other, and thus a change in $Neigh_{max}$ in DBHT does not significantly impact the accuracy of collective perception. This is because although a low $Neigh_{max}$ does reduce the average number of opinions that can be collected as shown in the bottom row in Figure 3.5, to meet the likelihood threshold on the final chosen hypothesis of 0.99, the robots have to collect more samples over longer periods of time, as shown in Figure 3.5 (middle row). Therefore, the limit on the maximum number of neighbors gives a trade-off between decision time, design complexity, and robustness of the system.

3.5 Summary and Discussion

In this chapter, Distributed Bayesian Hypothesis Testing (DBHT) has been proposed as a novel collective perception strategy. It has been shown that distributed observations can be

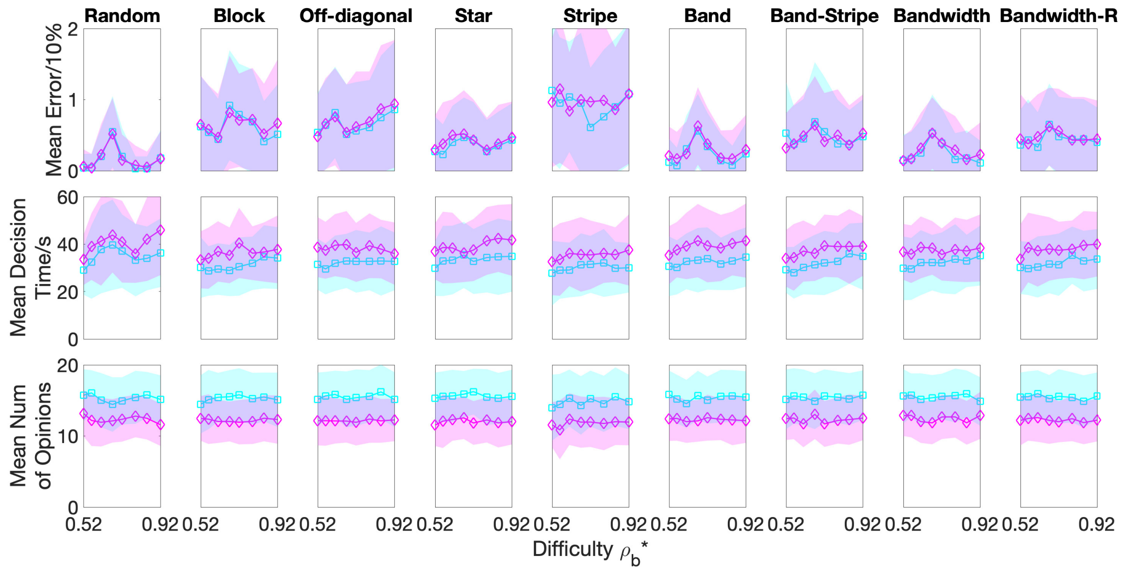


Figure 3.5: Mean error, mean decision time, and mean number of opinions collected for DBHT with different $Neigh_{max}$; Cyan □:DBHT $Neigh_{max} = 5$, Magenta ◇:DBHT $Neigh_{max} = 2$ [SM20]

aggregated centrally using an ad hoc communication network. The performance of DBHT is measured at different sampling and dissemination intervals. It can be concluded that, up to a limit, collecting sparse and uncorrelated samples could increase perception accuracy but also increase the decision time. Changing the dissemination interval presents a similar trade-off. DBHT's performances are then compared with those of 3 other state-of-the-art opinion-based collective decision-making strategies, DMVD, DMMD, and DC, in how well they determine which color is in the majority. It is shown that DBHT often has superior performance due to its resilience in high ρ_b^* and feature patterns with large clusters, as well as its stable decision time regardless of the difficulty of the environment. Finally, DBHT's ability to accurately estimate the proportion of colors is examined, together with the effect of limiting the maximum number of neighbors during dissemination. It is observed that the error not only generally increases with ρ_b^* and pattern difficulty, but also tends to spike around boundary cases between classes. Also, the limit on the maximum number of neighbors does not have a significant impact on estimation accuracy. The decrease in the number of opinions that can be collected is made up by a longer decision time, which means more uncorrelated samples. The ability of the DBHT approach to compute and fuse quality estimates in parallel provides a foundation for subsequent multi-option collective consensus-forming approaches in this thesis.

The main limitation of the proposed DBHT approach is the centralized communication paradigm and the dependence on a leader to collect information from the individual robots. This causes the collective to have a single point of failure and thus limited robustness. The construction of the communication links also requires the robots to have unique indices and thus limits the scalability of the system. In addition, the performances of the considered strategies are compared only at one set of algorithmic parameter settings, without considering the speed versus accuracy trade-off. This reduces the general applicability of the results.

Chapter 4

Multi-Option Collective Consensus Forming with Parallel Encoding of Option Preferences

This chapter is based on [SM21b; SHM21].

Beyond binary environmental classification, the computed likelihoods using the previously proposed Bayesian hypothesis testing can be employed by the swarm to perform more precise collective estimation of global environmental attributes. In order for the swarm to form a consensus on the estimated attribute, the agents need to process multiple potential options with similar levels of validity. Therefore, multi-option collective decision-making strategies that allow for effective pooling of information from individual agents need to be developed and tested.

This chapter builds on top of the Bayesian hypothesis testing approach for individual environmental exploration in collective perception and investigates ways to propagate the computed relative preferences of the options across the swarm in a decentralized way for consensus to be achieved. The proposed consensus-forming strategies are tested in the discrete collective estimation scenario, which is a multi-option extension to the collective perception scenario studied in Chapter 3. In order to account for the accuracy versus speed trade-off in collective consensus processes, the performances of the investigated strategies are gauged using a multi-criteria framework, considering decision error, consensus time, rate of failing to reach consensus, and number of messages passed. Existing opinion-based decentralized consensus-forming strategies are used as a baseline to measure the viability of the proposed decision mechanisms that employ parallel encoding of option preferences.

4.1 Best-of-n Discrete Collective Estimation Scenario

This section presents the details of the discrete collective estimation scenario, together with the generation of experimental environments with different difficulty levels. Subsequently, baseline opinion-based consensus strategies are modified to fit the investigated multi-option scenario.

4.1.1 Scenario Configuration

The environmental design in the discrete collective estimation scenario is unchanged compared to in the aforementioned binary collective perception scenario studied in Chapter 3, with the arena still filled by two features of different ratios. The difference is that, instead of just finding out which feature is in the majority, the robot collective seeks to determine, out of M fill ratio hypotheses h_m discretizing the continuous space $[0, 1]$, the hypothesis index m^* where h_m is the closest to the actual fill ratio of a chosen feature n:

$$m^* = \text{argmin}_m |h_m - r_n|. \tag{4.1}$$

This turns the binary collective perception into a multi-option scenario with a series of potential options and allows for the experimentation of multi-option collective consensus-forming strategies. Such scenario configuration enables the customization of many aspects of the decision-making problem faced by the robot swarm while preserving the randomness of the environment, so that a series of experimental environments with similar key characteristics can be generated to gauge the performances of the investigated consensus strategies accurately. Specifically, the difficulty of the decision-making problem can be adjusted via the pattern of distribution of the two features. At the same time, the relative prevalence of environmental exploration and inter-agent communication can be tuned via the rate of observation collection and message parsing respectively, as well as the range of communication.

4.1.2 Generation of Experimental Environments

The experiments investigating the performances of the considered strategies in the discrete collective estimation scenarios retain the basic environmental designs of the binary collective perception scenarios in the previous section. The experimental arena consists of $L_E \times W_E$ square tiles in black or white color, representing the two features in the environment. Without loss of generality, the robots are tasked with forming a consensus on the fill ratio of the feature represented by black tiles.

In the binary collective perception scenarios, the difficulty of the consensus task can be tuned by the fill ratio of the two features as well as the pattern of feature distribution in the environment. However, in the discrete collective estimation scenario investigated in this chapter, the fill ratio does not have a significant impact on the difficulty of the decision-making task. Therefore, the difficulty level is only tuned via the pattern of feature distribution. As observed in Chapter 3, distribution patterns that place the same feature in a cluster can negatively impact the accuracy of the collective decision-making task. Environments with random feature distribution, where the probability of a feature being in each position is independent of each other, can be easily generated by randomly placing individual tiles with its associated feature in the environment. On the other hand, in order to streamline the generation of environments with concentrated feature distributions, the procedure shown in Algorithm 2 is used.

Algorithm 2 Generation of environments with concentrated feature distributions

Input: Arena length L_E, arena width W_E, mean block width w_{mean}, targeted fill ratio of feature represented by color black r_{black}

Output: Matrix E with each entry $e_{\alpha,\beta}$ representing the feature placed in position (α, β); 0 represents white and 1 represents black

```
 1: Initialize $E$ as a matrix of zeros of size $L_E \times W_E$
 2: while $(\sum_{\alpha,\beta} e_{\alpha,\beta})/(L_E \times W_E) < r_{black}$ do
 3:     Sample block width $w$ from normal distribution $\mathcal{N}(w_{mean}, 1^2)$ rounded to integer
 4:     Sample $\alpha'$ and $\beta'$ separately from uniform distributions $\mathcal{U}(0, max(L_E - w, 0))$ and
        $\mathcal{U}(0, max(W_E - w, 0))$ rounded to integer
 5:     for $\alpha$=1..w do
 6:         for $\beta$=1..w do
 7:             $e_{\alpha'+\alpha,\beta'+\beta} = 1$
 8: while $(\sum_{\alpha,\beta} e_{\alpha,\beta})/(L_E \times W_E) > r_{black}$ do
 9:     Sample $(\alpha', \beta')$ to be the coordinate of a random edge tile of an existing block,
        with at least one of the eight neighboring tiles having the white feature while tile
        $(\alpha', \beta')$ having the black feature
10:     $e_{\alpha',\beta'} = 0$
```

The generation process starts by placing a series of square blocks of the feature in question with side lengths randomly sampled from a normal distribution $\mathcal{N}(w_{mean}, 1^2)$ (Algorithm 2 line 3). The square blocks are then placed successively at random positions in the arena (line 4), until the fill ratio of the black feature is higher than the targeted fill ratio. Individual tiles at the edges of the blocks are then successively flipped to white features until the fill ratio of the black feature is exactly as targeted (lines 8-10).

The level of concentration of the feature distribution can be adjusted via the mean width of feature blocks w_{mean}. Figure 4.1 shows example environments generated using different w_{mean} configurations when r_{black} is fixed at 0.45. As w_{mean} increases, the black features become more concentrated. This causes increasing confusion to the swarm by giving robots in different local areas of the environment increasingly different beliefs

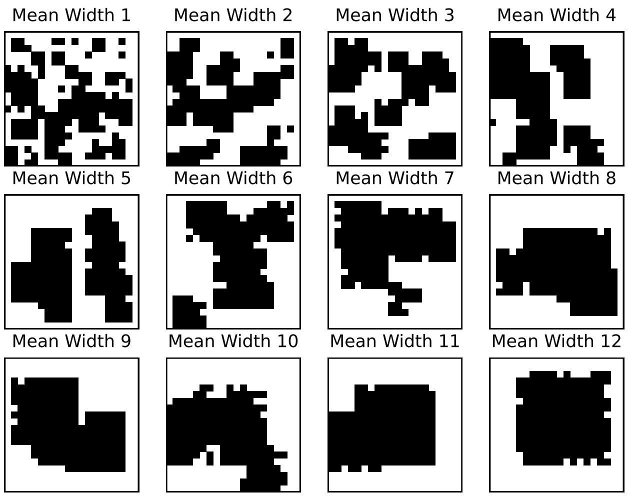


Figure 4.1: Example of various arenas with different mean widths of blocks. Each subfigure shows a randomly generated arena to be used in an experimental run ($r_{black} = 0.45$). As the mean widths of blocks increase, the black tiles become more concentrated. [SM21b]

regarding the fill ratio, and thus creates difficulty in achieving accurate consensus.

4.1.3 Baseline Opinion-Based Consensus Strategies for the Discrete Collective Estimation Scenario

This subsection introduces the baseline decision-making strategies used for comparison with the proposed parallel encoding approaches in the discrete collective estimation scenario. Among the investigated strategies, the Individual Exploration strategy is a baseline that does not utilize any communication. In addition, the aforementioned opinion-based collective decision-making strategies DC and DMVD for binary problems are straightforwardly modified to handle more than 2 potential options.

Individual Bayesian Hypothesis Testing Baseline Strategy

The Individual Exploration strategy shown in Algorithm 3 is used as a baseline, with each individual robot independently performing Bayesian hypothesis testing. At every control loop, the robot with index i samples the arena floor for an observation and updates its own likelihoods of the fill ratio hypotheses $\vec{l_i}$ using the Bayesian statistics-based hypothesis testing approach introduced in Chapter 3 (line 3-4). The difference in the implementation here is that an observation is only collected when the robot is moving forward in motion state A (shown in Table 3.1). This is to prevent duplicated observation being collected when the robot is rotating on a fixed spot in motion state B.

Algorithm 3 Individual Exploration baseline [SM21b]

Input: Hypothesis matrix H of size $M \times 2$; robot index $i \in \{1..N_{robot}\}$; initialized belief of robot i: $\vec{l_i}$ of length M

Output: Converged decision: d_i

```
1: while True do
2:     if Robot $i$ is moving forward in motion State A then
3:         Collect observation and compute $\vec{o}$
4:         $\vec{l_i}$ = Normalize($\vec{l_i} \circ (H \cdot \vec{o})$)
5:     $d_i$ = argmax($\vec{l_i}$)
```

The decision of the robot is computed independently at every control loop based on its updated likelihood array (line 5). Since the robots do not communicate with each other, they do not sense when a consensus is achieved. In implementation, an experimental run is terminated when the decisions of all robots converge to a single one. A collective decision-making strategy needs to perform significantly better than the Individual Exploration baseline approach to justify the extra infrastructure, as well as potential security and malfunction risks introduced by inter-robot communication.

Opinion-based Collective Consensus Strategies in the Investigated Multi-option Discrete Collective Estimation Scenario

The opinion-based collective consensus strategies Direct Comparison (DC) and Direct Modulation of Voter-based Decisions (DMVD), employed before in the binary collective perception problem, are also used as baseline strategies to gauge the effectiveness of the proposed parallel encoding approaches. Compared to the implementations in Chapter 3, modifications are made for them to handle the investigated discrete collective estimation scenario. The decision-making process begins with randomly initialized decisions d_i for every robot with indices $i \in N_{robot}$. The robots alternate between exploration and dissemination states following the probabilistic finite state machine shown in Figure 4.2.

The robot behavior during exploration states is shown in Algorithm 4. During exploration states, the robot computes the quality of its chosen decision with the same Bayesian hypothesis testing approach, taking the likelihoods of the hypotheses as option qualities (line 2-4). This allows for the exploration state behavior to be unified, where the robot computes the likelihoods of all hypotheses simultaneously and selects the corresponding entry for dissemination with its peers. At every control loop, the robot has a probability of $1/\sigma$ to switch from exploration states to dissemination states (line 5-7).

On the other hand, since each robot has only one chosen option, each of the M options needs to be separately advocated by a robot during the dissemination process, shown in the dissemination states in Figure 4.2.

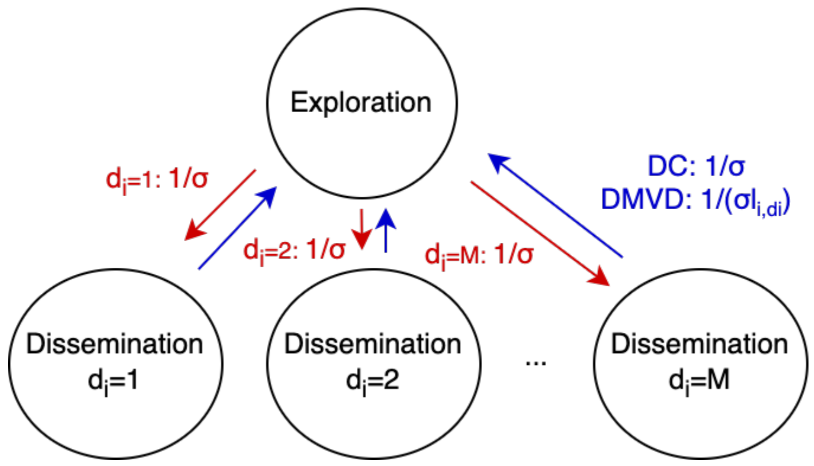


Figure 4.2: Probabilistic finite state machine illustrating the modified decision-making behaviors of the opinion-based strategies in the multi-option consensus scenario

Algorithm 4 Exploration state behavior of robots in opinion-based strategies

Input: Collective observation $\vec{o}$ at every control loop; current decision of robot d_i
Output: Likelihood of decision d_i given robot i's previous observations l_{i,d_i}

```
1: while Robot $i$ is in exploration state do
2:     if Robot $i$ is moving forward in motion State A then
3:         Collect observation and compute $\vec{o}$
4:         $\vec{l_i}$ = Normalize($\vec{l_i} \circ (H \cdot \vec{o})$)
5:     $v$ =sample from $\mathcal{U}[0,1)$
6:     if $v < 1/\sigma$ then
7:         Switch to dissemination state
```

Algorithm 5 shows the dissemination behavior of a robot under DC strategy. The implementation is similar to the binary collective perception scenario in Chapter 3. At every control loop, the robot receives the chosen decisions d_j and the computed quality l_{j,d_j} from a random robot j that is also disseminating within the communication range (lines 4-5). The robot also broadcasts its own decision d_i and computed quality l_{i,d_i} to its peers (line 6). The robot also has a probability of $1/\sigma$ at every control loop to conclude the dissemination period. If triggered, the robot switches its decision to the one with the highest associated quality among those collected from neighbors and held by itself (lines 9-10). Under DC strategy, the parameter σ controls the average length of exploration and dissemination periods, thus controlling the frequencies of inter-robot opinion transfer compared to individual environmental exploration. In addition, since a robot only switches to a particular decision if another robot is currently choosing it, it is important to ensure a diversity of opinions among the agents before enough samples are collected. In a collective decision-making scenario with only 2 options, if the agents have not converged to a single option, all options would have at least one agent choosing it. However, when applied to a

Algorithm 5 Dissemination state behavior of robots in DC strategy

Input: Current decision of robot d_i; computed likelihood during exploration l_{i,d_i}

Output: Updated decision of robot d_i after dissemination with its peers

```
1:  Set mean length of exploration/dissemination states $\sigma$ and mutation probability $\tau$
2:  $J = \{\}$
3:  while Robot $i$ is in dissemination state do
4:      ($d_j$, $l_{j,d_j}$) = CollectNeighborOpinion
5:      $J = J \cup j$
6:      Broadcast ($d_i$, $l_{i,d_i}$)
7:      $v$ =sample from $\mathcal{U}[0, 1)$
8:      if $v < 1/\sigma$ then
9:          if $max_{j \in J}(l_{j,d_j}) > l_{i,d_i}$ then
10:             $d_i$ = argmax$_{j \in J}(l_{j,d_j})$
11:             $J = \{\}$
12:             $v'$ =sample from $\mathcal{U}[0, 1)$
13:             if $v' < \tau$ then
14:                 $d_i$ = RandomChoice$(d_i + 1, d_i - 1)$ when valid
15:         Switch to exploration state
```

scenario with more than 2 options, there is an increasing probability that none of the agents pick one of the decisions, thus prematurely eliminating it in the decision-making process. Therefore, the swarm needs to keep a diversity of opinions among the robots during the decision-making process to adequately consider all of the options and reach an accurate consensus. This is implemented by having the robot randomly switch from d_i to a new hypothesis that is next to its old option and are also valid options (i.e., $d_i + 1$ or $d_i - 1$) with a probability defined by the parameter τ at the end of dissemination periods (line 14).

Algorithm 6 shows the dissemination behavior of the robot under DMVD strategy. It is distinguished from that under DC strategy by the robot only exchanging the current chosen option d_i with its peers (lines 4-6). The computed quality l_{i,d_i} is not transmitted directly but is used to modulate the probability of state transition $\sigma l_{i,d_i}$ from dissemination to exploration states. This means that on average, a robot that obtains a high quality for its current decision will be in the dissemination period for a longer time. This means it broadcasts its own opinion more frequently and changes its own decision less frequently per unit time. At the end of the dissemination periods, the robot switches to a random decision among those collected during dissemination, which has a higher chance of being a high-quality decision (line 10). This mechanism allows the swarm to converge to the optimal decision during the consensus process. Similar to in DC strategy, the robot randomly switches to an adjacent decision to d_i at the end of its dissemination period (line 14).

Algorithm 6 Dissemination state behavior of robots in DMVD strategy

Input: Current decision of robot d_i; computed likelihood during exploration l_{i,d_i}
Output: Updated decision of robot d_i after dissemination with its peers

```
1:  Set mean length of exploration/dissemination states $\sigma$ and mutation probability $\tau$
2:  $J = \{\}$
3:  while Robot $i$ is in dissemination state do
4:      $d_j$ = CollectNeighborOpinion
5:      $J = J \cup j$
6:      Broadcast $d_i$
7:      $v$ =sample from $\mathcal{U}[0, 1)$
8:      if $v < 1/(\sigma l_{i,d_i})$ then
9:          if $J \neq \emptyset$ then
10:             $d_i = d_{j'}$ where $j'$ is random index stored in set $J$
11:             $J = \{\}$
12:             $v'$ =sample from $\mathcal{U}[0, 1)$
13:             if $v' < \tau$ then
14:                 $d_i$ = RandomChoice$(d_i + 1, d_i - 1)$ when valid
15:         Switch to exploration state
```

4.2 Decentralized Multi-Option Collective Consensus Forming with Bayesian Belief Fusion

The first proposed multi-option collective consensus strategy enforces consensus via direct fusion of individual robots' Bayesian beliefs regarding the discretized decision space. This section presents its decision-making mechanism and its performances, compared with those of opinion-based strategies.

4.2.1 Distributed Bayesian Belief Sharing Strategy

The likelihood array computed by individual Bayesian hypothesis testing can be directly multiplied together to pool the information from every robot for an accurate global estimation of an environmental attribute, as in Chapter 3. However, the centralized communication paradigm limits the scalability of the system. Therefore, distributed Bayesian belief sharing (DBBS shown in Algorithm 7) is proposed as a decentralized way to pool individual beliefs within a swarm to accomplish discrete collective estimation.

In the proposed DBBS approach, individual agents do not have unique indices, and the communication is kept to a peer-to-peer manner. Thus, during a control loop, the robot with index i broadcasts its message and receives a single message from a random neighbor within communication distance. This means a robot can receive multiple identical messages from the same neighbor, leading to strong positive feedback that can skew the opinions in a local cluster of robots. To counter this, the robot i keeps 2 sets of beliefs: $\vec{l_i}$ and $\vec{l'_i}$. $\vec{l_i}$ denotes the belief computed from the robot's own observations, and $\vec{l'_i}$ denotes the combined belief received from its neighbors. The latter progressively decays at every control loop to weaken its influence. The two beliefs are combined when the robot is computing its decision.

Algorithm 7 Distributed Bayesian Belief Sharing (DBBS) [SM21b]

Input: Hypothesis matrix H of size $M \times 2$; robot index $i \in \{1..N_{robot}\}$; initialized belief of robot i: $\vec{l_i}$ of length M; initialized record of neighbors' beliefs : $\vec{l'_i}$ of length M
Output: Converged decisions: d_i

```
1:  Set decay parameters $\lambda$, $\mu$
2:  while Decisions in swarm have not converged do
3:      if Robot n is moving forward in motion State A then
4:          Collect observation and compute $\vec{o}$
5:          $\vec{l_i} = $ Normalize$(\vec{l_i} \circ (H \cdot \vec{o}))$
6:      $\vec{\xi_j}$ = CollectNeighborOpinion
7:      if $\vec{\xi_j}$ collected then
8:          $\vec{l'_i} = $ Normalize$(\vec{l'_i}^{\lambda} \circ \vec{\xi_j})$
9:      $\vec{\xi_i} = \vec{l_i} \circ \vec{l'_i}^{\mu}$
10:     Broadcast$(\vec{\xi_i})$
11:     $d_i = $ argmax$(\vec{l_i} \circ \vec{l'_i})$
```

Details of the proposed strategy are as follows. At every control loop, the robot makes an observation on the color of the arena floor and modifies $\vec{l_i}$ according to the distributed Bayesian hypothesis testing approach introduced in Chapter 3 (lines 3-5). The vector $\vec{\xi}$ is used to denote the messages passed among the robots. The robot attempts to collect a neighbor's message $\vec{\xi_j}$ from a random neighbor if there is one broadcasting (line 6). If $\vec{\xi_j}$ is received, the robot updates its record of its neighbors' belief $\vec{l'_i}$ with a weighted product of itself and the received message, as follows (line 8):

$$\vec{l'_i} = \text{Normalize}(\vec{l'_i}^{\lambda} \circ \vec{\xi_j}). \tag{4.2}$$

λ is the decay coefficient of past neighbors' beliefs and is applied to the old value of $\vec{l'_i}$ during its update. It serves to reduce the influence of older received messages on the decision making and opinion fusion of the swarm.

After updating both belief arrays, the robot computes its own message $\vec{\xi_i}$ that it will send to its neighbors (line 9). It is a weighted element-wise product of $\vec{l_i}$ and $\vec{l'_i}$, with the weight of $\vec{l'_i}$ being μ, as shown below:

$$\vec{\xi_i} = \vec{l_i} \circ \vec{l'_i}^{\mu}. \tag{4.3}$$

The parameter μ controls the weight of $\vec{l}_i'$ that is sent back to the other robots during communication. It serves to control directly the level of positive feedback in the collective decision-making process. The robot broadcasts $\vec{\xi}_i$ to its neighbors for the rest of the control loop (line 10). A robot's decision is made individually and is the option corresponding to the highest combined likelihood when multiplying $\vec{l}_i$ and $\vec{l}_i'$, as follows:

$$d_n = \text{argmax}(\vec{l}_i \circ \vec{l}_i'). \tag{4.4}$$

From a probabilistic inference point of view, both parameters λ and μ tunes the assumed level of independence between observations made by different robots. In practice, increasing either λ or μ strengthens the impact of neighboring robots' opinions relative to a robot's own opinion during the decision-making process, therefore increasing the positive feedback in the decision-making process.

Positive feedback has been used before in collective decision-making strategies. In DC and DMVD, as well as the Bayesian belief-based strategy proposed by Ebert et al. [Ebe+20], positive feedback is used, but only at the collective level, which means that the decision made by a particular robot induces other robots to make the same decision. The proposed DBBS approach also employs positive feedback at an individual level, which means that a robot's belief strengthens itself as it is passed back from its neighbors in a decayed form. A robot's belief influences not only its immediate neighbors but is also passed along in the swarm. Such design causes the beliefs of all agents in the swarm to be more closely and more directly linked, therefore making the decision-making process faster without needing to sacrifice accuracy.

4.2.2 Multi-Criteria Analysis Framework

In order to objectively compare the performances of the considered strategies, the speed versus accuracy trade-off during the consensus processes has to be accounted for. Therefore, the performances of the considered consensus strategies need to be analyzed using a multi-criteria framework.

The speed versus accuracy trade-off is displayed by the different performances at different algorithmic parameter settings for a particular consensus strategy. The multi-criteria performance $\vec{c}$ at a particular set of algorithmic parameters $\vec{\theta}$ can be estimated by taking the mean values of criteria measurements over many simulation runs in randomly generated environments. The relative optimality of two performances at two different parameter settings can be compared, without assigning weights to the criteria, using the principle of Pareto dominance [Cen77]. The multi-criteria performance at parameter setting $\vec{\theta}_1$ can be said to objectively outperform the performance at parameter setting $\vec{\theta}_2$, denoted as $\vec{c}_1$ and $\vec{c}_2$ respectively, when the former is at least as good as the latter in all criteria and outperforms the latter in at least one criterion:

$$\vec{c}_1 \prec \vec{c}_2 \text{ when } c_{1,x} \leq c_{2,x}\ \forall x \text{ and } \exists x : c_{1,x} < c_{2,x}. \tag{4.5}$$

A smaller value is assumed to be more desirable in all criteria.

Thus, given a set of mean performance measurements C at various parameter settings, the subset with the most optimal multi-criteria performances (the Pareto front set C'_{Pareto}) can be obtained as the minimal set where every entry not included in C'_{Pareto} is outperformed by an entry in C'_{Pareto}:

$$C'_{Pareto} = \underset{C' \subseteq C, \forall \vec{c}_2 \in C \setminus C'\ \exists \vec{c}_1 \in C' : \vec{c}_1 \prec \vec{c}_2}{\operatorname{argmin}} |C'|. \tag{4.6}$$

This allows the efficiency of speed versus accuracy trade-offs to be compared between different consensus strategies by comparing the Pareto fronts obtained while minimizing the influence of algorithmic parameter selection.

4.2.3 Experiments and Results

In this subsection, the proposed DBBS approach is tested using the discrete collective estimation scenario. The baseline opinion-based consensus strategies are used for comparison. The basic environmental design is kept the same as in Chapter 3, with the arena size set to $2m \times 2m$ filled by 400 square tiles. Following previous results, the frequency of observation collection needs to be less than once per second to prevent excessive observation collection by a robot on a single tile. Additionally, in order to keep the number of observations and exchanged messages among the robots at similar levels across the considered strategies, the control loops are set to different lengths for different investigated strategies. For the DMVD and DC strategies, an observation is collected on average once every 2 control loops. The rate of message transmission is also at most once every 2 control loops. The length of a control loop is thus set to $1s$. For the Individual Exploration baseline and DBBS strategies, an observation is made in every control loop and a message transmission happens in every control loop as well. Therefore, the length of a control loop for these two strategies is set to $2s$ to keep a fair comparison.

Three performance criteria are computed from 50 experimental runs for every set of parameter settings: absolute error, consensus time, and failure rate. Absolute error is defined as the mean deviation of the chosen fill ratio hypotheses by the agents from the true fill ratio of the environmental feature in question. Consensus time is defined as the time taken for all robots to converge to the same hypothesis decision during an experimental run. As consensus of all robots' opinions is not guaranteed, the failure rate is used to measure the probability of a considered strategy failing to reach a consensus in the given time limit of 1200 seconds. The time limit of 1200 seconds is selected to be well above the average consensus time for considered strategies to reduce the probability of premature rejection of an experimental run. In addition, the total number of message transfers until consensus is also recorded to compare the levels of efficiency of inter-robot communication in different strategies.

Scenarios with Random Distribution of Environmental Features

The considered consensus strategies are first tested in environments with random feature distributions and at different fill ratios ranging from 0.05 to 0.45. The average performance obtained across all five fill ratio scenarios is taken to be the performance of the considered strategy at the associated algorithmic parameter configuration.

For both DC and DMVD decision-making strategies, two important algorithmic parameters are the mean length of exploration and dissemination states σ and the probability of mutating the chosen option τ. Both parameters tune the performances of the two strategies in terms of the accuracy versus speed trade-off. Their impacts are investigated in this subsection.

The performances of DC at various σ and τ settings are shown in Table 4.1. It can be observed that as either σ or τ is increasing, the error generally decreases while the consensus time and failure rate increase. Also, the design configurations with no diversity enforcement, i.e. $\tau = 0$, produce the highest errors. Moreover, the decrease in error is steeper and the increase in consensus time is milder in the increasing τ direction. Therefore, when trying to increase the accuracy by sacrificing consensus time, it is better to have a high τ value than a high σ value. This is demonstrated in the following data points. At $\sigma = 1s$ and $\tau = 0.05$, the mean error is 0.196 and the mean consensus time is $162.0s$. However, to reach a similar consensus time of $166.4s$ by increasing σ to 8 while keeping τ at 0, a higher error of 0.534 is obtained. This is because the decision-making mechanism of DC can quickly lead the decisions in a neighborhood to converge to the option with the highest quality, and thus it is more beneficial for achieving a more accurate decision-making to maintain a high diversity of opinions. A high τ value means that the robots readily and randomly modify their opinions. If a modification is beneficial, it will soon spread across neighboring robots, and if a modification is not beneficial, the robot will quickly switch to a better decision in the next dissemination session. In addition, the failure rate is quite low and around 0 across all parameter settings except at high σ settings. This shows that DC is very reliable in providing a consensus even in a multi-option collective decision-making problem. Finally, the mean number of message transfers is largely proportional to the mean consensus time. This is because DC strategy has a constant rate of message transfer.

The performances of the DMVD strategy at various σ and τ settings are shown in Table 4.2. Here, the impacts of σ and τ are similar to in DC. The configurations with $\tau = 0$ also produce a higher error than other settings with larger τ values. However, for DMVD,

Table 4.1: Mean absolute error, mean consensus time, mean message transfer and failure rate of DC at different σ and τ settings; vertical-σ mean lengths of exploration/dissemination states, horizontal-τ probability of mutating the chosen option [SM21b]

		Mean Error/0.1						Mean Consensus Time/s					
	τ	0.00	0.01	0.02	0.03	0.04	0.05	0.00	0.01	0.02	0.03	0.04	0.05
σ/s	1	0.736	0.588	0.392	0.380	0.260	0.196	71.8	84.4	101.5	116.3	134.7	162.0
	2	0.804	0.472	0.452	0.332	0.244	0.180	77.4	98.8	112.0	135.2	150.3	168.7
	3	0.712	0.420	0.384	0.364	0.228	0.196	99.7	120.8	130.4	138.3	162.8	186.5
	4	0.588	0.460	0.360	0.292	0.184	0.152	116.4	131.1	138.1	151.4	174.8	198.0
	5	0.610	0.296	0.348	0.240	0.180	0.144	129.2	142.3	157.0	183.1	184.7	196.7
	6	0.470	0.348	0.320	0.224	0.196	0.140	150.0	152.4	163.1	177.1	188.3	203.8
	7	0.512	0.308	0.356	0.244	0.220	0.148	147.3	169.1	170.9	194.6	196.6	225.7
	8	0.534	0.316	0.280	0.188	0.192	0.116	166.4	171.8	182.2	203.7	210.3	243.5
	9	0.456	0.336	0.324	0.236	0.168	0.120	165.6	181.1	193.5	211.8	233.7	239.2
	10	0.400	0.364	0.272	0.156	0.200	0.156	179.0	189.4	208.3	230.0	224.3	244.5
	15	0.351	0.284	0.244	0.196	0.116	0.092	218.6	249.2	248.5	269.9	278.7	307.1
	20	0.294	0.229	0.217	0.192	0.100	0.084	248.0	267.7	291.7	295.7	334.1	357.0
	25	0.281	0.252	0.172	0.148	0.160	0.100	269.4	316.8	342.3	348.5	364.4	386.1
	30	0.305	0.208	0.180	0.136	0.120	0.104	326.0	348.3	366.5	393.6	398.1	425.7
	35	0.325	0.209	0.192	0.132	0.104	0.069	359.3	383.3	391.2	418.9	443.7	468.6
	40	0.292	0.241	0.150	0.137	0.116	0.073	400.9	405.9	426.6	452.7	458.4	506.9
	45	0.309	0.194	0.153	0.104	0.109	0.101	419.6	445.6	466.8	500.6	516.9	544.6
	50	0.193	0.245	0.174	0.094	0.112	0.094	433.8	464.6	491.1	516.3	547.6	577.2
		Mean Message Transfer						Failure Rate					
σ/s	1	955	1128	1357	1544	1792	2141	0.000	0.000	0.000	0.000	0.000	0.000
	2	1018	1313	1486	1788	1992	2218	0.000	0.000	0.000	0.000	0.000	0.000
	3	1283	1561	1709	1817	2148	2469	0.000	0.000	0.000	0.000	0.000	0.000
	4	1512	1716	1815	1993	2294	2614	0.000	0.000	0.000	0.000	0.000	0.000
	5	1674	1857	2031	2399	2416	2570	0.004	0.000	0.000	0.000	0.000	0.000
	6	1955	1971	2104	2299	2472	2660	0.004	0.000	0.000	0.000	0.000	0.000
	7	1916	2182	2217	2529	2551	2886	0.000	0.000	0.000	0.000	0.000	0.000
	8	2140	2214	2346	2650	2743	3157	0.004	0.000	0.000	0.000	0.000	0.000
	9	2118	2322	2480	2742	3014	3074	0.000	0.000	0.000	0.000	0.000	0.000
	10	2301	2425	2671	2984	2863	3184	0.000	0.000	0.000	0.000	0.000	0.000
	15	2725	3182	3201	3512	3592	3969	0.008	0.000	0.000	0.000	0.000	0.000
	20	3066	3336	3682	3723	4253	4570	0.008	0.004	0.004	0.000	0.000	0.000
	25	3388	3952	4382	4401	4539	4866	0.004	0.000	0.000	0.000	0.000	0.000
	30	4045	4341	4538	4934	4990	5366	0.004	0.000	0.000	0.000	0.000	0.004
	35	4469	4756	4904	5109	5561	5837	0.016	0.004	0.000	0.000	0.000	0.008
	40	4947	5083	5308	5723	5721	6356	0.000	0.004	0.012	0.004	0.004	0.008
	45	5191	5512	5845	6234	6477	6919	0.004	0.008	0.008	0.000	0.012	0.012
	50	5283	5680	6048	6389	6808	7238	0.004	0.004	0.012	0.020	0.032	0.024

increasing σ has a larger impact on reducing the error and increasing the consensus time than increasing τ. In addition, increasing either σ or τ introduces a significant failure rate. This is shown via the following data points. At $\sigma = 1s$ and $\tau = 0.05$, the mean error is 1.197 and the mean consensus time is $678.1s$. However, to reach a similar mean error by increasing σ to $8s$ while holding τ at 0, a mean consensus time of $428.6s$ is obtained, which is far smaller but at a similar mean error of 1.010. This is because the DMVD strategy uses

Table 4.2: Mean absolute error, mean consensus time, mean message transfer and failure rate of DMVD at different σ and τ settings; vertical-σ mean lengths of exploration/dissemination states, horizontal-τ probability of mutating the chosen option [SM21b]

		Mean Error/0.1						Mean Consensus Time/s					
	τ	0.00	0.01	0.02	0.03	0.04	0.05	0.00	0.01	0.02	0.03	0.04	0.05
σ/s	1	3.232	2.691	2.347	1.679	1.226	1.197	156.2	239.3	409.4	516.3	588.8	678.1
	2	2.444	1.858	1.093	0.603	0.309	0.099	233.3	336.8	427.8	465.4	537.4	619.9
	3	1.728	0.996	0.479	0.243	0.084	0.031	288.7	389.4	430.5	464.5	531.0	542.6
	4	1.500	0.979	0.502	0.053	0.057	0.000	351.0	403.3	420.1	448.3	505.0	565.8
	5	1.288	0.468	0.215	0.018	0.032	0.000	374.8	402.2	422.3	456.4	537.1	541.0
	6	1.335	0.267	0.100	0.000	0.000	0.000	426.8	404.7	457.2	490.3	530.7	550.9
	7	0.814	0.162	0.059	0.004	0.004	0.000	404.0	456.4	450.0	504.8	513.3	569.6
	8	1.010	0.151	0.018	0.009	0.000	0.000	443.8	456.8	478.3	532.7	558.3	599.0
		Mean Message Transfer						Failure Rate					
σ/s	1	1298	2011	3510	4520	5290	6151	0.000	0.004	0.032	0.104	0.292	0.472
	2	1169	1763	2463	2865	3420	4147	0.000	0.012	0.052	0.104	0.132	0.152
	3	1051	1644	2075	2435	2844	3134	0.000	0.048	0.124	0.112	0.100	0.108
	4	998	1454	1771	2190	2526	3051	0.008	0.052	0.156	0.092	0.092	0.096
	5	963	1417	1669	1979	2487	2723	0.040	0.120	0.088	0.092	0.116	0.120
	6	960	1468	1790	2059	2414	2680	0.068	0.132	0.080	0.080	0.052	0.052
	7	1046	1574	1733	2175	2261	2735	0.116	0.112	0.120	0.080	0.056	0.076
	8	973	1526	1808	2186	2478	2824	0.180	0.128	0.112	0.072	0.080	0.068

sampling to select options for individual robots, and therefore is able to maintain a high level of diversity in the opinions of the robots during the decision-making process. If more diversity is added via a high τ setting, the robot can have difficulty coming to a consensus, as shown in the high failure rate when τ is high. The mean number of message transfers for the DMVD strategy is not strictly proportional to the consensus time. This is because DMVD changes the lengths of the agents' dissemination states depending on the qualities of their decisions. Thus, agents often do not keep broadcasting their decisions for the full duration set by the parameter σ.

On the other hand, the performance of the DBBS strategy is tuned by the parameters λ and μ. λ controls the rate at which a robot forgets previously received opinions. μ controls the strength that a robot passes on opinions received from other robots. The performances of the DBBS strategy at various λ and μ settings are shown in Table 4.3.

Similar to DC and DMVD strategies, DBBS also faces a trade-off between the decision speed and the accuracy. With either λ or μ increasing, the error increases while the consensus time decreases. The increase in error is very significant when μ increases from 0.6 to 1.0 but is not significant before that. Also, the decrease in consensus time is significant when μ increases from 0 to 0.4, but is also not significant beyond that. Therefore, the selection of μ and hence the degree of positive feedback in the decision-making process has an optimum at around 0.4 and 0.6 that can achieve both fast and accurate decision-making relative to other μ values. In contrast, increasing λ has a steadier impact on the performance, and the speed versus accuracy trade-off is more prevalent here. As expected,

Table 4.3: Mean absolute error, mean consensus time, mean message transfer and failure rate of DBBS at different λ and μ settings; vertical-λ decay rate of past neighbors' beliefs, horizontal-μ decay rate of current neighbors' beliefs during communication [SM21b]

		Mean Error/0.1						Mean Consensus Time/s					
	μ	0.0	0.2	0.4	0.6	0.8	1.0	0.0	0.2	0.4	0.6	0.8	1.0
λ	0.5	0.000	0.000	0.008	0.172	0.580	1.320	180.3	119.8	81.1	46.2	39.7	36.3
	0.6	0.000	0.000	0.020	0.284	1.028	1.484	163.0	103.5	56.5	42.7	39.3	35.4
	0.7	0.000	0.004	0.112	0.500	1.280	1.560	151.9	80.9	49.3	40.0	39.9	34.1
	0.8	0.000	0.028	0.256	0.716	1.388	1.712	131.4	60.9	46.9	39.4	38.4	33.6
	0.9	0.000	0.112	0.448	1.096	1.332	1.736	119.2	53.9	41.7	40.6	35.3	33.9
		Mean Message Transfer						Failure Rate					
λ	0.5	1687	1126	769	447	386	355	0.000	0.000	0.000	0.000	0.000	0.000
	0.6	1525	977	542	414	383	347	0.000	0.000	0.000	0.000	0.000	0.000
	0.7	1425	768	474	389	388	335	0.000	0.000	0.000	0.000	0.000	0.000
	0.8	1236	584	452	383	374	330	0.000	0.000	0.000	0.000	0.000	0.000
	0.9	1121	518	405	395	346	333	0.000	0.000	0.000	0.000	0.000	0.000

the mean number of message transfers is also roughly proportional to the consensus time. In addition, DBBS has not had a failure in decision-making during the experiments in this scenario, which shows its robustness compared to the other considered strategies.

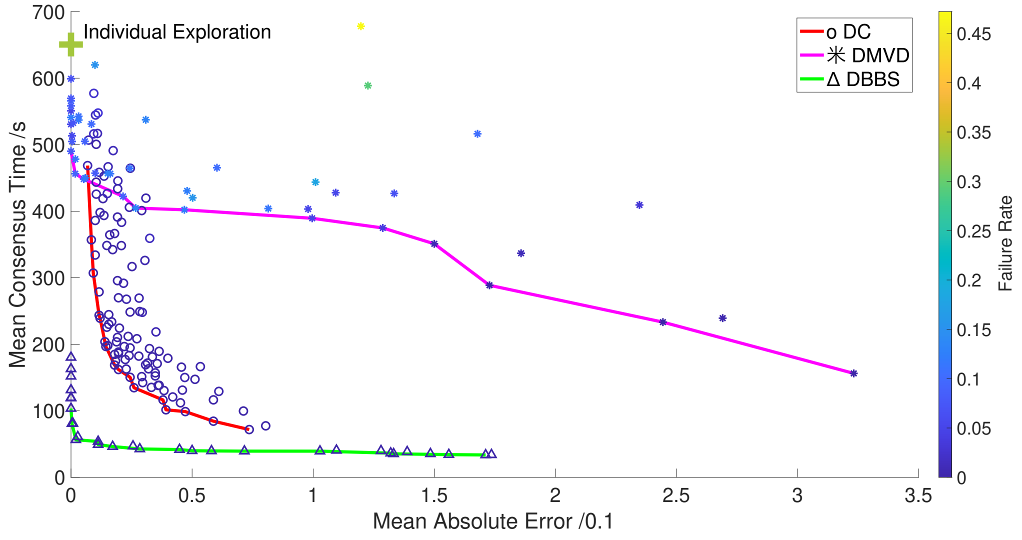


Figure 4.3: Plot of mean consensus time against mean absolute error for all considered strategies at all considered parameter settings when facing random environmental feature distribution; color coding of markers shows the failure rate; solid lines show the Pareto frontiers of speed versus accuracy performances [SM21b]

The above numerical results are depicted in Figure 4.3 and 4.4 so that the trade-offs between the considered metrics for the different strategies can be investigated. Figure 4.3 depicts the trade-off between consensus time and mean absolute error. The performance of the Individual Exploration baseline is also shown in the figure ($+$ Marker). The Individual Exploration baseline can ensure zero error in the final consensus when the decision-making process is successful. This is because, in Individual Exploration, robots do not communicate with each other. Therefore, for a consensus to be formed, all robots must pick that option based on their own observations. The probability for all robots to converge to a wrong option is almost zero. However, due to this design, Individual Exploration has a high mean consensus time of $651s$ and a high failure rate of 0.332, as there is no way to encourage the formation of a consensus among the robots.

As already mentioned above, the performances of the other collective decision-making strategies exhibit trade-offs at different parameter settings. The Pareto frontiers of the trade-offs between mean consensus time and mean absolute error in each strategy are shown in solid lines in Figure 4.3.

DMVD's performance is shown using $*$ markers in Figure 4.3, and its Pareto frontier is shown in magenta. It outperforms the Individual Exploration baseline, as it reaches the same error of 0. DVMD can reach a mean consensus time of $518s$ and a failure rate of 0.1. The Pareto frontier of DMVD has a low gradient, and hence it can obtain very low error values, but when the decision-making process needs to be faster, the error rapidly increases.

DC's performance is shown in $\circ$ markers in Figure 4.3, and its Pareto frontier is shown in red. DC is able to come to a consensus much faster and at a much lower failure rate than Individual Exploration and DMVD. It can be observed that DC is able to produce a mean error of 0.256 at a mean consensus time of $145s$ and a failure rate of 0, while DMVD requires a consensus time of $415s$ and a failure rate of 0.084 to produce a comparable error of 0.249. However, DC is not able to produce a mean error as low as DMVD. In the experiments, the lowest mean error achieved is 0.0569 at a mean consensus time of $525s$, which is higher than those produced by DMVD at a lower consensus time. However, the failure rate is smaller than those produced by the two previous strategies at 0.016. Tuning its parameters to extend the mean consensus time has an increasingly diminishing impact on reducing the error for DC, as shown by the increasingly vertical slope of the DC's Pareto frontier on the top-left side.

DBBS's performance is shown in Δ markers in Figure 4.3, and its Pareto frontier is shown in green. In contrast to Individual Exploration and DMVD, DBBS is able to reach an error of zero at a much smaller mean consensus time of $102s$ and without any failures. This is well beyond the Pareto frontiers of both DMVD and DC. On the other hand, the reduction of consensus time to lower than $60s$ causes a dramatic rise of error from close to 0 to 1.736. As shown in Table 4.3, these data points correspond to high values of the parameter μ. This is because at high settings of μ, the neighbors' beliefs are weighted heavily when computing an agent's own belief, and thus strong positive feedback can occur, hence a few agents can lead the swarm into making a premature consensus.

For all the considered collective decision-making strategies, there are trade-offs between accuracy and consensus speed in their decision-making process. Regarding the 3 benchmark strategies, both DMVD and DC clearly outperform the Individual Exploration baseline, while the two collective strategies exhibit different characteristics in the accuracy versus speed trade-off. DMVD can reach higher accuracy, but is inelastic in terms of decision speed. DC, on the other hand, can form a consensus quickly, but is inelastic in terms of accuracy. In contrast, DBBS is able to outperform the state-of-the-art decision-making strategies in all 3 metrics.

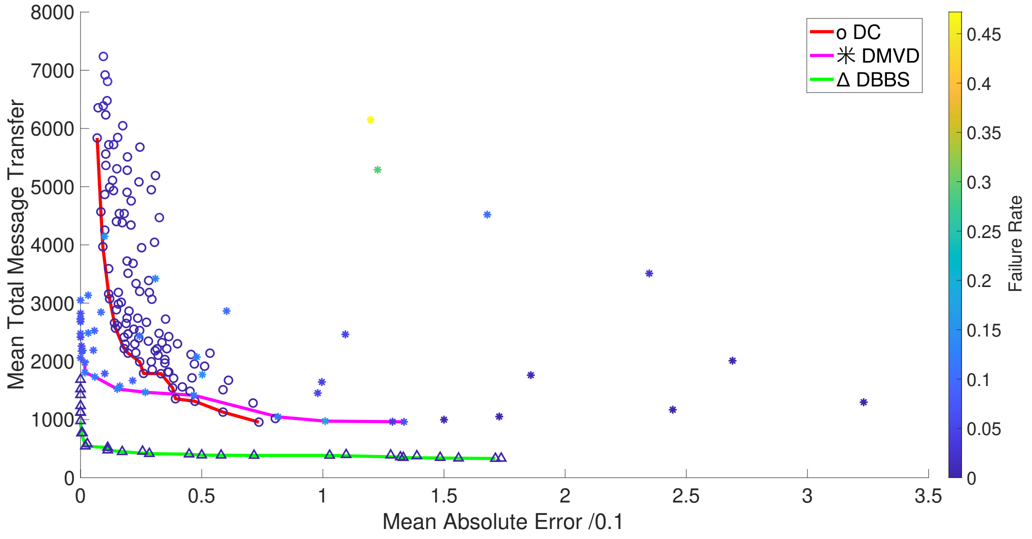


Figure 4.4: Plot of mean total message transfer against mean absolute error for all considered strategies at all considered parameter settings when facing random environmental feature distribution; color coding of markers shows the failure rate; solid lines show the Pareto frontiers of speed versus accuracy performances [SM21b]

Figure 4.4 plots the mean total number of message transfers against mean absolute error to investigate the performances of considered strategies from the point of view of the communication overhead. Among the 3 strategies, DMVD has the lowest average rate of message transfer at $4.96/s$. This causes DMVD to surpass DC with most configurations in terms of communication required. DBBS has an average rate of message transfer at $9.53/s$. This is close to the rate of message transfer intended in the experimental setup of 1 message per agent per $2s$, which produces an ideal rate of $10/s$ with 20 agents. The actual rate is likely to be lower because agents sometimes have no peers in their communication range. DC has the highest rate of message transfer at $12.69/s$. This is because in the experiments DC, similar to DMVD, does not restrict the neighborhood size. Hence, agents can receive all messages sent from neighbors within their communication radius during a fixed time period. While in DBBS, agents are restricted to receiving only 1 message per control loop.

Scenarios with Concentrated Distribution of Environmental Features

To gauge the negative effects of local information variance on decision-making processes in the investigated discrete collective estimation scenario, this subsection presents the performances of the considered decision-making strategies when facing concentrated feature distributions. The first set of experiments uses the algorithmic parameter settings shown in Table 4.4, which produce moderate results on the Pareto frontiers in the previous subsection. The considered strategies with these parameter settings are tested in a series of environments with progressively more concentrated distribution of black features. The changes in mean absolute error, mean consensus time, and failure rate are plotted in Figure 4.5. In this plot, a mean block width of 0 means that the tiles are distributed randomly.

Table 4.4: Parameter settings of considered strategies in Figure 4.5 [SM21b]

DC	$\sigma = 8s$	$\tau = 0.05$
DMVD	$\sigma = 4s$	$\tau = 0.02$
DBBS	$\lambda = 0.7$	$\mu = 0.4$

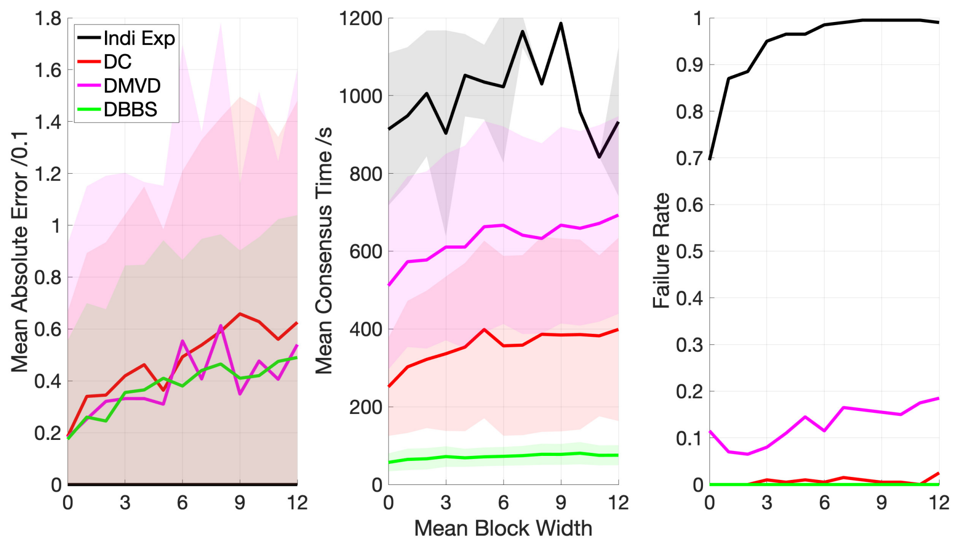


Figure 4.5: Progression of performances of considered strategies as the level of concentration of features increases; shaded areas represent standard deviations [SM21b]

In this figure, Individual Exploration's performances are shown in black. It can be observed that the mean error remains 0, when the mean block width increases from 0 to 12, while the failure rate increases rapidly to close to 1. DMVD and DC both experience an increase in mean error, mean consensus time, and failure rate as the mean block width increases as shown in magenta and red colors respectively. The increase in error is of a comparable degree. DC has a more significant increase in consensus time from $251s$

to $399s$, and DMVD has a more significant increase in failure rate from 0.115 to 0.185. DBBS also experiences a rise in error and a small rise in consensus time, but the failure rate remains zero even at the highest mean block width setting, as shown in green. This shows that DBBS's robustness remains even in scenarios with a high concentration of environmental features.

To obtain a full picture of the performances of considered strategies in more difficult environments, their performances across all considered parameter settings in arenas with a fill ratio of 0.45 and mean block width of 12 are shown in Figure 4.6. Individual Exploration has a failure rate close to 1 and therefore is not plotted. The Pareto frontiers obtained in the previous section in random environments are also shown in dotted lines to provide comparison.

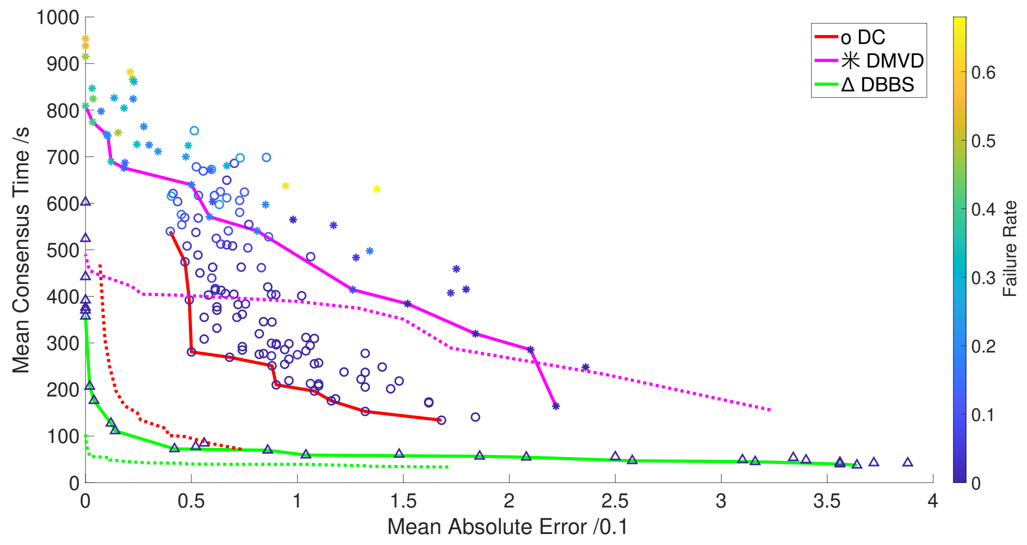


Figure 4.6: Plot of mean consensus time against mean absolute error for all considered strategies at all considered parameter settings when facing concentrated distribution of environmental features with $r_{black} = 0.45$, $w_{mean} = 12$; color coding of markers shows the failure rate; solid lines show the Pareto frontiers of speed versus accuracy performances; dotted lines show the Pareto frontiers obtained before in random environments [SM21b]

It can be observed in the magenta line in Figure 4.6 that there is a significant rise on the left-hand side of the Pareto frontier compared to random environments. Thus, a lot more consensus time needs to be sacrificed to reach a low error. An error of 0 can only be reached by extending the consensus time to $809s$. There is also a large increase in failure rate, which reaches above 0.5 for many settings. This is because DMVD has a poor ability to ensure the formation of a consensus, compared to DC and DBBS. This shortcoming is more prevalent in an environment with a high concentration of features, as the robots can hold conflicting opinions skewed to either extreme. Without enough pressure toward consensus, the probability of the robots reaching a consensus in the required time is low.

There is an increase in mean error and mean consensus time across all parameter settings for both DC and DBBS, compared to the previous subsection, as shown in the red and green lines respectively in Figure 4.6. DC has an even increase in both mean error and mean consensus time across all parameter settings. The failure rate also increases significantly when the mean consensus time is high. On the top left end of the Pareto front, the minimum mean error DC can achieve increases from 0.069 to 0.4. While the mean consensus time increases from $469s$ to $540s$, and the failure rate increases from 0.08 to 0.1.

DBBS has a high increase in error from 1.74 to 3.64 when mean consensus time is at its lowest, which is the range with high μ settings, but there is not a significant increase in consensus time in this section, only from $33.9s$ to $37.5s$. This is because high level of positive feedback can guarantee that the robots would come to a consensus very quickly. However, this comes at the cost that robots could only take very few samples. In an environment with a high concentration of features, these samples are likely to be skewed to one color and cause the final consensus to be inaccurate. At low μ settings on the top left end of the Pareto front, there is a mild increase in error and consensus time across most parameter settings. But zero error can still be achieved by extending the consensus time, which needs to be increased from $104s$ to $357s$, while the failure rate can still be kept at 0.

Analysis of DBBS under Sparse Communications

In order to gauge the performances of the proposed DBBS strategy under sparse communications, the communication range of the robots is varied and its effects on the performances of DBBS in environments with random feature distribution are plotted in Figure 4.7. Besides the default communication range of $0.5m$, settings $0.1m$ and $0.3m$ have also been tested. All three cases result in no failure in decision-making, and therefore the failure rate is not included in this figure.

When the communication range reduces from $0.5m$ to $0.3m$, there is a mild increase in the consensus time of DBBS. When the communication range is reduced further to $0.1m$, the consensus time significantly increases to almost twice that of a communication range of $0.3m$. On the other hand, the general shapes of the Pareto frontiers between consensus time and error are largely the same. The swarm is also able to reach an error of 0 for all communication ranges without much sacrifice of consensus time. This demonstrates that DBBS is resilient to the effects of sparse communications. A reduced communication range translates largely to longer consensus time, while the accuracy and reliability both remain high.

As a benchmark, the performances of DC at a reduced communication range are shown in Figure 4.8. A shorter communication range of 0.1 reduces the number of neighbors an agent is able to communicate within the dissemination periods. Thus, it makes the decision process slower, but also improves the accuracy of the results. However, the increase in the

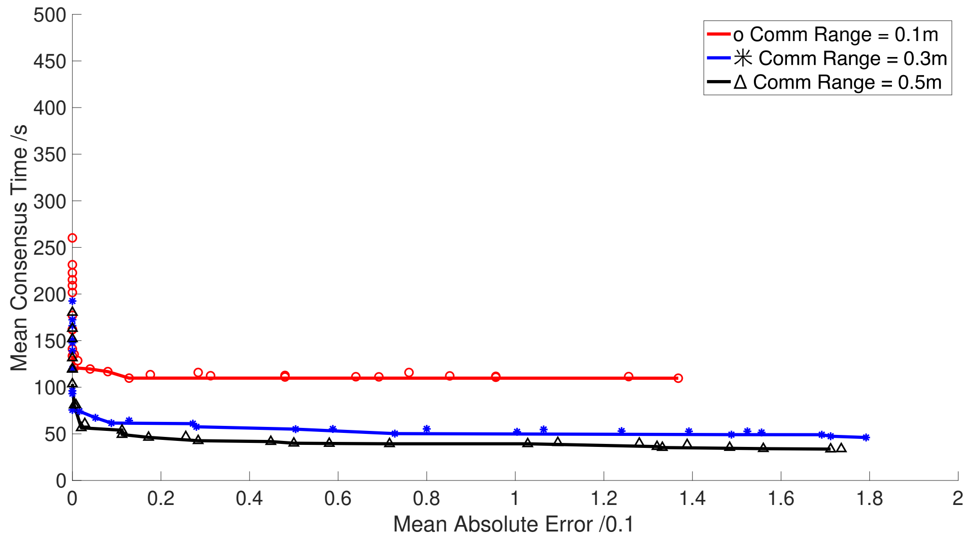


Figure 4.7: Plot of mean consensus time against mean absolute error for DBBS with different communication ranges when facing random distribution of environmental features [SM21b]

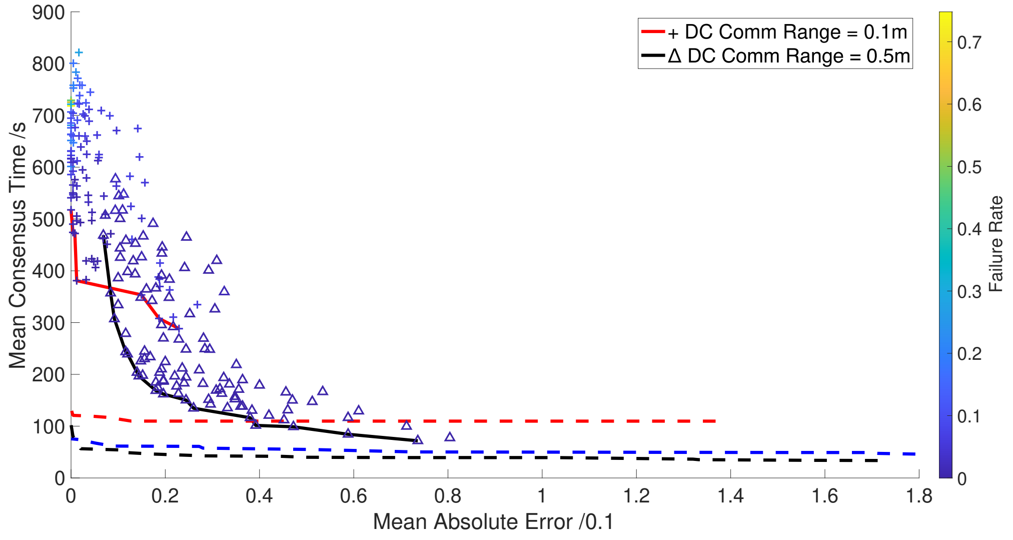


Figure 4.8: Plot of mean consensus time against mean absolute error for DC with different communication ranges when facing random distribution of environmental features; color coding of markers shows the failure rate; dashed lines show the Pareto frontiers produced by DBBS [SM21b]

consensus time is more significant than in DBBS, and DC is still outperformed in this situation.

The performances of the same communication range configurations in environments with concentrated feature distribution are shown in Figure 4.9. It can be observed that

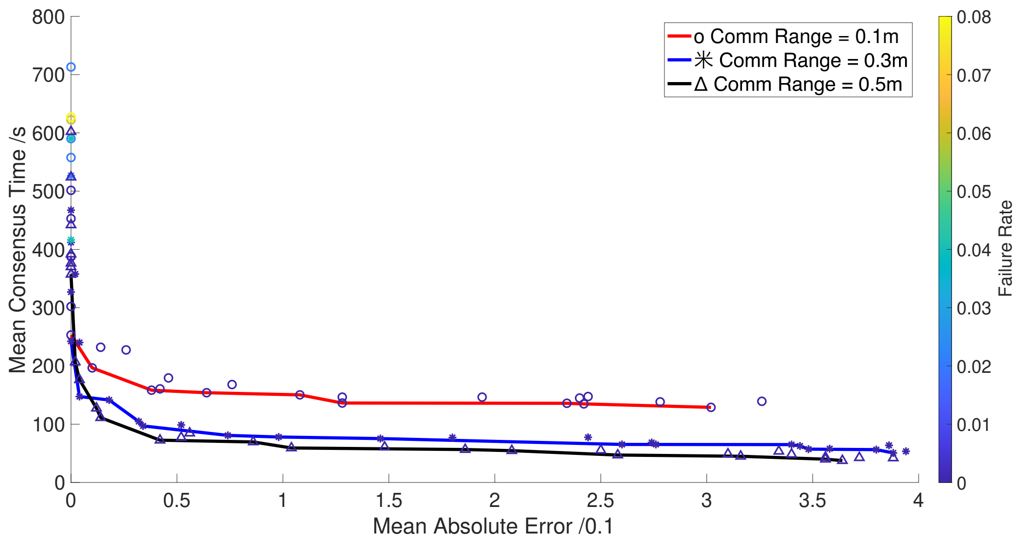


Figure 4.9: Plot of mean consensus time against mean absolute error for DBBS with different communication ranges when facing concentrated distribution of environmental features with $r_{black} = 0.45$, $w_{mean} = 12$; color coding of markers shows the failure rate [SM21b]

the change in performance as the communication range decreases is similar to the results in random environments. Due to the higher difficulty of decision-making in concentrated environments, the algorithm experiences some failures in decision-making when the communication range is at $0.3m$ and $0.1m$. However, the failure rate is very low and not exceeding 0.08. Also, the failures arise in mostly outlying parameter configurations that are not on the Pareto frontier. For a communication range of 0.1, an error of 0 can still be achieved at a consensus time of $253s$, which is well beyond the Pareto frontiers of other considered decision-making strategies.

4.2.4 Summary and Discussion

In this section, discrete collective consensus via direct fusion of Bayesian likelihoods is proposed and tested against opinion-based strategies in the discrete collective estimation scenario. The performances of the proposed Distributed Bayesian Belief Sharing (DBBS) strategy are analyzed and compared with those of existing opinion-based consensus strategies using a multi-criteria framework that takes into account the consensus accuracy, speed, and reliability.

As demonstrated in the experimental results, both the proposed DBBS approach and the baseline opinion-based strategies exhibit trade-offs between speed and accuracy in their parameter settings in operation. This is in line with related literature on binary collective perception [Ebe+20; Aus+22]. When applying collective decision-making strategies to scenarios with more than 2 options, the designers also need to contend with maintaining a diversity of opinions in the decision-making process for the agents to consider all of the available options without premature elimination, as well as coming to a consensus by the end of the run. This creates another trade-off on the design level between reliability and accuracy. This is demonstrated by the performances of DC, which is reliable but not accurate, and those of DMVD and Individual Exploration, which are accurate but unreliable.

It has been observed by [Tal+19] that, compared to stochastic switching methods, DC appears to be less accurate but faster, due to the higher amount of information transfer but also the propagation of errors across the swarm. A similar performance difference between DC and DMVD has been observed. From a multi-criteria perspective, the Pareto frontier for DC tends to have an overall high gradient in a consensus time versus error plot, causing the error to be inelastic and hard to be reduced by slowing down the consensus speed. Since in the implementation of DC, quality measurements are not adopted by other robots, a similar error propagation does not exist. Instead, DC is still able to enforce a consensus via an elitist selection mechanism. For example, a single robot that obtains a high option quality can make every robot with which it is in contact, change its decision. Thus, such a robot can possess a disproportionate amount of influence and can cause a premature convergence to a wrong option. In contrast, DMVD's Pareto frontier has an overall low gradient in a consensus versus error plot, causing the consensus time to be inelastic and not easily reduced by sacrificing accuracy.

DBBS is able to achieve better performances in terms of accuracy, speed, and reliability. Its better performance comes at a cost of higher communication complexity. The communication bandwidths required for DMVD and DC are both independent of the number of options. DMVD only needs the robots to transmit the chosen option, while DC needs the robots to transmit the chosen option and its corresponding quality. In contrast, DBBS requires the robots to transmit all estimated qualities. This causes the communication bandwidths required to scale with the number of potential options. In the scenario considered in this chapter, the ratio of communication bandwidths required for DMVD, DC, and DBBS is roughly $1 : 2 : 10$. Therefore, to implement DBBS on hardware, robots need higher communication bandwidths. On the other hand, DBBS has no need for robot identification, while in practice both DC and DMVD require the agents to broadcast a randomly generated 16-bit identifier to differentiate each other in a neighborhood [Val+16a].

DBBS's performance is also sensitive to its parameter settings. Especially, a high μ setting can cause strong positive feedback and produce a very high error. This effect is further magnified when the strategy is applied to an environment with a concentrated feature distribution. A potential solution is to implement a similar identifier as DC and DMVD, and use the memory of past received messages to enable the agents to avoid logging messages from the same sender in a short period of time.

When comparing the performances of considered strategies with their communication bandwidths required in mind, it can be concluded that generally, as the communication bandwidths increase between agents, there is a larger pressure on all agents to converge to a single consensus. This effect increases from Individual Exploration to DMVD and then to DC. This increased pressure for consensus has a positive influence on consensus speed and reliability, but can have a negative influence on the accuracy.

This shows the shortcomings of existing collective decision-making strategies based on finite state machines, in which agents hold only one option and try to converge to the optimal option through dissemination. The proposed strategy DBBS avoids this shortcoming by having agents communicate with each other using the belief of the options. Thus, the agents can hold multiple likely options simultaneously during dissemination. This feature of DBBS ensures a high diversity of opinions among agents and also reliable convergence to an optimal option.

When applied to more difficult environments with more concentrated features, DC and DMVD exhibit a significant increase in error, consensus time, and failure rate. While the performances of DBBS worsen mildly, with no increase in failure rate for most parameter settings. The increase in error can also be mitigated by parameter selection that slows the decision-making process. DBBS is also proven to be able to withstand the effects of sparse communications with only limited sacrifice of the consensus time.

When comparing the proposed DBBS strategies with other related collective decision-making strategies, DBBS shares much motivation with decision-making strategies based on opinion pooling [LLW18a; LLW18b; CLB19]. These strategies also attempt to perform multi-option collective decision-making by directly combining estimations of option qualities. The proposed approach differs firstly in that it is designed with a physics-based environment in mind, such that the bidirectional communication required for opinion pooling is abandoned in favor of a broadcast-based communication model. DBBS also takes advantage of the characteristics of the discrete collective estimation scenario and uses a Bayesian statistics-based method to compute option qualities. Therefore, DBBS has less communication and computational complexity than decision-making strategies based on opinion pooling.

4.3 Decentralized Multi-Option Collective Consensus Forming with Distributed Majority Voting

Majority-rule collective consensus-forming strategies, such as DMMD investigated in Chapter 3, form global consensus via first local consensus among agents that can directly communicate with each other with localized elections, where the majority opinion overwrites the minority opinion. With random interaction and enough mixture among the agents, the localized consensuses propagate across the swarm and lead to a global consensus. When extending the same decision mechanism into a multi-option best-of-n problem with $n > 2$, an increasingly likely scenario is that no single option reaches a majority in the localized elections. An important design choice is therefore the vote tallying system used to interpret the election results and compute the winning options. This section investigates the performances of different voting systems in creating a collective consensus in the multi-option best-of-n discrete collective estimation scenario.

In this section, the opinion-based consensus strategy of DMMD is extended into the multi-option scenario by adding a multi-candidate voting system into the decision mechanism. 3 voting systems have been investigated: first past the post (FPTP), single transferable vote (STV), and Borda count (BC). Majority-rule strategy with FPTP voting system is a straightforward extension of DMMD into the multi-option scenario that uses only the modulation of dissemination period lengths to express a robot's relative preferences of the options. The latter two are ranked voting systems where the preferences are also expressed with the rankings in the ballot.

4.3.1 Multi-Option Distributed Majority Voting Strategies

The basic underlying decision-making behaviors of the investigated distributed iterative voting strategies retain the probabilistic finite state machine consisting of exploration and dissemination states shown in Figure 4.2. The exploration state behavior uses the same Bayesian hypothesis testing approach to compute the option qualities as other investigated opinion-based strategies in the discrete collective estimation scenario.

The dissemination state behavior for the multi-option majority voting approach is shown in Algorithm 8. Similar to implementations of opinion-based strategies in binary consensus problems, the robot still always holds a chosen option d_i. This is then used to perform local majority voting under FPTP voting system. On the other hand, under ranked voting systems STV and BC, the robot expresses its relative preferences among the options in its ballot $\vec{b}_i$, which is computed in line 4 and subsequently broadcasted in line 7. At the start of its dissemination period, the robot collects ballots from its neighboring peers (line 5). It then tallies all ballots received using its voting system and switches to the winning option (line 11). The random mutation of decisions is retained from the previously investigated opinion-based strategies, to ensure diversity of opinions during the decision process (lines 13-15). The decision-making behavior of the proposed majority rule strategy is controlled by the same parameters σ and τ as the already investigated opinion-based strategies. σ controls the mean length of the exploration and dissemination periods, while τ controls the mutation probability of d_i at every control loop.

Algorithm 9 shows the computation of the ballot from every robot's decision and beliefs, for every considered voting system. Among them, a robot's ballot under first-past-the-post (FPTP) only includes the currently chosen option d_i (line 2). In contrast, for single transferable vote (STV) and Borda count (BC) voting systems, the robot produces a ranking of all options as its ballot. The robot's chosen option d_i is always ranked first

Algorithm 8 Basic dissemination behavior utilizing distributed majority voting [SHM21]

Input: Current decision of robot d_i; computed likelihood array during exploration $\vec{l_i}$ of length M
Output: Updated decision of robot d_i after dissemination with its peers

```
1:  Set parameters $\sigma$, $\tau$
2:  $J = \{\}$
3:  **while** Robot $i$ is in dissemination state **do**
4:      $\vec{b_i}$ = ComputeBallot($d_i, \vec{l_i}$) (shown in Algorithm 9)
5:      $\vec{b_j}$ = CollectNeighborBallot
6:      **if** $\vec{b_j}$ is collected **then**
7:          $J = J \cup j$
8:      Broadcast $\vec{b_i}$
9:      $v$ =sample from $\mathcal{U}[0, 1)$
10:     **if** $v < 1/(\sigma l_{i,d_i})$ **then**
11:         **if** $J \neq \emptyset$ **then**
12:             $d_i$ = VoteTally($\{\vec{b_j} | j \in (J \cup i)\}$) (shown in Algorithm 10)
13:             $J = \{\}$
14:             $v'$ =sample from $\mathcal{U}[0, 1)$
15:             **if** $v' < \tau$ **then**
16:                 $d_i$ = RandomChoice($d_i + 1, d_i - 1$) when valid
17:         Switch to exploration state
```

Algorithm 9 Function to generate ballot based on individual information $ComputeBallot(d_i, \vec{l_i})$ [SHM21]

Input: Chosen decision d_i, computed likelihood array $\vec{l_i}$ of length M
Output: Ballot $\vec{b_i}$ of agent i

```
1:  **if** Voting system is FPTP **then**
2:      $b_i = d_i$
3:  **else**
4:      # Voting system is STV or BC
5:      $\vec{l'_i} = \vec{l_i}$
6:      $\vec{l'_i}[d_i] = \infty$
7:      $\vec{b_i}$ = argsort($-\vec{l'_i}$)
```

to ensure the malleability of the robot's opinion (line 6). The other options are ranked according to the robot's own computed likelihoods from environmental exploration (line 7).

Algorithm 10 shows the vote tallying mechanism of the investigated voting systems. FPTP shown in lines 1-5 is a naive implementation of majority rule decision-making in multi-option best-of-n scenarios and serves as a baseline for the performances of the ranked voting strategies. Under FPTP, the robot simply counts the number of votes each option receives in vector $\vec{v}$ (lines 2-4) and then selects the option with the most number of votes regardless of whether an absolute majority is achieved (line 5). In case of a tie, a random option that receives the most number of votes is selected.

Algorithm 10 Function to tally collected ballots $VoteTally(B)$ [SHM21]

Input: Collected ballots during dissemination stored in set B
Output: Updated decision of robot d_i after tallying of ballots

```
1:  if Voting system is FPTP then
2:      $\vec{v} = 0s$ array of length equal to the number of options $M$
3:      for all $b \in B$ do
4:          $v_b = v_b + 1$
5:      $d_i = \text{argmax}_m(v_m)$
6:  else if Voting system is STV then
7:      while $max(v_m) \leq |B|/2$ do
8:          $\vec{v} = 0s$ array of length equal to the number of options $M$
9:          for all $\vec{b} \in B$ do $v_{b_1} = v_{b_1} + 1$
10:         Eliminate option $\hat{m} = \text{argmin}_m(v_m)$
11:         for all $\vec{b} \in B$ do
12:             for all $b_m$ in $\vec{b}$ do
13:                 if $b_m = \hat{m}$ then delete $b_m$ from $\vec{b}$
14:             if $\vec{b}$ is empty then delete $\vec{b}$ from $B$
15:     $d_i = \text{argmax}_m(v_m)$
16: else
17:     # Voting system is BC
18:     $\vec{v} = 0s$ array of length equal to the number of options $M$
19:     for all $\vec{b} \in B$ do
20:         $\vec{v}' = 0s$ array of length equal to the number of options $M$
21:         for all $b_m$ in $\vec{b}$ do $v'_{b_m} = m$
22:         $\vec{v} = \vec{v} + \vec{v}'$
23:     $d_i = \text{argmin}_m(v_m)$
```

The two ranked voting systems investigated are STV (lines 6-15) and BC (lines 17-23). Under STV, the robot first counts the number of votes received by each option when only considering the options ranked first on every ballot (line 9). When none of the options receives an absolute majority, the least picked option in terms of first choice is eliminated, and removed from all ballots (lines 10-14). This causes the ballots that picked the eliminated options as their first choice to transfer to their next choice in the ranking. This process continues until an option receives an absolute majority of first-choice votes, and that option will be picked as the new decision of robot i (line 15). Under BC, the robot first converts the ranked ballots into points (line 21). Thus, an option that is placed the first in a ballot receives one point, and an option that is placed the second receives two points, etc. The points received are then summed for every option separately (line 22). The winning option is the one with the lowest number of points (line 23).

4.3.2 Experiments and Results

In order to determine the effectiveness of the proposed majority rule consensus strategies in multi-option best-of-n problems, experiments are carried out in the same discrete collective estimation scenarios as before. The same performance metrics of absolute error, consensus time, and failure rate are measured. The performance of the Direct Comparison (DC) strategy is used as a baseline.

Parameter Settings of Ranked Voting Strategies

The decision accuracy and speed performances of the majority-rule strategy with STV voting system at various σ and τ settings are shown in Figure 4.10. The solid lines show the Pareto frontiers in terms of accuracy and speed at every τ setting. The markers correspond to different σ settings in their color codings.

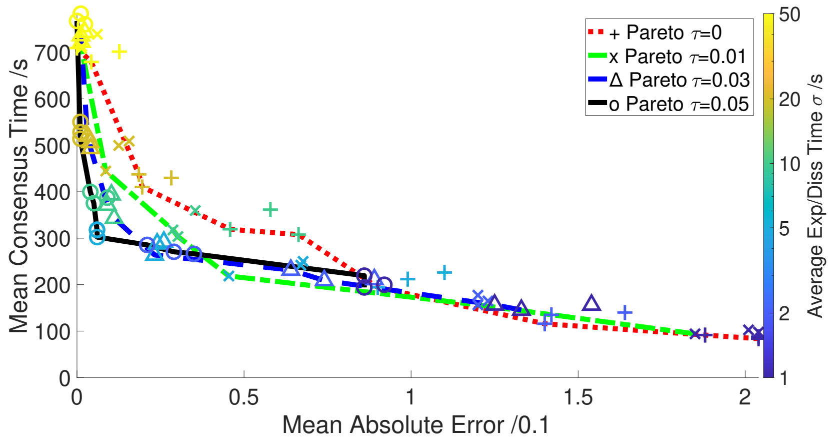


Figure 4.10: Performances of STV in environments with random feature distribution with respect to the mutation rate τ and mean exp/diss time σ [SHM21]

It can be observed that, similar to DC and DMVD in the previous section, adding the random opinion mutation mechanism controlled by τ leads to an increase in decision accuracy. In addition, although for a single σ setting, a higher τ setting usually leads to a longer decision time, the Pareto frontiers at higher τ settings outperform those at lower τ settings in most sections. A high τ setting of 0.05 is able to achieve a good performance of 0.0606 error and $302s$ consensus time at the bottom left. On the other hand, as shown by the color coding of markers, an increasing σ raises the mean consensus time significantly and also reduces the error produced. The performances also become insensitive to other parameters if σ is large, as shown in the clustering of data points at the top left when σ is $20s$ or $50s$.

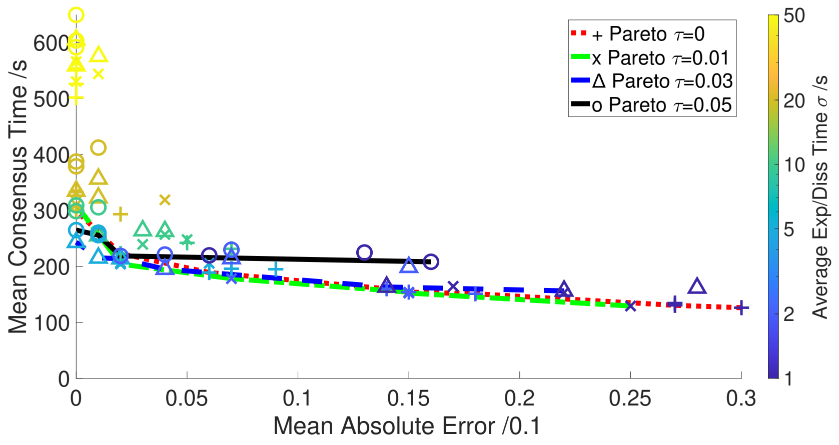


Figure 4.11: Performances of BC in environments with random feature distribution regarding the mutation rate τ and mean exp/diss time σ [SHM21]

The same plot is made for the majority-rule strategy with BC voting system in Figure 4.11. When comparing the performances of BC here with those of STV, it can be noticed that BC can achieve much lower errors of less than 0.05 within a shorter decision time of around $200s$. It can also be seen that σ has similar effects compared to STV. However, increasing τ does not straightforwardly increase accuracy as under STV. Notably, the Pareto frontier produced when $\tau = 0.05$ is completely dominated by that produced when $\tau = 0.03$. This shows a BC voting system inherently introduces diversity of opinions and thus does not need as high a setting of τ. On the other hand, increasing τ has the similar effects of reducing the lower bound of error obtained and increasing the lower bound of consensus time as in STV.

Comparison of Considered Strategies in Environments with Random Distribution of Features

The Pareto frontiers in terms of decision speed and accuracy 3 investigated majority-rule decision-making strategies together with that of DC baseline in environments with random feature distribution is shown in Figure 4.12.

As a naive implementation of the majority-rule strategy, FPTP outperforms DC at high consensus times of beyond $400s$. It, however, produces slightly higher failure rates, as shown by the color of the data points. In addition, when a decision needs to be reached quickly, the errors produced by FPTP increase rapidly at the bottom right portion of the Pareto frontier.

Compared to the two benchmark strategies, STV can achieve superior performance in the middle of its Pareto frontier, around the consensus time of $300s$. However, at a very

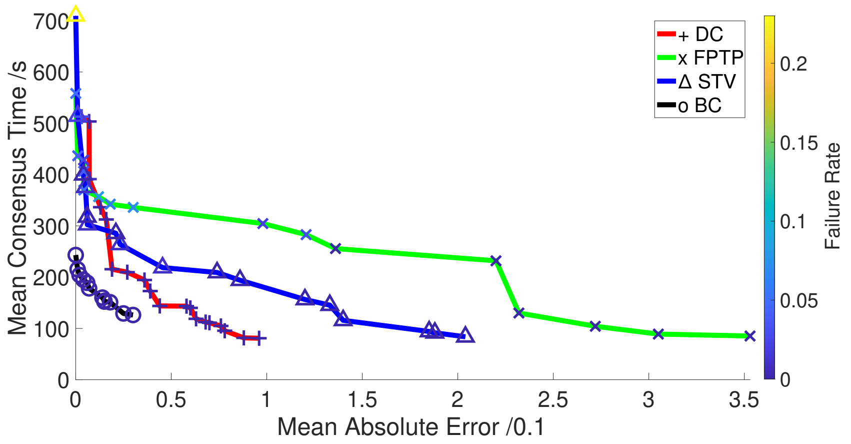


Figure 4.12: Pareto frontiers of consensus time versus absolute error for all considered strategies; color coding of markers shows the failure rates [SHM21]

high consensus time, its performance is dominated by that of FPTP, and at a very low consensus time, its performance is dominated by DC. In addition, it can achieve lower failure rates than FPTP when consensus time is high. On the other hand, BC can dominate most results produced by the other strategies, except at a very low consensus time.

Additionally, it is necessary to consider that the amount of required communication in these decision-making strategies is different. In DC, agents exchange both the chosen option and the corresponding quality estimate. In FPTP, agents exchange only the chosen option. While in the STV and BC, agents exchange the rankings of all options. Therefore, the communication bandwidth required of the 4 considered strategies would have the relationship of $FPTP < DC < STV = BC$. With this in mind, it can be seen that STV only provides a situational improvement in performance over both benchmarks. In contrast, BC can display superior performance and reach an error of 0 at a far lower consensus time than the others.

Comparison of Considered Strategies in Environments with Concentrated Distribution of Features

This subsection investigates the performances of the same majority-rule strategies in environments with concentrated feature distribution. The same test environments as in the previous section are used. The Pareto frontiers of the considered strategies' performances are shown in Figure 4.13. The Pareto frontiers obtained earlier in random environments are shown in dashed lines.

There is a significant drop in performances for all considered strategies compared to in random environments. Both FPTP and STV experience an increase in the error primarily. However, fast convergences can still be achieved, as shown on the bottom right end of the Pareto frontiers. On the other hand, DC and BC experience an increase in both decision

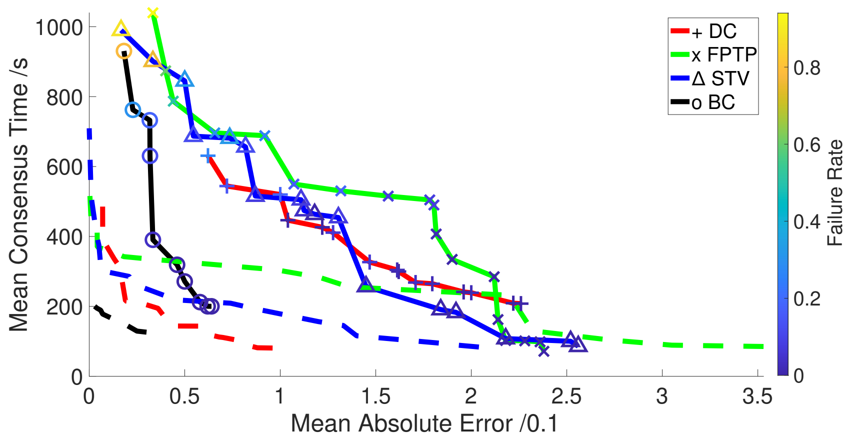


Figure 4.13: Pareto frontiers of consensus time versus absolute error for all considered strategies in environments with concentrated feature distribution; color coding of markers shows the failure rates; dashed lines show the Pareto frontiers achieved previously in random environments [SHM21]

time and error.

Among the considered strategies, BC's performances are the least elastic in terms of error, and extending the decision time has very little effect in reducing the error, while for all other strategies, there are apparent and linear trade-offs between the two metrics. On the other hand, BC still has superior performance compared to the other strategies. However, at higher decision times, the performances of BC come very close to those of FPTP and STV.

4.3.3 Summary and Discussion

In the experimental results above, it is demonstrated that BC is a promising technique in multi-option collective decision-making problems. It can significantly outperform the baseline approaches of DC and FPTP in the scenarios investigated. There is a parallel between the ranked voting mechanism of BC used here and the direct Bayesian belief fusion approach proposed in the previous section. The ranked voting mechanism of BC achieves a limited form of opinion fusion with a predetermined set of beliefs, which are the point allocations used in the tallying process. Compared with full opinion fusion, this design choice has two advantages. First, transmitting the ranking of options takes up less communication bandwidth than transmitting the associated qualities, and thus can be achieved with cheaper equipment. Second, limiting the propagation of option qualities can minimize the impact of extreme or faulty estimations on the whole swarm, as indicated in [Tal+19].

On the other hand, STV fails to significantly outperform the benchmarks while using more communication bandwidth. It is caused by the stochasticity in the decision-making process of STV. In real-life elections, STV rarely deals with situations with more candidates than voters, which is frequently the case in the investigated scenario. When multiple options receive no first preferences during voting, the elimination process will eliminate a random option among them. This can cause valid options to be prematurely eliminated. In a typical swarm intelligence setting, the decision-making strategy needs to form a decision based on the information in a small locality and thus STV struggles in such environment. It is however capable of faster convergence than BC, as it is better at eliminating unfavorable options quickly.

FPTP is frequently the worst-performing strategy among the 4 considered. In FPTP, the chosen options are only selected from the first choices of the voters, causing inadequate information transfer among the agents. However, it uses the least communication bandwidth, and therefore should only be considered a viable consensus strategy when the communication needs to be minimized. Otherwise, a ranked voting system should be utilized in a similar collective decision-making scenario.

In the experiments in this chapter, the frequency of message passing is kept the same across different considered strategies. This comparison framework does not take into consideration the different message sizes under the different considered strategies, and therefore cannot address quantitatively the trade-off between communication complexity and decision performance. This problem is magnified when facing multi-option consensus problems in larger decision spaces, where the message sizes of both parallel encoding consensus approaches scale linearly with the decision space size. The next two chapters seek to address this limitation via bandwidth-controlled experiments.

Chapter 5

Investigating the Scalability of Parallel Encoding Consensus Approaches with Bandwidth-Controlled Experimentation

This chapter is based on [SM22].

Under the two previously proposed parallel encoding consensus approaches, the agents need to transmit their relative preferences regarding all options to each other during the decision-making process. This limits such approaches' scalability when facing a high number of discrete potential options. This effect has not been fully shown in Chapter 4, when the comparison among the considered strategies was done with the frequency of message passing kept the same. In this chapter, a new comparison framework is introduced, where the rate of information transfer is kept the same across the considered strategies during experiments by adjusting the frequency of message passing. This gives more insights into the relative efficiency of the considered consensus strategies when utilizing inter-agent communication, and shows the changes to their relative viabilities when facing discrete decision spaces of different sizes.

5.1 Bandwidth-Controlled Comparison Framework

This section shows in detail the framework used to compare the performances of the considered strategies with the communication bandwidths and paradigm kept the same. The implementations of the considered strategies in Chapter 4 are changed to adjust the communication bandwidth by tuning the probability of message passing at every control loop. In addition, the communication paradigm is kept strictly peer-to-peer without the need for unique identifications on the agents to maintain a fair comparison.

5.1.1 Modified Implementations of the Considered Consensus Strategies

This subsection presents the updated implementations of the considered strategies and explains how the communication bandwidths are controlled in each strategy.

Opinion-Based Strategies: DC and DMVD

As discussed in Chapter 4, the finite state machine that governs the decision-making behaviors of the studied opinion-based strategies requires the robots to have unique ID numbers so that duplicated messages are not collected during dissemination periods. Therefore, in order to make their communication paradigm in line with that of DBBS and peer-to-peer, the implementations of DC and DMVD strategies are modified as shown in Algorithm 11.

Algorithm 11 Modified implementation of DC and DMVD [SM22]

Input: Hypothesis matrix H of size $M \times 2$; robot index $i \in \{1..N_{robot}\}$; initialized belief of robot i: $\vec{l_i}$ of length M; initialized decision of robot i: d_i

Output: Converged decisions: d_i

```
1:  Set broadcasting probability $\phi$ and mutation probability $\tau$
2:  while Decisions in swarm have not converged do
3:      if Robot $i$ is moving forward in motion State A then
4:          Collect observation and compute $\vec{o}$
5:          $\vec{l_i} = \text{Normalize}(\vec{l_i} \circ (H \cdot \vec{o}))$
6:      DC: $(d_j, l_{j,d_j})$ = CollectNeighborOpinion
7:      DMVD: $d_j$ = CollectNeighborOpinion
8:      if $d_j$ is collected then
9:          DC : If $l_{j,d_j} > l_{i,d_i}$: $d_i = d_j$
10:         DMVD: $d_i = d_j$
11:     $v$ =sample from $\mathcal{U}[0,1)$
12:     if $v < \tau$ then
13:         $d_i = \text{RandomChoice}(d_i + 1, d_i - 1)$ when valid
14:     Broadcast $d_i$, $l_{i,d_i}$ randomly at probability DC: $\phi$; DMVD: $\phi \times l_{i,d_i}$
```

The separate exploration and dissemination strategies are combined here into one control loop, with lines 3-5 performing exploration and lines 6-14 performing dissemination. The parameter ϕ controls the probability of the robot to broadcast its opinion at every control loop (line 14). For DC strategy, a setting of $\phi = 1$ means that the robot i broadcasts its opinions (d_i, l_{i,d_i}) at every control loop. Lower ϕ values cause the broadcasting probability to decrease, thus reducing the communication bandwidth between robots. For DMVD strategy, the broadcasting probability is further scaled by the quality of the robot's chosen option l_{i,d_i}. This replaces the mechanism of scaling the average length of dissemination periods, and ensures convergence to the option with the highest quality.

Parallel Encoding Approaches: RV and DBBS

Based on the findings in Chapter 4, the ranked voting consensus strategy with Borda count tallying system (RV-BC) is selected for further study because it outperforms the other investigated voting systems. Similar to the opinion-based strategies, it is also modified to have a unified exploration and dissemination behavior in one control loop, as shown in Algorithm 12.

Algorithm 12 Ranked Voting with Borda Count (RV-BC) [SM22]

Input: Hypothesis matrix H of size $M \times 2$; robot index $i \in \{1..N_{robot}\}$; initialized belief of robot i: $\vec{l_i}$ of length M; initialized decision of robot i: d_i

Output: Converged decisions: d_i

```
1:  Set broadcasting probability $\phi$, ballot length $\eta$ and mutation probability $\tau$
2:  Initialize $\vec{b_i}$ with zeros
3:  while Decisions in swarm have not converged do
4:      if Robot $i$ is moving forward in motion State A then
5:          Collect observation and compute $\vec{o}$
6:          $\vec{l_i}$ = Normalize($\vec{l_i} \circ (H \cdot \vec{o})$)
7:      $\vec{b'_j}$ = CollectNeighborBallot
8:      if $\vec{b'_j}$ is collected then
9:          $d_i$ = VoteTally($\{\vec{b_i}, \vec{b_j}\}$)
10:     $v$ =sample from $\mathcal{U}[0, 1)$
11:     if $v < \tau$ then
12:         $d_i$ = RandomChoice($d_i + 1, d_i - 1$) when valid
13:     $\vec{b_i}$ = ComputeBallot($d_i, \vec{l_i}$)
14:     $\vec{b'_i} = [b_{i,1}..b_{i,\eta}]$
15:     Broadcast $\vec{b'_i}$ randomly at probability $\phi \times l_{i,d_i}$
```

The communication bandwidth between robots under RV-BC is tuned via two parameters: broadcasting probability ϕ and ballot length η. The effect of ϕ is the same as in the previous subsection. On the other hand, when the ballot length η is smaller than the number of options, the robot shortens its ballot to length η by removing the least preferred options in lines 13-14 of Algorithm 12. The shortened ballot is referred to as $\vec{b'}$ in the pseudocode.

The bandwidth-controlled implementation of the DBBS approach, shown in Algorithm 13, is largely the same as in Chapter 4. The only difference is the addition of the same broadcasting probability parameter ϕ being added (line 10).

Algorithm 13 Distributed Bayesian Belief Sharing (DBBS) with bandwidth control [SM22]

```
Input: Hypothesis matrix $H$ of size $M \times 2$; robot index $i \in \{1..N_{robot}\}$; initialized belief of robot $i$: $\vec{l_i}$ of length $M$; initialized record of neighbors' beliefs : $\vec{l'_i}$ of length $M$
Output: Converged decisions: $d_i$
1:  Set decay parameters $\lambda$, $\mu$ and broadcasting probability $\phi$
2:  while Decisions in swarm have not converged do
3:      if Robot n is moving forward in motion State A then
4:          Collect observation and compute $\vec{o}$
5:          $\vec{l_i} = \text{Normalize}(\vec{l_i} \circ (H \cdot \vec{o}))$
6:      $\vec{\xi_j}$ = CollectNeighborOpinion
7:      if $\vec{\xi_j}$ collected then
8:          $\vec{l'_i} = \text{Normalize}(\vec{l'_i}^{\lambda} \circ \vec{\xi_j})$
9:      $\vec{\xi_i} = \vec{l_i} \circ \vec{l'_i}^{\mu}$
10:     Broadcast $\vec{\xi_i}$ randomly at probability $\phi$
11:     $d_i = \text{MaxIndex}(\vec{l_i} \circ \vec{l'_i})$
```

5.1.2 Assumptions on Message Sizes for Each Considered Collective Consensus Strategy

The message formats during inter-agent communication under the considered consensus strategies are shown in Table 5.1. The corresponding message sizes in $bits$ are computed based on available data types in c++.

Table 5.1: Assumptions on the message format and sizes for all considered strategies [SM22]

Decision-making Strategy	Message Format
DMVD	short int (16 bits)
DC	short int + float (48 bits)
DBBS	$M\times$ float ($32M$ bits)
RV-BC	$\eta\times$ short int (16η bits)

In DMVD, the robots exchange only the index of their current decisions during dissemination, and the assumption is made that the information is stored in a $short\ int$ variable of 16 bits. Similarly, in DC, the robots exchange the index of their current decisions as well as the estimated qualities, which are stored using a $short\ int$ and a $float$ respectively, adding to a total of 48 bits. In DBBS, the robots exchange the estimated qualities of all considered options, which consists of M $float$ variables, where M is the number of options, making the message size $32M$ bits. Finally, in RV-BC, the robots exchange the rankings of the options, which are stored in η $short\ int$ variables, with η being the ballot length parameter. The message size is thus 16η bits.

5.2 Experiments and Results

This section covers the experimental setup and the parameter settings used to control the communication bandwidths of the considered strategies. Afterward, the experimental results are presented.

Table 5.2: Parameter settings for each investigated strategy to reach the desired bandwidths levels [SM22]

	Bandwidths			
Decision-making Strategy	1.6 bits/s	3.2 bits/s	8 bits/s	16 bits/s
DMVD	$\phi = 0.1$	$\phi = 0.2$	$\phi = 0.5$	$\phi = 1$
DC	$\phi = 0.1/3$	$\phi = 0.2/3$	$\phi = 0.5/3$	$\phi = 1/3$
DBBS	$\phi = 0.005$	$\phi = 0.01$	$\phi = 0.025$	$\phi = 0.05$
RV-BC $\eta = 10$	$\phi = 0.01$	$\phi = 0.02$	$\phi = 0.05$	$\phi = 0.1$
RV-BC $\phi = 0.1$	$\eta = 1$	$\eta = 2$	$\eta = 5$	$\eta = 10$

The performances of considered strategies are measured at 4 different bandwidth levels as shown in Table 5.2, together with the parameter settings for all considered algorithms to limit the communication bandwidths to those levels. The experimental environments are unchanged compared to in Chapter 4. At each bandwidth level, the same multi-criteria framework is used, and the Pareto frontiers of performances in terms of mean absolute error and mean consensus time are plotted for different levels of communication. For DC, DMVD and RV-BC, the Pareto frontiers are obtained using the settings of $\tau = \{0, 0.01, 0.02, ...0.08\}$, while for DBBS, the Pareto frontiers are obtained using the settings of $\lambda = \{0.5, 0.6, 0.7, 0.8, 0.9\}$ and $\mu = \{0, 0.2, 0.4, 0.6, 0.8, 1\}$.

5.2.1 Comparison of the Considered Strategies

The obtained Pareto frontiers between mean consensus time and mean absolute error are plotted in Figure 5.1. The color coding of markers represents the failure rates.

As shown in Figure 5.1 (a), at the lowest considered bandwidth of $1.6\ bits/s$, all considered strategies display clear trade-offs between decision speed and accuracy. They are also able to keep failure rates low at less than 0.1 for most parameter settings, except for DC with τ settings higher than 0.03 where the failure rates increase rapidly and approach 1 when $\tau >= 0.05$. This is contrary to the more reliable performances of DC observed in Chapter 4. Compared to DC, DMVD has comparable performances at high consensus time and can outperform DC at low consensus time. It also has lower failure rates across all parameter settings. Among the considered strategies, DMVD can reach a consensus

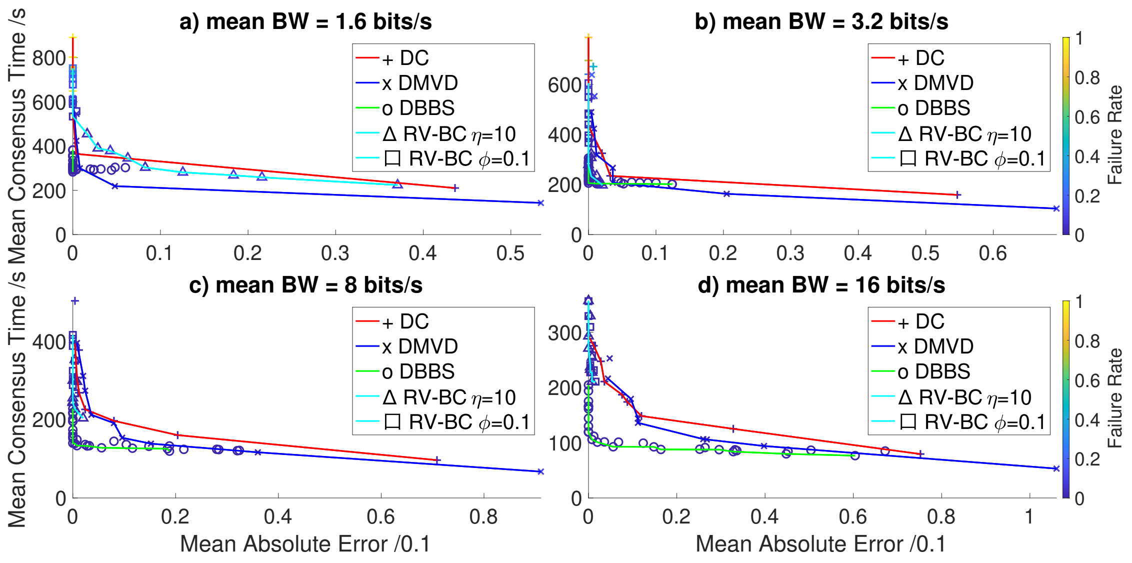


Figure 5.1: Experimental results obtained by the considered strategies when facing random distribution of environmental features [SM22]

the fastest at low τ settings, although at the cost of higher error. DBBS is able to reach an error of zero in a shorter time than other considered strategies, but has a hard time further reducing its consensus time. RV-BC has relatively poor performances in terms of decision time at error at this bandwidth level and its Pareto frontier is largely dominated by those of DBBS and DMVD. The upper left section of the Pareto frontier, produced by limiting the communication probability ϕ, results in very low error, but the consensus time is relatively high, while the lower right section, produced by limiting the ballot size η, results in relatively high error with not enough reduction in consensus time to outperform other considered strategies significantly.

As shown in Figure 5.1 (b-d), as the communication bandwidth increases, all considered strategies experience improvements in their performances, and are able to achieve progressively lower error at shorter decision times and lower failure rates. Among them, DBBS has the most significant improvements and progressively outperforms the other algorithms at the entire Pareto frontier at higher bandwidths. On the other hand, RV-BC has significant improvements in its error, but the algorithm has difficulties coming to a fast decision even at the highest communication bandwidth of $16\ bits/s$. At the same time, the performances of RV-BC become increasingly inelastic regarding the parameters and the results largely cluster together.

The performances of considered strategies in environments with concentrated feature distribution are shown in Figure 5.2. It can be observed that all considered strategies experience a significant performance drop in all three metrics compared to in random environments. It is difficult for all strategies to achieve an error of zero without experiencing high failure rates. This is especially apparent at low communication bandwidths shown

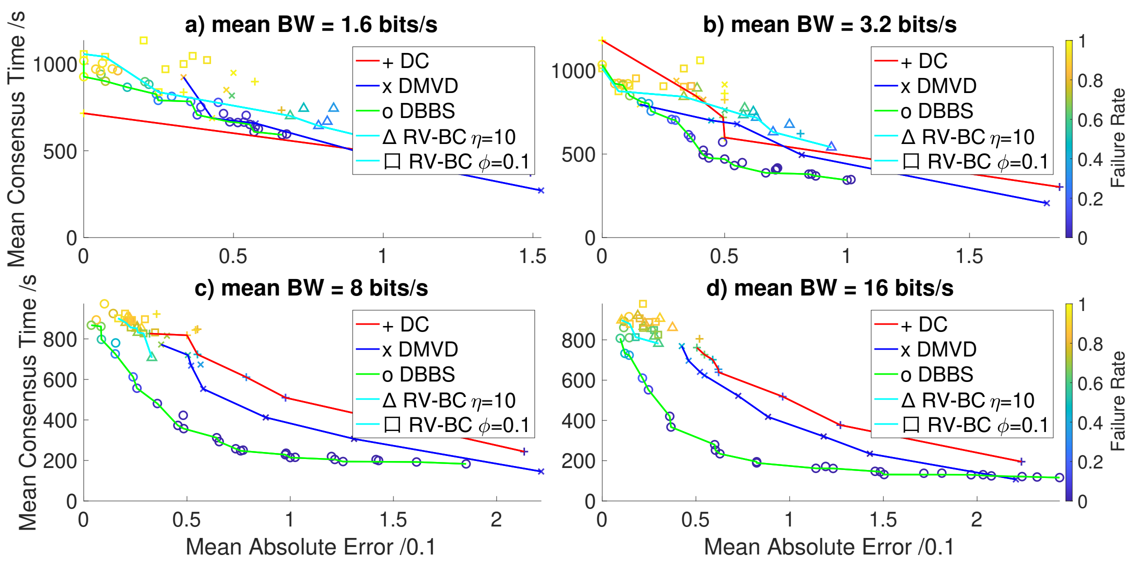


Figure 5.2: Experimental results obtained by the considered strategies when facing concentrated distribution of environmental features [SM22]

in Figure 5.2 (a, b), where the data points at the top-left side of the Pareto frontiers have failure rates approaching 1. As the communication bandwidth increases, all considered strategies experience a decrease in failure rate and decision time, but an increase in error.

Among the considered strategies, DBBS still outperforms the others during most of the Pareto frontiers in terms of decision speed and accuracy, especially at higher communication bandwidths. It is also able to achieve lower failure rates at equivalent decision time, demonstrating its reliability in more difficult environments. On the other hand, RV-BC displays slightly worse performances at lower bandwidths compared to DC and DMVD, while continuing to display inelastic performances at higher bandwidths.

5.2.2 Scalability to Number of Options

The number of hypotheses M is then varied to investigate the scalability of the considered strategies to the decision space size. Figure 5.3 shows the changes in performances of the considered strategies with the number of options in random test environments with the communication bandwidth being $8\ bits/s$.

It can be observed that all considered strategies face an increase in consensus time, error, and failure rate when facing a higher number of options. This is especially prevalent for the opinion-based strategies DC and DMVD (Figure 5.3 a,b), both of which are unable to achieve an accurate and reliable decision when $M = 40$, as shown on the top-left sections of their Pareto frontiers. However, by tuning the mutation probability τ lower, both strategies are still able to come to a fast and reliable decision at the bottom-right sections of their Pareto frontiers, albeit at a lower accuracy. This is because the communication

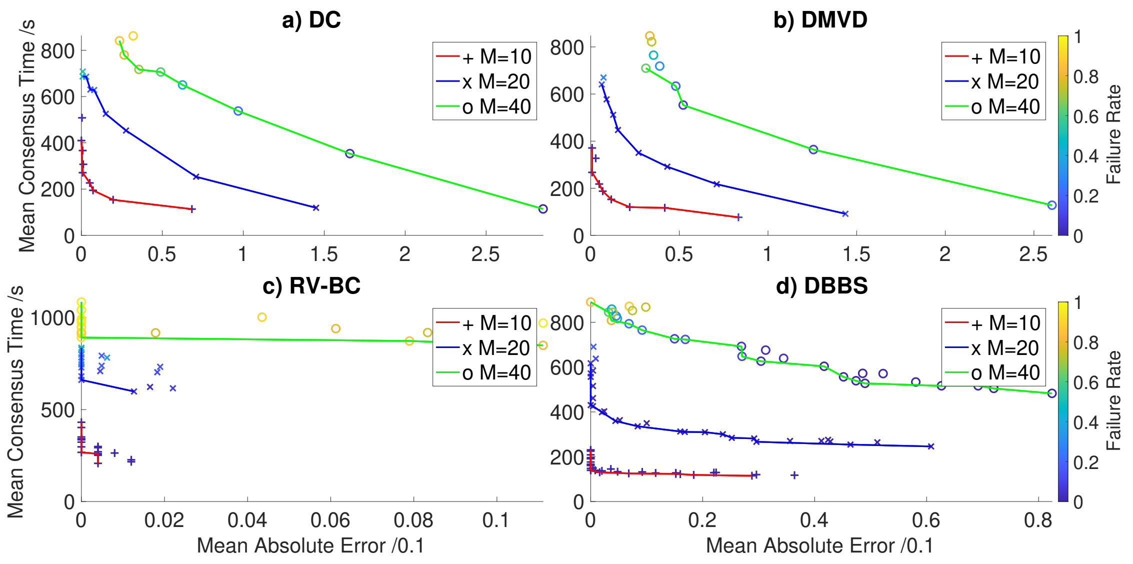


Figure 5.3: Decision-making speed and accuracy trade-off of the considered strategies when facing random distribution of environmental features with M varied ($BW = 8\,bits/s$); color coding of markers shows the failure rate

behaviors of DC and DMVD are not affected by the number of potential options, so a fast consensus can be achieved by limiting the degree of randomness in the decision-making process and having a low τ setting. However, when an accurate decision is required, a high M value means that there are more incorrect options to be eliminated, therefore causing higher consensus time and failure rate at higher τ settings.

RV-BC and DBBS (Figure 5.3 c,d), on the other hand, both experience an increase in consensus time and failure rate across their entire Pareto frontiers as M increases. Both strategies are still able to achieve accurate decisions by sacrificing some speed and reliability, as shown by the top-left sections of their Pareto frontiers. In contrast, at the bottom-right sections, both strategies are unable to achieve a fast decision similar to DC and DMVD at higher M values. The reason for this is the experimental setup that scales the broadcasting probability ϕ and ballot size η with M for both RV-BC and DBBS. This design aims to keep the bandwidth constant, but it also drastically reduces the decision speed of both strategies. However, the more sporadic messages carry more information than those under DC and DMVD, enabling the swarm to consider the potential options in parallel and leading to more accurate decisions at higher M values.

Figure 5.4 shows the performances of the considered strategies at different considered bandwidth levels when M is set to 40. Comparing the performances demonstrated here with those in Figure 5.1, there is an expected drop in all metrics. Across the considered strategies, the opinion-based strategies, DC and DMVD, have a more clear advantage over RV-BC and DBBS at lower communication bandwidths. DBBS is only able to overtake DC and DMVD in Pareto frontiers at $BW = 8\,bits/s$. The results here show that in-

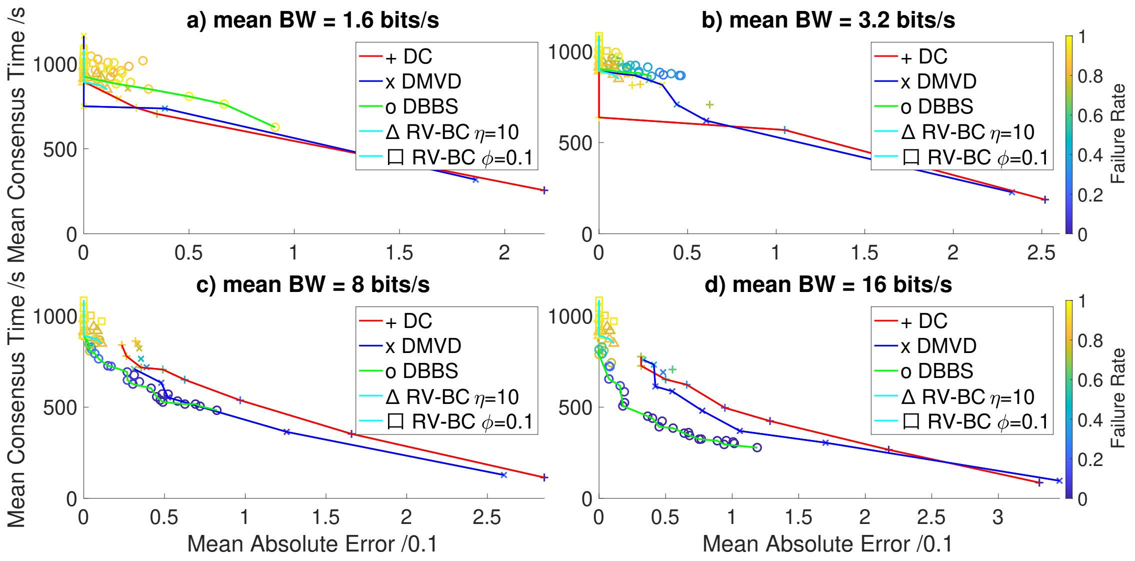


Figure 5.4: Decision-making speed and accuracy trade-off of the considered strategies when facing random distribution of environmental features ($M = 40$); color-coding of markers shows the failure rate

creasing the number of options has a bigger impact on the performances of RV-BC and DBBS than the opinion-based approaches, DC and DMVD. The impact on DBBS is more controllable and can be mitigated by increasing the broadcasting probability ϕ and hence communication bandwidth. Nonetheless, the opinion-based approaches have a clearer advantage at low bandwidth levels. At the same time, RV-BC is very unreliable for this scenario, with the failure rate across the Pareto frontier beyond 80%. RV-BC is limited by the small election size dictated by the communication paradigm defined in this chapter, which produces a high level of stochasticity in the decision-making process, leading to accurate but very slow decision making. This effect is multiplied as the number of options increases, causing poor convergence capabilities.

5.3 Summary and Discussion

Comparing the performances observed in this chapter with those of the same consensus strategies in Chapter 4, it can be observed that when controlling the communication bandwidth and paradigm, DC experiences a significant performance drop and is consistently comparable or worse compared to DMVD all along the Pareto frontier. This demonstrates that directly transmitting option qualities for comparison is not an efficient use of limited communication bandwidth. On the other hand, DC is more adversely affected by the peer-to-peer communication paradigm used in this chapter, which limits the number of options seen by individual robots and increases the number of message transmissions needed for high-quality options to propagate among the robots. This slows down the consensus process. The same effects are also observed for RV-BC, which displays slower but still accurate consensus performances compared to the results shown in Chapter 4. This is mainly caused by a reduced neighborhood size, which makes the voting process prone to ties. However, despite its reduced performances, it is still often able to outperform the opinion-based approaches at higher consensus times, especially at higher communication bandwidths. In contrast, DMVD is less impacted by the restriction in communication bandwidth and paradigm. This is because it only transmits the index of the chosen option to its peers, and the modulation of communication probability enables good options to propagate in a locality even when communication is only restricted to peer-to-peer. DBBS still clearly outperforms the other considered strategies when controlling the communication bandwidth, especially at higher bandwidths of 8 $bits/s$ and 16 $bits/s$. On the other hand, the difference is not as prevalent at lower bandwidths.

Both opinion-based approaches also experience an oversaturation of communication at high bandwidth levels, which leads to a faster consensus but higher error in decision making. This reflects the finding in [Tal+21; Aus+22] where an increase in communication has an adverse effect on the decision-making process in limiting the adaptability of the swarm. Although both aforementioned studies address decision-making scenarios with dynamic environments, the finding translates to the context of multi-option decision making in static environments, where adaptability is crucial in convergence towards the correct option out of multiple suboptimal ones. Without a high level of adaptability, a swarm can easily converge to a suboptimal consensus without adequate exploration of the environment. In contrast, DBBS experiences oversaturation of communication at high bandwidth levels to a smaller degree. The positive feedback mechanism used in belief fusion also increases the enforcement of consensus with more frequent communication. However, with parallel consideration of all available options, DBBS is able to partially mitigate the increase in error and achieve substantial strides in its Pareto frontiers of speed vs accuracy relationship compared to at lower bandwidth levels. The strategy does become more sensitive in terms of parameter settings when bandwidth increases, as too strong positive feedback can significantly increase the error in decision making.

As the number of available options increases, the viabilities of both multi-option approaches, RV-BC and DBBS, are more adversely affected than those of the opinion-based approaches. This is because for RV-BC and DBBS, the size of an individual message passed scales with the number of available options. This shortcoming is especially prevalent at lower bandwidth levels. Opinion-based approaches of DC and DMVD also face challenges in scenarios with an increased number of options, namely the increased number of incorrect options to be eliminated. Overall, a higher number of options gives advantages to the opinion-based approaches. Although DBBS can still achieve a superior performance at higher bandwidth levels, the bandwidth required for it to be the optimal strategy is increased when facing a higher number of options. The next chapter further investigates the effects of large decision spaces on the performances of investigated parallel encoding approaches to consensus forming and seeks to map the upper limit of their viabilities.

Chapter 6

Towards Many-Option Collective Decision Making: Impact of Number of Options and Quality Correlation on the Performances of Parallel Encoding Discrete Consensus Approaches

This chapter is based on [SM24].

As the number of options in a discrete consensus problem increases, it becomes closer to a continuous consensus problem. Thus, the proposed parallel encoding discrete consensus approaches need to be compared against continuous consensus approaches to determine the limit of their viabilities. In addition, the discrete collective estimation scenario studied in Chapters 4 and 5 focuses on discrete options that are strongly correlated to each other in terms of qualities, thus limiting the applicability of the results on general best-of-n problems. In this chapter, the multi-feature many-option discrete collective estimation scenario is introduced to address both points. It tests the performances of the proposed parallel encoding consensus approaches when the number of options significantly exceeds that of the agents. Their performances are compared with that of the Linear Consensus Protocol (LCP), which treats the decision space as continuous. This chapter also extends the aforementioned discrete collective estimation scenario to include more environmental features that serve to add weakly correlated options in terms of quality, facilitating investigation into the impact of the quality distribution on collective decision-making processes.

6.1 Multi-Feature Many-Option Collective Estimation Scenario

This section extends the discrete collective estimation scenario investigated in Chapters 4 and 5 in terms of the problem complexity to include more features and discretized hypotheses for the fill ratio of every feature. This causes changes to both the problem definition and the algorithm used to generate the required environments.

6.1.1 Scenario Configuration

The investigated multi-feature collective estimation scenario keeps most of the characteristics of the base version presented in Chapter 4. The extended scenario adds flexibility in the environmental settings by allowing C different features, with the fill ratio hypotheses of each feature discretized to a precision P. Both parameters can be freely adjusted. C can take the value of any integer greater than 1, while P can be any divisible fraction of 1. Examples of the new experimental environments are shown in Figure 6.1, with red, yellow and blue colors representing the features.

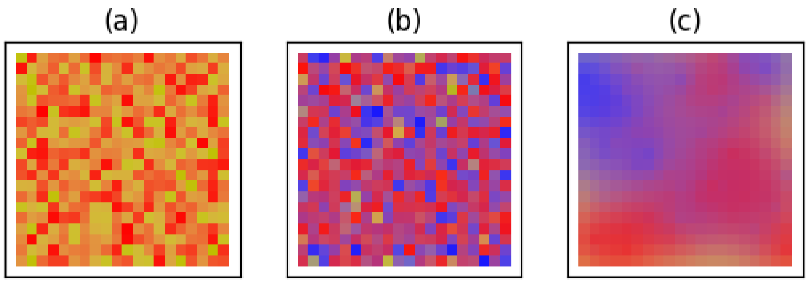


Figure 6.1: Examples of the experimental environments used (a) environment with 2 features (b) environment with three features (c) environment with concentrated distribution of features [SM24]

The tiles in the experimental arena can be expressed in a matrix as follows:

$$E = \begin{bmatrix} \vec{e}_{1,1} & \vec{e}_{1,2} & \vec{e}_{1,3} & \dots \\ \vec{e}_{2,1} & \vec{e}_{2,2} & \vec{e}_{2,3} & \dots \\ \vec{e}_{3,1} & \vec{e}_{3,2} & \vec{e}_{3,3} & \dots \\ & \dots & & \end{bmatrix}. \tag{6.1}$$

In order to accommodate the flexibility in environmental settings, each tile has a mixture of multiple features present. For a tile on the α^{th} row and β^{th} column, the proportions of the features are expressed using vector $\vec{e}_{\alpha,\beta}$. The real number entry $s_{\alpha,\beta,c}$ represents the proportion of the feature c among all C features in tile (α,β). All feature proportions in a single tile sum up to 1:

$$\vec{e}_{\alpha,\beta} = \begin{bmatrix} s_{\alpha,\beta,1} & s_{\alpha,\beta,2} & s_{\alpha,\beta,3} & \dots & s_{\alpha,\beta,C} \end{bmatrix}, \sum_{c \in C} s_{\alpha,\beta,c} = 1. \tag{6.2}$$

The robot swarm's goal in this scenario is to collectively determine the fill ratios for all present features. For the discrete collective decision-making strategies investigated, this is done to a predetermined decision space discretization precision P, which is varied together with the number of features C to change the number of options the robots face. A single decision by robot i regarding the fill ratios of the features is an estimation of those of all present features in the form of $\vec{d_i} = \begin{bmatrix} d_{i,1} & d_{i,2} & \dots & d_{i,C} \end{bmatrix}$. The discrete decision space is a high-dimensional right-angle simplex, where the number of dimensions is $C-1$ and the number of possible options per dimension is $1/P$. The whole decision space has M distinct fill ratio options. M has the scale of $O(\frac{1}{P}^C)$.

The number of potential options scales exponentially with the number of features C and inversely with the level of discretization precision P. Both methods of changing the number of options are investigated. The key distinction between increasing the number of features and the level of discretization precision is that the former introduces unordered and weakly correlated decisions in terms of quality by expanding the decision space, while the latter introduces ordered and strongly correlated decisions in terms of quality by more finely dividing an existing decision space.

6.1.2 Generation of Experimental Environment

In line with previous chapters, the considered consensus strategies are experimented in environments with different distributions of features, namely random distribution (Figure 6.1 (a,b)) and concentrated distribution (Figure 6.1 (c)). The generation process of the two categories of environments is shown in Algorithm 14. The distribution pattern is controlled by the parameter Pattern, while the level of concentration is controlled by the parameter γ.

Environments with randomly distributed features are produced by randomly generating a feature composition for every feature at every tile that has the mean equal to the desired fill ratio across the entire arena (lines 3-8), and then normalizing all feature proportions in every tile such that they sum up to 1 (line 9). On the other hand, a concentrated pattern of distribution is generated through an iterative process, where layers of radially distributed feature blocks are placed at random locations and bring the fill ratios progressively closer to the required fill ratio (lines 13-22).

Algorithm 14 Generation of experimental environment [SM24]

Input: Number of features C; required fill ratio of every feature $\vec{r} = [r_1, r_2, ..., r_C]$; Pattern=Random/Concentrated; level of concentration γ; arena size=$[L_E, W_E]$

Output: Feature composition of every tile in the environment E of size $L_E \times W_E \times C$

```
 1: $E = L_E \times W_E \times$ array of zeros with size $C$
 2: **if** Pattern=Random **then**
 3:     **for** c=1:C **do**
 4:         Array$_1$ $=$ sample $L_E \times W_E$ numbers from $\mathcal{U}(0,1)$
 5:         Array$_2$ $=$ empty array of equal size to Array$_1$
 6:         Array$_2$[Array$_1$ $\geq r_c$] $=$ sample for every entry from $\mathcal{U}(0, r_c)$
 7:         Array$_2$[Array$_1$ $< r_c$] $=$ sample for every entry from $\mathcal{U}(r_c, 1)$
 8:         $E[:,:,c] =$ Array$_2$
 9:     Normalize the feature proportion in every tile in $E$ so that they sum up to 1
10: **else**
11:     # Pattern=Concentrated
12:     MaxDiff $= 1$ ; $c = 1$
13:     **while** MaxDiff $> 0.01$ **do**
14:         Array$_1$ $=$ empty array of size $L_E \times W_E$
15:         $(\alpha^*, \beta^*) =$ random position in the arena
16:         **for** every entry in Array$_1$ with coordinates $(\alpha, \beta)$ **do**
17:             Dist $=$ Euclidean distance from $(\alpha, \beta)$ to ($\alpha^*$,$\beta^*$)
18:             Array$_1$$[\alpha, \beta] =$ MaxDiff $\times$ 10 $\times$ exp($-$Dist $\times$ $\gamma$)
19:         $E[:,:,c] = E[:,:,c] +$ Array$_1$
20:         Normalize the feature proportion in every tile in $E$ so that they sum up to 1
21:         MaxDiff $=$ compute the highest deviation between the current fill ratio and
            required fill ratio among considered features
22:         $c =$ feature with the highest deviation between the current fill ratio and the
            required fill ratio
```

6.2 Collective Consensus Strategies for the Many-Option Scenario

The proposed parallel encoding discrete consensus strategies, DBBS and RV, are modified to suit a many-option scenario. For comparison, linear consensus protocol (LCP) is used as a baseline, where the decision space is interpreted as continuous.

6.2.1 Low-Level Control Mechanisms for Multi-Feature Environmental Exploration

For all considered decision-making strategies, the robots use the same random walk mechanism as in previous chapters to explore the environment and obtain their own estimates of the fill ratio of the observed features. Due to the expanded decision space, the quality computation is modified compared to the previous Chapters as follows.

For both discrete decision-making strategies, RV and DBBS, the decision space changes with the number of features C and the level of precision P. The hypothesis matrix H contains all valid hypotheses is stored by every robot and has size $M \times C$ as shown:

$$H = \begin{bmatrix} \vec{h}_1 \\ \vec{h}_2 \\ ... \\ \vec{h}_M \end{bmatrix} \text{ where } \vec{h}_m = \begin{bmatrix} h_{m,1} & h_{m,2} & ... & h_{m,C} \end{bmatrix}. \tag{6.3}$$

N is the size of the decision space and is the number of possible discrete options. Every row is a hypothesis consisting of the fill ratio combinations of all C considered features, and is a potential option by the agents. They are computed using both the number of features C and the discretization precision P by enumerating all possible fill ratios for individual features from 0 to 1 and excluding the entries where the sum of fill ratios exceeds 1. It is assumed that the number of features C and the discretization precision P are known beforehand, and thus this calculation is done before deployment.

During its environmental exploration, robot i stores beliefs for every hypothesis computed from its environmental exploration in array $\vec{l_i}$ of length M as follows:

$$\vec{l_i} = \begin{bmatrix} l_1 & l_2 & l_3 & ... & l_M \end{bmatrix}^T \tag{6.4}$$

$$\vec{l}_{i,0} = \begin{bmatrix} 1/M & 1/M & 1/M & ... & 1/M \end{bmatrix}^T. \tag{6.5}$$

Every entry l represents the computed likelihood of a single hypothesis. All entries are initialized to value $1/M$ at the start of an experimental run when the robot does not have any information regarding the environment.

At every control loop, a robot makes an observation of the features present at its current position, if it is moving forward in motion state A. Observation collection is limited to during forward motion in order to prevent an agent from collecting multiple observations at the same location. The robot detects a random feature present in the tile beneath it. The intensities of the features on that tile are used as weights to determine which feature is detected. The result is stored in an array of size C, with only the entry associated with the detected feature being 1 while the others being 0. This can be represented as a weighted sampling operation from a set of vectors as follows:

$$\text{When the robot is on coordinate } (\alpha, \beta)\text{: } \vec{o} \in_R \{ \begin{bmatrix} 1 \\ 0 \\ ... \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ ... \\ 0 \end{bmatrix} ... \begin{bmatrix} 0 \\ 0 \\ ... \\ 1 \end{bmatrix} \} \tag{6.6}$$

$$\text{Sampling weights: } \vec{e}_{\alpha,\beta} = \begin{bmatrix} s_{\alpha,\beta,1} & s_{\alpha,\beta,1} & ... & s_{\alpha,\beta,C} \end{bmatrix} \text{ respectively.} \tag{6.7}$$

The likelihood of making a feature observation, given that a particular hypothesis is true, is the fill ratio of the feature in the hypothesis, hence can be computed as $H \cdot \vec{o}$. Therefore, after each observation, the belief of robot i $\vec{l_i}$ is modified by iteratively performing element-wise product (represented as $\circ$) of the observation likelihoods as follows:

$$\vec{l}_{i,t} = \text{Normalize}(\vec{l}_{i,t-1} \circ (H \cdot \vec{o})). \tag{6.8}$$

The iterative updates allow the belief of individual robots to gradually converge to the most likely fill ratio hypothesis.

On the other hand, under LCP, the robots simply use the frequencies of features being observed as their estimations of the features' distribution. Robot i's belief $\vec{\rho_i}$ records the number of times each feature has been observed. It has size C and is updated after every observation as follows:

$$\vec{\rho}_{i,0} = \begin{bmatrix} 0 & 0 & 0 & ... & 0 \end{bmatrix}^T \tag{6.9}$$

$$\vec{\rho}_{i,t} = \vec{\rho}_{i,t-1} + \vec{o}. \tag{6.10}$$

From here, the robot's opinion on the proportion of features based on its observations can be easily calculated as:

$$\vec{\xi_i} = \vec{\rho_i}/||\vec{\rho_i}||_1. \tag{6.11}$$

6.2.2 Decision-Making Strategies Investigated

This subsection presents the investigated collective consensus-forming strategies in the many-option scenario.

Iterative Many-Option Ranked Voting

The implementation of ranked voting (RV) consensus approaches in Chapter 4 and 5 kept the opinion-based decision mechanism, where each robot has a chosen option and the ranked ballots serve to encode its preferences of the other options during opinion exchange. This design has poor scalability to the number of discrete options. Algorithm 15 shows the modified ranked voting consensus approach for the many-option scenario.

The robot i encodes its ranking of all M options in the ranking array $\vec{v_i}$. Every entry $v_{i,m}$ where $m \in \{1..M\}$ represents the rank of option m in terms of preference by robot i. $\vec{v_i}$ is initialized as an array of 0s representing that none of the options has been ranked. Array $\vec{v_i}$ is updated by two processes during a control loop: environmental exploration (line 2-5) and dissemination with the other robots (lines 6-10). During exploration, the robot first performs the individual Bayesian hypothesis testing and updates its belief array $\vec{l_i}$ (lines 3-4). The belief array $\vec{l_i}$ is then transformed into a ranking array via two consecu-

Algorithm 15 Iterative many-option Ranked Voting (RV) [SM24]

Input: Hypothesis matrix H of size $M \times C$; robot index $i \in \{1..N_{robot}\}$; initialized belief of robot i: $\vec{l_i}$ of length M

Output: Converged decisions: d_i

```
1:  Set evidence rate $r_e$, ballot size $\eta$ and broadcasting probability $\phi$
2:  Initialize $\vec{v_i}$ as an array of 0s with length $M$
3:  **while** Decisions in swarm have not converged **do**
4:      **if** Robot $i$ is moving forward in motion State A **then**
5:          $\vec{o}$ = Collect observation and compute $\vec{o}$
6:          $\vec{l_i} = \text{Normalize}(\vec{l_i} \circ (H \cdot \vec{o}))$
7:          With probability $r_e$: $\vec{v_i} = \text{argsort}^2(\vec{v_i} + \text{argsort}^2(-\vec{l_i}))$
8:      $\vec{b_j}$ = CollectNeighborBallot
9:      **if** $\vec{b_j}$ collected **then**
10:         $\vec{v}' = 0$s array of length equal to $M$
11:         **for** all $b_{j,m}$ in $\vec{b_j}$ **do** $v'_{b_{j,m}} = m$
12:         $\vec{v_i} = \text{argsort}^2(\vec{v_i} + \vec{v}')$
13:     $\vec{b_i} = \text{argsort}(\vec{v_i})$
14:     $\vec{b}'_i = [b_{i,1}..b_{i,\eta M}]$
15:     $d_i = \text{argmin}(\vec{v_i})$
16:     Broadcast $\vec{b}'_i$ randomly at probability $\phi$
```

tive $argsort$ operations, and combined with $\vec{v_i}$ via a Borda count ranked voting system to update the robot's opinion. This update is done with a probability set by the evidence rate parameter r_e (line 5).

The robots exchange opinions with each other by transmitting the ballot array $\vec{b}$. It is an ordered list of indices of options in terms of their rankings in $\vec{v}$ for every robot, and can be truncated to preserve information on only the options ranked at the front. The robot i begins its dissemination process by trying to pick up ballot $\vec{b_j}$ from another robot j within the communication radius and is broadcasting (line 6). If a ballot is received, it is used to update $\vec{v_i}$ using the same Borda count ranked voting system (line 10). The robot's own ballot $\vec{b_i}$ is computed from $\vec{v_i}$ (line 11) and truncated according to the ballot size parameter η at position ηM (line 12). The truncated ballot $\vec{b}'_i$ is then broadcasted at probability ϕ during the rest of the control loop (line 14). ϕ is not freely adjusted, but computed based on the communication bandwidth available and the ballot size η. The robot's decision is independently computed from its ranking array $\vec{v_i}$ (line 13).

The behavior of the RV consensus strategy is controlled via the two parameters r_e and η. r_e controls the relative frequency, and thus importance, between environmental exploration and communication with peers in the decision-making process. A higher r_e leads to a greater emphasis on individual exploration and vice versa. While η controls the strength of consensus enforcement within the robot swarm. As η increases, the number of considered options, thus weakening the consensus enforcement. At the same time, since the communication frequency ϕ is inversely correlated with η, a higher η value reduces the frequency of communication between robots and has the similar effect of weakening consensus enforcement.

Discrete Bayesian Belief Sharing in the Many-Option Scenario

The implementation of the Bayesian belief fusion approach for a many-option consensus scenario shown in Algorithm 16 is similar to in previous chapters, as the strategy is shown to have good scalability. Modifications are made to improve the stability of the strategy in many-option scenarios.

Algorithm 16 Modified implementation of Discrete Bayesian Belief Sharing (DBBS) in many-option scenario [SM24]

Input: Hypothesis matrix H of size $M \times C$; robot index $i \in \{1..N_{robot}\}$; initialized belief of robot i: $\vec{l_i}$ of length M; initialized record of neighbors' beliefs : $\vec{l'_i}$ of length M
Output: Converged decisions: d_i

```
1:  Set decay parameters $\lambda$, $\mu$ and broadcasting probability $\phi$
2:  while Decisions in swarm have not converged do
3:      if Robot $i$ is moving forward in motion State A then
4:          Collect observation and compute $\vec{o}$
5:          $\vec{l_i} = \text{Normalize}(\vec{l_i} \circ (H \cdot \vec{o}))$
6:          $\vec{l_i}[\vec{l_i} < 0.001/M] = 0.001/M$
7:      $\vec{\xi_j}$ = CollectNeighborOpinion
8:      if $\vec{\xi_j}$ collected then
9:          $\vec{l'_i} = (\vec{l'_i} \times \lambda + \vec{\xi_j}) - \text{mean}(\vec{l'_i} \times \lambda + \vec{\xi_j})$
10:     $\vec{\xi_i} = ln(\vec{l_i}) + \vec{l'_i} \times \mu$
11:     Broadcast $\vec{\xi_i}$ randomly at probability $\phi$
12:     $d_i = \text{argmax}(ln(\vec{l_i}) + \vec{l'_i})$
```

When facing a high number of discrete options, the original implementation of DBBS experiences underflow in the likelihood array entries. To address this, a minimum value of $0.001/M$ has been added for entries in the robot's own belief array $\vec{l_i}$ (line 6). In addition, the record of neighbors' beliefs $\vec{l'_i}$ and the outgoing message $\vec{\xi_i}$ are transformed to logarithmic values. Thus, the updating operation of $\vec{l'_i}$ becomes as shown in line 9, and the entries are normalized such that the mean value is zero. Both decay parameters also now have multiplicative relationships with $\vec{l'_i}$ instead of exponential ones (lines 9-10). The computation of the robot's decision d_i is likewise changed (line 12).

Linear Consensus Protocol

In order to determine the validity of treating the investigated many-option collective estimation scenario as a discrete consensus problem, the continuous consensus approach Linear Consensus Protocol (LCP) shown in Algorithm 17 is used as a baseline.

Algorithm 17 Linear Consensus Protocol (LCP) [SM24]

Input: Robot index $i \in \{1..N_{robot}\}$
Output: Converged decisions: d_i

```
1:  Set decision-making cycle length $\zeta$ and broadcasting probability $\phi$
2:  Initialize $\vec{\rho_i}$ to empty array with length $C$
3:  $J = \{\}$
4:  while Convergence not detected do
5:      if Robot $i$ is moving forward in motion State A then
6:          Collect observation and compute $\vec{o}$
7:          $\vec{\rho_i} = \vec{\rho_i} + \vec{o}$
8:          $\vec{\xi_i} = \vec{\rho_i}/||\vec{\rho_i}||_1$
9:          if $\vec{d_i}$ is empty then
10:             $\vec{d_i} = \vec{\xi_i}$
11:     $\vec{d_j}$ = CollectNeighborOpinion
12:     if $\vec{d_j}$ collected then
13:         $J = J \cup \vec{d_j}$
14:         if Size of $R \geq \zeta$ then
15:             $\vec{d_i} = \text{mean}(\vec{d_i}, \vec{\xi_i}, \text{every element in } J)$
16:             Clear $R$
17:     Broadcast $\vec{d_i}$ randomly at probability $\phi$
```

Different from the aforementioned discrete decision-making strategies, under LCP robot i stores its collected observations in the array $\vec{\rho_i}$ of length C. When an observation is collected during environmental exploration, $\vec{\rho_i}$ is updated by adding the observation array $\vec{o}$, which has the effect of adding 1 to the number of occurrences of the observed color (line 7). The estimated fill ratios from the robot's own observations $\vec{\xi_i}$ are obtained by computing the proportion of each feature observed (line 8). The robot's decision $\vec{d_i}$ is also in the form of a real number array of size C. Without input from neighbors, d_i is initialized with the same value as $\vec{\xi_i}$. At every control loop, the robot i broadcasts its own decision array $\vec{d_i}$ (line 17) and tries to receive a decision array $\vec{d_j}$ from its peers in the neighborhood (line 11). Received decision arrays are stored in the set J without identification of the sending robot. The maximum size of J is the parameter ζ and is used to control the decision-making frequency of the swarm. When the size of J reaches ζ, the robot updates its decision array $\vec{d_i}$ by taking an average value for each entry individually across $\vec{d_i}$, $\vec{\xi_i}$ and every $\vec{d_j}$ in set J (line 15).

6.3 Experiments and Results

This section presents the experiments of the considered consensus strategies in the investigated many-option collective estimation scenario.

6.3.1 Experimental Setup

The experiments in this chapter carry over the basic environmental designs of the previous chapters, with the arena still being size $2m \times 2m$, filled by 400 square tiles, and explored by 20 mobile robots. The bandwidth-controlled comparison framework introduced in Chapter 5 is used here as well. The assumptions on the message formats and sizes are shown in Table 6.1.

Table 6.1: Assumptions on the message formats and sizes for all considered strategies [SM24]

Decision-making Strategy	Message Format
RV	$\eta \times M \times$ short int ($16M\eta$ bits)
DBBS	$M\times$ float ($32M$ bits)
LCP	$C\times$ float ($32C$ bits)

Before the decision-making process, the decision space for every robot is computed. For LCP, it is the real number space of $\vec{d_i} \in \Re, \sum \vec{d_i} = 1$. For RV and DBBS, the decision space is discretized with respect to the required decision precision P into $\begin{bmatrix}0 & 1P & 2P & ... & 1\end{bmatrix}$ representing the fill ratio hypotheses for each individual color. The hypotheses are concatenated to form the fill ratio hypotheses of all considered colors, with those having a sum greater than 1 across all colors removed from the hypotheses list.

The expanded decision space in the many-option scenario makes an exact convergence more difficult. Thus, the consensus performance in this chapter is measured using three new metrics: scatter at convergence, error at convergence, and convergence time. The scatter of all robots' opinions at time t is defined as the sum of Euclidean distances between the centroid of all decisions regarding the fill ratios within the swarm and the individual decisions of the robots:

$$\text{Scatter}_t = \sum_{\forall i'} (\text{mean}_{\forall i}(\vec{d}_{i,t}) - \vec{d}_{i',t})^2. \tag{6.12}$$

Convergence is achieved when the scatter is at a minimum during an experimental run, limited to a time limit of $1200\ s$. The scatter at convergence can be represented as follows:

$$\text{ConvScatter} = \min(\text{Scatter}_t). \tag{6.13}$$

The error at convergence is defined as the total absolute error between all robot's fill ratio decisions and the true fill ratios $\vec{r}$ at the time of convergence:

$$t* = \text{argmin}_t(\text{Scatter}_t) \tag{6.14}$$

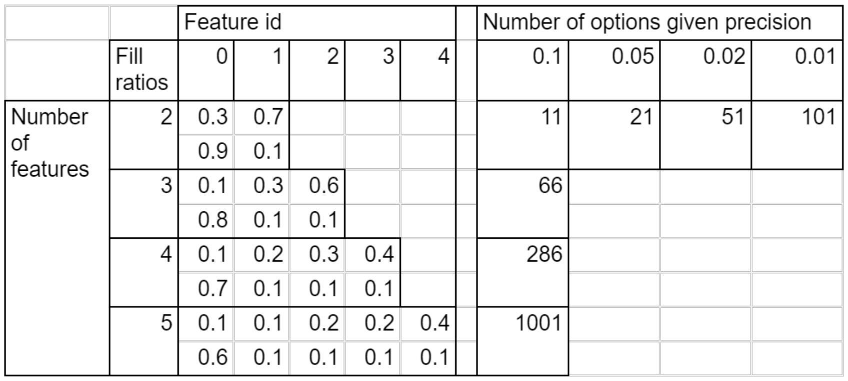
$$\text{ConvError} = \sum_{\forall i} \text{abs}(\vec{r} - \vec{d}_{i,t*}). \tag{6.15}$$

Finally, convergence time is defined as the time taken to reach 90% of the minimum scatter value from the maximum scatter value during the experiment:

$$\text{ConvTime} = \min(t)\forall t\text{: Scatter}_t \leq 0.9\min(\text{Scatter}_t) + 0.1\max(\text{Scatter}_t). \tag{6.16}$$

The fill ratios tested for every considered number of features are shown in Table 6.2. For every number of features, two different fill ratio configurations are tested, with 40 experimental runs conducted for each fill ratio scenario at every parameter configuration of the considered strategies, as shown in Table 6.3. The aggregate across the two fill ratio configurations is used to determine the performances of the considered strategies at the corresponding parameter configurations.

Table 6.2: Fill ratios tested for every considered number of features and the resulting number of options at different discretization precision levels [SM24]



		Feature id					Number of options given precision			
	Fill ratios	0	1	2	3	4	0.1	0.05	0.02	0.01
Number of features	2	0.3	0.7				11	21	51	101
		0.9	0.1							
	3	0.1	0.3	0.6			66			
		0.8	0.1	0.1						
	4	0.1	0.2	0.3	0.4		286			
		0.7	0.1	0.1	0.1					
	5	0.1	0.1	0.2	0.2	0.4	1001			
		0.6	0.1	0.1	0.1	0.1				

Table 6.3: Parameter values used in the experiments for the considered strategies [SM24]

Decision-making strategy	Parameter settings used
RV	$r_e = [0.2, 0.5, 1]; \eta = [0.05, 0.1, 0.2, 0.5]$
DBBS	$\lambda = [0.5, 0.7, 0.9]; \mu = [0.2, 0.4, 0.6, 0.8]$
LCP	$l = [5, 10, 15, 20, 30, 40, 60, 80]$

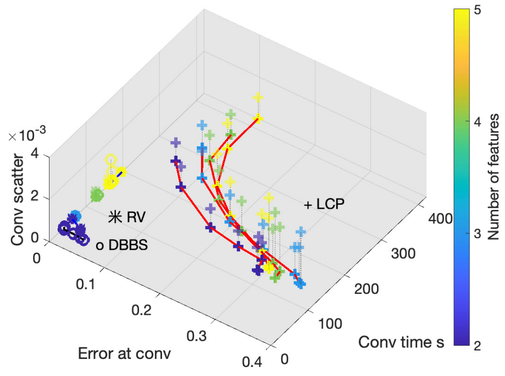
The performances of the considered strategies at different parameter settings are used to gauge the trade-offs between different performance metrics at different decision-making difficulty levels, with the same multi-criteria framework as in previous chapters. The extent of the multi-criteria trade-off is quantified with the spread of the Pareto frontiers, which is computed as the area bounded by the extreme points on the Pareto front. A higher spread means a more significant trade-off between speed and accuracy in the decision-making process, while a spread of 0 means the existence of a single best parameter configuration for the algorithm in that particular scenario.

6.3.2 Performances of the Considered Strategies with Respect to Different Numbers of Environmental Features

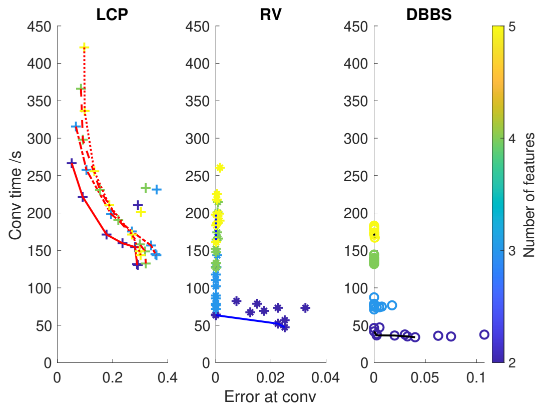
This subsection presents the performances of the considered strategies at different numbers of environmental features, i.e. different numbers of colors in the arena. The error versus convergence speed Pareto fronts of the considered strategies' mean performances at different parameter configurations are shown in Figure $6.2(a)$. The solid markers show the error versus convergence time performances at parameter configurations on the Pareto fronts, with the solid lines showing the Pareto fronts. The transparent markers show the position of the performances in the $3D$ space when considering scatter.

Comparing the performances of both discrete strategies RV and DBBS versus those of LCP in Figure $6.2(a)$, it can be shown that discrete decision-making strategies have superior performances in terms of the error versus convergence time trade-off, as they outperform the Pareto fronts of LCP for all tested scenarios. RV and DBBS also produce lower scatter at convergence, showing a stronger ability to reliably reach consensus compared to LCP. In Figure $6.2(a)$ when the environmental features are randomly distributed, as the number of environmental features increases, the performances of both discrete strategies experience a significant drop in terms of convergence speed. On the other hand, LCP's performances experience a drop in convergence speed on the top-left side of the Pareto fronts. As observed in Figure $6.2(b)$, its performances also experience a slight drop in accuracy when facing more environmental features.

In order to get a clearer view of the impact of the number of features on the performances of the individual strategies, Figure $6.2(b)$ shows the scatter plot of error versus convergence time performances at different axis scales in terms of error. It can be seen more clearly that for both discrete strategies RV and DBBS, a higher number of environmental features diminishes the extent of the speed versus accuracy trade-off. For a higher number of features, there exists a singular best parameter configuration, as opposed to a Pareto front of equally good configurations observed at a lower number of features. On the other hand, LCP consistently displays speed versus accuracy trade-offs for all numbers of features. Between the two discrete strategies, RV sees more variations in its performances at different parameter settings when facing a higher number of features compared to DBBS, which produces performances that are clustered together for different parameter settings.



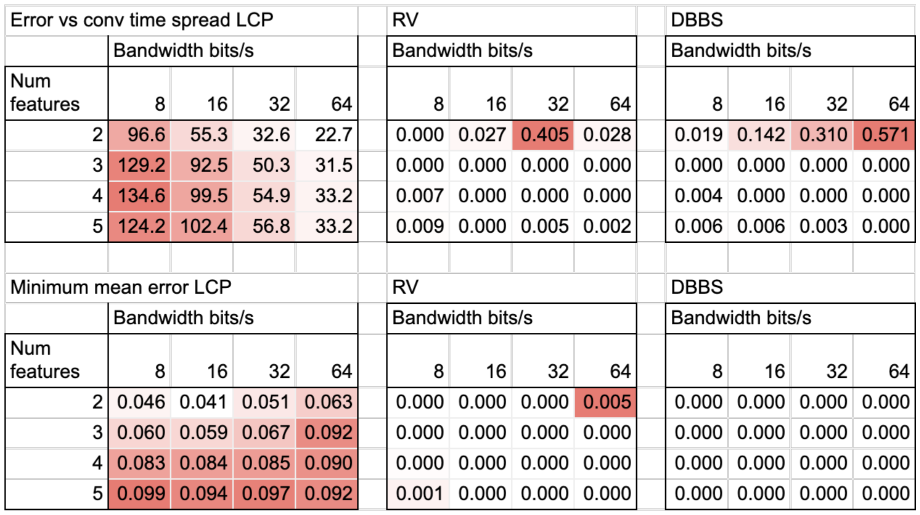
(a)



(b)

Figure 6.2: Performances of the considered strategies in all metrics when facing different numbers of randomly distributed environmental features (a) Error versus convergence time Pareto fronts, $3D$ space also includes scatter at convergence (b) Scatter plot for all tested parameter settings in terms of error versus convergence time; bandwidth=32 $bits/s$, decision space discretization precision=0.1 [SM24]

Table 6.4: Error vs convergence time spread and minimum mean error of the Pareto front of the considered strategies at different number of features; decision space discretization precision=0.1 [SM24]



Error vs conv time spread LCP					RV				DBBS			
	Bandwidth bits/s				Bandwidth bits/s				Bandwidth bits/s			
Num features	8	16	32	64	8	16	32	64	8	16	32	64
2	96.6	55.3	32.6	22.7	0.000	0.027	0.405	0.028	0.019	0.142	0.310	0.571
3	129.2	92.5	50.3	31.5	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
4	134.6	99.5	54.9	33.2	0.007	0.000	0.000	0.000	0.004	0.000	0.000	0.000
5	124.2	102.4	56.8	33.2	0.009	0.000	0.005	0.002	0.006	0.006	0.003	0.000

Minimum mean error LCP					RV				DBBS			
	Bandwidth bits/s				Bandwidth bits/s				Bandwidth bits/s			
Num features	8	16	32	64	8	16	32	64	8	16	32	64
2	0.046	0.041	0.051	0.063	0.000	0.000	0.000	0.005	0.000	0.000	0.000	0.000
3	0.060	0.059	0.067	0.092	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
4	0.083	0.084	0.085	0.090	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
5	0.099	0.094	0.097	0.092	0.001	0.000	0.000	0.000	0.000	0.000	0.000	0.000

To quantify the impact of the number of environmental features on the extent of speed versus accuracy trade-off, Table 6.4 shows the spread of the Pareto fronts for the considered strategies at different number of features and at different bandwidth levels. In addition, the minimum mean error values indicate the positions of the Pareto front and show the limits of the performances of the considered strategies. For LCP, it can be observed that the main influence on the spread of Pareto front and the extent of speed versus accuracy trade-off comes from the bandwidth level. It also sees a slight increase in the spread of the Pareto front and a worsening of the accuracy at the extreme point of the Pareto front, as the number of features increases. This trend is, however, reversed for RV and DBBS, both of which show a drop in Pareto front spread when the number of features increases. Both discrete decision-making strategies can also maintain very low error at the extreme point of the Pareto fronts as the number of features increases.

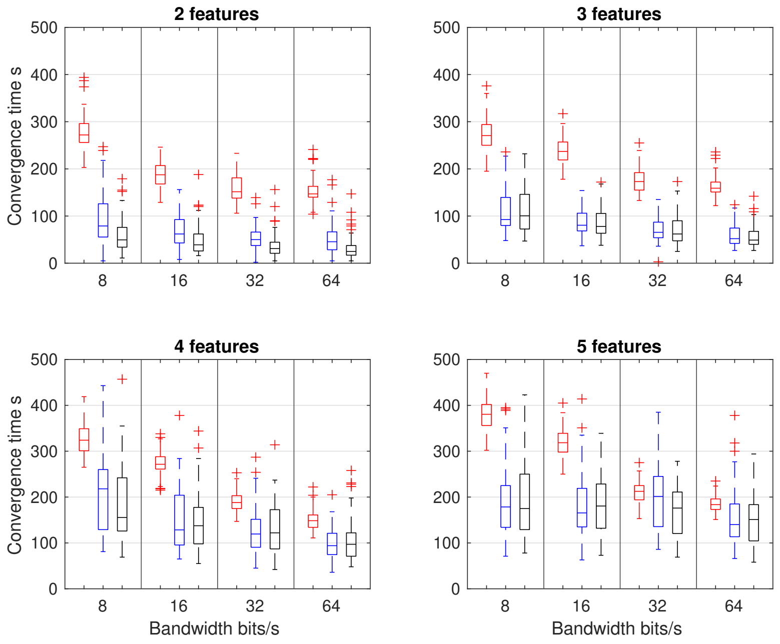
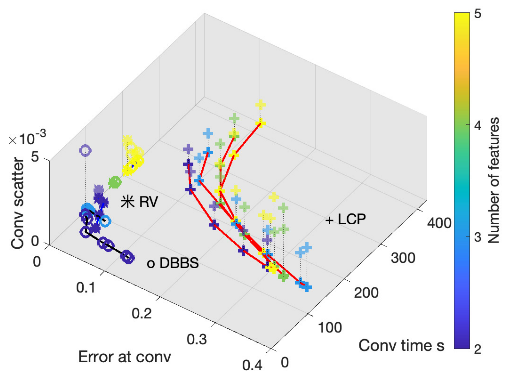
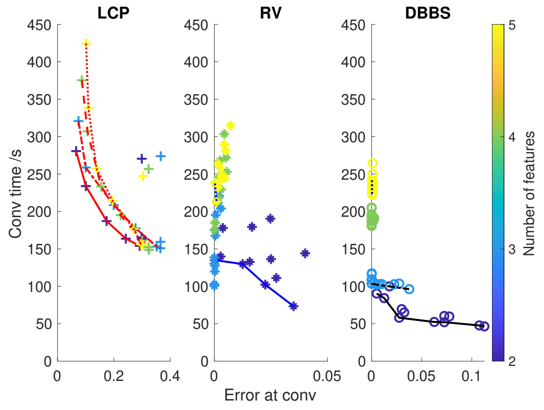


Figure 6.3: Box plots showing the distribution of the convergence time of the considered strategies at the best parameter configurations facing different number of features at different bandwidth levels; Red: LCP, Blue: RV, Black: DBBS; decision space discretization precision=0.1 [SM24]

The box plots of the convergence times produced by the considered strategies under the best parameter configurations in random environments are shown in Figure 6.3. The parameter configurations chosen here are those corresponding to the center points on the Pareto frontiers of error versus convergence time performances. As a baseline, LCP generally experiences an increase in convergence speed when the communication bandwidth increases, and a decrease in convergence speed when the number of features increases. The latter holds for both multi-objective decision-making strategies, coupled with an increase in the variation of the convergence time. On the other hand, an increase in communication bandwidth does not have as strong a positive influence on the convergence speed of both discrete strategies, especially when facing a higher number of features. This causes the convergence speed of RV and DBBS to be close to that of LCP at high number of features and high bandwidths.



(a)



(b)

Figure 6.4: Performances of the considered strategies in all metrics when facing different numbers of concentrated environmental features (a) Error versus convergence time Pareto fronts, $3D$ space also includes scatter at convergence (b) Scatter plot for all tested parameter settings in terms of error versus convergence time; bandwidth=32 $bits/s$, decision space discretization precision=0.1 [SM24]

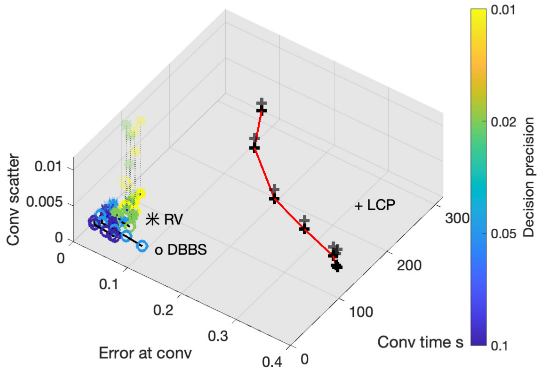
Figure 6.4 shows the performances of the considered strategies when facing different numbers of concentrated features. A similar trend to that in random environments is observed. Concentrated feature distribution further reduces the convergence speed of the discrete decision-making strategies compared to LCP. For RV, further instability with respect to parameter settings is introduced, causing the bi-objective performances in error and convergence time when facing three features to exceed those when facing two features. Both discrete decision-making strategies also see a greater extent of error versus convergence time trade-off compared to random environments, while LCP does not see a significant reduction in performances.

The results above have shown that when considering error, convergence time, and scatter, both tested discrete approaches, RV and DBBS, have superior performances compared to the LCP baseline decision space with a high number of features. This indicates that the parallel consideration of multiple potential options by RV and DBBS, when accounting for the increased bandwidth costs, is justified in terms of the decision-making performances.

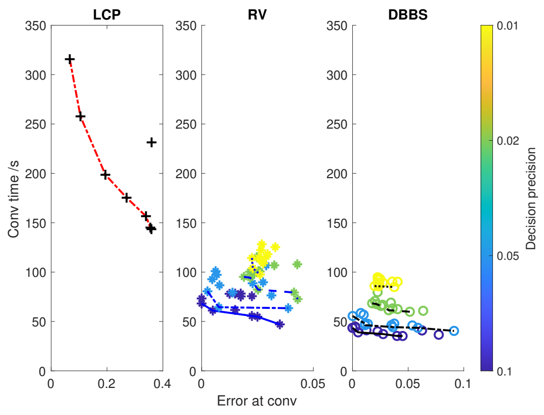
At the same time, it is observed that an increase in the number of environmental features has a positive effect on the accuracy performances of the considered discrete decision-making strategies, especially DBBS. This is seen together with a significant decrease in the spread of the observed Pareto fronts, hence making the algorithms more sensitive in terms of parameter settings, but also presenting singular best configurations and reducing the trade-offs faced. This is in contrast to the behavior displayed by LCP, which sees a worsening in performance across all metrics when facing a higher number of features.

6.3.3 Performances of the Considered Strategies with Respect to Different Decision Space Discretization Precision

This subsection presents the performances of the considered strategies at different levels of decision space discretization. It is the second way where the number of potential options can be changed for the discrete decision-making strategies. For LCP, with its continuous decision space, the level of discretization has no effect on the decision-making performances.



(a)



(b)

Figure 6.5: Performances of the considered strategies in all metrics when randomly distributed environmental features at different levels of discretization precision (a) Error versus convergence time Pareto fronts, $3D$ space also includes scatter at convergence (b) Scatter plot for all tested parameter settings in terms of error versus convergence time; bandwidth=32 $bits/s$, number of features=2 [SM24]

Figure 6.5 shows the performances of the considered strategies in randomly distributed environments when facing different levels of discretization precision. It can be observed that compared to the previous subsection for different numbers of features, the level of discretization precision has a smaller impact on the decision speed for the discrete decision-making strategies, but a bigger impact on the scatter and error. Notably, as shown in Figure $6.5(a)$, at finer discretization precision, both RV and DBBS show a significant increase in scatter at convergence, exceeding those produced by LCP. As shown in Figure $6.5(b)$, as the discretization precision becomes finer, both discrete strategies produce higher errors and shorter error versus convergence time Pareto fronts.

Table 6.5 shows the quantification of the changes in the Pareto fronts for RV and DBBS. It can be observed that for both RV and DBBS, there is a region with respect to precision and bandwidth settings where the error versus convergence time spread reaches a

Table 6.5: Error vs convergence time spread and minimum mean error of the Pareto front of the considered strategies at different levels of discretization precision; number of features=2 [SM24]

Error vs conv time spread RV					DBBS			
	Bandwidth bits/s				Bandwidth bits/s			
Precision	8	16	32	64	8	16	32	64
0.1	0.013	0.030	0.741	0.139	0.000	0.013	0.371	0.380
0.05	0.016	0.021	0.619	0.809	0.141	0.287	1.375	0.774
0.02	0.045	0.119	0.543	0.072	0.002	0.046	0.289	0.409
0.01	0.120	0.053	0.060	0.136	0.117	0.009	0.018	0.073

Minimum mean error RV					DBBS			
	Bandwidth bits/s				Bandwidth bits/s			
Precision	8	16	32	64	8	16	32	64
0.1	0.000	0.000	0.000	0.005	0.000	0.000	0.000	0.000
0.05	0.004	0.005	0.003	0.001	0.001	0.003	0.000	0.004
0.02	0.025	0.022	0.019	0.020	0.020	0.019	0.018	0.021
0.01	0.028	0.025	0.023	0.022	0.020	0.021	0.020	0.022

maximum, centered around precision of 0.05 and bandwidth of 32 $bits/s$. While for other scenarios, the spread of the Pareto fronts decreases. On the other hand, the minimum mean error increases steadily as precision decreases for both discrete strategies, while not being significantly affected by the bandwidth levels. These results show that a finer discretization precision has a negative impact on the decision-making accuracy of both discrete strategies. This is opposite to the impact of a higher number of features shown in the previous subsection. This negative impact is also not easily mitigated by allowing a higher frequency of communication or by parameter tuning of the decision-making strategy.

The distribution of the convergence time of the considered strategies is shown in Figure 6.6. It can be confirmed that compared to different number of features shown in the previous subsection, decision precision has a smaller impact on both the median and the variation of the decision speed of the discrete strategies. As such, even at the smallest tested decision precision of 0.01, both discrete strategies are still significantly faster than the LCP baseline. For every decision precision tested, increasing the communication bandwidth also has a positive impact on the decision speed.

Lastly, the performances of the considered strategies when facing concentrated environmental features at different levels of discretization precision are shown in Figure 6.7. For both discrete strategies, as discretization precision becomes finer, a similar trend of increasing error and convergence time is observed. The increase in convergence time at finer discretization precision is more significant than observed when facing random environmental features and makes the decision speed of the discrete strategies on par with that of LCP at the discretization precision of 0.01. On the other hand, LCP's performances still do not significantly decrease compared to when facing random environmental features.

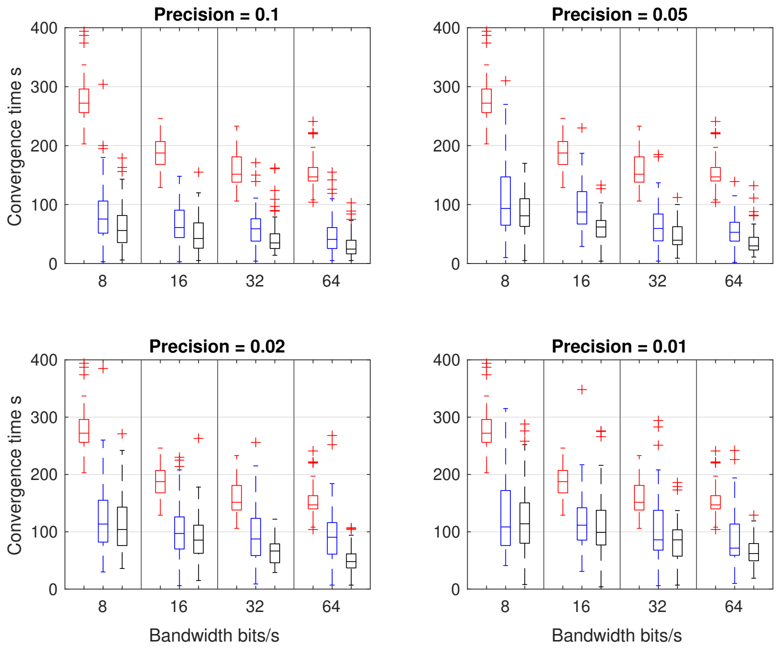
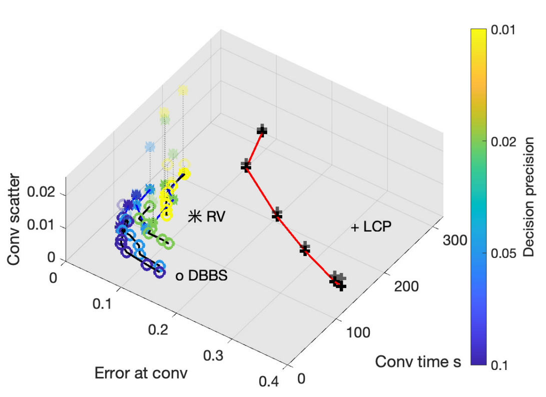
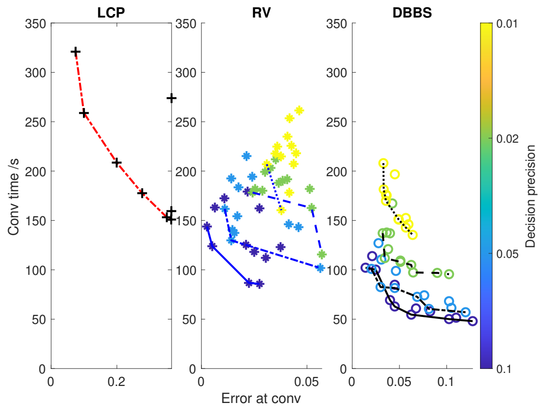


Figure 6.6: Box plots showing the distribution of the convergence time of the considered strategies at the best parameter configurations facing random features at different bandwidth and discretization precision levels; Red: LCP, Blue: RV, Black: DBBS; number of features=2 [SM24]





(a) (b)

Figure 6.7: Performances of the considered strategies in all metrics when facing concentrated environmental features at different levels of discretization precision (a) Error versus convergence time Pareto fronts, $3D$ space also includes scatter at convergence (b) Scatter plot for all tested parameter settings in terms of error versus convergence time; bandwidth=32 $bits/s$, number of features=2 [SM24]

6.4 Summary and Discussion

This chapter examined the performances of the considered parallel encoding consensus-forming strategies in large discrete decision spaces, as well as facing different quality distributions. Based on the experimental results presented, the following judgments can be made: Firstly, in the decision-making scenarios investigated that go up to 101 options with two features and 1001 options with five features, parallel encoding consensus-forming strategies have an edge in terms of accuracy and speed compared to LCP tested. This, coupled with the fact that continuous consensus-forming strategies have an inherent difficulty in forming an exact consensus in distributed systems [EB10], means, that for continuous consensus forming, the discretization of the decision space and the adoption of discrete decision-making strategies can often be the best approach in reaching an accurate consensus, even if a moderate level of discretization precision is required.

The operation of parallel discrete decision-making strategies in a larger decision space, however, can be negatively impacted by the number of potential options. Based on the experimental results, both investigated discrete decision-making strategies, RV and DBBS, produce a smaller error and a higher convergence time when facing a larger decision space caused by a higher number of features, while showing a larger error and a less significant increase in the convergence time when facing a larger decision space caused by finer discretization precision. This distinction is caused by the differences in the correlations between the existing options and the new options introduced by the two different expansions to the decision space. When the number of features increases, the decision space expands in the number of dimensions, leading to unordered and weakly correlated options being added to the existing pool of potential options. It is thus less likely to mislead the swarm to erroneous options, but rather only increases the number of options the robots need to process. Thus, the impact of a higher number of features is similar to that of a smaller communication bandwidth, in that the decision-making process is slowed down but made more accurate due to the added time to explore the environment, as observed in Chapters 4 and 5.

On the other hand, a finer discretization precision increases the number of options in the existing decision space, thus introducing many options that are correlated with the existing ones in terms of option qualities. These options can easily mislead the swarm and cause it to converge to an erroneous option, thus increasing the error, which is a similar effect to that caused by concentrated feature distribution. In addition, both sources of expansion of the decision space also reduce the extent of the speed versus accuracy trade-off, thus making the decision-making process more invariant with respect to parameter settings. This is a direct result of the expanded list of options, which makes it increasingly difficult to reach a fast consensus even when not considering the accuracy. This distinguishes scenarios with larger decision spaces from those with concentrated distribution of features. Its impact on the viability of discrete decision-making strategies in large decision spaces is two-fold: it reduces the need to make an additional judgment regarding parameter settings with respect to trade-offs between different decision-making metrics; however, it is also more important to select the best parameter settings to ensure good performances of the concerning strategies.

In Chapters 4 and 5, it has been observed that belief fusion approaches tend to have a stronger level of positive feedback than ranked voting approaches and can thus lead to faster but less accurate consensus. This trend is confirmed here, as DBBS tends to be faster and less accurate than RV in all experiments. However, it is more prevalent when facing more finely discretized options compared to a higher number of features. This highlights the importance of parameter tuning to restrict the strength of positive feedback in consensus-forming processes when facing ordered and correlated options.

Lastly, both discrete consensus strategies see more significant performance drops in terms of decision speed and accuracy when facing concentrated feature distributions compared to random feature distributions. As observed previously, the effect of a concentrated feature distribution is a dispersion of the individual robots' opinions obtained from environmental exploration. In the results, LCP displays a more stable process in unifying the robots' opinions, while both discrete strategies face increased difficulty as shown by an increased decision time and scatter at convergence. This is also caused by the stronger positive feedback effect in the discrete consensus strategies, which causes agents in physical proximity to reinforce the opinions of each other, thus preventing consensus with the rest of the swarm.

Chapter 7

Decision Space Reduction in Collective Preference Learning Problem via Iterative Ranked Voting

This chapter is based on [SM23].

In Chapter 6, the effects of large decision space on the proposed discrete collective decision-making strategies are investigated. It has been observed that a larger decision space generally increases the difficulty of the decision-making scenario. It is therefore a beneficial practice in many-option collective decision problems to restrict the decision space during the decision-making process. This chapter presents an independent many-option collective decision-making scenario, collective preference learning, where the agents are tasked with collectively ranking a few spatially distributed sites. Two different consensus-forming strategies are examined in this scenario: one allows the agents to pick any opinion in the discrete decision space filled by all combinations of binary comparison between sites; the other enforces a strict ordering among the considered sites, thus reducing the decision space size. The aim is to investigate the benefits and costs of restricting the decision space during the collective decision-making process.

7.1 Collective Preference Learning Scenario

In embodied intelligent systems, learning the ranking of a number of objects according to their relative preference is an important operation, that has many real-life applications and can also serve as building blocks for more complex behaviors. When the objects are spatially distributed and cannot be sampled by a single robot, a swarm robotics system needs a collective strategy to combine the incomplete ranking information from each individual robot. This section presents a collective preference learning scenario based on the non-physics-based scenario studied in [CL21] and is illustrated in Figure 7.1. There are N sites distributed in an arena marked in gray. Each site has a fixed detectable index $n \in \{1..N\}$ and a predetermined quality $q_1..q_N$, shown via the color intensities, that can be measured with an added noise. A group of robots marked in red are tasked with ranking the sites in terms of quality.

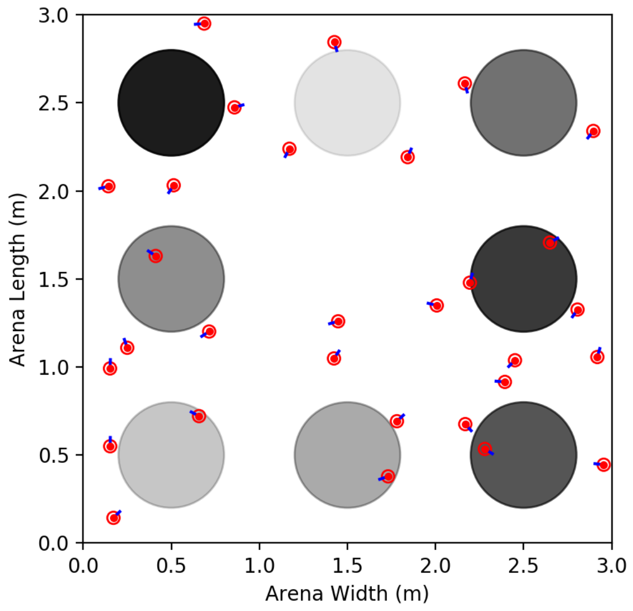


Figure 7.1: Illustration of the collective preference learning scenario; gray circular areas represent the $N = 8$ sites, their color intensities represent their qualities; red dots indicate mobile robots roaming the arena [SM23]

Each robot has only access to local information individually and has to rely on its peers to get a full view of the environment. To simulate this, the robots are programmed to perform random walk in the arena and collect quality observations on the sites spontaneously when they come into range. A single robot only holds a single pairwise comparison between two sites that it has visited at a time. This information is pooled by all robots to deduce and form a consensus regarding the ranking of the sites.

From the point of view of a single robot, a pairwise comparison between sites n_1 and n_2 can take three possibilities: site n_1 is better than n_2, site n_1 is worse than n_2, or the comparison is unknown. In order to pool this information across all N sites, the number of possible combinations describing every pair of quality comparison is expressed as $3^{N(N-1)/2}$, which scales exponentially with N^2. However, since comparison is transitive, the sites can also be ranked into an ordered list of increasing qualities. This reduces the decision space size to $N!$. Both approaches are examined in this chapter, and the differences in performances as a result of the different decision spaces are investigated.

7.2 Decentralized Strategies for Preference Learning

To tackle the presented preference learning scenario, two consensus strategies are used. The first is a direct fusion of the pairwise comparison information by the robots, while the second uses distributed ranked voting to achieve consensus of the rankings. This section presents the two consensus strategies in detail, while also showing the evaluation metrics used for the performances of the considered strategies.

7.2.1 Obtaining Pairwise Quality Comparisons

For the two investigated consensus strategies, the robots use the same mechanism individually to collect observations on the environment and turn the observations into pairwise comparisons between the two sites. Robot i walks randomly in the environment. When it is in the area of a site during a control loop, it has a probability (evidence rate r_e) to detect the site and updates its recorded pairwise comparison using Algorithm 18. The robot keeps track of the indices and qualities of two sites, expressed as follows:

$$\text{Indices} : \vec{d_i} = [d_{i,1}, d_{i,2}]$$

$$\text{Qualities} : \vec{q_i} = [q_{i,1}, q_{i,2}].$$

All 4 variables are initialized to -1.

Algorithm 18 Procedure to update pairwise comparison stored by a single robot $UpdatePair(i, d*, q*)$ [SM23]

Input: Robot index $i \in \{1..N_{robot}\}$; stored site indices $\vec{d_i}$; stored site qualities $\vec{q_i}$
Output: Updated site indices $\vec{d_i}$; updated site qualities $\vec{q_i}$

```
1: if $d_{i,1} = d*$ or $d_{i,1} = -1$ then
2:     $d_{i,1} = d*$; $q_{i,1} = q*$
3: else if $d_{i,2} = d*$ or $d_{i,2} = -1$ then
4:     $d_{i,2} = d*$; $q_{i,2} = q*$
5: else
6:     $ind$ = RandomChoice$\{1, 2\}$
7:     $d_{i,ind} = d*$; $q_{i,ind} = q*$
8: if $q_{i,1} < q_{i,2}$ then
9:     Switch$(d_{i,1}, d_{i,2})$; Switch$(q_{i,1}, q_{i,2})$
```

When a site is detected, the index $d*$ and measured quality $q*$ are recorded. If $d*$ is present in $\vec{d_i}$, the robot updates the associated quality value with $q*$. If one value in $\vec{d_i}$ is -1, indicating the position is empty, a new $d*$ also fills the position (lines 1-4). If both values in $\vec{d_i}$ are filled and are not equal to $d*$, then one of the two positions is selected at random and filled with $d*$ and $q*$ (lines 6-7). Finally, the robot always preserves the ordering $q_{i,1} \geq q_{i,2}$, thus if this is no longer the case after updating, then the values in both $\vec{d_i}$ and $\vec{q_i}$ will be switched (line 9).

7.2.2 Baseline Consensus Strategy: Pairwise Belief Fusion

The baseline consensus strategy investigated relies on direct fusion of pairwise comparisons recorded by the robots [CL21]. The implementation here is presented in Algorithm 19.

Algorithm 19 Collective preference learning using belief fusion [SM23]

Input: Robot index $i \in \{1..N_{robot}\}$; Initialized belief matrix of robot i B_i as zeros matrix of size $N \times N$

Output: Converged belief matrix of robot i B_i

```
1:  Set evidence rate $r_e$, environmental noise $\sigma_{noise}$, flag indicating transitivity preserving
    operation $f_t$
2:  while Decisions in swarm have not converged do
3:      if Robot i is on site n & Site detected with probability $r_e$ then
4:          UpdatePair($i, n$, sample($\mathcal{N}(q_n, \sigma^2_{noise})$)) # shown in Algorithm 18
5:          if $d_{i,1} > 0$ and $d_{i,2} > 0$ then
6:              $B_i[d_{i,1}, d_{i,2}] = 1$; $B_i[d_{i,2}, d_{i,1}] = -1$
7:      else if Other robots in communication radius of robot i then
8:          $j =$ RandomChoice{Indices of neighboring robots}
9:          $B_i = B_i + B_j$
10:         $B_i[B_i \neq 0] = B_i[B_i \neq 0]/abs(B_i[B_i \neq 0])$
11:     if $f_t$ i.e. transitivity needs to be preserved then
12:         for $n = 1..N$ do
13:             for $n_1 = 1..N$ do
14:                 for $n_2 = 1..N$ do
15:                     if $B_i[n, n_1] = 1$ & $B_i[n, n_2] = -1$ & $B_i[n_1, n_2] = 0$ then
16:                         $B_i[n_1, n_2] = -1$; $B_i[n_2, n_1] = 1$
17:     Broadcast upper triangular half of $B_i$ to neighboring robots
```

The robot records the pairwise relationship between all possible pairs of sites in the belief matrix B_i. The entry $B_i[n_1, n_2]$ can take one of 3 values, 1 when $q_{n_1} > q_{n_2}$, -1 when $q_{n_1} < q_{n_2}$, and 0 when the pairwise relationship is unknown or when $n_1 = n_2$. At every control loop, the robot first updates its recorded site pair stored in $\vec{d_i}$ and $\vec{q_i}$ using its environmental observation (line 4). The recorded pairwise comparison is then stored in the belief matrix B_i (lines 5-6). The robot then starts broadcasting its belief to its peers and picks up the belief matrix of a random neighbor, with which it performs belief fusion to update its own belief matrix (lines 9-10).

Depending on the initial settings from the user, the robot checks the preservation of transitivity in the belief matrix at the end of every control loop (lines 11-16). f_t is a Boolean variable that marks this setting. The operation ensures that when the belief matrix B_i records $q_a > q_b$ and $q_b > q_c$, it will automatically also record $q_a > q_c$. Since the operation needs to traverse the whole matrix N times, it is an expensive operation with complexity scaling to N^3, and presents a trade-off between performance and computational resources needed. Both versions of the belief fusion algorithm, with and without transitivity preservation, have been experimented with to obtain a full view of its performances.

7.2.3 Distributed Iterative Ranked Voting for Preference Learning

The other experimented approach employs the ranked voting consensus approach investigated in the previous chapters. The decision-making behavior is shown in Algorithm 20, while Algorithms 21 and 22 are subroutines used in the algorithm. In this approach, the robot i stores the indices of all known sites in the order of decreasing quality with the ordered list $\vec{r_i}$, which is empty at initialization:

$$\text{Initial: } \vec{r_i} = [].$$

The list grows as more sites are observed by the robot itself and its peers, up to the maximum size of N. At every control loop, the robot attempts to detect a potential site using the same environmental observation procedure introduced previously in Algorithm 18 (Algorithm 20 line 4).

Algorithm 20 Collective preference learning using ranked voting [SM23]

Input: Robot index $i \in \{1..N_{robot}\}$; Initialized ranking list $\vec{r_i} = []$
Output: Converged ranking vector of robot i $\vec{r_i}$

```
1:  Set evidence rate $r_e$, environmental noise $\sigma_{noise}$
2:  while Decisions in swarm have not converged do
3:      if Robot i is on site n & Site detected with probability $r_e$ then
4:          UpdatePair($i, n$, sample($\mathcal{N}(q_n, \sigma^2_{noise})$)) # shown in Algorithm 18
5:          if $d_{i,1} >= 0$ and $d_{i,2} >= 0$ then
6:              UpdateRanking($i, \vec{d_i}$) # shown in Algorithm 21
7:      else if Other robots in communication radius of robot n then
8:          $j$ = RandomChoice{Indices of neighboring robots}
9:          $\vec{r_i}$ = election($\vec{r_i}, \vec{r_j}$) # shown in Algorithm 22
10:         if $d_{i,1} >= 0$ and $d_{i,2} >= 0$ then
11:             UpdateRanking($i, \vec{d_i}$) # shown in Algorithm 21
12:     Broadcast $\vec{r_i}$ to neighboring robots
```

Algorithm 21 Procedure to update the rankings of known sites based on store pairwise comparison in a robot $UpdateRanking(i, \vec{d_i})$ [SM23]

```
Input: Robot index $i \in \{1..N_{robot}\}$; stored site indices $\vec{d_i}$
Output: Updated ranking vector of robot i $\vec{r_i}$
 1: if $d_{i,1}$ is not in $\vec{r_i}$ then
 2:     if $d_{i,2}$ is not in $\vec{r_i}$ then
 3:         Randomly insert $d_{i,1}$; Randomly insert $d_{i,2}$ after $d_{i,1}$
 4:     else
 5:         Randomly insert $d_{i,1}$ before $d_{i,2}$
 6: else
 7:     if $d_{i,2}$ is not in $\vec{r_i}$ then
 8:         Randomly insert $d_{i,2}$ after $d_{i,1}$
 9:     else if rank of $d_{i,1} >$ rank of $d_{i,2}$ in $\vec{r_i}$ then
10:         Switch $d_{i,1}$ and $d_{i,2}$ in $\vec{r_i}$
```

Afterward, if both positions in its pairwise comparison are filled, the robot uses it to update its own computed ranking of all sites $\vec{r_i}$ (Algorithm 20 lines 5-6). This operation follows the procedure $update\_ranking(i, \vec{d_i})$. As shown in Algorithm 21, the robot seeks to insert the two sites in its recorded pairwise comparison $\vec{d_i}$ into its computed ranking $\vec{r_i}$, while preserving the pairwise ordering. Thus, if one site in $\vec{d_i}$ is recorded in $\vec{r_i}$, the other is placed before or after it to preserve the pairwise ordering (Algorithm 21 lines 5 and 8). For example, if $d_{i,1}$ is present in $\vec{r_i}$ but not $d_{i,2}$, the latter would be inserted in a random position after the former, as follows marked with downward arrows:

$$\text{Inserting after } d_{i,1} : \vec{r_i} = [s_a\ s_b\ s_c\ d_{i,1} \downarrow s_d \downarrow s_e \downarrow].$$

If both sites are present in $\vec{r_i}$, the robot checks if the rankings comply with the pairwise relationship in $\vec{d_i}$, and switches the rankings if not (Algorithm 21 line 10). An example of the switching operation is as follows:

$$\text{Switching } d_{i,1}, d_{i,2} : \vec{r_i} = [s_a\ s_b\ s_c\ d_{i,2}\ s_d\ d_{i,1}\ s_e]$$

$$\Rightarrow \vec{r_i} = [s_a\ s_b\ s_c\ d_{n,1}\ s_d\ d_{n,2}\ s_e].$$

The robot constantly broadcasts its current computed $\vec{r_i}$ to its neighbors in its communication radius. If a site is not detected, it randomly picks up a message sent by its neighbor j, if one is present, and it performs an election to generate a new $\vec{r_i}$ (Algorithm 20 lines 7-9). All interactions among the robots are kept to a peer-to-peer and pairwise fashion similar to in the baseline algorithm, such that the communication paradigms of the considered algorithms in this study can be roughly similar.

The pooling of opinions is conducted via a localized election with Borda count tallying

Algorithm 22 Function describing the pairwise election producing new rankings by combining opinions from two robots $election(\vec{r}_i, \vec{r}_j)$ [SM23]

Input: Ranking vectors from two robots $\vec{r}_i$, $\vec{r}_j$
Output: Combined ranking vector $\vec{r}*$

```
1:  $\vec{c}$ = ordered list of unique site indices in $\vec{r}_i$ and $\vec{r}_j$
2:  Initialize $\vec{v}_i$ and $\vec{v}_j$ as $-1s$ arrays of length equal to $\vec{c}$
3:  for all $c$ in $\vec{c}$ do
4:      if $c$ is in $\vec{r}_i$ then
5:          $\vec{v}_i[c]$ = rank of $c$ in $\vec{r}_i$
6:      if $c$ is in $\vec{r}_j$ then
7:          $\vec{v}_j[c]$ = rank of $c$ in $\vec{r}_j$
8:  $\vec{v}_i[\vec{v}_i < 0]$ = length$(\vec{r}_i)/2$
9:  $\vec{v}_j[\vec{v}_j < 0]$ = length$(\vec{r}_j)/2$
10: $\vec{r}* = \vec{c}$[argsort$(\vec{v}_i + \vec{v}_j)$]
11: Return $\vec{r}*$
```

system between the robot i and its neighbor j. The detailed process is shown in Algorithm 22. In the election process, the index rankings $\vec{r}_i$ and $\vec{r}_j$ need to be transformed into the scores of all considered sites, which are stored in $\vec{v}_i$ and $\vec{v}_j$ for the two voters respectively. The transformation is done in Algorithm 22 lines 2-7. The corresponding score of a considered site is the ranking of it in $\vec{r}_i$ or $\vec{r}_j$ (Algorithm 22 lines 5,7). The two score vectors must then be padded to contain the same sites, which are tracked by the vector $\vec{c}$. The unranked candidates' indices are selected using boolean indexing in Algorithm 22 lines 8-9. This is different from when ranked voting is utilized in real-life elections. This is because when a real-life ranked voting ballot has missing entries, it means that the unranked candidates have lower preferences than all ranked candidates, and hence can be ranked last. However, in the proposed algorithm, an unranked site has an unknown quality relative to the ranked sites. Therefore, they are assigned a temporary ranking that is half of the number of ranked sites, such that the resulting ranking of unranked sites only considers the opinion of the other robot.

The following example illustrates the aforementioned operations.

$$\vec{r}_i = [3, 1, 5, 6, 2],\ \vec{r}_j = [5, 1, 2, 7]$$

$$\vec{c} = [1, 2, 3, 5, 6, 7]$$

$$\text{Ranking} \Rightarrow \text{score:}$$

$$\vec{v}_i = [1, 4, 0, 2, 3, -1],\ \vec{v}_j = [1, 2, -1, 0, -1, 3]$$

$$\text{Pad score vectors:}$$

$$\vec{v}_i = [1, 4, 0, 2, 3, 2.5],\ \vec{v}_j = [1, 2, 2, 0, 2, 3]$$

The padded score vectors are then added together, applied with an $argsort$ operation, and used as indices in an indexing operation of $\vec{c}$ vector to obtain the new ranking vector $\vec{r}*$ (Algorithm 22 line 10). Here it is performing the inverse of traditional Borda count [Eme13] and using the rankings directly as the associated points and sorting the candidates in ascending order of their received points. The produced ranking is also randomized in the event of a tie anywhere in the ranking of received points.

Keeping up with the example above, the following is an example of how $\vec{r}*$ is produced.

Adding score vectors:

$$\vec{v}_{total} = [2, 6, 2, 2, 5, 5.5]$$

Ranking of considered sites' indices in $\vec{c}$:

$$argsort(\vec{v}_{total}) = [2, 3, 0, 4, 5, 1]$$

Resulting ranking of considered sites:

$$\vec{r}* = \vec{c}[argsort(\vec{v}_{total})] = [3, 5, 1, 6, 7, 2]$$

Finally, the election results also need to be checked if they comply with the recorded pairwise comparison by calling the $UpdateRanking$ procedure again (Algorithm 20 line 10-11).

Overall, at the design level, the proposed algorithm uses less communication, storage and computational resources compared to the benchmark algorithm based on belief fusion, especially the variant of it with the transitivity preserving operation.

7.3 Experiments and Results

The experiments are conducted in physics-based simulation. The experimental arena has a fixed size of $3m \times 3m$. Individual robots have the same random walk routines as in previous chapters. $N = 8$ sites are placed in the experimental arena in fixed positions of $(0.5, 0.5)$, $(1.5, 0.5)$, $(2.5, 0.5)$, $(0.5, 1.5)$, $(2.5, 1.5)$, $(0.5, 2.5)$, $(1.5, 2.5)$, and $(2.5, 2.5)$, all with radii of $0.3m$, as shown before in Figure 7.1. In every experimental instance, their qualities are chosen from the array $[0, 1, ..., 7]$ in random order. Noise $\mathcal{N}(0, \sigma_{noise}^2)$ is added to the true qualities of the sites during environmental observation to simulate different levels of inaccuracies in the cognitive abilities of the robots. The performances of the considered strategies are measured under different sets of experimental parameters: noise level σ_{noise}, evidence rate r_e, and number of robots N_{robot}. For every combination of parameters, 20 independent experiments are conducted.

In order to evaluate the performances of the two considered consensus strategies, their outputs need to be in the same format. The proposed ranked voting strategy encodes the ranking in a vector with a maximum size equal to the number of sites N, while the benchmark belief fusion strategy records all pairwise relationships using a $N \times N$ matrix. Since the conversion from the latter to the former can result in information loss, the rankings produced by the proposed ranked voting algorithm are converted into a belief matrix containing all known pairwise relationships for evaluation. The conversion is done using Algorithm 23.

Algorithm 23 Function converting ranking vector into belief matrix $conversion(\vec{r_i})$ [SM23]

```
1: Initialize $B_i$ as zeros matrix of size $N \times N$
2: for $i = 1..N$ do
3:     for $j = 1..N$ do
4:         if $\vec{r_i}[i] < \vec{r_i}[j]$ then
5:             $B_i[i, j] = 1$
6:         else if $\vec{r_i}[i] > \vec{r_i}[j]$ then
7:             $B_i[i, j] = -1$
8: Return $B_i$
```

After unifying the outputs from the two considered algorithms, the output is compared to the belief matrix produced by the pairwise relationships of the true values of the sites B^*. Due to the large discrete decision space where the belief matrix can vary, the accuracy and uniformity of the decision-making process are measured using a similar approach as during the evaluation of the many-option discrete collective estimation strategies studied in Chapter 6.

The error is computed as the mean absolute error between $B*$ and B_i from every robot:

$$Error = (\Sigma_{i=1..N_{robot}} sum(abs(B^* - B_i)))/N_{robot}. \tag{7.1}$$

At initialization, all elements in B_i are set to 0, hence the error at initialization is $Error_0 = N(N-1)$. In this study $N = 8$, thus the error at initialization is 56. The maximum error that can theoretically be reached is $2N(N-1)$, where every pairwise relationship in the matrix is the opposite of the correct value. In this study, this value is 112. The lowest error that can be achieved is 0, where the ranking of every robot is exactly correct. During an experimental instance, the lowest error is achieved within a time limit of $2400s$ as the peak performance at convergence. The convergence time is computed as the time taken for the whole swarm to reach 90% of its peak performance from the initial condition, i.e. reach an error lower than $Error_{Peak} + (Error_0 - Error_{Peak}) * 0.1$.

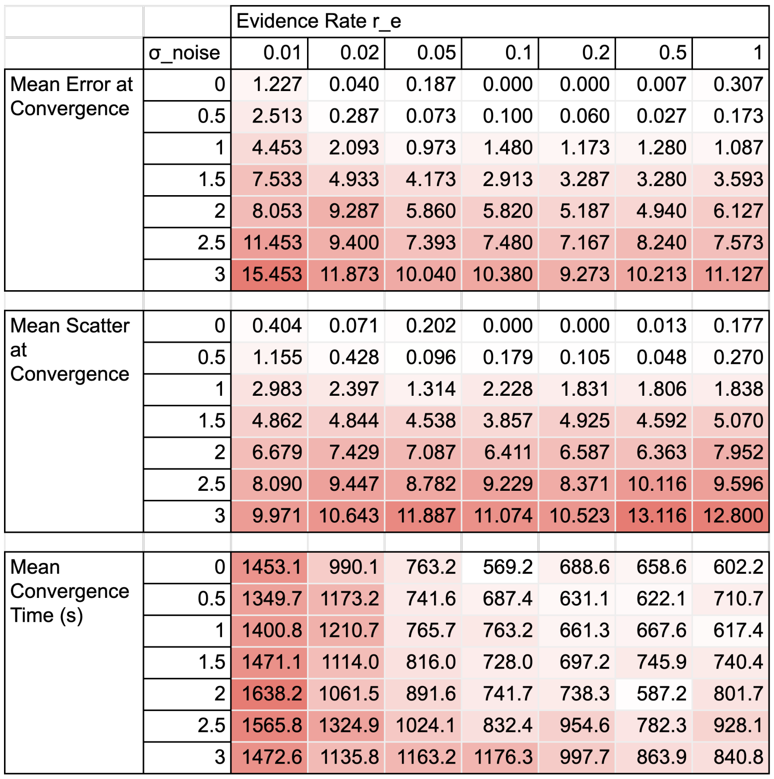
To measure the level of disagreement among the robots, the quantity $Scatter$ is defined as the average error between the belief matrices computed by every robot and those of every other robot, as follows:

$$Scatter = \frac{\Sigma_{i=1..N_{robot}} \Sigma_{j=1..N_{robot}} sum(abs(B_i - B_j))}{(N_{robot}-1)N_{robot}}. \tag{7.2}$$

7.3.1 Performances of Ranked Voting Strategy with Respect to Noise and Evidence Rates

The mean error and scatter at convergence, together with the mean convergence time, across 20 experiments at every parameter combination for the proposed ranked voting algorithm at various noise level σ_{noise} and evidence rate r_e settings are shown in Table 7.1.

Table 7.1: Performances of proposed ranked voting algorithm at different noise levels σ_{noise} and evidences rates r_e; $N_{robot} = 30$ [SM23]



		Evidence Rate r_e						
	σ_noise	0.01	0.02	0.05	0.1	0.2	0.5	1
Mean Error at Convergence	0	1.227	0.040	0.187	0.000	0.000	0.007	0.307
	0.5	2.513	0.287	0.073	0.100	0.060	0.027	0.173
	1	4.453	2.093	0.973	1.480	1.173	1.280	1.087
	1.5	7.533	4.933	4.173	2.913	3.287	3.280	3.593
	2	8.053	9.287	5.860	5.820	5.187	4.940	6.127
	2.5	11.453	9.400	7.393	7.480	7.167	8.240	7.573
	3	15.453	11.873	10.040	10.380	9.273	10.213	11.127
Mean Scatter at Convergence	0	0.404	0.071	0.202	0.000	0.000	0.013	0.177
	0.5	1.155	0.428	0.096	0.179	0.105	0.048	0.270
	1	2.983	2.397	1.314	2.228	1.831	1.806	1.838
	1.5	4.862	4.844	4.538	3.857	4.925	4.592	5.070
	2	6.679	7.429	7.087	6.411	6.587	6.363	7.952
	2.5	8.090	9.447	8.782	9.229	8.371	10.116	9.596
	3	9.971	10.643	11.887	11.074	10.523	13.116	12.800
Mean Convergence Time (s)	0	1453.1	990.1	763.2	569.2	688.6	658.6	602.2
	0.5	1349.7	1173.2	741.6	687.4	631.1	622.1	710.7
	1	1400.8	1210.7	765.7	763.2	661.3	667.6	617.4
	1.5	1471.1	1114.0	816.0	728.0	697.2	745.9	740.4
	2	1638.2	1061.5	891.6	741.7	738.3	587.2	801.7
	2.5	1565.8	1324.9	1024.1	832.4	954.6	782.3	928.1
	3	1472.6	1135.8	1163.2	1176.3	997.7	863.9	840.8

It can be observed from the mean error and mean scatter results that the noise level has a significant impact on the accuracy and precision performances of the proposed algorithm. As the noise level σ_{noise} increases, there is a very clear increase in both mean error and mean scatter at convergence. However, for most noise level and evidence rate combinations, the mean scatter is consistently higher than the mean error at convergence. This shows an accurate but imprecise decision distribution from the proposed algorithm.

At low r_e values below 0.02 and at especially high noise levels, the relationship above can be reversed, and the error could be higher than the scatter at convergence. This is to be expected as at these r_e values, the robots get very few observations. Coupled with a high noise level, erroneous pairwise observations tend not to be challenged, leading to inaccurate results.

At a particular noise level, the lowest mean errors and mean scatters are quite likely to be found in the middle range of evidence rates from 0.05 to 0.5, while both too low and too high an evidence rate can negatively affect the decision-making accuracy. Due to the stochasticity in the proposed algorithm's decision-making process, especially the random inserting of observed pairwise relationships in Algorithm 21, the proposed algorithm needs a certain number of pairwise opinion combinations relative to the evidence input to enforce a consensus, which is harder to meet when the evidence rate is too high.

On the other hand, the mean convergence time is more affected by the evidence rate r_e than by the noise level. When r_e increases from 0.01 to 0.1, there is a very apparent drop in mean convergence time at every noise level. However, beyond an evidence rate of 0.1, the change in mean convergence time is more irregular. This, combined with evidence rate's effects on errors and scatters at convergence, shows that for the proposed algorithm, a lack of evidence can hamper the decision-making process, but too high an influx of evidence does not necessarily have a positive effect.

7.3.2 Comparison with the Belief Fusion Benchmark at Different Noise Levels

The performance distribution across 20 experimental runs of considered algorithms under different noise levels are shown in Figure 7.2. The evidence rate r_e is set to 0.2. The swarm size N_{robot} is set to 30. Linear regression is also performed on the mean performances across all experimental runs at individual parameter settings against noise level, and computed the gradient of the best-fitting linear function and the coefficient of determination (R^2), the latter of which measures the level of linear relationship observed in the data. The results are shown in Table 7.2.

Table 7.2: Gradient and R^2 values obtained from linear regression of mean performances against σ_{noise} [SM23]

Algo	LR of Error	LR of Scatter	LR of Conv Time
BF w Tr	$G = 9.4$ $R^2 = 0.918$	$G = 1.48$ $R^2 = 0.916$	$G = 276$ $R^2 = 0.881$
BF w/o Tr	$G = 7.39$ $R^2 = 0.936$	$G = 1.58$ $R^2 = 0.979$	$G = 209$ $R^2 = 0.839$
RV	$G = 3.77$ $R^2 = 0.994$	$G = 3.78$ $R^2 = 0.975$	$G = 118$ $R^2 = 0.75$

In Figure 7.2 (a,b), it can be observed that all 3 algorithms produce comparable errors and scatters at convergence when the noise is low at 0 or 0.5. As shown in Figure 7.2

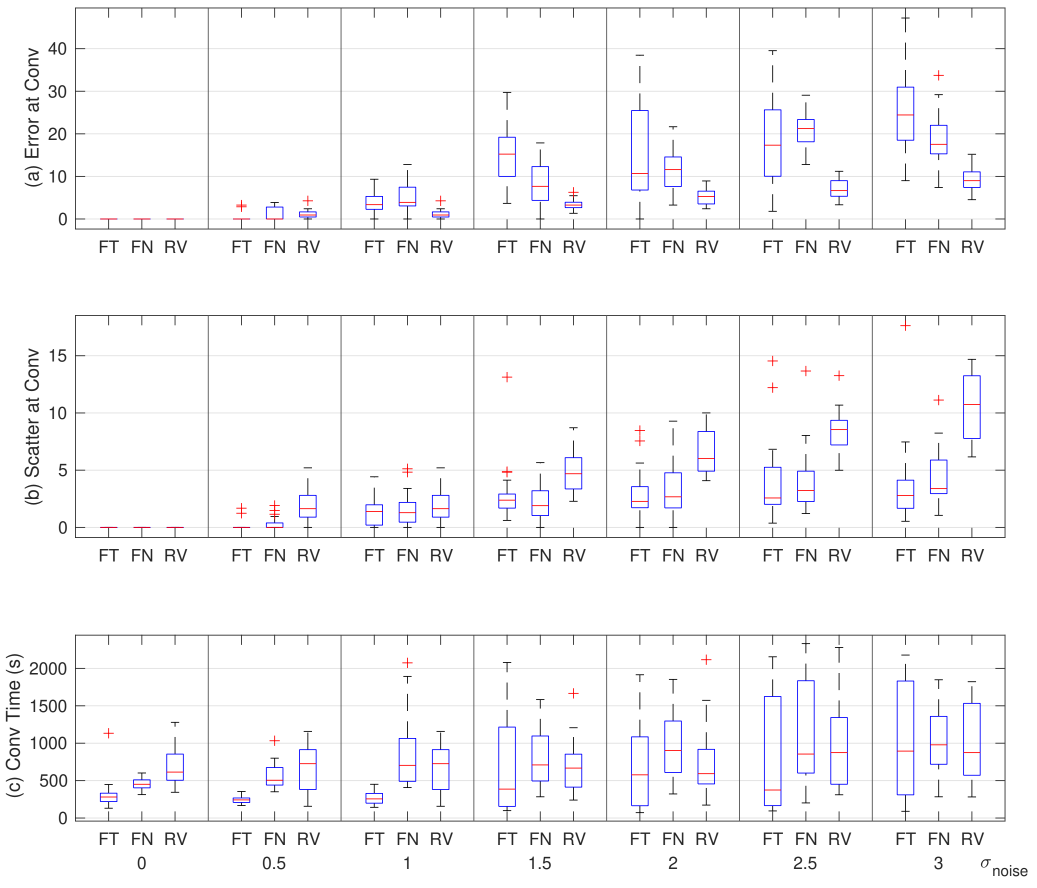


Figure 7.2: Box plots of (a) error at convergence (b) scatter at convergence (c) convergence time (s) for the proposed ranked voting algorithm (RV) and both variants of the benchmark algorithm (FT:fusion with transitivity preserved, FN:fusion without transitivity preserved) at different σ_{noise} settings; $r_e = 0.2$, $N_{robot} = 30$ [SM23]

(c) both variants of belief fusion are also able to converge within a shorter time compared to the proposed ranked voting algorithm. Among them, belief fusion with transitivity-preserving operations is the fastest.

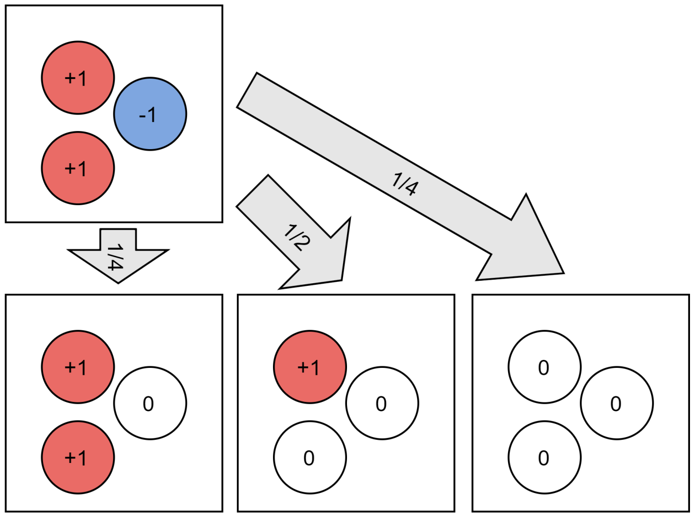
However, when the noise increases, the advantages of both belief-fusion-based algorithms begin to diminish. As shown in Figure 7.2 (a), when noise level σ_{noise} is in the range between 1 and 3, the error at convergence increases significantly for both variants of belief fusion, as the median error increases from around 0 to 24.4 for belief fusion with transitivity-preserving operation, and 17.5 without. The reduction in accuracy in the face of noise is also observed in the proposed ranked voting algorithm, however, the increase in median error at convergence here is much milder and the median value only hit 9 at the highest experimented noise level of 3. This is substantiated by the statistical analysis in Table 7.2, where the proposed ranked voting algorithm obtains the lowest gradient of mean error with respect to σ_{noise} at 3.77.

On the other hand, as observed in Figure 7.2 (b), the proposed ranked voting algorithm produces a progressively higher scatter than the two variants of belief fusion as the noise level increases, reaching a median value of 10.7. As noted in the previous subsection, the scatter produced by the proposed ranked voting algorithm is consistently on roughly the same scale as the error. However, both variants of belief fusion, although experiencing a significant increase in error, only have a mild increase in scatter, to a median of 2.79 when transitivity is preserved and 3.39 when it is not, as noise increases. This is also shown in Table 7.2, where the proposed ranked voting algorithm obtains the highest gradient of mean scatter with respect to σ_{noise} at 3.78.

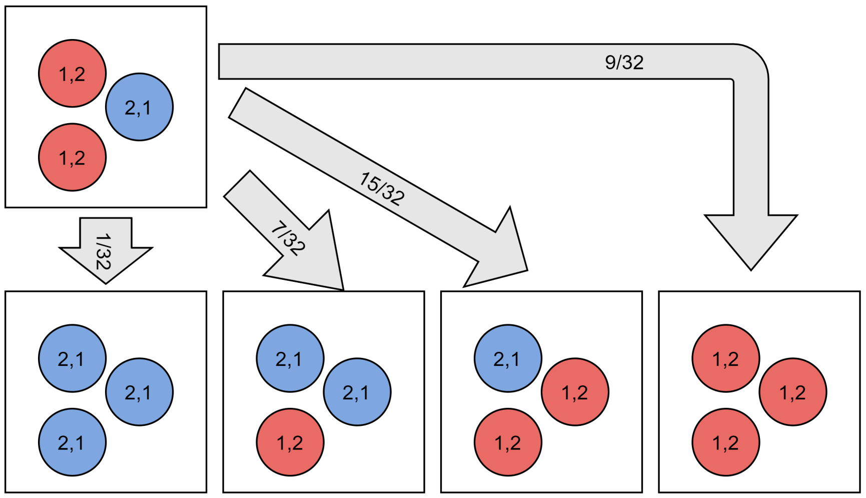
From the aforementioned experimental data, it can be concluded that as noise increases, the proposed ranked voting algorithm experiences a drop in precision, producing a higher scatter as the noise increases. Although the error also increases, it is consistently on the same scale or smaller than the scatter, confirming the fact that the proposed ranked voting algorithm keeps a high accuracy and much of the increasing error can be ascribed to scatter. In contrast, both variants of belief fusion experience a smaller increase in scatter, but they experience a much larger increase in error compared to the proposed ranked voting algorithm, demonstrating the fact that belief fusion can lead to consistent consensus among the swarm but is unable to reliably obtain the correct ranking at high noise scenarios.

As shown in Figure 7.2 (c), the convergence time for both variants of belief fusion experiences in general an increase as the noise level increases. Its variance also rises for both algorithms. At higher levels of noise from 2 to 3, the convergence time of all 3 algorithms are roughly on the same level and the advantage in fast convergence of belief fusion does not hold anymore. As shown in 7.2, the linear relationships between convergence time and noise level are not as strong as for the previous two performance metrics, shown by lower R^2 values. However, the proposed ranked voting algorithm still obtains the lowest gradient at 118.

Taking an integrated look at the performances of the considered algorithms with respect to the noise level, the differences in their performances can be explained by looking at their decision-making mechanisms. Both variants of belief fusion use a deterministic fusion function that encodes every pairwise relationship, making it easy for the whole swarm to converge their individual beliefs. However, it is also vulnerable to being misled by erroneous information at high noise scenarios. On the other hand, the proposed ranked voting algorithm limits the number of decision variables faced by the individual robots by using a more compact way of encoding the decisions. Its method of opinion combination also introduces a degree of stochasticity into the decision-making process, hence allowing the swarm to correct itself from wrong ordering easily, albeit at the cost of reducing the precision of the decisions made.



(a) Belief fusion



(b) Ranked voting

Figure 7.3: Toy examples of the state transition only considering 1 pairwise comparison within a small locality of 3 robots of (a) the benchmark belief fusion algorithm and (b) the proposed ranked voting algorithm; circles represent robots, numbers, and color codings represent robot opinions, arrows & fractions represent possible next states and transition rates [SM23]

To better illustrate the differences in the decision-making mechanisms of the considered algorithms, Figure 7.3 shows two toy examples of the benchmark belief fusion algorithm when there is only one pairwise relationship and three robots considered, and also in the absence of evidence input. The three robots are assumed to be within communication distance of each other. Every robot randomly receives a belief message from a random neighbor and performs its decision-making process. The top rows in both subfigures show the initial state in the locality, and the bottom rows show the possible states in the next time step. It can be seen in Figure 7.3 (a) for belief fusion that all three possible transitions eliminate the minority opinion -1, and the first two transitions will result in all three robots picking the opinion $+1$ in the following time steps. In contrast, in Figure 7.3 (b) for ranked voting it can be observed that only the first and last outcome with a combined probability of $5/16$ result in loss of information. In addition, no robots are left with the unknown status of 0, and the spread of a particular single opinion is significantly slowed.

7.3.3 Comparison with the Belief Fusion Benchmark at Different Evidence Rates

The impact on the operations of the considered algorithms from evidence rate r_e is then compared. The performance distribution at different r_e values are plotted in Figure 7.4. The noise level σ_{noise} is set to 1.5, and the swarm size N_{robot} is set to 30. The results from linear regression of the mean performances against the natural log of the evidence rate $ln(r_e)$ is shown in Table 7.3.

Table 7.3: Gradient and R^2 values obtained from linear regression of mean performances against $ln(r_e)$ [SM23]

Algo	LR of Error	LR of Scatter	LR of Conv Time
BF w Tr	$G = -0.42$ $R^2 = 0.146$	$G = 0.386$ $R^2 = 0.791$	$G = -34.2$ $R^2 = 0.144$
BF w/o Tr	$G = -1.91$ $R^2 = 0.901$	$G = 0.409$ $R^2 = 0.738$	$G = -144$ $R^2 = 0.522$
RV	$G = -0.333$ $R^2 = 0.429$	$G = 0.0204$ $R^2 = 0.007$	$G = -141$ $R^2 = 0.665$

From Figure 7.4 (a), it can be observed that all considered algorithms experience a general reduction in error when the evidence rate increases. The reduction is the least apparent in belief fusion with transitivity-preserving operations. For the proposed ranked voting algorithm, there is also a significant drop in the variance of the error at convergence. This is also substantiated by the statistical analysis shown in Table 7.3, where belief fusion with transitivity preserved obtains a very weak linear relationship between mean error and $ln(r_e)$ with $R^2 = 0.146$, as well as between mean convergence time and $ln(r_e)$ with $R^2 = 0.144$. Figure 7.4 (b) shows that both variants of belief fusion see higher scatter in their results as the evidence rate increases. There is also more variance in the scatter observed. However, this feature is not observed in the proposed ranked voting algorithm. Instead, the median scatter decreases when the evidence rate increases from 0.01 to 0.1 and starts increasing beyond that. There is also an observable increase in the variance of the scatter when the evidence rate reduces beyond 0.1. As shown in Table 7.3, both variants of belief fusion obtain moderately strong linear relationships between mean scatter and $ln(r_e)$ with R^2 being 0.791 and 0.738 respectively. On the other hand, for the proposed ranked

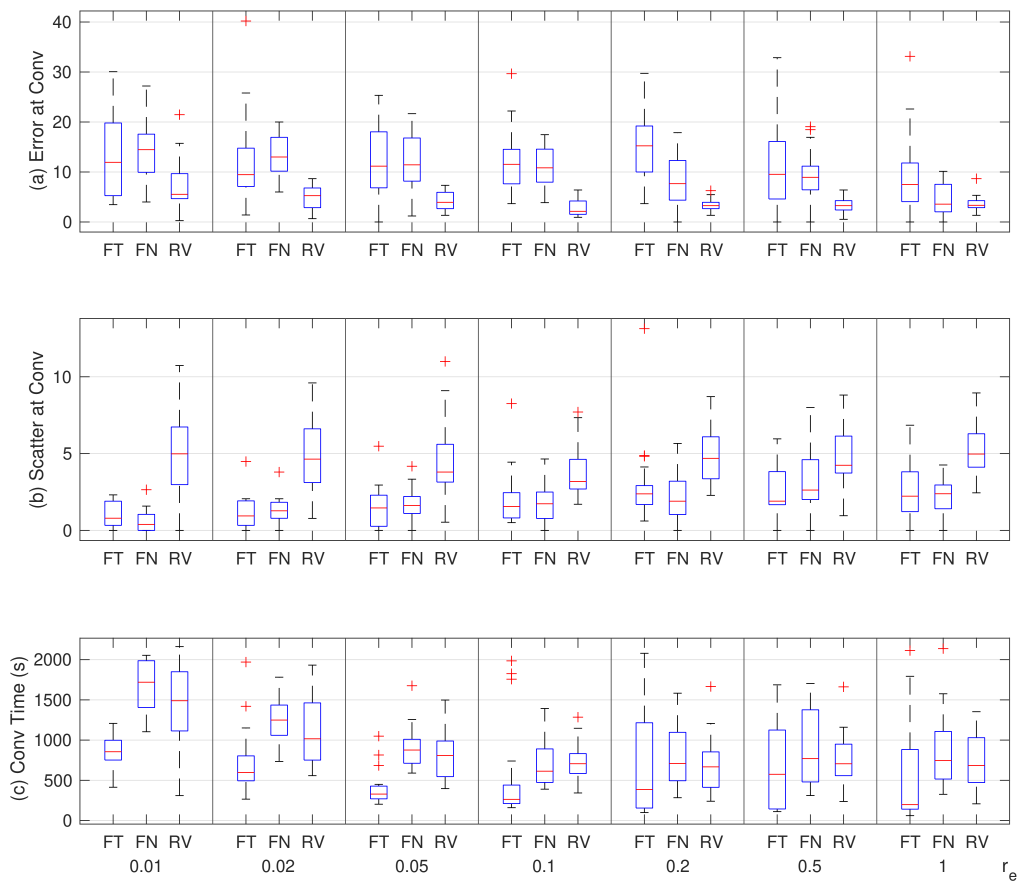


Figure 7.4: Box plots of (a) error at convergence (b) scatter at convergence (c) convergence time (s) for the proposed ranked voting algorithm (RV) and both variants of the benchmark algorithm (FT:fusion with transitivity preserved, FN:fusion without transitivity preserved) at different r_e settings; $\sigma_{noise} = 1.5$, $N_{robot} = 30$ [SM23]

voting algorithm, mean scatter is largely independent of evidence rates with $G = 0.0204$ and $R^2 = 0.007$.

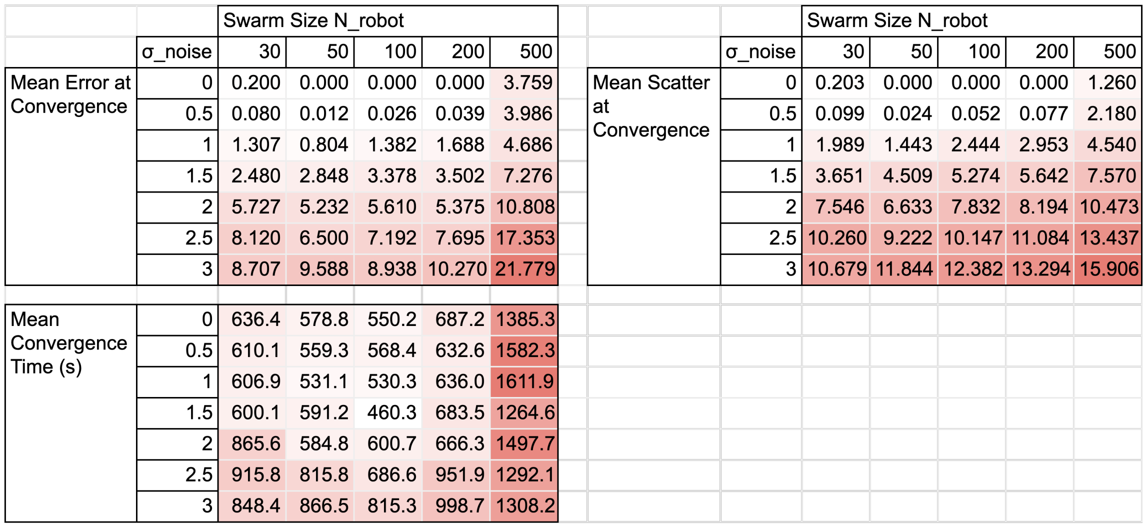
In terms of convergence time, all considered algorithms experience a significant increase in decision speed when the evidence rate increases from 0.01 to 0.1. Beyond 0.1, the median convergence time either experiences a slight increase as in the case of the two variants of belief fusion, or does not see much change as in the case of the proposed ranked voting algorithm. At the same time, both variants of belief fusion experience an increase in the variance of the convergence time at a high evidence rate. The same holds true for the proposed ranked voting algorithm when compared to the variance at $r_e = 0.1$.

Overall, the performances of the proposed ranked voting approach generally improve as the evidence rate increases, with a reduced error and convergence time. It is also more resistant to the effects of low evidence rates in terms of error compared to both variants of belief fusion. Its convergence time also increases at a slower rate than belief fusion without transitivity preserved when the evidence rate reduces, while being more vulnerable in this aspect compared to belief fusion with transitivity preserved. For the belief fusion benchmark, both variants see reducing error when the evidence rate increases, but both also see increasing scatter and a much higher uncertainty in convergence time as the evidence rate increases beyond 0.2.

7.3.4 Performances of Ranked Voting Strategy with Respect to Swarm Sizes

Afterward, the impact of swarm sizes N_{robot} on the performances of the proposed ranked voting algorithm is examined. The mean performances across 20 experimental runs at every parameter combination are shown in Table 7.4.

Table 7.4: Performances of proposed ranked voting algorithm at different noise levels σ_{noise} and swarm sizes N_{robot}; $r_e = 0.2$ [SM23]



		Swarm Size N_robot				
	σ_noise	30	50	100	200	500
Mean Error at Convergence	0	0.200	0.000	0.000	0.000	3.759
	0.5	0.080	0.012	0.026	0.039	3.986
	1	1.307	0.804	1.382	1.688	4.686
	1.5	2.480	2.848	3.378	3.502	7.276
	2	5.727	5.232	5.610	5.375	10.808
	2.5	8.120	6.500	7.192	7.695	17.353
	3	8.707	9.588	8.938	10.270	21.779
Mean Convergence Time (s)	0	636.4	578.8	550.2	687.2	1385.3
	0.5	610.1	559.3	568.4	632.6	1582.3
	1	606.9	531.1	530.3	636.0	1611.9
	1.5	600.1	591.2	460.3	683.5	1264.6
	2	865.6	584.8	600.7	666.3	1497.7
	2.5	915.8	815.8	686.6	951.9	1292.1
	3	848.4	866.5	815.3	998.7	1308.2

		Swarm Size N_robot				
	σ_noise	30	50	100	200	500
Mean Scatter at Convergence	0	0.203	0.000	0.000	0.000	1.260
	0.5	0.099	0.024	0.052	0.077	2.180
	1	1.989	1.443	2.444	2.953	4.540
	1.5	3.651	4.509	5.274	5.642	7.570
	2	7.546	6.633	7.832	8.194	10.473
	2.5	10.260	9.222	10.147	11.084	13.437
	3	10.679	11.844	12.382	13.294	15.906

It can be observed that for all three metrics, optimal behaviors are more likely to be observed at medium ranges of swarm sizes of 50 and 100, while the performances at extreme swarm sizes are often worse off. This is similar to the effects produced by varying the evidence rate r_e. However, there is a more clear worsening of all considered metrics at higher swarm sizes compared to evidence rates. This is to be expected as a higher swarm size not only introduces more evidence but also introduces more agents that need to be brought into convergence for a consensus to form.

7.3.5 Comparison with the Belief Fusion Benchmark at Different Swarm Sizes

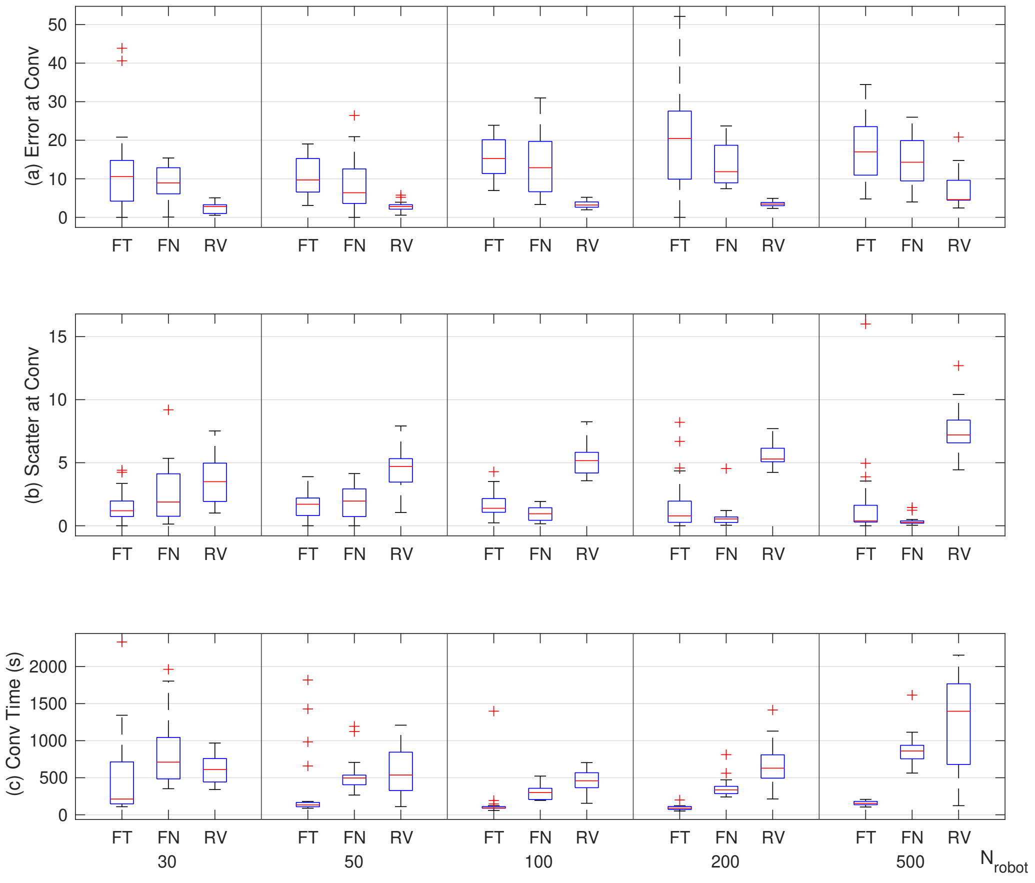


Figure 7.5: Box plots of (a) error at convergence (b) scatter at convergence (c) convergence time (s) for the proposed ranked voting algorithm (RV) and both variants of the benchmark algorithm (FT:fusion with transitivity preserved, FN:fusion without transitivity preserved) at different N_{robot} settings; $\sigma_{noise} = 1.5$, $r_e = 0.2$ [SM23]

This subsection compares the impact of swarm size N_{robot} on the performances of the considered algorithms, shown in Figure 7.5. The noise level σ_{noise} is set to 1.5, and the evidence rate r_e is set to 0.2. The results from linear regression of the mean performance against the natural log of the swarm size $ln(N_{robot})$ is shown in Table 7.5.

It can be observed that all considered algorithms experience an increase in error when the swarm size increases. This is substantiated by the statistical analysis in Table 7.5, where the proposed ranked voting algorithm obtains the lowest gradient of mean error

Table 7.5: Gradient and R^2 values obtained from linear regression of mean performances against $ln(N_{robot})$ [SM23]

Algo	LR of Error	LR of Scatter	LR of Conv Time
BF w Tr	$G = 2.74$ $R^2 = 0.53$	$G = -0.138$ $R^2 = 0.247$	$G = -137$ $R^2 = 0.689$
BF w/o Tr	$G = 2.37$ $R^2 = 0.782$	$G = -0.777$ $R^2 = 0.892$	$G = 5.38$ $R^2 = 0.001$
RV	$G = 1.53$ $R^2 = 0.777$	$G = 1.29$ $R^2 = 0.961$	$G = 218$ $R^2 = 0.598$

against $ln(N_{robot})$ at 1.53. Both variants of belief fusion also see a general reduction in scatter as the swarm size increases, while for the proposed ranked voting algorithm, there is still a clear linear relationship between scatter and swarm size. It is thus shown that as the number of agents increases, both variants of belief fusion see a stronger push towards consensus, which produces lower scatter but higher error. For belief fusion with transitivity preserved, this also translates to a lower convergence time, with a gradient of -137. The same effects are not observed in the proposed ranked voting algorithm, which sees its error scales much slower to swarm size. However, this comes at the cost of a higher and scaling convergence time with a gradient of 218. The proposed ranked voting algorithm also uses less communication bandwidth, storage, and processing power compared to the benchmarks, which makes it viable in large swarm sizes.

7.4 Summary and Discussion

This chapter focuses on the collective preference learning scenario and investigates the effects of restricting the available decision space during the consensus-forming process with the iterative ranked voting approach. Based on the experimental results, the performances of the proposed ranked voting approach can be characterized as being, in general, slower and less precise, but more accurate and cheaper than the benchmark belief fusion algorithms. There is especially a clear advantage of the ranked voting at high noise and high swarm size scenarios.

The differences in their performances are due to the different decision-making mechanisms used. The proposed ranked voting algorithm uses a more compact encoding method to represent the ranking among the sites. It is able to give a compromising result when agents of different opinions are combining their opinions. In contrast, the benchmark belief fusion algorithms revert all entries in conflict back to the initial unknown status of value 0, resulting in information loss. This feature, combined with the mechanism in belief fusion operation to always assign any available $+1$ or -1 entry values to entries with the unknown status, results in a positive feedback loop within the swarm. Thus, swarms using the belief fusion algorithm can come to a consensus rapidly, but when most of the belief matrices are filled, it is very hard for dissenting agents to spread their opinions, even if they hold correct pairwise information.

This mirrors the fast consensus forming by opinion-based and parallel encoding consensus approaches in the discrete collective estimation problem investigated in the previous chapters. However, it has been previously assumed that all potential options can be tracked by the robots. In contrast, in the collective preference learning problem among 8 sites investigated in this chapter, there are $8! = 40320$ possible ranking combinations. Therefore, it is impossible for the robots to accurately track all possible options, rather, both investigated approaches in this chapter try to approach the collective preference learning problem as an optimization problem and the individual agents seek to make incremental changes in the form of single pairwise relationships to approach the true preference order. In such an approach, the existence of a positive feedback loop in the decision-making process can cause the swarm to be stuck on a local optimum, where a few agents have more accurate ranking information, but could not overpower the established consensus, leading to premature convergence. The impact of such premature convergence on the accuracy of the consensus depends on two factors, the level of dynamism in the environment and the level of sensory capabilities of individual agents. In a dynamic environment, the established consensus can potentially prevent the swarm from responding to changes in the environment. On the other hand, this could also negatively impact the accuracy in a static environment when the individual agents have poor sensory capabilities, in terms of the environment being noisy or observations being hard to collect, due to establishing a consensus before the agents can make enough observations. This is substantiated by the performances of considered algorithms at high noise levels and low evidence rates respectively.

On the other hand, the proposed ranked voting algorithm restricts the decision space faced by the agents to only possible strict ordering combinations. This makes the decision space more convex and reduces the occurrences of local optima. In addition, it employs a degree of stochasticity in its election process. The ordering among options with tied points in the election result is random. Since there are only two voters, ties are fairly common. This leads to a higher scatter in the final result, as conflicting information needs many pairwise robot interactions to be eliminated. However, it also means that dissenting opinions have an opportunity to spread within the swarm. The whole swarm can thus readily shift in opinions and has a much better chance of approaching the true result. It is also less likely for a pairwise robot interaction to result in a loss of information, and the swarm can thus avoid being dominated by a single opinion.

Chapter 8

Multi-Objective Dynamic Decision Space Restriction in Swarm Task Allocation Problem

This chapter is based on [SM21a].

The previous chapter has shown how a many-option consensus-forming problem can be more effectively handled by choosing a decision-making strategy that minimizes the decision space size. This principle can also be applied to decision-making problems with continuous decision spaces, which offer greater freedom when customizing the quality functions and allow for deeper investigation of collective decision-making behaviors when facing complex quality distributions.

Previous investigated scenarios in this thesis all have static unimodal quality functions with a single correct decision, from which the more the swarm deviates, the worse the quality is. This chapter seeks to extend previous scenarios and investigate a collective decision-making scenario with a composite quality function, whose output depends on both the individual agent's decision and the decisions of all other agents in the vicinity. Depending on the multi-modality of the quality function, the optimal collective configuration can be states where the behaviors of the robots are not in uniformity, instead favoring specialization. An online search for the optimal collective configuration in a complex quality distribution has been studied in previous literature using various distributed optimization frameworks, especially embodied evolution, where evolutionary optimization algorithms are implemented on distributed platforms in a decentralized, online, and parallel manner [BHP18]. In embodied evolution, one important approach to encourage the emergence of specialized behaviors is with reproductive isolation via either geographical and communicative isolation [MCB16], or by considering the affinity between individuals during recombination such that an individual is recombined with others close to its own genotype [PBD09; Pri+10; Tru+13; TP18]. This study investigates the performance of the embodied evolutionary approach when facing the composite quality function, especially its ability to produce optimal specialized behaviors when facing a multi-modal quality function. In addition, a novel approach based on dynamic reproductive isolation via a collective decision mechanism has been proposed and tested.

8.1 Continuous Task Allocation Scenario Based on Local and Collective Quality Functions

Many biological swarm intelligence systems produce divergent and specialized behaviors out of homogeneous agents when facing complex tasks. The classical approach to explaining such behaviors is with response threshold models [BTD96]. Such models express the interactions between the environment and the agents during task allocation by having the agents respond to task-specific stimuli from the environment. If the stimulus exceeds the corresponding threshold, the agent has a high chance to engage in a task and vice versa. A minority of literature studies task allocation via interaction among the agents [GGT92; PGG96]. Recently, there has been a growing interest in such models [Gor16]. Notably, Chen et al. [CMG20] have introduced a game-theoretical model that explains the emergence of specialized behaviors in social insects via inter-agent interactions under reward functions that favor global payoff. This model serves as the basis for the construction of the composite quality function used in this study.

In the investigated continuous task allocation scenario, a group of agents with indices $i = 1..N_{robot}$ need to be allocated in a self-organized way to two tasks: A and B. The agents have to optimize a parameter $x_i \in [0, 1]$ which indicates the proportion of time the agent i spends on performing task A. The variable x_i needs to be optimized with respect to 2 quality functions $f(x_i)$ and $g(\bar{x})$. Here $f(x_i)$ is the local quality function that depends only on the agent's own decision, while $g(\bar{x})$ is the global quality function that depends on the total proportion of time among all agents devoted to the two tasks, hence $\bar{x}$. The agents are assumed to have only reactive behaviors with no planning capability. They also have limited communication ranges and can only exchange information with their peers nearby. They can measure the value of the local quality function $f(x_i)$, but can only measure a local estimate of the global quality function $g(\bar{x})$, denoted as g_i^*. It is computed using only the genomes within the communication range of robot n:

$$g_i^* = g(\bar{x}_j). \tag{8.1}$$

$$\{j | j \in \{1..N_{robot}\}; \text{robot } j \text{ is within communication range of robot } i\}$$

The total measured reward is therefore expressed as follows:

$$\Pi_i = (1-w)f(x_i) + wg_i^*. \tag{8.2}$$

The agents thus need to find:

$$x_i^* = argmax_{x_i}[(1-w)f(x_i) + wg_i^*]. \tag{8.3}$$

where $w \in [0,1]$ is the weight of g relative to the total reward and is used as an environmental parameter controlling the relative importance of the two tasks.

In this chapter, $g(\bar{x})$ is assumed to be unimodal and thus have a singular optimum. It models the optimality of the distribution of the whole swarm's efforts between tasks A and B. It gives the highest reward when the current effort allocation is at the desired allocation, and gives less reward the further the current effort allocation deviates from the desired allocation. On the other hand, $f(x_i)$ can be multimodal, and models the local characteristics of the agents. For example, when $f(x_i)$ is unimodal, the individual agents have an optimal distribution of efforts between the two tasks in question, and the characteristics of the agents do not favor specialization. In case $f(x_i)$ is multimodal, the agents can operate at optimal or near-optimal efficiency at multiple distributions of effort between the two tasks, therefore the characteristics of agents favor specialization.

8.2 Investigated Distributed Optimization Approaches

This section introduces four distributed optimization algorithms investigated in this chapter. Individual random walk is a baseline optimization algorithm that does not utilize inter-agent interactions. Two variants of the basic embodied evolutionary algorithm have also been implemented, with and without static reproductive isolation. Lastly, dynamic reproductive isolation based on collective decision-making has been attempted with the bi-objective embodied evolutionary approach.

8.2.1 Individual Random Walk Optimization

Algorithm 24 shows the individual random walk optimization approach. Under this approach, the agents do not exchange information with their peers and adopt a purely greedy approach to maximize its reward Π_i.

The agents start from randomly initialized x_i values within the decision space $[0,1]$. At every control loop, each agent i measures the fitness Π_i which denotes the current total obtained reward indicating the fitness of its current genome. The agent then compares Π_i with the fitness from its last control loop Π_i', and adopts the genome with the higher fitness. It then mutates its chosen genome using a Gaussian exploration noise with a standard

Algorithm 24 Individual Random Walk [SM21a]

```
Input: Agent index $i \in \{1..N_{robot}\}$; initial $x_i$
Output: Optimized $x_i$
1: Set exploration noise level $\sigma_{\text{exp}}$
2: $x'_i = 0$; $\Pi'_i = -\infty$
3: while True do
4:     Current Weighted Total Fitness: $\Pi_i = (1 - w)f(x_i) + wg^*_i$
5:     if $\Pi_i >= \Pi'_i$ then
6:         $x'_i = x_i$
7:         $\Pi'_i = \Pi_i$
8:         $x_i = \mathcal{N}(x_i, {\sigma_{\text{exp}}}^2)$, restricted between [0, 1]
9:     else
10:        $x_i = \mathcal{N}(x'_i, {\sigma_{\text{exp}}}^2)$, restricted between [0, 1]
```

deviation of σ_{exp} (lines 8 and 10). This baseline algorithm achieves optimization via only the information available to the individual agents themselves, with no communication. It is expected that a viable collective approach needs to perform better than this baseline algorithm to justify the added communication complexity.

8.2.2 Baseline Embodied Evolutionary Algorithms

This subsection introduces the implementations of the embodied evolutionary approach of producing optimized task allocations.

Algorithm 25 Baseline Embodied Evolution [SM21a]

```
Input: Agent index $i \in \{1..N_{robot}\}$; initial $x_i$
Output: Optimized $x_i$
1: Set exploration noise level $\sigma_{\text{exp}}$
2: while True do
3:     Current Weighted Total Fitness: $\Pi_i = (1 - w)f(x_i) + wg^*_i$
4:     Broadcast $x_i$, $\Pi_i$ to neighbors
5:     Collect $x_j$, $\Pi_j$ from $J$ agents in communication radius
6:     if $J > 0$ & max$(\Pi_j) > \Pi_i$ then
7:         $x_i = x_{\text{j*}}, j* = \text{argmax}_j(\Pi_j)$
8:         $x_i = N(x_i, {\sigma_{\text{exp}}}^2)$, restricted between [0, 1]
9:     Clear recorded $x_j$ and $\Pi_j$
```

Algorithm 25 shows the outline for a baseline Embodied Evolution (EE) algorithm similar to the one presented in [BHP18]. At every control loop, the agents exchange genomes and their corresponding fitness with their neighbors (lines 4-5). An agent selects the genome that corresponds to the highest fitness value among its neighbors (including itself), denoted by the index $j = 1, \cdots, J$ (line 7). After that, a Gaussian exploration noise with standard deviation σ_{exp} is added to mutate the genome (line 8). This is a very standard implementation of embodied evolution, and is widely used in other studies in the field.

Algorithm 26 EE with Affinity Bias [SM21a]

```
Input: Agent index $i \in \{1..N_{robot}\}$; initial $x_i$
Output: Optimized $x_i$
 1: Set exploration noise level $\sigma_{\text{exp}}$
 2: while True do
 3:     Current Weighted Total Fitness: $\Pi_i = (1-w)f(x_i) + wg_i^*$
 4:     Broadcast $x_i$, $\Pi_i$ to neighbors
 5:     Collect $x_j$, $\Pi_j$ from $J$ agents in communication radius
 6:     if $J > 0$ & $\max(\Pi_j) > \Pi_i$ then
 7:         $\Pi'_j = \Pi_j * \exp(-|x_i - x_j|)$
 8:         $x_i = x_{\text{j*}}, j* = \text{argmax}_j(\Pi'_j)$
 9:         $x_i = N(x_i, {\sigma_{\text{exp}}}^2)$, restricted between [0, 1]
10:     Clear recorded $x_j$ and $\Pi_j$
```

Algorithm 26 uses a similar affinity bias in the selection process as proposed in [PBD09; Pri+10]. Most of the mechanisms are the same as in Algorithm 25. However, in this algorithm, the fitness of neighboring robots which are compared during selection, is multiplied by an affinity factor. This factor decreases as the difference between the two genomes increases (line 7). In this way, an agent selects the optimal genome using the fitness adjusted with the affinity factor (line 8). Therefore, the agents are encouraged to select genomes close to their own. This enforces a static reproductive isolation, and is able to encourage the whole population to diverge into multiple subgroups to suit the requirements of an environment. Similar use of affinity bias is popular in embodied evolution to evolve divergent cooperative behaviors, and is used in various state-of-the-art literature [Tru+13; TP18].

8.2.3 Bi-Objective Embodied Evolution

Algorithm 27 is proposed as an alternative algorithm to perform optimal task allocation in the considered scenario. It seeks to navigate the composite quality function using a collective decision mechanism that dynamically restricts exploration in the decision space.

The variable $y_i \in \{-1, 1\}$ is introduced which indicates an agent's preferred direction to change the current genome x_i. $y_i = 1$ means that agent i prefers to increase x_i and $y_i = -1$ means that agent n prefers to decrease x_i. The estimated reward of the preferred direction is denoted by ρ_{i}.

Algorithm 27 Bi-Objective Embodied Evolution [SM21a]

```
Input: Agent index $i \in 1..N_{robot}$; initial $x_i, y_i$
Output: Optimized $x_i$
1: Define exploration noise level $\sigma_{\text{exp}}$
2: $\rho_{\text{i}} = 0$, $\Pi'_i = -\infty$
3: $\Pi_{mean} = 0, \Pi'_{mean} = -\infty$
4: $x'_i = 0$, $state_i = 0$
5: while True do
6:     Current Weighted Total Fitness: $\Pi_i = (1-w)f(x_i) + wg^*_i$
7:     Broadcast $x_i$, $y_i$, $\Pi_i$, $\rho_i$ to neighbors
8:     if $state_i$=0 then
9:         # Local random walk, searching for local optimum
10:        if $\Pi_i \geq \Pi'_i$ then
11:            $x'_i = x_i$, $\Pi'_i = \Pi_i$
12:            $x_i = N(x_i, \sigma_{\text{exp}}{}^2)$, if $y_i = -1$ restricted between $[0, x_i]$, else between $[x_i, 1]$
13:        else
14:            $x_i = N(x'_i, \sigma_{\text{exp}}{}^2)$, if $y_i = -1$ restricted between $[0, x_i]$, else between $[x_i, 1]$
15:        $state_i = 1$
16:    else
17:        Collect $x_j$, $y_j$, $\Pi_j$, $\rho_{\text{j}}$ from $J$ neighbors in communication radius
18:        # CDM on expected direction of option change that leads to improvement
19:        $y_{\text{Mode}} = \text{Mode}(y_{j,\, j\,=\,1,\ldots,\,J}, y_i)$
20:        $\Pi_{mean} = \text{Mean}(\Pi_{j,j=1,\ldots,J}, \Pi_i)$
21:        $\Delta_{\text{fitness}} = \Pi_{mean} - \Pi'_{mean}$
22:        $\rho_{\text{i}} = \Delta_{\text{fitness}} \times y_i \times y_{\text{Mode}} \times M$
23:        $\Pi'_{mean} = \Pi_{mean}$
24:        if $\text{max}(\rho_m) > \rho_i$ then
25:            $y_i = y_j, j = \text{argmax}_j(\rho_{j,\, j\,=\,1,\cdots,\,J})$
26:        if $\Delta_{\text{fitness}} < 0$ and $y_i = y_{\text{Mode}}$ then
27:            $y_i = -y_i$
28:        # Decision making on individual opinion
29:        if $y_i = -1$ then $\vec{j}_{\text{available}} = \vec{j}[x_j < x_i]$
30:        else $\vec{j}_{\text{available}} = \vec{j}[x_j > x_i]$
31:        if $\vec{j}_{\text{available}} \neq \emptyset$ then
32:            $x'_i = x_i$, $\Pi'_i = \Pi_i$
33:            $x_i = x_j, j = \text{RandomChoice}(\vec{j}_{\text{available}})$
34:        $state_i = 0$
35:        Clear recorded $x_j$, $y_j$, $\Pi_j$ and $\rho_j$
```

In this algorithm, a collective decision-making process that determines the more suitable y_i value is running simultaneously along with the optimization of genome x_i. Optimization of x_i is done in two alternating states, indicated by $state_i$. In State 0 (lines 8-15), the agent performs a random walk similar to Algorithm 24. However, the agent will only sample a new value of x_i in the preferred direction indicated by y_i. In State 1 (lines 17,29-35), the agent receives x_j from its J neighbors and uses them to update its own opinion x_i similar to Algorithm 25 and 26, but will only select a random genome within its preferred direction indicated by y_i. The random walk in State 0, ensures that the random selection process in State 1 produces better offspring than the previous generation.

The preferred direction y_i is determined in another decision-making process (lines 18-27) that is also executed in State 1. The agents exchange their current y_i values and their corresponding qualities ρ_i with their neighbors. Each agent then computes y_{Mode}, which is the prevailing y value in all neighboring agents and the agent itself. y_{Mode} is used to estimate the prevailing y in the whole swarm. Π_{mean} is computed as the mean of current total fitness obtained by all agents in the vicinity of agent n. The change in Π_{mean} (denoted as Δ_{fitness}) between consecutive control loops is then computed. Δ_{fitness} is used to estimate the effect of the current prevailing y on the total fitness. It is expected that when the current prevailing y in the whole swarm indicates a direction that improves $g(\bar{x})$, agents are more likely to experience an increase in their own total fitness. Also, if the agent's individual y_i is the same as the prevailing y in the whole swarm, then it has contributed positively to the effect of the latter, while if they disagree, then the contribution is negative. Thus, the decision-making mechanism is designed such that when y_i is the same as y_{Mode}, Δ_{fitness} contributes positively to the computation of y_i's quality, and vice versa. The quality of y_i (denoted as ρ_i) is computed in line 22 as follows:

$$\rho_{\text{i}} = \Delta_{\text{fitness}} \times y_i \times y_{\text{Mode}} \times M. \tag{8.4}$$

ρ_i scales with both Δ_{fitness} and the number of neighbors M. The latter is because sampling the y values of more neighbors leads to a more accurate estimation of the prevailing y of the whole population, therefore making the estimation of the quality of y_i more accurate. After that, an optimal new y_i is chosen from the y values of an agent's neighbors and that of itself based on their qualities ρ (lines 24-25). Finally, if $\Delta_{fitness}$ is negative and y_i is equal to the prevailing y, the performances of the whole swarm is likely decreasing and the agents are moving away from optimal genome selections. Therefore, y_i value would be flipped to reverse this trend (lines 26-27).

Overall, the proposed bi-objective embodied evolutionary algorithm does not freely copy high-fitness genomes from other individuals in a population, as in traditional embodied evolutionary algorithms. In contrast, agents use the genome and fitness information from their neighbors to form an estimation of the more potentially optimal direction y_i to change its own genome x_i, and thus restrict the neighbors available for selection. This enforces a dynamic reproductive isolation which is both able to select a genome with high combined fitness and to maintain a separation of specialized behaviors if required.

8.3 Experiments and Results

In order to freely control the shape of the quality functions faced by the intelligent swarm, the experiments are conducted in an abstract manner without actual simulation of the tasks. The control loops of the agents are simulated to be distributed and asynchronous processes with lengths following an exponential distribution $exp(0.1)s$. The sporadic connectivity among the swarm agents is simulated in a physics-based environment where 20 mobile agents constantly perform random walks in a $2m \times 2m$ arena, with the default communication and interaction range being $0.5m$,

The rest of this section presents the experimental results. The optimal parameter settings for the considered algorithms are first determined. Then, w is varied between 0 and 1 to observe the performance of all considered algorithms with different levels of consideration of global and local reward. At each environmental configuration, 20 experiments are conducted. The performance is measured via the peak total local fitness $\Sigma f(x_i)$ and total global fitness $N_{robot} g(\bar{x})$. The performances of considered algorithms will be compared among each other and with the Pareto frontier produced by NSGA-II, which is used to determine the theoretical optimum of the experiment scenarios. The average total fitness obtained over a 60-second period is also measured to determine the long-term performances of the considered algorithms. Finally, a stochastic term is added to the quality functions to see the performances of considered algorithms when facing uncertainties and noise.

8.3.1 Selection of σ_{exp}

An important parameter for all considered algorithms is σ_{exp}. It is selected in the following scenario. Both $f(x_i)$ and $g(\bar{x})$ are set to be simple unimodal functions defined as follows and shown in Figure 8.1:

$$g(\bar{x}) = exp(-(\frac{\bar{x} - 0.8}{0.2})^2) \tag{8.5}$$

$$f(x_i) = exp(-(\frac{x_i - 0.6}{0.2})^2). \tag{8.6}$$

This scenario is meant to model task allocation problems in an environment that does not favor specialization. Since both of these quality functions have only a single optimum which is at different positions, there is a clear trade-off. An agent that moves its option closer to the optimum of $f(x_i)$, its corresponding $\bar{x}$ will move further away from the optimum of $g(\bar{x})$.

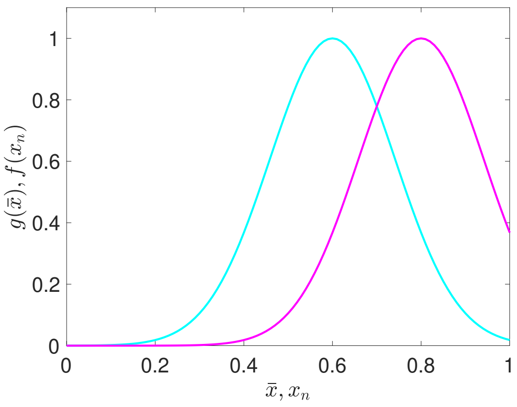


Figure 8.1: Illustration of the global $g(\bar{x})$ (magenta) and the local $f(x_i)$ (cyan) quality functions in Scenario 1 [SM21a]

Table 8.1: Fitness performances at different σ_{exp} for all considered algorithms, $w = 0.5$ [SM21a]

	σ_{exp}	0.01	0.02	0.05	0.1	0.2	0.5
Individual Baseline	Peak Global Fitness	6.703	9.620	12.511	14.440	17.436	19.229
	Peak Local Fitness	15.228	17.235	17.155	15.418	12.385	8.613
	Mean Total Fitness	9.624	11.971	13.142	12.224	9.709	5.925
Baseline EE	Peak Global Fitness	10.464	11.292	13.303	15.775	18.054	19.357
	Peak Local Fitness	18.808	18.264	16.733	14.385	11.921	8.410
	Mean Total Fitness	13.847	13.654	12.852	11.456	9.197	6.092
EE with Affinity Bias	Peak Global Fitness	9.779	10.634	12.724	15.500	17.915	19.263
	Peak Local Fitness	19.043	18.463	16.983	14.477	11.909	8.193
	Mean Total Fitness	13.637	13.413	12.720	11.355	9.169	5.804
Bi-Objective EE	Peak Global Fitness	15.326	15.703	17.834	18.411	18.932	19.588
	Peak Local Fitness	15.818	15.403	13.595	12.897	11.723	8.825
	Mean Total Fitness	14.792	14.637	13.923	12.877	11.288	9.130

In order to find the optimal σ_{exp} value, both the peak performances and the continuous performances of considered algorithms are measured. The peak performances are measured via the global fitness and the local fitness when the total fitness obtained by the population reaches maximum during an experimental run. The weight of global fitness w is set to be 0.5 to select a σ_{exp} value that can deliver results with balanced global and local fitness. The performances at different σ_{exp} values are shown in Table 8.1.

It can be observed in Table 8.1 that the peak global fitness and the peak local fitness are conflicting objectives with respect to the selection of σ_{exp}, as for all considered algorithms, a higher σ_{exp} produces a higher peak global fitness and a lower peak local fitness, and vice versa. This is because a high local fitness requires all individuals to have similar genomes that are close to the optimum of the local quality function $f(x_i)$. This is hard to achieve when the individuals undergo drastic random mutations with a high σ_{exp} after every control loop. On the other hand, a high σ_{exp} enables the population to explore the global quality function $g(\bar{x})$ more effectively and hence attain a higher global fitness. Given the conflicting nature of the two objectives, it is ideal to balance the two quality functions when the weight w is set to 0.5.

It is also desirable for the considered algorithms to stay on high-fitness genomes continuously, given that the algorithms are designed to run online. Therefore, the long-term continuous performance of the algorithms is considered at different parameter settings, and the mean total fitness during a 60-second period is measured. It can be observed that the individual random walk baseline algorithm reaches maximum mean total fitness at $\sigma_{exp} = 0.05$, while for all variants of embodied evolution, mean total fitness decreases as σ_{exp} increases. The former is because the agents in individual random walk baseline do not copy genomes from each other, and thus all have to approach the optimum via exploratory mutations. Therefore, σ_{exp} needs to have a larger value to ensure a fast convergence. For other embodied evolution algorithms, the ability to copy genomes from other individuals ensures that convergence is quick even with a small σ_{exp}, while a large value of σ_{exp} can introduce instability to the population and reduce the mean total fitness.

Taking an integrated look at both aspects of the algorithms' performances, σ_{exp} values are picked for the considered algorithms that do not compromise the mean total fitness too much while seeking a balance between global and local peak fitness. The chosen σ_{exp} values are shown in Table 8.2.

Table 8.2: Choice of σ_{exp} for different algorithms [SM21a]

	Individual	Base EE	EE Affinity	Bi-Ob
σ_{exp}	0.05	0.02	0.02	0.02

8.3.2 Optimal Task Allocation

In this subsection, w is varied from 0 to 1 in an interval of 0.02 and the local and global fitness obtained across the considered algorithms have been compared.

Scenario 1: Unimodal $f(x_i)$

The first scenario uses the same unimodal $f(x_i)$ and $g(\bar{x})$ as in 8.3.1. The results are shown in Figures 8.2, 8.3 and 8.4.

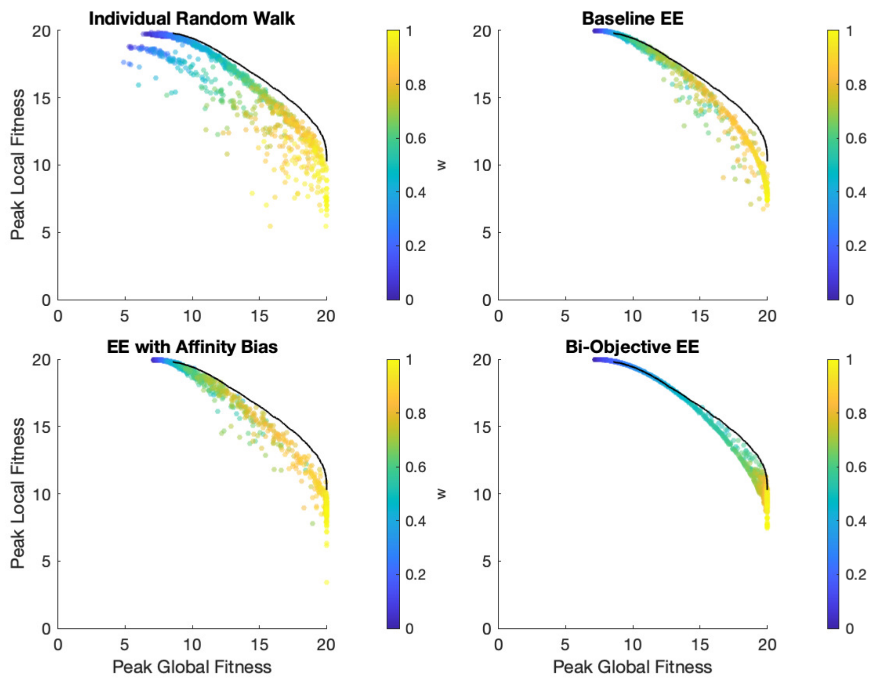


Figure 8.2: Optimal global and local fitness obtained during each experimental run for all considered algorithms in Scenario 1; color scale indicates weight of global fitness w; black line indicates the Pareto frontier computed by NSGA-II [SM21a]

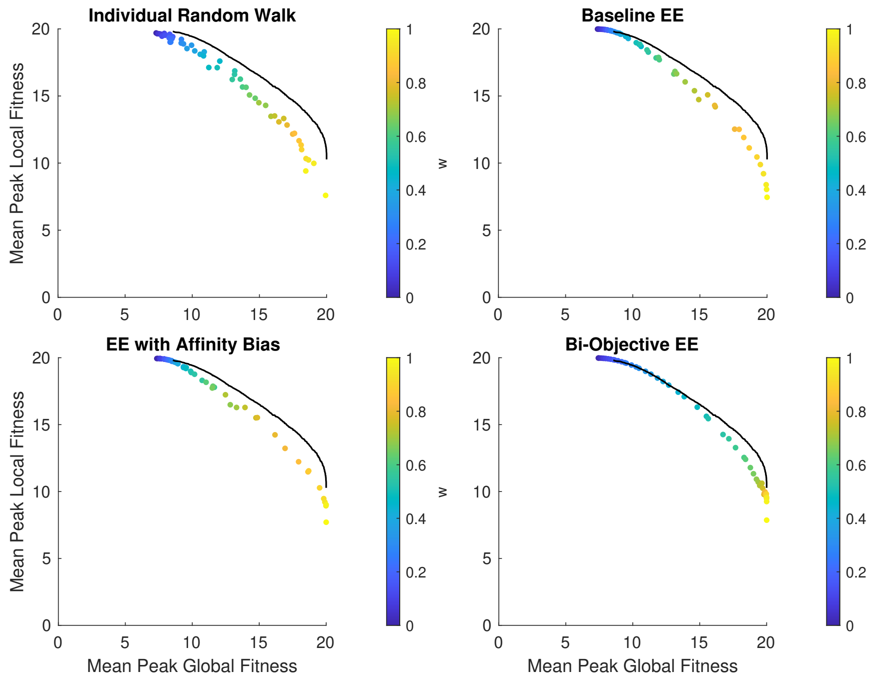
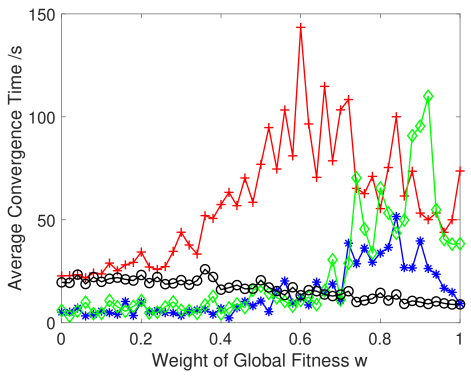
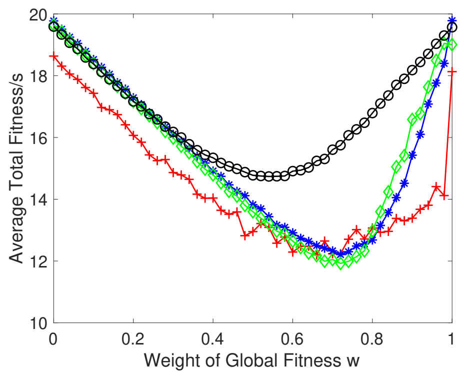


Figure 8.3: Average optimal global and local fitness at each w value in Scenario 1 [SM21a]



(a)



(b)

Figure 8.4: (a) Average convergence time and (b) average total fitness obtained per second of all considered algorithms in Scenario 1; red+:Individual BL, blue*:BL EE, green◇:EE Affinity, black○:Bi-Objective EE [SM21a]

As shown in Figure 8.2 (top left), the Individual Random Walk approach is sometimes able to find the optimal global and local fitness quite well in its decision-making process. However, it also frequently converges to a suboptimal solution, indicated by the scattered data points below the Pareto frontier. Therefore, as shown in Figure 8.3 (top left), the mean output at medium to high w values are quite far from the Pareto frontier. Regarding the convergence time (Figure 8.4(a)), Individual Random Walk is usually slower than other algorithms and its convergence time increases with w. Its long convergence time is due to the fact that the agents do not learn genomes with high fitness from each other, and they have to approach the optimal point using the random exploration steps. Its increase with respect to w is because for the total reward $(1-w)f(x_i)+wg_i^*$, x_i has a much larger impact on $f(x_i)$ than on g_i^*. Therefore, the higher w is, the less impact x_i has on the total fitness, and less accurate an estimate of the fitness improvement of an exploration step is.

The peak performances of the Baseline EE (Figure 8.2 top right) are mostly clustered together with high local fitness and low global fitness, except at high w values beyond 0.7. This is due to the fact that the decision mechanism of traditional evolutionary algorithms relies on comparing the fitness among the individuals. In this case, the agents compare the total reward received $\Pi_i = (1-w)f(x_i)+wg_i^*$. Since g_i^* values are very close within a local group of agents, it is often ignored in the comparison. Thus, the agents will frequently converge to the solution that maximizes $f(x_i)$. At high w values beyond 0.7, the high weight causes the small differences in g_i^* to be magnified and enables the algorithm to consider the global reward. As shown in Figure 8.4(a), Baseline EE is the fastest algorithm for most of the cases, since the agents learn good options from each other and will quickly converge to an optimal option as long as it is reached by one agent.

EE with Affinity Bias (Figure 8.2 bottom left) has similar peak performances to Baseline EE, as the affinity bias is mainly designed to create behavioral specialization and does not affect the evolution process significantly when the environment does not encourage specialization [PBD09].

Finally, Bi-Objective EE (Figure 8.2 bottom right) obtains peak performances that are very close to the Pareto frontier. The results are considerably more consistent than those of Individual Random Walk. They are also more balanced than those produced by Baseline EE and EE with Affinity Bias, as the results at different w values are evenly spread along the Pareto frontier. Bi-objective EE avoids excessively focusing on local fitness like the other two variants of EE. Since it simultaneously decides on the preferred direction to change the agents' options y_i as well as the agents' genomes x_i. The decision-making process on y_i is able to keep track of the temporal changes of the agent's own fitness and the distribution of allocated tasks in the agent's locality $\bar{x}_{n^*}$, and therefore is able to efficiently optimize g_i^*. As shown in Figure 8.4(a), Bi-Objective EE is faster than the individual baseline, but is slightly slower than the other two variants of EE except at high w settings.

The average total fitness values obtained per second over a whole experimental run of $60s$ by all considered algorithms are shown in Figure 8.4(b). They are plotted against the weight of global quality function w. It can be observed that all 4 algorithms have performances that are close to each other at both extreme ends of the values of w, while Bi-Objective EE outperforms the other algorithms, especially both variants of EE, at medium to high values of w from 0.4 to 1. As shown in Figures 8.2 and 8.3, these are the values where the peak performances of both variants of EE are skewed towards the local reward.

Overall, it can be observed that when the characteristics of the agents do not favor specialization, the agents need to be able to find an optimal balance between their individual local quality function and the global quality function. Therefore, although the two existing variants of embodied evolution is able to exceed the performance of the individual baseline when w is at either extreme ends of its values, they are unable to effectively consider the interactions between the two quality functions, and therefore are unable to achieve the optimal behaviors when w is at a medium value. On the contrary, Bi-Objective EE is able to attain optimal behavior consistently. It outperforms the two existing variants of embodied evolution in its bi-objective performances, while outperforming Individual Random Walk in reliability.

Scenario 2: Bimodal $f(x_i)$

The second experiment scenario keeps $g(\bar{x})$ as in Scenario 1 and defines a bimodal $f(x_i)$ as follows:

$$f(x_i) = exp(-(\frac{x_i - 1}{0.2})^2) * 0.9 + exp(-(\frac{x_i - 0.3}{0.2})^2). \tag{8.7}$$

This scenario significantly differs from Scenario 1 in two ways: First, it is not guaranteed that the agents reach the optimum of $f(x_i)$ by taking incremental steps of improvement from a random initial position. Second, this scenario is meant to model task allo-

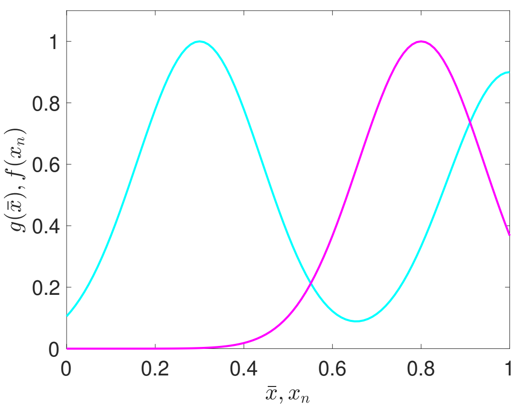


Figure 8.5: Illustration of the global $g(\bar{x})$ (magenta) and local $f(x_i)$ (cyan) quality functions in Scenario 2 [SM21a]

cation scenarios that favor specialization and test the ability of the agents to self-organize into multiple subgroups. Since there are 2 local optima in $f(x_i)$, the agents' x_i can be split between the two optima in a particular proportion so that $\bar{x}$ lands on the optimum of $g(\bar{x})$. In addition, given the fact that the optimization problem is multi-modal with a lot of variables, NSGA-II has some difficulties in reaching the true Pareto frontier.

In the peak performances produced by the Individual Random Walk (Figure 8.6 top left), there are a large number of data points that are below the Pareto frontier and arranged in stripes. This is because of the characteristics of this multi-modal scenario. Since there are two local optima, the population needs to converge to one of them when w is low and achieves a particular proportion between them otherwise. Since the individuals can only modify their genomes via exploratory random mutations, if an incorrect proportion of agents converges to one local optimum of $f(x_i)$, there is no way to change x_i from one local optimum to another. Therefore, from the perspective of the whole population, there are many local optima in terms of total fitness to which the swarm can converge. In Figure 8.7 top left, it can be observed that this approach works well at low w values but is unable to stay close to the Pareto frontier at high w values. This is a significant drop in performance compared to the results of the same algorithm in Scenario 1. It can be additionally observed that the Individual Random Walk fails to achieve the theoretical optimal performance at very low w values, which should have an optimal local fitness of close to 20 and global fitness of close to 0, compared to the results produced by Baseline EE and Bi-Objective EE.

Baseline EE (Figure 8.6 top right) produces peak performances that mostly stay close to the Pareto frontier, with fewer suboptimal data points than Individual Random Walk. As can be seen in the plot of mean peak fitness (Figure 8.7 top right), Baseline EE is able to reach optimal performances consistently at low w values. However, at medium and high w values, the results are unable to converge to the desired top-right corner of the graph. Here,

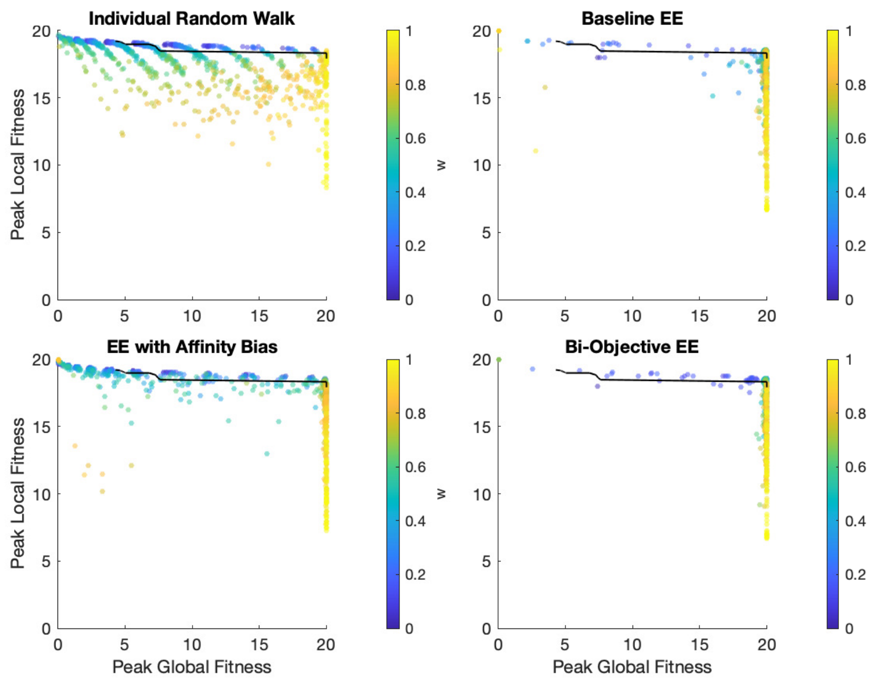


Figure 8.6: Optimal global and local fitness obtained during each experimental run for all considered algorithms in Scenario 2; color scale indicates weight of global fitness w; black line indicates the Pareto frontier computed by NSGA-II [SM21a]

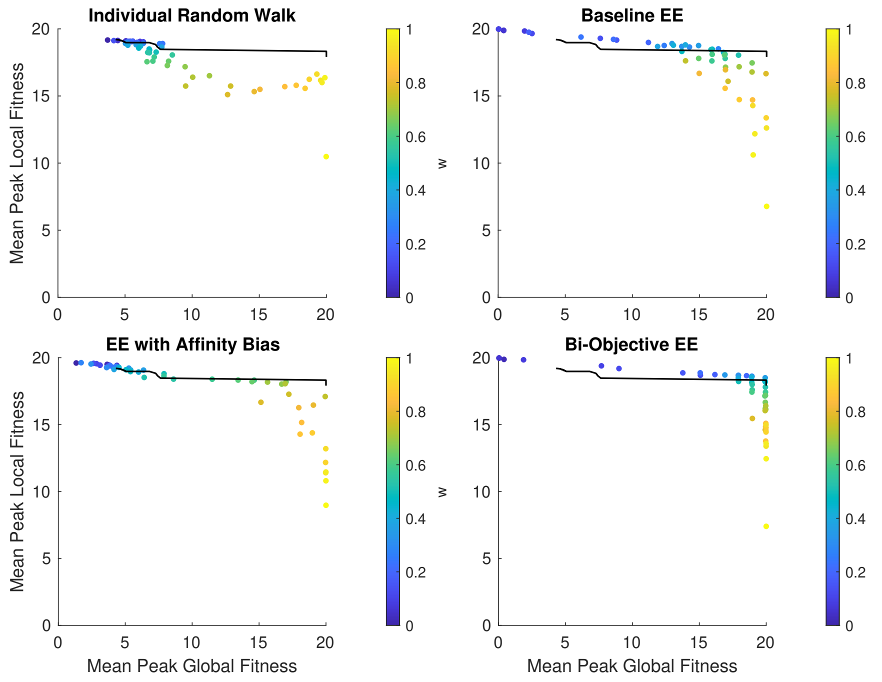


Figure 8.7: Average optimal global and local fitness at each w value in Scenario 2 [SM21a]

the same limitations in achieving optimal performances at high w values as in Scenario 1 is displayed for Baseline EE.

EE with Affinity Bias (Figure 8.6 bottom left) produces less suboptimal data points than Individual Random Walk but more than Baseline EE. This is because the reproductive isolation created by the affinity bias limits the collective optimization among the agents, and produces many potential local minima similar to Individual Random Walk. From the mean peak fitness plot (Figure 8.6 bottom left), it can be observed that EE with Affinity Bias only achieves optimal performances consistently at low w values, while the results can only approach the right section of the Pareto frontier at w values beyond 0.7.

As shown in Figure 8.6 (bottom right), Bi-Objective EE has far fewer outlying data points below the Pareto frontier, with most data points clustering at either the top left and the top-right corner of the graph. As can be seen in Figure 8.7 (bottom right), Bi-Objective EE is not only able to consistently deliver optimal local fitness of 20 at very low w values, comparable to Baseline EE, but also to produce higher global fitness as w increases, which significantly outperforms other considered algorithms.

To understand the differences between the performances of considered algorithms, the progression of individual genomes during an experimental run when $w = 0.5$ is shown in Figure 8.8. The three approaches, Individual Random Walk, EE with Affinity Bias, and Bi-Objective EE are able to split the whole population into 2 subgroups centering around the two local optima of the local quality function in Figure 8.5. In contrast, the whole population in Baseline EE eventually converges to one of the two local optima, thus ignoring the global quality function entirely. This is why Baseline EE is not effective at dealing with a global quality function.

On the other hand, although both the Individual Random Walk and the EE with Affinity Bias can split the individuals into two subgroups, the split does not consider the optimal proportion of agents in each subgroup and is thus heavily influenced by the initial distribution of genomes. As shown in Figure 8.9, in both algorithms $\bar{x}$ remains around 0.5 despite the changes in individuals' genomes. This is because the genomes are initialized with a random value between 0 and 1, and the initial $\bar{x}$ is likely to be around 0.5. As the individuals converge to the local optima of the local quality function closest to their initial genomes, the mean of all genomes $\bar{x}$ is unlikely to change much and thus still stays around 0.5, which is unlikely to be the optimal value for the global quality function.

Finally, Bi-Objective EE is able to split the population into two subgroups, and to regulate the proportion of individuals in each subgroup so that the global distribution of specialized behaviors is optimal. As shown in Figure 8.9, $\bar{x}$ under Bi-Objective EE starts near 0.5 but moves to around 0.8, which is the optimal value of the global quality function, and oscillates around it. This ensures not only the individual genomes converge to optima of the local quality function, but also the global distribution behavior is optimal as well.

Overall, when the characteristics of the agents favor specialization, Bi-Objective EE outperforms the other considered algorithms in reaching the optimal behaviors. This is due

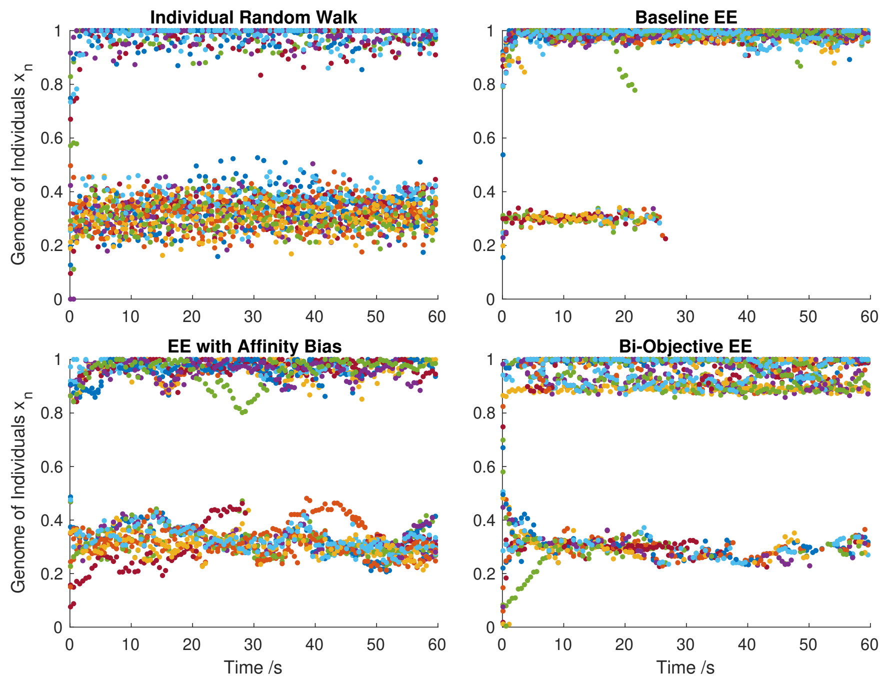


Figure 8.8: Progression of all agents' genomes x_i through time ($w = 0.5$); colors represent different agents [SM21a]

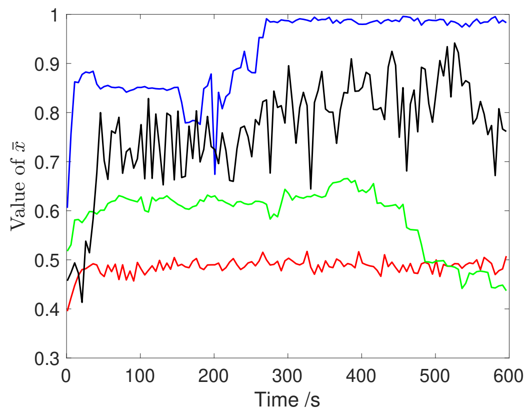
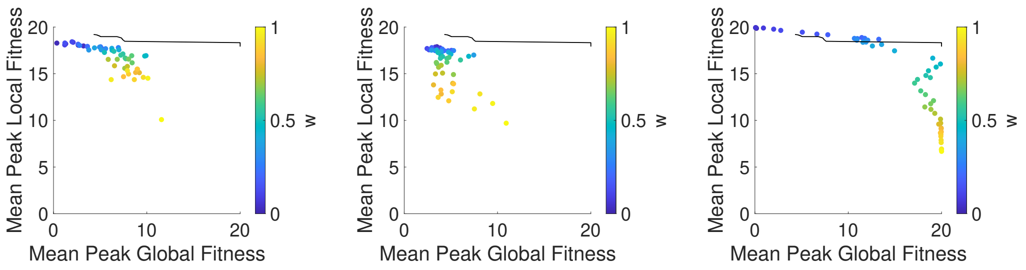


Figure 8.9: Progression of $\bar{x}$ through time ($w = 0.5$); red:Individual BL, blue:BL EE, green:EE Affinity, black:Bi-Objective EE [SM21a]

to its ability to consider the two objectives while using a collective approach; therefore, it overcomes the multi-modality of the local quality function.

Performance under Sparse Communications

In order to investigate the performances of considered algorithms when communications are sparse, the communication range of the simulated agents is reduced to $0.1m$ for this experiment and the optimal global and local fitness obtained is observed. As shown in Figure 8.10 (a,b), when communications are sparse, existing variants of embodied evolution experience a significant decrease in performance. This is because these algorithms rely on selecting the fittest genome from their neighbors to converge to an optimal behavior. When the communications become sparse, the number of available neighbors decreases, and thus it becomes harder for the whole population to reach a high total fitness.



(a) Baseline EE (b) EE with Affinity Bias (c) Bi-Objective EE

Figure 8.10: Average optimal global and local fitness at each w value in Scenario 2 under sparse communications, color scale indicates weight of global fitness w; black line indicates the Pareto frontier computed by NSGA-II [SM21a]

On the other hand, as shown in Figure 8.10 (c), Bi-Objective EE is able to maintain a high performance under sparse communications. Different from the other two embodied evolutionary algorithms, Bi-Objective EE does not directly copy high-fitness genomes from their neighbors, but also relies on the temporal changes of their own fitness values to determine the optimal behavior. Therefore, it is more resilient to a reduction in the number of neighbors. However, information from neighbors is important in Bi-Objective EE to accurately determine the fitness of its y values. Thus, when comparing its performance here with the previous section in Figure 8.7 (bottom right), the performance here experiences a drop in both the global and the local fitness obtained, as fewer data points concentrate in the top-right corner.

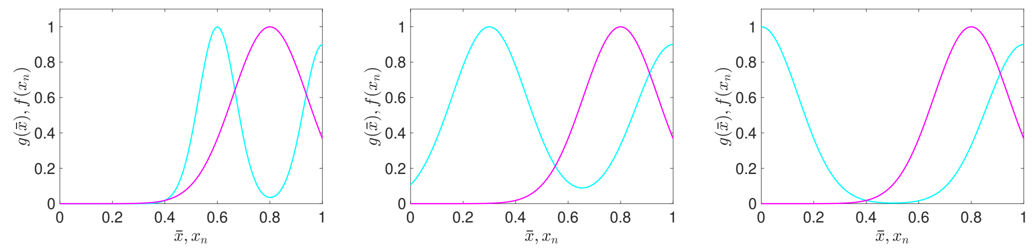
8.3.3 Long-Term Performances of Specialized Behaviors

This subsection focuses on the long-term performances of the considered algorithms in producing and maintaining specialized behaviors. This aspect of the algorithms is important given that they are designed to run online. In order to gauge the continuous performances, the mean total fitness attained by the considered algorithms is compared. As discussed in [MCB16], behavioral specialization is more easily done when the targeted distribution is roughly equal. Therefore, three different scenarios that encourage a division of labor are introduced, with each requiring progressively more imbalanced distribution of subgroups, as shown in Table 8.3. The optimal proportions for the two subgroups are $1:1$, $2:5$, and, $1:4$ respectively. Illustrations of these quality functions are shown in Figure 8.11. The average fitness obtained by the considered algorithms is examined in all

3 scenarios and with different w values. The same experiments are also repeated with a stochastic term added to the quality functions to measure the performances when facing uncertainty.

Table 8.3: Local quality functions in scenarios 3.1 - 3.3 [SM21a]

Scenario	Local quality function
3.1	$f(x_i) = exp(-(\frac{x-1}{0.1})^2) * 0.9 + exp(-(\frac{x-0.6}{0.1})^2)$
3.2	$f(x_i) = exp(-(\frac{x-1}{0.2})^2) * 0.9 + exp(-(\frac{x-0.3}{0.2})^2)$
3.3	$f(x_i) = exp(-(\frac{x-1}{0.2})^2) * 0.9 + exp(-(\frac{x-0.0}{0.2})^2)$



(a) Quality functions 3.1 (b) Quality functions 3.2 (c) Quality functions 3.3

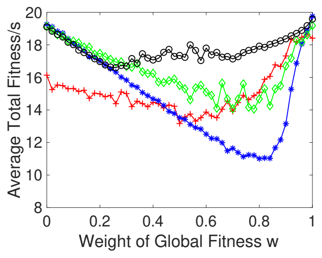
Figure 8.11: Illustrations of the global $g(\bar{x})$ (magenta) and local $f(x_i)$ (cyan) quality functions in Scenario 3.1 - 3.3 [SM21a]

Average Fitness Obtained in the Absence of Noise

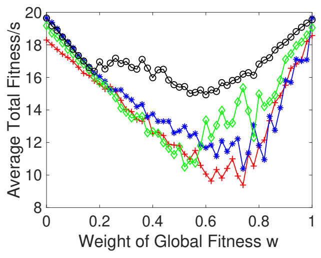
The mean total fitness obtained at every second is shown in Figure 8.12. It can be observed that both Baseline EE and EE with Affinity Bias outperform Individual Random Walk at small w values, but are overtaken at medium and large w values. This is because both variants of EE can quickly converge to the optimum of the local quality function, but they are unable to effectively consider the global quality function, and therefore the mean fitness decreases steadily as w increases.

EE with Affinity Bias performs the best and significantly outperforms Baseline EE at medium w in Scenario 3.1 (Figure 8.12(a)), where a balanced distribution of subgroups is encouraged. However, its advantage over Baseline EE diminishes in Scenario 3.2 and 3.3 (Figure 8.12(b,c)), as there is no mechanism to regulate the distribution of individuals between subgroups, and the produced distribution is often not optimal.

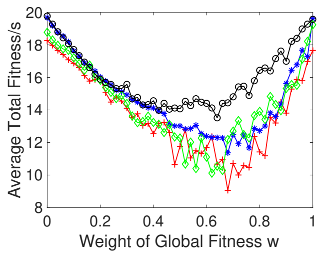
The proposed algorithm Bi-Objective EE has slightly worse performances than the other two EE variants in Scenario 3.1 (Figure 8.12(a)) at low w values, but has comparable or better performances in all other situations. This demonstrates that Bi-Objective EE algorithm is able to deliver good continuous performances compared to other considered



(a) Scenario 3.1



(b) Scenario 3.2



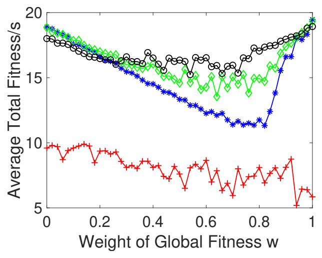
(c) Scenario 3.3

Figure 8.12: Average total fitness obtained per second by the population at all w values for all considered algorithms; red+:Individual BL, blue*:BL EE, green◇:EE Affinity, black○:Bi-Objective EE [SM21a]

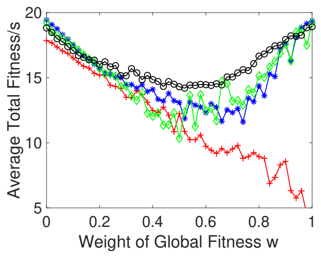
algorithms. However, its advantage over the other algorithms also diminishes as the desired distribution of tasks becomes more imbalanced. This shows that task allocation into imbalanced subgroups remains challenging for Bi-Objective EE.

Average Fitness Obtained with the Presence of Noise

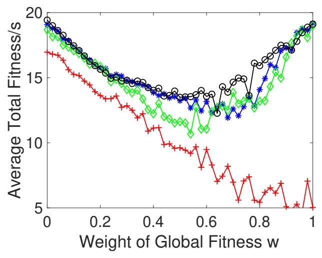
In the next step, a Gaussian noise is added with a standard deviation of 0.1 to the quality functions measured by the individual agents. The performances of considered algorithms are shown in Figure 8.13.



(a) Scenario 3.1



(b) Scenario 3.2



(c) Scenario 3.3

Figure 8.13: Average total fitness obtained per second by the population at all w values for all considered algorithms with stochasticity added; red+:Individual BL, blue*:BL EE, green◇:EE Affinity, black○:Bi-Objective EE [SM21a]

Individual Random Walk's performance is reduced more significantly by the noise, compared to all EE variants. This is due to the fact that the collective learning in EE algorithms mitigates the effects of noise, as individual agents can pool their observations to gain more effective genome selection in the evolutionary process.

Bi-Objective EE is slightly more susceptible to the effects of noise than the other two variants, as it relies more on the individual observations. However, its performances are still consistently equivalent or better than other considered algorithms, especially at w

values between 0.4 and 0.8.

8.4 Summary and Discussion

This chapter looks at a collective task allocation problem in a continuous decision space. The generalized problem setting allows for more complex quality distribution in the decision space. Bi-Objective EE is proposed as an approach to achieve dynamic decision space restriction in the embodied evolution process, and is shown to be able to handle multimodal quality distribution functions and produce well-performing specialized behaviors.

Taking an integrated look at the experimental results, the following observations can be made regarding the considered algorithms. Both variants of EE are effective at producing an optimal robot controller when only a local quality function is considered. Baseline EE is the best among the considered algorithms in such scenarios. It enables the individuals to quickly converge to the optimal genome regardless of the multi-modality of the considered quality function. Baseline EE also has full reproductive freedom among the agents, often producing better peak performances than algorithms with limited or no reproduction, such as EE with Affinity Bias and Individual Random Walk respectively. However, when a composite quality function is considered, the local component is given a much bigger emphasis than the global component. Furthermore, when a division of labor is encouraged by the interactions between global and local quality functions, Baseline EE still focuses on finding the singular optimum of the local quality function, often ultimately producing a total consensus rather than the desired division of labor and thus often has poor continuous performances.

EE with Affinity Bias is able to encourage a division of labor, and therefore, it sometimes has superior continuous behaviors than Baseline EE. However, it is still unable to effectively optimize the proportions of specialized behaviors according to the global quality function. Thus, it has an advantage over Baseline EE only when the desired distribution of tasks is roughly equal, but this advantage diminishes when the desired distribution becomes more imbalanced. At the same time, the static reproductive isolation limits the copying of genomes between the agents, causing EE with Affinity Bias to often have worse peak performance than Baseline EE.

To improve the performances of existing embodied evolutionary algorithms in a similar task allocation scenario, the agents should take into account both the local reward of their actions and the effectiveness of the cooperation between agents with different individual configurations. The embodied evolutionary algorithms implemented should also move beyond enforcing static reproductive isolation, such as the affinity bias used in this chapter, to generate divergent behaviors. The proposed Bi-Objective EE algorithm addresses these two points by using a decision-making mechanism on the preferred direction to change the genomes of individual agents, thus achieving dynamic reproductive isolation. This mechanism of optimization can better regulate the proportion of specialized behaviors than the static reproductive isolation achieved via the affinity bias. Based on the experimental results, Bi-Objective EE is consistently equal to or better than other algorithms in terms of performance. Its peak fitness obtained is close to the Pareto frontier computed by NSGA-II. Its continuous performance is only overshadowed by other variants of EE when a high weight is given to the local quality function as opposed to the global quality function. Compared to the other variants of EE, the proposed algorithm's mechanism of genome optimization relies more on the individual agents' local information and especially on the temporal changes of the agents' obtained fitness values. Thus, it is more resistant to the effects of sparse communication but less resistant to stochastic quality functions. Also, the creation of optimal specialized behaviors remains challenging when the desired distribution is imbalanced.

Chapter 9

Conclusion and Future Work

The content of this thesis addresses three main research objectives: The first is on how an intelligent swarm can efficiently come to a consensus in a best-of-n problem when $n > 2$. To this end, two novel consensus strategies have been proposed, both of which employ parallel encoding of option preferences during the decision-making process. Their performances are then compared with those of opinion-based strategies in the discrete collective estimation scenario. The second research objective arises out of the linearly increasing communication complexity with the number of options under parallel encoding of option preferences. It is thus worth investigating the scalability of their performances when facing many-option consensus-forming scenarios where the number of discrete options far exceeds the number of agents. Through the performances of the considered strategies in many-option scenarios, the impact of an expanding decision space on the collective decision-making process can also be observed. This leads to the last research objective, investigating how the performances of collective decision-making strategies can be improved by actively reducing the decision space size. This has been attempted in two independent scenarios, with many-option and continuous decision spaces, respectively.

9.1 Summary of Research Contributions

Much of the research into collective decision making in artificial swarm intelligence has been inspired by the self-organized decision-making behaviors in natural intelligent swarms. The characteristics of biological agents lead to the prevalent opinion-based consensus strategies in the study of best-of-2 problems. However, this design has poor scalability when facing best-of-n consensus problems with $n > 2$.

In order to deal with collective consensus scenarios with more than 2 options, this thesis presents 2 novel decision-making strategies that focus on exchanging the agents' preferences of the options via explicit parallel encoding. The first proposed approach, distributed Bayesian belief sharing (DBBS), seeks to compute the exact Bayesian likelihood of a given option being the best option and let agents perform belief fusion by transmitting the raw likelihood array. The second proposed approach, ranked voting (RV), encodes the agents' relative preferences of the potential options via their rankings. The agents' opinions are combined using ranked voting systems. The two proposed approaches are compared with the classical opinion-based strategies in the discrete collective estimation scenario. The performances of the considered consensus strategies are analyzed using a multi-criteria framework, with attention paid to the trade-offs between different performance metrics measured at different parameter settings. The proposed consensus strategies using parallel encodings of option preferences have been shown to outperform the opinion-based strategies, when considering convergence speed, accuracy, and reliability.

On the other hand, under the proposed parallel encoding consensus approaches, the amount of information needed to be sent between the agents increases linearly with the number of potential options. This limits their scalability on real hardware platforms when facing a large number of potential discrete options. To measure this effect, communication bandwidth-controlled experiments are conducted in both multi-option and many-option consensus scenarios. The communication bandwidth available and the number of potential options have been observed to have different effects on opinion-based and parallel consensus approaches.

Experiments in many-option discrete collective estimation scenarios have also demonstrated the importance of limiting the decision space size during the decision-making process. Thus, two independent collective decision-making scenarios have been looked at to investigate the ways decision space reduction can be done for different problems. In the investigated collective preference learning scenario, two decision-making strategies that encode the preference order in different ways are experimented and the benefits and drawbacks of limiting the discrete decision space size are discussed. In the continuous task allocation scenario, dynamic decision space reduction has been implemented using a collective decision mechanism to better fit a continuous multi-modal quality function.

9.2 Limitations and Future Research Directions

The main limitation of this thesis is the lack of experiments on real robot hardware. Much of the investigated decision-making scenarios are also restricted to collective perception and the related collective estimation problems, with other important collective decision-making problems unexplored. In addition, agents are assumed to move individually and randomly in the conducted experiments, causing the investigation of collective decision making to have weak integration with collective motion behaviors.

Besides addressing the aforementioned limitations, many open questions still need investigation in the area of multi-option collective decision making. A couple of examples are as follows:

In order to ensure the applicability of swarm decision-making approaches in real-world problems, mathematical modeling of the decision-making process is crucial to establish the generality of the methodology. Best-of-2 collective consensus processes can be modeled mathematically by formulating the ordinary differential equation describing the proportion of agents holding one of the options [Rei+24]. However, if the same approach is applied to a multi-option scenario, the opinion dynamics need to be modeled by a matrix differential equation, whose complexity increases with the number of discrete options available. Thus, novel encoding approaches for mathematical modeling of multi-option collective decision-making processes need to be developed for the design paradigm to be applicable as engineering solutions.

Another crucial area of future research to improve the applicability of collective intelligence approaches in real-world problems is on their performances in dynamic environments. Best-of-2 collective consensus problems have already been studied in dynamic environments with changing environmental features [Aus+22]. The same can be done with multi-option collective decision-making strategies. In addition, it is important to consider dynamic swarm composition during the collective decision-making process, where older agents can stop functioning, and new uninformed agents are progressive added to the swarm. This will enable a greater applicability of swarm intelligence techniques to complex real-world tasks.

Bibliography

[Ala+04] M. Alanyali et al. “Distributed Bayesian hypothesis testing in sensor networks”. In: *Proceedings of the 2004 American Control Conference*. Vol. 6. 2004, 5369–5374 vol.6. DOI: 10.23919/ACC.2004.1384706.

[AB22] Abdollah Amirkhani and Amir Hossein Barshooi. “Consensus in multi-agent systems: a review”. In: *Artificial Intelligence Review* 55.5 (2022), pp. 3897–3935. DOI: 10.1007/s10462-021-10097-x.

[Ant+09] Pavlos Antoniou et al. “Employing the flocking behavior of birds for controlling congestion in autonomous decentralized networks”. In: *2009 IEEE Congress on Evolutionary Computation*. 2009, pp. 1753–1761. DOI: 10.1109/CEC.2009.4983153.

[Aus+22] Till Aust et al. “The Hidden Benefits of Limited Communication and Slow Sensing in Collective Monitoring of Dynamic Environments”. In: *Swarm Intelligence. ANTS 2022. Lecture Notes in Computer Science*. Ed. by Marco Dorigo et al. Springer Cham, 2022, pp. 234–247. ISBN: 978-3-031-20176-9. DOI: 10.1007/978-3-031-20176-9_19.

[BM19] Palina Bartashevich and Sanaz Mostaghim. “Benchmarking Collective Perception: New Task Difficulty Metrics for Collective Decision-Making”. In: *Progress in Artificial Intelligence. EPIA 2019. Lecture Notes in Computer Science*. Ed. by Paulo Moura Oliveira, Paulo Novais, and Luís Paulo Reis. Springer Cham, 2019, pp. 699–711. ISBN: 978-3-030-30241-2. DOI: 10.1007/978-3-030-30241-2_58.

[BM21] Palina Bartashevich and Sanaz Mostaghim. “Multi-featured collective perception with evidence theory: tackling spatial correlations”. In: *Swarm Intelligence* 15.1-2 (2021), pp. 83–110. DOI: 10.1007/s11721-021-00192-8.

[BF01] Samuel N. Beshers and Jennifer H. Fewell. “Models of division of labor in social insects”. In: *Annual review of entomology* 46.1 (2001), pp. 413–440. DOI: 10.1146/annurev.ento.46.1.413.

[Bla48] Duncan Black. “On the Rationale of Group Decision-making”. In: *Journal of Political Economy* 56.1 (1948), pp. 23–34. DOI: 10.1086/256633.

[BTD96] Eric Bonabeau, Guy Theraulaz, and Jean-Louis Deneubourg. “Quantitative study of the fixed threshold model for the regulation of division of labour in insect societies”. In: *Proceedings of the Royal Society of London. Series B: Biological Sciences* 263.1376 (1996), pp. 1565–1569. DOI: `10.1098/rspb.1996.0229`.

[Bon+99] Eric Bonabeau et al. *Swarm intelligence: from natural to artificial systems*. 1. Oxford university press, 1999.

[BRM17] Thomas Bose, Andreagiovanni Reina, and James AR Marshall. “Collective decision-making”. In: *Current Opinion in Behavioral Sciences* 16 (2017), pp. 30–34. DOI: `10.1016/j.cobeha.2017.03.004`.

[Bra+13] Manuele Brambilla et al. “Swarm robotics: a review from the swarm engineering perspective”. In: *Swarm Intelligence* 7 (2013), pp. 1–41. DOI: `10.1007/s11721-012-0075-2`.

[BHP18] Nicolas Bredeche, Evert Haasdijk, and Abraham Prieto. “Embodied evolution in collective robotics: A review”. In: *Frontiers in Robotics and AI* 5 (2018), p. 12. DOI: `10.3389/frobt.2018.00012`.

[Bri+16] Markus Brill et al. “Pairwise Diffusion of Preference Rankings in Social Networks”. In: *International Joint Conference on Artificial Intelligence (IJCAI 2016)*. 2016, pp. 130–136.

[Cam+20] Scott Camazine et al. “Self-organization in biological systems”. In: *Self-Organization in Biological Systems*. Princeton university press, 2020.

[Cam+11] Alexandre Campo et al. “Self-organized discrimination of resources”. In: *PLoS One* 6.5 (2011), e19888. DOI: `10.1371/journal.pone.0019888`.

[Cen77] Yair Censor. “Pareto optimality in multiobjective problems”. In: *Applied Mathematics and Optimization* 4.1 (1977), pp. 41–59. DOI: `10.1007/BF01442131`.

[CMG20] Rui Chen, Bernd Meyer, and Julian Garcia. “A computational model of task allocation in social insects: ecology and interactions alone can drive specialisation”. In: *Swarm Intelligence* 14 (2020), pp. 143–170. DOI: `10.1007/s11721-020-00180-4`.

[CM09] Lars Chittka and Helene Muller. “Learning, specialization, efficiency and task allocation in social insects”. In: *Communicative & Integrative Biology* 2.2 (2009), pp. 151–154. DOI: `10.4161/cib.7600`.

[CL09] Larissa Conradt and Christian List. “Group decisions in humans and animals: a survey”. In: *Philosophical Transactions of the Royal Society B: Biological Sciences* 364.1518 (2009), pp. 719–742. DOI: `10.1098/rstb.2008.0276`.

[CR03] Larissa Conradt and Timothy J. Roper. “Group decision-making in animals”. In: *Nature* 421.6919 (2003), pp. 155–158. DOI: `10.1038/nature01294`.

[CB13] Chelsea N. Cook and Michael D. Breed. “Social context influences the initiation and threshold of thermoregulatory behaviour in honeybees”. In: *Animal Behaviour* 86.2 (2013), pp. 323–329. DOI: 10.1016/j.anbehav.2013.05.021.

[CL21] Michael Crosscombe and Jonathan Lawry. “Collective preference learning in the best-of-n problem”. In: *Swarm Intelligence* 15 (2021), pp. 145–170. DOI: 10.1007/s11721-021-00191-9.

[CLB19] Michael Crosscombe, Jonathan Lawry, and Palina Bartashevich. “Evidence Propagation and Consensus Formation in Noisy Environments”. In: *Scalable Uncertainty Management. SUM 2019. Lecture Notes in Computer Science*. Ed. by Nahla Ben Amor, Benjamin Quost, and Martin Theobald. Vol. 11940. Springer, Cham, 2019, pp. 310–323. ISBN: 978-3-030-35514-2. URL: https://doi.org/10.1007/978-3-030-35514-2%5C_23.

[Den+90] Jean-Louis Deneubourg et al. “The self-organizing exploratory pattern of the argentine ant”. In: *Journal of Insect Behavior* 3 (1990), pp. 159–168. DOI: 10.1007/BF01417909.

[Dia+06] Mary B. Dias et al. “Market-Based Multirobot Coordination: A Survey and Analysis”. In: *Proceedings of the IEEE* 94.7 (2006), pp. 1257–1270. DOI: 10.1109/JPROC.2006.876939.

[DTT21] Marco Dorigo, Guy Theraulaz, and Vito Trianni. “Swarm robotics: Past, present, and future [point of view]”. In: *Proceedings of the IEEE* 109.7 (2021), pp. 1152–1165. DOI: 10.1109/JPROC.2021.3072740.

[Dua+11] Ana Duarte et al. “An evolutionary perspective on self-organized division of labor in social insects”. In: *Annual review of ecology, evolution, and systematics* 42.1 (2011), pp. 91–110. DOI: 10.1146/annurev-ecolsys-102710-145017.

[Dua+12] Ana Duarte et al. “Evolution of self-organized division of labor in a response threshold model”. In: *Behavioral ecology and sociobiology* 66.6 (2012), pp. 947–957. DOI: 10.1007/s00265-012-1343-2.

[EGN18] Julia T. Ebert, Melvin Gauci, and Radhika Nagpal. “Multi-feature collective decision making in robot swarms”. In: *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*. 2018, pp. 1711–1719.

[Ebe+20] Julia T. Ebert et al. “Bayes Bots: Collective Bayesian Decision-Making in Decentralized Robot Swarms”. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. 2020, pp. 7186–7192. DOI: 10.1109/ICRA40945.2020.9196584.

[EB10] Nelson Elhage and Jacob Beal. “Laplacian-based consensus on spatial computers”. In: *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1-Volume 1*. 2010, pp. 907–914.

[Eme13] Peter Emerson. "The original Borda count and partial voting". In: *Social Choice and Welfare* 40.2 (2013), pp. 353–358. DOI: 10.1007/s00355-011-0603-9.

[Eml52] John T. Emlen. "Flocking Behavior in Birds". In: *The Auk* 69.2 (1952), pp. 160–170. DOI: 10.2307/4081266.

[Fer+12] Eliseo Ferrante et al. "Self-organized flocking with a mobile robot swarm: a novel motion control method". In: *Adaptive Behavior* 20.6 (2012), pp. 460–477. DOI: 10.1177/1059712312462248.

[Fra+12] Gianpiero Francesca et al. "Analysing an evolved robotic behaviour using a biological model of collegial decision making". In: *International Conference on Simulation of Adaptive Behavior*. Springer. 2012, pp. 381–390. DOI: 10.1007/978-3-642-33093-3_38.

[Fra+03] Nigel R. Franks et al. "Speed versus accuracy in collective decision making". In: *Proceedings of the Royal Society of London. Series B: Biological Sciences* 270.1532 (2003), pp. 2457–2463. DOI: 10.1098/rspb.2003.2527.

[Gar+13] Simon Garnier et al. "Do ants need to estimate the geometrical properties of trail bifurcations to find an efficient route? A swarm robotics test bed". In: *PLoS computational biology* 9.3 (2013), e1002903. DOI: 10.1371/journal.pcbi.1002903.

[Gor16] Deborah M. Gordon. "From division of labor to the collective behavior of social insects". In: *Behavioral Ecology and Sociobiology* 70 (2016), pp. 1101–1108. DOI: 10.1007/s00265-015-2045-3.

[GGT92] Deborah M. Gordon, Brian C. Goodwin, and Lynn E.H. Trainor. "A parallel distributed model of the behaviour of ant colonies". In: *Journal of Theoretical Biology* 156.3 (1992), pp. 293–307. DOI: 10.1016/S0022-5193(05)80677-0.

[Gos+89] Simon Goss et al. "Self-organized shortcuts in the Argentine ant". In: *Naturwissenschaften* 76.12 (1989), pp. 579–581.

[Has+13] Farzad Farnoud Hassanzadeh et al. "Building consensus via iterative voting". In: *2013 IEEE International Symposium on Information Theory*. 2013, pp. 1082–1086. DOI: 10.1109/ISIT.2013.6620393.

[Ibr+24] Rusul Ibrahim et al. "Review of Collective Decision Making in Swarm Robotics". In: *Journal of Al-Qadisiyah for Computer Science and Mathematics* 16.1 (2024), pp. 72–80. DOI: 10.29304/jqcsm.2024.16.11519.

[Jia16] Yichuan Jiang. "A Survey of Task Allocation and Load Balancing in Distributed Systems". In: *IEEE Transactions on Parallel and Distributed Systems* 27.2 (2016), pp. 585–599. DOI: 10.1109/TPDS.2015.2407900.

[Joh+02] R. A. Johnstone et al. "Information flow, opinion polling and collective intelligence in house–hunting social insects". In: *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences* 357.1427 (2002), pp. 1567–1583. DOI: 10.1098/rstb.2002.1066.

[KHE15] Alaa Khamis, Ahmed Hussein, and Ahmed Elmogy. "Multi-robot task allocation: A review of the state-of-the-art". In: *Cooperative Robots and Sensor Networks 2015* (2015), pp. 31–51. DOI: 10.1007/978-3-319-18299-5_2.

[LLW18a] Chanelle Lee, Jonathan Lawry, and Alan F.T. Winfield. "Combining Opinion Pooling and Evidential Updating for Multi-Agent Consensus". In: *Proceedings of the 27th International Joint Conference on Artificial Intelligence*. IJCAI'18. Stockholm, Sweden: AAAI Press, 2018, pp. 347–353. DOI: 10.5555/3304415.3304465.

[LLW18b] Chanelle Lee, Jonathan Lawry, and Alan F.T. Winfield. "Negative Updating Combined with Opinion Pooling in the Best-of-n Problem in Swarm Robotics". In: *Swarm Intelligence. ANTS 2018. Lecture Notes in Computer Science*. Ed. by Marco Dorigo et al. Vol. 11172. Springer Cham, 2018, pp. 97–108. DOI: 10.1007/978-3-030-00533-7_8.

[LLW21] Chanelle Lee, Jonathan Lawry, and Alan F.T. Winfield. "Negative updating applied to the best-of-n problem with noisy qualities". In: *Swarm Intelligence* 15.1-2 (2021), pp. 111–143. DOI: 10.1007/s11721-021-00188-4.

[LD19] Nicole Leitner and Anna Dornhaus. "Dynamic task allocation: how and why do social insect workers take on new tasks?" In: *Animal Behaviour* 158 (2019), pp. 47–63. DOI: 10.1016/j.anbehav.2019.09.021.

[Lin10] Shili Lin. "Rank aggregation methods". In: *Wiley Interdisciplinary Reviews: Computational Statistics* 2.5 (2010), pp. 555–570. DOI: 10.1002/wics.111.

[MJ11] Yan Meng and Yaochu Jin. *Bio-Inspired Self-Organizing Robotic Systems*. 255. Springer, 2011. DOI: 10.1007/978-3-642-20760-0.

[Mon+09] Francesco Mondada et al. "The e-puck, a Robot Designed for Education in Engineering". In: *Proceedings of the 9th Conference on Autonomous Robot Systems and Competitions* 1.1 (2009), pp. 59–65. URL: http://infoscience.epfl.ch/record/135236.

[MCB16] Jean-Marc Montanier, Simon Carrignon, and Nicolas Bredeche. "Behavioral specialization in embodied evolutionary robotics: Why so Difficult?" In: *Frontiers in Robotics and AI* 3 (2016), p. 38. DOI: 10.3389/frobt.2016.00038.

[Mon+11] Marco A. Montes de Oca et al. "Majority-rule opinion dynamics with differential latency: a mechanism for self-organized collective decision-making". In: *Swarm Intelligence* 5 (2011), pp. 305–327. DOI: 10.1007/s11721-011-0062-z.

[Nei13] Masatoshi Nei. *Mutation-Driven Evolution*. OUP Oxford, 2013.

[Olf06] R. Olfati-Saber. "Flocking for multi-agent dynamic systems: algorithms and theory". In: *IEEE Transactions on Automatic Control* 51.3 (2006), pp. 401–420. DOI: 10.1109/TAC.2005.864190.

[OM04] Reza Olfati-Saber and Richard M Murray. "Consensus problems in networks of agents with switching topology and time-delays". In: *IEEE Transactions on automatic control* 49.9 (2004), pp. 1520–1533. DOI: 10.1109/TAC.2004.834113.

[Olf+06] Reza Olfati-Saber et al. "Belief consensus and distributed hypothesis testing in sensor networks". In: *Networked Embedded Sensing and Control: Workshop NESC'05: University of Notre Dame, USA October 2005 Proceedings*. Springer. 2006, pp. 169–182. DOI: 10.1007/11533382_11.

[PGG96] Stephen W. Pacala, Deborah M. Gordon, and H.C.J. Godfray. "Effects of social group size on information transfer and task allocation". In: *Evolutionary Ecology* 10.2 (1996), pp. 127–165. DOI: 10.1007/BF01241782.

[PZ11] Chris A C Parker and Hong Zhang. "Biologically Inspired Collective Comparisons by Robotic Swarms". In: *Int. J. Rob. Res.* 30.5 (Apr. 2011), pp. 524–535. ISSN: 0278-3649. DOI: 10.1177/0278364910397621.

[PZ09] Chris A. C. Parker and Hong Zhang. "Cooperative Decision-Making in Decentralized Multiple-Robot Systems: The Best-of-N Problem". In: *IEEE/ASME Transactions on Mechatronics* 14.2 (2009), pp. 240–251. DOI: 10.1109/TMECH.2009.2014370.

[PS06] Kevin M. Passino and Thomas D. Seeley. "Modeling and analysis of nest-site selection by honeybee swarms: the speed and accuracy trade-off". In: *Behavioral Ecology and Sociobiology* 59 (2006), pp. 427–442. DOI: 10.1007/s00265-005-0067-y.

[PBD09] Abraham Prieto, Francisco Bellas, and Richard J. Duro. "Adaptively coordinating heterogeneous robot teams through asynchronous situated coevolution". In: *International Conference on Neural Information Processing*. Vol. 5864. Springer. 2009, pp. 75–82. DOI: 10.1007/978-3-642-10684-2_9.

[Pri+10] Abraham Prieto et al. "Open-ended evolution as a means to self-organize heterogeneous multi-robot systems in real time". In: *Robotics and Autonomous Systems* 58.12 (2010), pp. 1282–1291. DOI: 10.1016/j.robot.2010.08.004.

[Pri78] Ilya Prigogine. "Time, structure, and fluctuations". In: *Science* 201.4358 (1978), pp. 777–785. DOI: 10.1126/science.201.4358.777.

[RHR21] Mohsen Raoufi, Heiko Hamann, and Pawel Romanczuk. "Speed-vs-Accuracy Tradeoff in Collective Estimation: An Adaptive Exploration-Exploitation Case". In: *2021 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*. 2021, pp. 47–55. DOI: 10.1109/MRS50823.2021.9620695.

[RRH23] Mohsen Raoufi, Pawel Romanczuk, and Heiko Hamann. “Estimation of continuous environments by robot swarms: Correlated networks and decision-making”. In: *2023 IEEE International Conference on Robotics and Automation (ICRA)*. 2023, pp. 5486–5492. DOI: 10.1109/ICRA48891.2023.10161354.

[Rei+17] Andreagiovanni Reina et al. “Model of the best-of-N nest-site selection process in honeybees”. In: *Physical Review E* 95.5 (2017), p. 052411.

[Rei+24] Andreagiovanni Reina et al. “Speed-accuracy trade-offs in best-of-n collective decision making through heterogeneous mean-field modeling”. In: *Phys. Rev. E* 109 (5 May 2024), p. 054307. DOI: 10.1103/PhysRevE.109.054307.

[Rei+] Andreagiovanni Reina et al. “A Design Pattern for Decentralised Decision Making”. In: *PLOS ONE* 10 (10), e0140950. DOI: 10.1371/journal.pone.0140950.

[Rey87] Craig 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. DOI: 10.1145/37401.37406.

[SP13] Thomas L. Saaty and Kirti Peniwati. *Group decision making: drawing out and reconciling differences*. RWS publications, 2013.

[SKG16] Nicole Salomons, Gabriel Kapellmann-Zafra, and Roderich Groß. “Human management of a robotic swarm”. In: *Annual Conference Towards Autonomous Robotic Systems*. Vol. 9716. Springer. 2016, pp. 282–287. DOI: 10.1007/978-3-319-40379-3_29.

[San21] Jack Santucci. “Variants of ranked-choice voting from a strategic perspective”. In: *Politics and Governance* 9.2 (2021), pp. 344–353. DOI: 10.17645/pag.v9i2.3955.

[Sch+16] Alexander Scheidler et al. “The k -Unanimity Rule for Self-Organized Decision-Making in Swarms of Robots”. In: *IEEE Transactions on Cybernetics* 46.5 (2016), pp. 1175–1188. DOI: 10.1109/TCYB.2015.2429118.

[SC06] Thomas Schmickl and Karl Crailsheim. “Trophallaxis among swarm-robots: A biologically inspired strategy for swarm robotics”. In: *The First IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomechatronics, 2006. BioRob 2006.* 2006, pp. 377–382. DOI: 10.1109/BIOROB.2006.1639116.

[SC08] Thomas Schmickl and Karl Crailsheim. “Trophallaxis within a robotic swarm: bio-inspired communication among robots in a swarm”. In: *Autonomous Robots* 25 (2008), pp. 171–188. DOI: 10.1007/s10514-007-9073-4.

[SMC06] Thomas Schmickl, Christoph Möslinger, and Karl Crailsheim. “Collective perception in a robot swarm”. In: *International Workshop on Swarm Robotics*. Vol. 4433. Springer. 2006, pp. 144–157. DOI: 10.1007/978-3-540-71541-2_10.

[Sch+09a] Thomas Schmickl et al. “Get in touch: cooperative decision making based on robot-to-robot collisions”. In: *Autonomous Agents and Multi-Agent Systems* 18 (2009), pp. 133–155. DOI: 10.1007/s10458-008-9058-5.

[Sch+09b] Thomas Schmickl et al. “Two different approaches to a macroscopic model of a bio-inspired robotic swarm”. In: *Robotics and Autonomous Systems* 57.9 (2009), pp. 913–921. DOI: 10.1016/j.robot.2009.06.002.

[Sch44] Erwin Schrödinger. *What is life? The physical aspect of the living cell and mind*. Cambridge university press Cambridge, 1944.

[SB01] Thomas D. Seeley and Susannah C. Buhrman. “Nest-site selection in honey bees: how well do swarms implement the ”best-of-N” decision rule?” In: *Behavioral Ecology and Sociobiology* 49 (2001), pp. 416–427. DOI: 10.1007/s002650000299.

[SHM21] Qihao Shan, Alexander Heck, and Sanaz Mostaghim. “Discrete Collective Estimation in Swarm Robotics with Ranked Voting Systems”. In: *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*. 2021, pp. 1–8. DOI: 10.1109/SSCI50451.2021.9659868.

[SM20] Qihao Shan and Sanaz Mostaghim. “Collective Decision Making in Swarm Robotics with Distributed Bayesian Hypothesis Testing”. In: *Swarm Intelligence. ANTS 2020. Lecture Notes in Computer Science*. Ed. by Marco Dorigo et al. Vol. 12421. Cham: Springer International Publishing, 2020, pp. 55–67. ISBN: 978-3-030-60376-2. DOI: 10.1007/978-3-030-60376-2_5.

[SM21a] Qihao Shan and Sanaz Mostaghim. “Achieving task allocation in swarm intelligence with bi-objective embodied evolution”. In: *Swarm Intelligence* 15 (2021), pp. 287–310. DOI: 10.1007/s11721-021-00198-2.

[SM21b] Qihao Shan and Sanaz Mostaghim. “Discrete collective estimation in swarm robotics with distributed Bayesian belief sharing”. In: *Swarm Intelligence* 15 (2021), pp. 377–402. DOI: 10.1007/s11721-021-00201-w.

[SM22] Qihao Shan and Sanaz Mostaghim. “Benchmarking Performances of Collective Decision-Making Strategies with Respect to Communication Bandwidths in Discrete Collective Estimation”. In: *Swarm Intelligence. ANTS 2022. Lecture Notes in Computer Science*. Ed. by Marco Dorigo et al. Vol. 13491. Cham: Springer International Publishing, 2022, pp. 54–65. ISBN: 978-3-031-20176-9. DOI: 10.1007/978-3-031-20176-9_5.

[SM23] Qihao Shan and Sanaz Mostaghim. “Noise-resistant and scalable collective preference learning via ranked voting in swarm robotics”. In: *Swarm Intelligence* 17 (2023), pp. 5–26. DOI: 10.1007/s11721-022-00214-z.

[SM24] Qihao Shan and Sanaz Mostaghim. "Many-option Collective Decision Making: Discrete Collective Estimation in Large Decision Spaces". In: *Swarm Intelligence* 18 (2024), pp. 215–241. DOI: 10.1007/s11721-024-00239-6.

[SK98] Onn Shehory and Sarit Kraus. "Methods for task allocation via agent coalition formation". In: *Artificial intelligence* 101.1-2 (1998), pp. 165–200. DOI: 10.1016/S0004-3702(98)00045-9.

[SA14] Aaron Smith and Janna Anderson. "AI, Robotics, and the Future of Jobs". In: *Pew Research Center* 6 (2014).

[SCD18] Volker Strobel, Eduardo Castelló Ferrer, and Marco Dorigo. "Managing Byzantine Robots via Blockchain Technology in a Swarm Robotics Collective Decision Making Scenario". In: *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*. AAMAS '18. Stockholm, Sweden: International Foundation for Autonomous Agents and Multiagent Systems, 2018, pp. 541–549. DOI: 10.5555/3237383.3237464.

[SCD20] Volker Strobel, Eduardo Castelló Ferrer, and Marco Dorigo. "Blockchain technology secures robot swarms: A comparison of consensus protocols and their resilience to byzantine robots". In: *Frontiers in Robotics and AI* 7 (2020), p. 54. DOI: 10.3389/frobt.2020.00054.

[Tal+21] Mohamed S Talamali et al. "When less is more: Robot swarms adapt better to changes with constrained communication". In: *Science Robotics* 6.56 (2021). DOI: 10.1126/scirobotics.abf1416.

[Tal+19] Mohamed S. Talamali et al. "Improving collective decision accuracy via time-varying cross-inhibition". In: *2019 International Conference on Robotics and Automation (ICRA)*. 2019, pp. 9652–9659. DOI: 10.1109/ICRA.2019.8794284.

[TVD18] Marco Trabattoni, Gabriele Valentini, and Marco Dorigo. "Hybrid control of swarms for resource selection". In: *International Conference on Swarm Intelligence*. Vol. 11172. Springer. 2018, pp. 57–70. DOI: 10.1007/978-3-030-00533-7_5.

[TN04] Frederic Tripet and Peter Nonacs. "Foraging for work and age-based polyethism: the roles of age and previous experience on task choice in ants". In: *Ethology* 110.11 (2004), pp. 863–877. DOI: 10.1111/j.1439-0310.2004.01023.x.

[TP18] Pedro Trueba and Abraham Prieto. "Improving performance in distributed embodied evolution: Distributed Differential Embodied Evolution". In: *Artificial Life Conference Proceedings*. MIT Press. 2018, pp. 222–223. DOI: 10.1162/isal_a_00046.

[Tru+13] Pedro Trueba et al. "Specialization analysis of embodied evolution for robotic collective tasks". In: *Robotics and Autonomous Systems* 61.7 (2013), pp. 682–693. DOI: 10.1016/j.robot.2012.08.005.

[Tur+08] Ali E. Turgut et al. “Self-organized flocking in mobile robot swarms”. In: *Swarm Intelligence* 2 (2008), pp. 97–120. DOI: 10.1007/s11721-008-0016-2.

[VFD17] Gabriele Valentini, Eliseo Ferrante, and Marco Dorigo. “The Best-of-n Problem in Robot Swarms: Formalization, State of the Art, and Novel Perspectives”. In: *Frontiers in Robotics and AI* 4 (2017). DOI: 10.3389/frobt.2017.00009.

[VHD14] Gabriele Valentini, Heiko Hamann, and Marco Dorigo. “Self-Organized Collective Decision Making: The Weighted Voter Model”. In: *Proceedings of the 2014 International Conference on Autonomous Agents and Multi-Agent Systems*. AAMAS ’14. Paris, France: International Foundation for Autonomous Agents and Multiagent Systems, 2014, pp. 45–52. ISBN: 9781450327381. DOI: 10.5555/2615731.2615742.

[VHD15] Gabriele Valentini, Heiko Hamann, and Marco Dorigo. “Efficient Decision-Making in a Self-Organizing Robot Swarm: On the Speed Versus Accuracy Trade-Off”. In: *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*. AAMAS ’15. Istanbul, Turkey: International Foundation for Autonomous Agents and Multiagent Systems, 2015, pp. 1305–1314. ISBN: 9781450334136. DOI: 10.5555/2772879.2773319.

[Val+16a] Gabriele Valentini et al. “Collective Decision with 100 Kilobots: Speed versus Accuracy in Binary Discrimination Problems”. In: *Autonomous Agents and Multi-Agent Systems* 30.3 (May 2016), pp. 553–580. ISSN: 1387-2532. DOI: 10.1007/s10458-015-9323-3.

[Val+16b] Gabriele Valentini et al. “Collective Perception of Environmental Features in a Robot Swarm”. In: *Swarm Intelligence. ANTS 2016. Lecture Notes in Computer Science*. Ed. by Marco Dorigo et al. Vol. 9882. Springer, Cham, 2016, pp. 65–76. ISBN: 978-3-319-44427-7. DOI: 10.1007/978-3-319-44427-7_6.

[VA07] Lovekesh Vig and Julie A. Adams. “Coalition formation: From software agents to robots”. In: *Journal of Intelligent and Robotic Systems* 50.1 (2007), pp. 85–118. DOI: 10.1007/s10846-007-9150-0.

[Wes+13] Claudia Westhus et al. “Behavioural plasticity in the fanning response of bumblebee workers: impact of experience and rate of temperature change”. In: *Animal Behaviour* 85.1 (2013), pp. 27–34. DOI: 10.1016/j.anbehav.2012.10.003.

[Yan52] Chen Ning Yang. “The spontaneous magnetization of a two-dimensional Ising model”. In: *Physical Review* 85.5 (1952), p. 808. DOI: 10.1103/PhysRev.85.808.