Titre: Title:	Risk Awareness in Swarm Robotics
Auteur: Author:	Samuel Arseneault
Date:	2021
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Arseneault, S. (2021). Risk Awareness in Swarm Robotics [Mémoire de maîtrise, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/9971/

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Risk Awareness in Swarm Robotics

SAMUEL ARSENEAULT
Département de génie informatique et génie logiciel

Mémoire présenté en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
Génie informatique

Décembre 2021

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Ce mémoire intitulé :

Risk Awareness in Swarm Robotics

présenté par **Samuel ARSENEAULT**
en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
a été dûment accepté par le jury d'examen constitué de :

Michel DAGENAIS, président
Giovanni BELTRAME, membre et directeur de recherche
Giuliano ANTONIOL, membre et codirecteur de recherche
Soumaya CHERKAOUI, membre

DEDICATION

There is strength in numbers,
but organizing those numbers is one of the great challenges.
—John C. Mather

ACKNOWLEDGEMENTS

I was introduced to Swarm Intelligence research in 2019 when Professors Giovanni Beltrame, David St-Onge and Nicolas Reeves gave me the opportunity to work on the creation of an artistic and technological installation for Chambord Castle's 500^{th} anniversary. This sparked my interest for distributed systems and research, and I would like to thank them for their trust which later led me to pursuing a master's degree.

This work and my master's would not have been possible without the great help and guidance from Professor Giovanni Beltrame. Throughout the last two years, he taught me invaluable research and engineering skills. I greatly appreciate his patience and availability to answer all my questions.

I was lucky enough to work with David Vielfaure on many of my projects, and I thank him for his insights, collaboration and support.

I would also like to thank Jérôme Collin and Benjamin De Leneer for trusting me respectively as their laboratory teacher assistant and their lecturer. Through my passion for teaching, they allowed me to transmit my interest of computer engineering to future generations of engineers.

I must also thank Polytechnique Montréal, Groupe Deschênes and the Ministry of Higher Education for providing financial support which allowed me to pursue my master's degree.

Finally, I want to thank my friends and family for their support and encouragements throughout the last years.

RÉSUMÉ

Le risque est omniprésent dans nos vies; il est impossible de l'ignorer. Nous pouvons soit l'éviter ou l'affronter. Selon la situation, les deux approches peuvent avoir leur mérite. D'une part, en évitant toujours le danger, nous accomplirons forcément bien peu. D'autre part, en le bravant continuellement, nous courrons à notre perte. Nous choisissons donc le meilleur plan d'action selon la gravité de la situation et selon notre tolérance au risque. Or, le risque n'est pas uniquement présent dans notre quotidien, mais aussi dans des domaines d'applications scientifiques. Plus particulièrement, on s'intéresse ici à sa présence dans le domaine de la robotique en essaim, où des collectivités de robots doivent accomplir des missions dans des environnements souvent périlleux. Sachant ce contexte, comment peut-on inculquer à ces robots une politique de gestion de risque équilibrée afin de les rendre plus résilients?

La stratégie adoptée dans ce mémoire pour répondre à la question précédente est de créer une politique de stockage distribué tenant compte du risque ainsi qu'une manière collaborative d'explorer un environnement inconnu sans trop exposer les individus d'un essaim au danger. Ce faisant, on vérifie la pertinence de la gestion de risque dans deux contextes d'intelligence en essaim différents mais reliés. Le premier système développé dans ce mémoire, *Risk Aware Swarm Storage (RASS)*, se base sur le potentiel des robots à stocker de l'information sécuritairement ainsi que sur un mécanisme de percolation des données vers un point central. La seconde idée exposée est *Distributed Online Risk Aware Explorer (DORA)*, qui optimise des gradients locaux d'information et de risque pour explorer son environnement optimalement.

Afin d'évaluer chaque système conçu, deux phases sont nécessaires. D'abord, des expériences sont réalisées en simulation pour vérifier conceptuellement les systèmes ainsi que leur capacité d'être mis à l'échelle. Puis, des tests sont effectués en environnement physique (sur de vrais robots) pour s'assurer de leur applicabilité en scénarios réels. Pour RASS ainsi que pour DORA, les résultats montrent que tout comme avec les humains, une prise en considération modérée du risque selon la situation engendre des résultats optimaux.

RASS et DORA proposent des pistes pour rendre des applications de robotique en essaim plus résilientes. Conséquemment, ils pourront être adaptés et bonifiés pour des scénarios plus diversifiés et complexes que ceux présentés dans les expériences réalisées ici, tel que des scénarios de recherche et de sauvetage élaborés.

ABSTRACT

Risk is omnipresent in our lives; it is quite simply impossible to ignore it. We may either avoid or confront it. Depending on the situation, both approaches have their merit. On one hand, by always avoiding danger, one will accomplish very little. On the other hand, constantly defying it may lead to one's ruin. Humans choose the best course of action based on the gravity of the situation and on their personal risk tolerance. Yet risk is not only present in our everyday lives, but also in scientific application domains. This is the case of swarm robotics, in which groups of robots often accomplish missions in dangerous environments. So how would it be possible to instill a risk management policy to these robots to make them more resilient?

The strategy adopted in the thesis in order to answer the previous question is to create a distributed storage policy which takes risk into account, as well as to create a collaborative solution to explore an unknown environment without needlessly exposing robots to danger. This way, the relevance of risk awareness is verified in two different but related swarm robotics contexts, adding diversity and value to this work's contribution. The first system described in this thesis is *Risk Aware Swarm Storage*, which is based on robots' potential to safely store data. It also relies on gradient-based routing to achieve data percolation towards a central point. The second idea explored in this thesis is *Distributed Online Risk Aware Explorer*, which optimizes local information and risk gradients to explore an unknown environment.

Two evaluation phases are required for each developed system. First, simulations are performed to verify the conceptual sanity of the designs as well as their scaling capacity. Second, the systems are tested on physical robots to ensure they have real-world applicability. For both RASS and DORA, results show that just like with human behavior, reasonably factoring risk in decision processes often yields optimal results.

RASS and DORA offer ideas to make swarm robotics applications safer and more resilient. Therefore, by adapting and possibly improving them where needed, they could be used in diverse and complex situations which were not studied in this work, such as search and rescue scenarios or cavern exploration.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ACRONYMS

ASN	Autonomous System Number
B.A.T.M.A.N.	Better Approach to Mobile Ad-hoc Networking
CRDT	Conflict-Free Replicated Data Types
DBM	Distributed Belief Map
DHT	Distributed Hash Table
DORA	Distributed Online Risk Aware Explorer
FBE	Frontier-Based Exploration
IoT	Internet of Things
LRU	Least Recently Used
MDP	Markov Decision Process
RASS	Risk Aware Swarm Storage
WSN	Wireless Sensor Network

CHAPTER 1 INTRODUCTION

Swarm intelligence can be defined as a collective behavior in a group of agents aiming to achieve objectives. Swarm intelligence algorithms have seen diverse applications for a few decades [1], such as in the medical field [2, 3] and more recently in data mining [4]. On the other hand, swarm robotics, a subset of swarm intelligence applied to robots, has seen few real-world applications and has so far been mostly limited to laboratory research projects. However, research in this field is accelerating, and techniques and use cases making swarm robotics more useful are constantly appearing. Among the considerations brought forward by ongoing research is robustness and resilience, which are crucial if industrial applications are to be considered [5, 6]. This is where this work seeks to make an impact, by providing ideas which will help swarm robotics move forward from laboratory experiments to real-world applications by increasing swarm robustness.

It should be noted that both RASS and DORA, the systems presented in Ch. 3 and Ch. 4 respectively, are products of the research conducted for articles which are not yet published at the moment of writing this thesis. Because of intellectual property reasons, the presentation approaches used in each case differed. For RASS, the article was reused and slightly adapted to add details and clarity. In DORA's case, the article was summarized.

1.1 Definitions and Basic Concepts

In the following section, concepts central to this work and useful to its comprehension are briefly explained to provide some context.

1.1.1 Swarm Intelligence

Swarm intelligence takes inspiration from biology and behaviors observed in nature. Of notable interest in this field are groups of animals in which emergent behaviors can be observed. Bee, ant, and termite colonies are known to show self-organization and coordination properties. Many algorithms based (sometimes loosely) on those emergent behaviors have been created and are the foundation of swarm intelligence. Notable examples include Ant Colony Optimization [7], the Bees algorithm [8] and Artificial Bee Colony algorithm [9].

As is the case for insects, performance and robustness of these algorithms results from the strength of numbers present in large swarms of agents. An important property of swarm intelligence is that agents should not be aware of the existence of a larger system, so as to re-

quire little coordination. Consequently, a criterion devised as part of these algorithms is that all agent behaviors must be based on local interactions to avoid centralized control. Indeed, such an outcome could result in bottlenecks or single point of failures being introduced. Relying solely on local interactions often has the added benefit of making these solutions simple and elegant.

1.1.2 Swarm Robotics

Swarm robotics emerged from the domain of swarm intelligence, from which algorithms are adapted and applied to groups of robots. Not only do algorithms transpose from one field to the other, but so do more general key principles. In both fields, solutions must be as decentralized as possible (sometimes, a base station may be required for robots). Moreover, the need for local interaction implies that robots must be able to communicate with one another or to at least sense environmental information indicative of the others' work. Exchange of information, whatever the shape it assumes, is required.

The key difference with swarm intelligence is that the latter's algorithms have been extensively applied in many concrete applications, which is not yet the case of swarm robotics. Simple tasks like pattern formation have been studied extensively [10–12], but do not translate directly to real world applications. Also, because of the physical nature of robots, specific challenges and limitations arise: storage and bandwidth restrictions, mechanical failures, etc. all affect the designed systems.

1.1.3 Decentralized/Distributed

Having explained the distinction between swarm intelligence and swarm robotics, a clarification regarding the definition of a *distributed* system as opposed to a *decentralized* one is required.

Distributed systems imply multiple connected devices which interact to perform a common set of tasks. This interaction may occur through various means and be coordinated by one or more central devices. This would be the case for an Internet of Things (IoT) application in which edge nodes collect data and forward it to local servers where it can be processed.

Decentralized systems also involve many connected devices which share a common goal. However, they are not coordinated by a central authority; cooperation happens only between peers. This is the case for BitTorrent, which relies on a peer-to-peer architecture in which no centralized activity is involved [13].

A decentralized system is distributed by nature, but the opposite is not necessarily true.

1.1.4 Scalability

Scalability can be defined as the ability of a system to adapt to changing amounts of work. In the context of swarm robotics, these can fit in two categories. First, additional loads can result from a changing (increasing) number of agents involved in the system, which necessarily impacts its performance because of communication overhead, among other things. Second, a change in burden may happen because of evolving tasks. In any case, both must be considered when designing robust swarm applications.

1.1.5 Occupancy and Belief Maps

It is a common practice to split an environment in a grid composed of cells. Discretization helps in several ways. First, it allows for easier addressing of areas of interest within it, often making algorithms more intuitive. Second, it eliminates unnecessary computational costs by giving control over how fine grained the description of the environment should be. These cells, aside from being useful for positioning, can be used to store information about the state of the environment. They can have any number of dimensions, and usually fall in two main categories: occupancy and belief maps.

Occupancy maps use these cells to store binary values indicating whether something is present or not at a given location. That element could be an obstacle, a source of danger, a target and so on.

Belief maps are a generalization of occupancy maps: instead of storing binary values, cells contain a number representing a probability. It indicates the confidence level, or the belief, that the cell contains a given element. Of course, storing probabilities instead of binary values implies more memory usage, but the trade-off is generally worth it as it opens up possibilities for more designs.

1.2 Problem Elements

Problem elements can be thought of as requirements which need to be fulfilled by the current research project. These fall within two main categories: functional and non-functional requirements. The first are the perceivable characteristics of the system, those which could be demanded by a client. They are system-specific. In the current work, they are the need for good storage/routing performance for RASS, and a potent exploration strategy for DORA. The second type of requirement, non-functional ones, are less system-specific and can be applied to both designed items. They include low failure rates, local information usage, tolerance to limited resources, and scalability.

1.2.1 Efficient Information Storage and Transmission

Broadly, the problem which much be solved here is to allow an extensible and efficient storage. To do so, the decentralization assumption must be slightly relaxed by allowing agents to offload some data to a base station, otherwise the total storage capacity would be static and equal to the sum of the individual capacities. This consideration is realistic in the context of swarm robotics, where the robots often need to contact a base station for other reasons like recharging. To achieve this objective, three items are required.

1. The designed storage policy must allow data to be stored until it is ready to be moved. Consequently, collective swarm storage must be allocated with care to avoid unnecessary duplication which would reduce the storage and data acquisition capacities.

2. A simple decentralized routing policy is necessary, and it must not rely on the base station, because robots cannot be assumed to maintain a stable connection with it.

3. The routing, besides transmitting data between peers, must also result in data percolation towards the base station.

1.2.2 Efficient Exploration

Of course, a system which has for main objective to explore should do well at this task. Thus, DORA has to obtain performances at least as good as state-of-the-art systems of the same nature. This means that for the duration of the exploration experiments, DORA should cover more unknown terrain than the benchmark solutions. This includes dealing with terrains ridden with obstacles. Furthermore, the chosen strategy cannot rely on centralized planning and coordination, meaning exploration efficiency has to emerge from peers.

1.2.3 Avoiding Failures

Robustness has been studied in single-robot systems, where robots are designed to be as reliable as possible though component redundancy [14] or through verified control systems [15–17]. However, even with these precautions, robots can still fail. This is where the inherent resilience of robot swarms becomes interesting, by allowing some individuals to fail while still maintaining group collective functionality. It has been shown that multi-robot teams are usually resilient to a reasonable number of robot failures [18–20]. Even though resilience through numbers is good, swarm systems should aim to reduce individual failures as they can create performance and material losses. Even worse, failure in an individual may cause a propagation of failures through the group [5]. Resilience is a quality which goes further than robustness. Indeed, systems designed with that quality in mind should accommodate failures by allowing the swarm to perform its normal functions (albeit with a possible loss of performance).

1.2.4 Using Local Information

The challenge with this requirement is twofold. First, because it is necessary to select which information is relevant to transmit between peers. This ensures that no unnecessary communication burden is present. It also serves as a way to keep the solution simple, making it easier to integrate in other projects. Second, the means of transmission must be appropriately selected to fit the needs of the solution, that is is must be simple to use and efficient.

1.2.5 Limited Resources

Some recently developed robots, like the Kilobot [21], were created with the first category of scalability in mind. Indeed, their low unit price combined with their small form factor makes the simultaneous use of hundreds or thousands of them possible (hence their name). This capacity has been verified in tasks like self-assembly [22]. However, this comes with a trade-off: they are severely limited in their individual capacity, which makes them susceptible to increased workloads if the swarm size remains constant. The solution to this weakness would be to use more powerful robots, but this obviously incurs higher and possibly unreasonable financial costs. The dilemma is thus that robots are usually either small in size and capacity, or are limited in numbers. Therefore, it is essential to conceive algorithms adapted for swarms of resource-limited robots. These algorithms must be as simple and efficient as possible. For example, [23] show how to perform distributed on-line learning with limited bandwidth. Another added benefit of designing simple algorithms is that by reducing computational load,

they reduce energy consumption, which mitigates the problem of limited energy availability most robots face due to their use of batteries [24].

1.2.6 Verifying Scalability

Because scalability is such an important characteristic of swarm systems, it is important to devise a strategy to verify the designs proposed in this work include it properly. Therefore, this verification has to include two steps: an assertion of whether the solutions can work at all with an increasing number of robots, and an evaluation of how this increase in number helps the systems' performance in their respective functional goals. In this regard, the use of a physics-based simulator is absolutely necessary to ensure testing realism. However, the robot-simulation gap states that while simulated agents can represent the general behavior of robots, they cannot account for all variables which would be present in real-world testing, thus possibly affecting result validity and accuracy [25]. That might lessen the value of the conducted tests, particularly for swarms of robots [26]. A simulator reducing the impact of this problem is desirable. Testing on physical robots also needs to be conducted to further compound this issue.

1.3 Research Objectives

The broad objective pursued in this research is to introduce risk awareness into swarm robotics. To do so in a significant way, RASS and DORA need to collectively address all the problem elements previously presented. Some of the research objectives are common to both. For instance, they must reduce the failure rate comparatively to their related state-of-the-art applications. Moreover, they must show that this is achievable by relying solely on local communication information processing. Additionally, they must be conceived with both limited resources and scalability in mind. Some of the objectives are not shared. First, RASS needs to perform well in terms of storage efficiency as well as data transmission speeds. Second, DORA has to exhibit good exploration coverage.

For RASS, the specific objectives are the following:

1. Have a lower data corruption rate than competing algorithms;
2. Be entirely decentralized and rely on local information only;
3. Work well on resource-limited platforms;
4. Obtain a better storage efficiency in comparison to benchmark algorithms;

5. Achieve similar transmission speeds compared to benchmark algorithms.

For DORA, the specific objectives are slightly different:

1. Have a lower robot failure rate than competing algorithms;
2. Be entirely decentralized and rely on local information only;
3. Work well on resource-limited platforms;
4. Achieve a better or similar terrain exploration coverage than benchmark algorithms.

This thesis therefore seeks to elaborate on how both algorithms comply with the set objectives by detailing their implementation as well as by verifying that they respect the desired properties through thorough experimentation.

1.4 Thesis Outline

This thesis is structured as follows. First, In Ch. 2, a detailed review of relevant scientific advances is provided. It explores the following themes: distributed information sharing, distributed storage, risk assessment, routing, swarm programming, and path planning and exploration strategies. Second, in Ch. 3, RASS, a method of introducing risk awareness into swarm storage is presented. This section contains a detailed explanation of the system as well as experimental results derived from its implementation. Third, in Ch. 4, a second risk-aware swarm system is presented in the form of DORA Explorer, a distributed algorithm leveraging swarm collaboration to safely explore unknown environments. In this section, an architectural description is given and is followed by experimental results. Fourth, in Ch. 5, the work conducted in the context of this thesis is summarized. Finally, in Ch. 5.2, the limitations of this thesis' work are presented and followed by recommendations for future research.

CHAPTER 2 LITERATURE REVIEW

This chapter explores relevant state of the art literature related to the domain of this thesis. This includes insights into the following topics: swarm robotics, decentralized storage, communication and consensus, distributed storage, risk assessment in robot swarms, and path planning and exploration strategies. Not all articles explored in this review were directly used, but they certainly all provide good background information. The direct contribution of those which were used for either RASS or DORA is explained in detail in their respective chapters.

2.1 Swarm Robotics

There has been a growing interest for swarm robotics [24, 27], showing they have some potential. A good starting point to explore literature relevant to this work is therefore the overview of swarm robotics published by Dorigo et al. [6], which highlights the benefits and challenges related to the field. A key insight is that individual robots sometimes might not be able to perform tasks on their own, such as surveillance of large areas [28]. Another scenario is time-sensitive applications like search and rescue, in which the increased speed provided by parallel task execution might be crucial [29]. In these cases where multiple robots are required, coordination through a central source might be considered a simpler solution, but it has drawbacks and may be impossible to use at times. This is why decentralized swarm control is often considered, but it too has challenges, like storage, communication and consensus mechanisms [30]. The last problematic is out of the scope of this work, but the other two deserve attention.

2.2 Decentralized Storage

To support collaboration within a swarm, efficient communication is necessary [31]. To that end, relevant articles supporting distributed or decentralized ways of sharing data are presented here. One of the first accounts of decentralized data transmission in the context of swarm intelligence was proposed in Ant Colony Optimization [32], in which agents leave pheromone trails in their environment which can be interpreted by their peers. This mechanism is a form of stigmergy.

Using the same concept to store data, the virtual stigmergy [33] is a distributed key-value storage mechanism that allows a swarm of robots to efficiently share information at runtime,

and effectively constitutes a shared memory space. It was designed to work with robots with limited bandwidth, lossy communication mediums, and low processing capabilities. It ensures full data replication across all of its storage nodes (i.e. the robots), meaning it is particularly robust to failures. It also aims for eventual data consistency [1] to increase data availability and access speed. Another advantage of virtual stigmergy is its inherent integration with Buzz. SwarmMesh [35], another distributed key-value storage mechanism, introduces the idea of storing data on the less vulnerable members of the swarm to increase robustness. This is done by calculating keys by hashing the data needing to be stored and then partitioning the keys based on the fitness (connectivity, available memory, etc.) of a given robot to store information. Access to stored information is done through a custom routing algorithm based on the keys. The system also allows for data replication to further improve robustness. Its usefulness has been proven in situations where robot resources and capabilities are limited [36]. Both virtual stigmergy and SwarmMesh are scalable to large swarms of robots with dynamic connection topologies. However, their main limitation resides mainly in the small size of the data that can be stored in the swarm. This is an issue if larger quantities of data need to be shared. For example, storing pictures, videos or complex data collected during a mission with these systems might not be feasible. This is because they lack a mechanism to split data in chunks to reference and reconstruct stored files. SOUL [37] addresses this by adapting existing peer-to-peer file sharing mechanisms (which are already decentralized in nature and support large file sharing) to the context of robotic swarms. The key principle in SOUL is that files are "split into a series of smaller chunks of data, referred to as datagrams, spread throughout the swarm". This spreading is done through a bidding process among the members of the swarm to determine which storage configuration will minimize the cost of reconstructing data (i.e. retrieving it) from the distributed datagrams. It might be necessary to use distributed storage mechanisms which are optimized for certain metrics. In [38], a method for scenarios in which bandwidth is limited is suggested. In situations where the location of the storage nodes is important, geographic hash tables can be used [39–41]. For other constraints such as requirements for low energy usage, Cell Hash Routing [42] may be considered as a starting point. It is well suited to networks with dynamic topologies and varying densities. All the previously mentioned storage solutions are part of a wider category of storage designs called Conflict-Free Replicated Data Types (CRDT), which aim to provide fast data availability and eventual data consistency.

[1] Eventual consistency guarantees that if a given datum is not modified for a certain time, all accesses to it should eventually become consistent [34].

2.3 Routing

To transmit data efficiently between two devices which are not directly connected, a routing strategy is necessary. That type of situation occurs in swarm robotics applications, where robots are often spread over a terrain and do not form a fully connected topology. Finding the best path between two points can be done with Dijkstra's algorithm [43]. However, such a method requires knowing the network topology in advance, and is quite computationally expensive, in the order of $\Theta((|V| + |E|) \log |V|)$ if V vertices and E edges are involved in the network. It is a prohibitive cost, especially if it needs to be computed periodically in dynamic topologies. Similarly, voting-based routing algorithms cannot be considered, as the voting process might introduce significant delays.

For these reasons, it is worth considering algorithms designed specifically for swarm robotics applications. By far one of the most prevalent approaches is to route data following a gradient based on one metric [44, 45]. The most used metric, perhaps because of its simplicity and efficiency, is the hop count between a source node and a target node. It is used notably in [45–48] for routing in ad hoc networks and in Wireless Sensor Network (WSN)s. Designs based on classic swarm intelligence algorithms can be used to find optimal routes, such as [49–53], but are more effective in static topologies, which makes their applications somewhat limited. Methods inspired by epidemics were shown to be efficient for disseminating data [54, 55]. For data aggregation purposes, which is an application similar to the percolation RASS seeks to achieve, the techniques in [51, 52, 56] were all shown to provide good results and could thus serve as inspiration.

2.4 Risk Assessment

Many methods to detect failures have been proposed. Endogenous (from internal origin) fault detection aims for robots to detect errors that occur to themselves. These include the like of malfunction of a particular sensor or a disconnection from the *ad hoc* network to which they are connected. Methods to this effect include those proposed by [57], which uses Kalman Filters [58] to detect outliers in sensor behavior, as well as others based on Bayesian networks [59] or particle filters [60]. Another algorithm inspired by fireflies allows exogenous fault detection, that is sensing failures in other members of the swarm [61]. A more recent exogenous (from external origin) anomaly detection solution was proposed in [62] and takes inspiration from the immune system. Taken separately or even in a possible combination, these fault detection have the potential to improve swarm resilience to failures. However, reaching a state where robots malfunction or crash is simply undesirable.

While some failures might be purely stochastic and unpredictable, some arise due to unnecessary exposure to risk. Therefore, of particular importance to the research objective of minimizing failures in the designed systems is the consideration of risk, which is directly correlated to breakdowns. Thus, strategies on how to assess the presence of danger and on how to share this information among the members of the swarm are necessary for RASS and for DORA. The definitions and categorizations of risk suggested in [63] are good starting points for creating risk assertions. These categories are locale-dependent, action-dependent and traverse-dependent. They respectively denote risk that does not depend on previous history (states and actions of the robots), risk that depends on recent history and risk which is linked to the whole history. An example of locale-dependent risk is given in [64], in which local terrain elevation may pose a threat to aircrafts if they fly too close. Traverse-dependent risk would be involved in situations where a cumulative effect is observed, like in prolonged exposition to radiation. These definitions can be useful to model the effects of long term exposure to danger. However, they definitions were created and used in the context of a formal path planner which is too resource-demanding in this current context. Also, this planner assumes prior knowledge of the environment's state, meaning that risk is not discovered by the robots themselves, but rather by a central operator. In SPIDER [65], agents are tasked with chain formation in dangerous environments. They adopt shy or bold behaviors, which allows them to adapt to varying levels of risk. This behavioral adaptability allows them to be resilient to significant failures and member loss. It is also interesting to note that many risk types can be combined through weighted sums to account for diverse scenarios [66]. Some association may also be made between risk and reward, as some exposure to reasonable amounts of the former may result in significant gains. For this reason, path planners described in [67, 68] allocate a "risk budget" to their agents. In other words, a balance between risk and reward can be used to guide robots through optimization, given that the risk tolerance of the system is properly defined.

2.5 Exploration Strategies

One possible strategy for exploration is based on ant colony stigmergy, in which virtual pheromones are used to dissuade robots from going to certain locations [69]. This mechanism can be used to guide robots away from each other (also know as dispersal) or away from uninteresting areas and therefore to steer exploration. However, the authors report its results are limited in dense topology scenarios.

The second possible strategy for terrain coverage is path planning. The planners suggested by [63, 70, 71] are based on a Markov Decision Process (MDP) and are risk-oriented. Their

shortcoming in relation to this work's objective are twofold. First, they assume at least a partial knowledge of the global state of the environment is available. This is not a valid assumption when exploring unknown environments, which is precisely what is defined in DORA's main functional requirement. Second, they are only applied to single-robot systems, and therefore fall short of the problem elements this works seeks to resolve.

Addressing the last point, several terrain coverage maximization strategies which are distributed in nature have been proposed. Many distributed exploration strategies that maximize the amount of covered terrain have been proposed. Voronoi-based coverage control techniques [72, 73] achieve interesting results, but are more useful when prior knowledge of the environment exists. The same issue applies to methods using time-varying domains (i.e. dynamic environments) [73, 74]. A perhaps simpler form of exploration which offers optimal terrain coverage (if not total, in the absence of failures), and consequently more suited to DORA's requirements, is FBE [75]. In this method, robots build a common knowledge of the furthest state of exploration their environment, and seek to widen this frontier at every time step, eventually resulting in full environment coverage. No prior knowledge is required by the agents. Several FBE refinements have been developed: Particle Swarm Optimization [76] and Wavefront Frontier Detector [77] are two of them. However, none of FBE-based strategies take risk into account. Because of their good performance in terms of terrain coverage, they could be used as benchmark solutions.

The system which is closest to DORA's objectives is the multi-robot control algorithm presented in [78, 79], because it maximizes the information gain during exploration in the presence of unknown hazards. Yet, it is not perfectly aligned with all non-functional requirements because it seeks an optimal exploration solution, which entails a computational complexity greater than $O(|V|2^{|V|})$ for every time step, where V is the set of robots. Using it would require introducing several approximations to lower the computation load of the robots.

2.6 Swarm Programming and Simulation

Swarm programming focuses on four key properties: decentralized control, absence of leaders, absence of predefined roles, and reliance on simple and local interactions. In this sense, it differs greatly from conventional centralized cloud-based applications or even from centralized sensor arrays which are common in IoT applications. Many programming paradigms for robot swarms have their shortcomings to address the issue of coordinating swarms of robots. This is the case for robot oriented programming, which is embodied by ROS, [80], because it focuses on single-robot scenarios. Also, aggregate programming like Protelis [81] suffers from offering little control over individual robot behavior. Granted, swarm robotics should avoid

behavior specialization, but sometimes it might be necessary, for example when hierarchies are needed or when location-specific tasks must be completed. For task-oriented programming frameworks such as Voltron [82], the issue is that they allow little control over low-level operations which might be required when operating robots.

Buzz [83], another programming language, addresses these issues and aims to meet the key properties mentioned earlier, therefore enabling an easier approach to swarm programming. Among other things, Buzz uses a custom virtual machine to run code in isolation of the operating system. It is designed to be an extensible language with high-level primitives facilitating robot interactions. A recently developed language, Koord [84], could also inspire software testing mechanisms, as it is built around a strong formal method philosophy. This means it allows swarms of robots to be programmed with verifiability in mind, that is each part of the distributed algorithms should be tested in a modular way with fixed guarantees in place.

Because validating various aspects of the proposed systems requires simulations, it is necessary to use a simulator well suited to scenarios in which multiple agents are present. It is also necessary to mitigate the previously mentioned robot-simulation gap. In this regard, ARGoS [85] is particularly well suited, because it has the capacity to run agents on separate threads, therefore improving simulation execution speed. This has the effect of making large scale simulations accessible. Moreover, because ARGoS is a physics-based simulator, it allows to take into account the effects of physical interactions between robots, their environment and themselves. Furthermore, ARGoS integrates particularly well with Buzz, with the possibility of directly using compiled Buzz scripts.

2.7 Grid Mapping

In the task of map building which is required for (or is the purpose of) robotic exploration [86, 87], belief maps offer a simple and powerful tool. They show significant improvements for exploration performance over occupancy maps because of their increased precision which allows for more informed decisions [88]. Their use in robotic exploration is far from new [89, 90]. In these, robots share sensor measurements and then build and update the belief maps independently. The disadvantages of the three previous methods are that they are limited to fixed grid sizes and are tested only with two robots, which make their application limited. Recently, in [91], improvements for multi-robot grid mapping have been suggested. In that article, the robots consider both the current and expected beliefs to collaborate. However, there seems to be no usage of them for storing risk beliefs. These maps are well suited to be stored in CRDTs, because cell locations can be used as keys and beliefs as values.

CHAPTER 3 RISK-AWARE SWARM STORAGE

In robotics, data acquisition often plays a key part in unknown environment exploration. For example, storing information about the topography of the explored terrain or the natural dangers in the environment can inform the decision-making process of the robots. Therefore, it is crucial to store these data safely and to make it available quickly to the operators of the robotic system. In a decentralized system like a swarm of robots, this entails several challenges. To address them, we propose RASS, a decentralized risk-aware swarm storage and routing mechanism, which relies exclusively on local information sharing between neighbours to establish storage and routing fitness. We test our system through thorough experiments in a physics-based simulator and test its real-world applicability with physical experiments. We obtain convincing reliability, routing speeds, and swarm storage capacity results.

3.1 Introduction

Using multi-robot systems for the exploration of unknown environments is very appealing. Indeed, if robots do not overlap in their exploration task, the amount of terrain covered increases proportionally with the number of robots in the system [92]. This can, for example, be particularly useful for search and rescue scenarios [93] where the speed at which the environment is covered is of critical importance. However, as the number of robots increases so does the amount of data collected, which puts pressure on the data storage infrastructure. Unfortunately, multi-robot systems usually suffer from unreliable connectivity [38], and directly sending the information to an external storage system (e.g., the cloud) may not always be feasible. An alternative to overcome this issue is to use the swarm of robots as means of distributed data storage. However, because the robots composing the swarm often have very limited memory and storage capacities, saving large amounts of data across the swarm can prove challenging. Furthermore, the robots in a swarm are not necessarily reliable [20]: even in controlled environments, they are usually meant to be easily replaced. This issue is aggravated when they must face situations that decrease their reliability, such as exposure to dangerous environments like search and rescue scenarios, forest fire reckoning, or nuclear inspection and cleanup. Therefore, giving the robots a way to eventually relay the information they acquired during their mission to a control station for more permanent storage is a definite advantage: it not only allows the information to be stored in a safe location accessible by human operators, it also alleviates the memory usage of the robots and enables continuous operation. Nonetheless, because in practice robots have a finite communication

range, the information collected at the periphery of the swarm usually needs to be routed through other robots before reaching a base station. Although forwarding the data through the shortest path towards the base station enables prompt retrieval [31], in real-world scenarios, this approach can be problematic for two reasons: first, such a path might not exist at all, for example if the robots are in an area completely cut off from external communication (e.g. a cave); second, if that path does exist, it might be too dangerous to use. For example, a robot trying to relay information by going through a highly radioactive area might cause a physical failure or data corruption. Furthermore, the base station cannot request the data, as communication with the robots is never guaranteed. Instead, data must be autonomously forwarded to the base station by the robots. For the same reason, and because of the overhead it entails, it is not practical to use leaders to coordinate the storage and routing processes. In addition, the system must take into account that the robots are constantly moving and acquiring new data, meaning that storage and routing conditions are dynamic. These constraints motivate the need for a completely decentralized, risk-aware swarm storage system that can safely and efficiently store data and route it towards a base station in a percolating fashion whenever possible. In this paper, we make the following contributions to multi-robot storage:

- A distributed, dynamic and risk-aware storage system
- A fully decentralized risk-aware routing algorithm

We name the combination of these two contributions RASS. The rest of the article is structured as follows: in section 3.2, we present related work and useful concepts upon which we build RASS; in section 3.3 we present our system model; in section 3.4 and 3.5 we detail our simulations and physical experiments along with their respective results; finally in section 3.6 we draw some conclusions.

3.2 Related Work and Background

3.2.1 Distributed Storage

A Distributed Hash Table (DHT) is a data structures specifically designed to partition data across a network of storage nodes in a key-value format. These structures come with a set of issues susceptible of hindering their performance, consistency and partial connectivity being some of them [38]. The virtual stigmergy presented in [33] tackles the problem using CRDTs and allowing decentralized robust information sharing for multi-robot systems. This virtual

stigmergy is implemented directly into the Buzz programming language [83]. However, in the virtual stigmergy, each node stores a full copy of the data (in an eventually consistent way). This means it has full redundancy and no partitioning, thus offering good reliability but poor storage capacity optimization.

A memory-efficient storage mechanism designed for multi-robot systems was proposed in SwarmMesh [35]. It uses the available memory of an agent and its connectivity with its peers to generate unique identifiers, which are the basis of its partitioning scheme and which represent an agent's fitness to store data. Storage of larger items (such as binary files) in DHTs has been studied in [37], where an auction process is used to partition data. Other partitioning systems have been proposed, such as Locality-aware Distributed Hash Tables [39] which posits that routing data through nodes that are close in a network will reduce latency. Therefore, they use information from the nodes' Autonomous System Number (ASN)s to partition data. Geographic Hash Tables [40], Cell Hash Routing [42] and [41] all use position-based information in their partitioning systems, and the second specifically addresses DHT implementation for ad-hoc networks of resource-constrained nodes.

These systems show various metrics that can be used to devise a partitioning system for a DHT. However, none of them takes into account the effects of environmental dangers on the reliability of the distributed storage system. RASS, our decentralized risk-aware storage and routing mechanism, aims to address this issue by including a risk measurement in its partitioning scheme.

3.2.2 Risk Assessment

Taking risk into account when designing autonomous systems is of high importance as excessive exposure to risk can lead to system failures. In an unknown environment, risks are usually tied to a certain location. Information about these locations could therefore provide increased situational awareness for robots to effectively perform a given mission. Belief maps and occupancy maps provide this situational awareness by assigning values to cells of a discretized environment. Whereas occupancy maps define the presence or absence of a feature (e.g. fire), belief maps assign probabilities to the cells of the discretized environment and usually offer better performance [88]. Belief maps have been extensively used in the past for robotic exploration [89–91].

A risk-aware exploration algorithm leveraging a belief map of the risk associated with the environment has been recently proposed in [94]. In this work, robots combine efforts in building a shared belief map of the environmental risks and use it to spatially avoid the dangerous pitfalls of the environment. The belief map is used to provide risk awareness

to an exploration algorithm which ultimately results in fewer robot failures. In RASS, we seek to use similar shared risk awareness to establish a reliable distributed data storage and routing mechanism, while improving its memory consumption by relying on more localized information.

3.2.3 Routing

There exist many types of routing algorithms for wireless sensor networks. For example, packets can be explicitly routed towards a known destination by forwarding messages towards a neighbour which optimizes a given metric. One simple approach is to send messages through the shortest path in terms of Euclidean distance. However, this assumes a complete knowledge of the nodes' Cartesian coordinates, which is far from realistic in swarm robotics. Indeed, robots might be operating into GPS-denied environments, or they might not even be equipped with GPS at all. A comparative study of metrics for multi-hop wireless networks shows that hop-count performs the best in scenarios where nodes are mobile [44] because it is robust in dynamic topologies [45], which is the case for RASS's nodes. Notable implementations using this metric are given in [46–48], which are respectively greedy, power-adaptive, and grid-based routing algorithms. Routing algorithms have also been inspired by biology: some are based on slime molds [49, 50] and others on ant colonies [51, 52]. However, the former category has applications mostly in static topology networks, which makes it ill-suited to our needs. The latter category specifically addresses data aggregation, which is particularly relevant to our objective of percolating data towards a base station.

3.3 System Model

We consider a fully decentralized multi-robot system tasked with the exploration of a dangerous environment. The multi-robot system is denoted as the collection of agents $a_i \in A$. We assume the robots to have limited storage and communication capabilities. We also assume the presence of an operator or base station who is interested in collecting the data generated by the swarm. Note that this last assumption does not mean that our system is centralized: it is reasonable to assume that the autonomous system has to produce some value (i.e. data) for the humans deploying it, and the base station does not influence the decision-making process of the swarm, but rather act as an information sink.

We consider the amount of data needed to be stored to be greater than the individual storage capabilities of the robots. It is therefore impossible for the individual agents to store a complete copy of the system's data, but the data can be fully stored by the base station.

Also, we consider that the robots are only able to communicate if they are within a certain communication radius R. As a result, if a message needs to be sent to a distant location (e.g. to the base station), it might need to be routed through multiple nodes before reaching the desired destination.

3.3.1 Risk Modelling

Radiation is known to cause performance loss and failures in robots [95, 96]. We therefore adapt the risk modelling from [94], which is based on a set of independent point radiation sources S with individual intensity $I_j \sim \mathcal{U}(0, 1)$. Note that we use radiation as a *model* of risk, but our method could be applied to other sources of danger: vertical air currents, high temperature areas, etc.

Risk is assumed to be dynamic, meaning that the position of the point radiation sources $\boldsymbol{s}_j(t) \in E$ can vary across time. For example, radiation can spread to new areas if radioactive particles are transported by wind. It is important to account for the dynamic nature of risk as it gives the system the capability to adapt to changing environments. We achieve adaptability by having each agent sense the radiation at every time step at its current position, which creates a step-by-step estimate of the risk at each robot's location, as shown in Fig. 3.1.

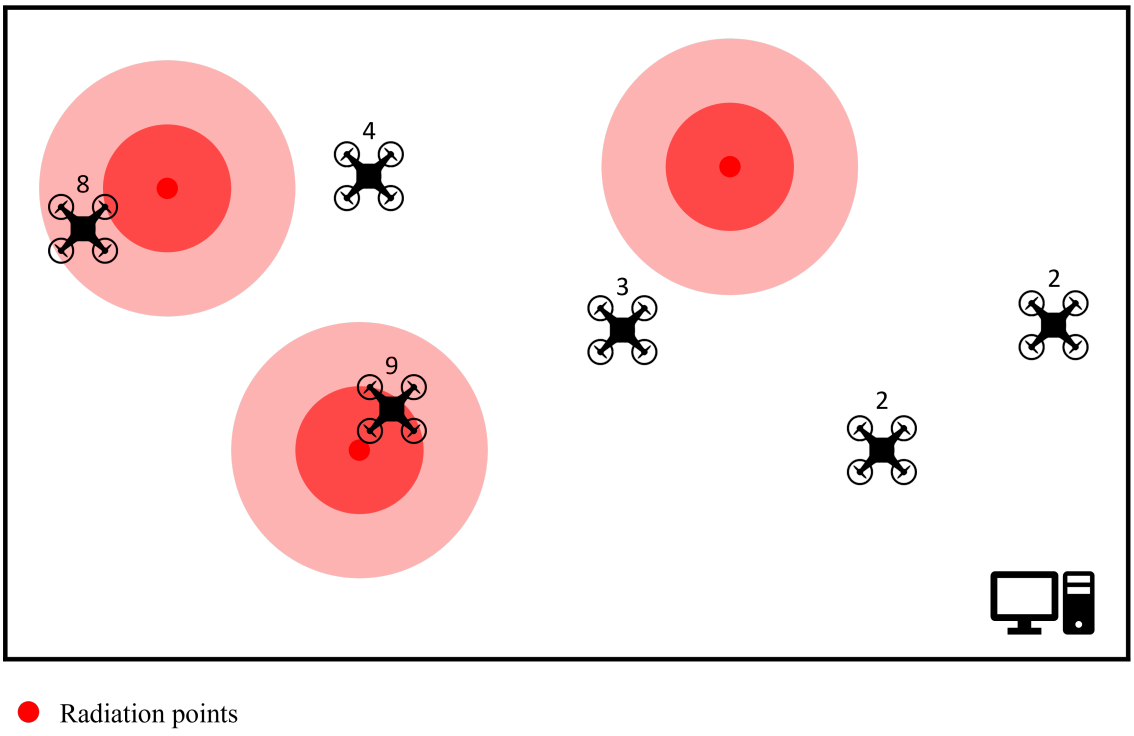

Figure 3.1 An example of risk values measured and held by robots at a given time step.

The perceived intensity decays exponentially (with λ as a decay parameter) as the Euclidean distance $\rho(\boldsymbol{x}_i)$ between $\boldsymbol{s}_j$ and $\boldsymbol{x}_i$ increases. The total perceived radiation level by a robot a_i at position $\boldsymbol{x}_i \in E$ is given by:

$$r(\boldsymbol{x}_i) = b + \sum_{\boldsymbol{s_j} \in S} \frac{I_{\boldsymbol{s_j}}}{1 + \lambda \rho(\boldsymbol{x}_i)^2} \tag{3.1}$$

and is measured by an on-board sensor with Gaussian measurement noise $b \sim \mathcal{N}(0, 0.05)$. We posit the radiation's effect on the system is to cause data corruption (which is one of its possible effects [96]). Let the probability of the event of a datum d getting corrupted while stored on robot a_i due to an individual radiation source $\boldsymbol{s}_j$ be $\mathbb{P}(c_i = 1|\boldsymbol{s}_j) \sim \mathcal{B}(r_{\boldsymbol{s}_j}(\boldsymbol{x}_i))$, which follows a Bernoulli distribution. Because we assume that the sources of radiation affect the robots (and thus the stored data) independently, the probability of a datum being corrupted due to the combined effect of radiation sources also follows a Bernoulli distribution given by:

$$\mathbb{P}(c_i = 1|S) \sim \mathcal{B}(r(\boldsymbol{x}_i)) = 1 - \prod_{\boldsymbol{s_j} \in S} 1 - \mathbb{P}(c_i = 1|\boldsymbol{s}_j) \tag{3.2}$$

3.3.2 Distributed Risk-Aware Storage

Our risk-aware storage system is built upon three assumptions:

1. Because nodes exposed to a higher level of risk also have a higher failure probability, they should be used less, thus maximizing overall storage reliability
2. Efficiently moving data away from the periphery of the swarm and towards the base station will increase the storage capacity of the system and the persistence potential of the data, because the base station usually has more storage and reliability than the swarm
3. Percolating data from edge nodes to the base station should be done by choosing routes devoid of risk to minimize data loss

From these assumptions, we derive RASS' two principal mechanisms: the routing table and the fitness-based percolation. We combine them to obtain a high-level algorithm described in Alg. 1, in which $\mathcal{N}$ is the set of neighbours of a given agent. Because of the distributed nature of the algorithm and because all robots execute the same code (except the base station), we forego the indices notation to simplify the algorithms.

Algorithm 1: RASS Execution Loop

```
while True do
    update_routing_table()
    update_fitness()

    if not is_fit() and |N| > 0 then
        evict_data()
    end

    store_measurements()
end
```

Routing Table

We assume that the swarm can be represented by a *connected* graph $\mathcal{G}$ with nodes $\mathcal{A}$ and edges $\mathcal{L}$ respectively representing agents and their wireless communication links. In practice, this connectivity cannot be maintained at all times, because of the quality of the links in $\mathcal{L}$ and because of the movements of nodes $\mathcal{A}$. However, this assumption can be mostly maintained through mechanisms for connectivity maintenance [97] if necessary. Furthermore, the need for this assumption can be relaxed because RASS does not need a constant link with the base station; it can opportunistically move data towards it when possible, and store them locally in the meantime.

Building a routing table with the aim of prioritizing higher capacity nodes is similar to the Gateway Optimization [98] implemented in the Better Approach to Mobile Ad-hoc Networking (B.A.T.M.A.N.) mesh networking protocol [99], where in our case the base station acts as a gateway. Because peer communication is not the focus of this work, the routing table held by each node only contains the minimal hop count between its neighbours and the base station and can therefore be implemented with a hash table. Therefore, given the existence of at least one (multi-hop) path between any node and the base station, we can establish a routing table based on hop count. The process to construct and periodically update this table involves exchanging messages between the base station and the nodes as suggested by [100], as we implement in Alg. 2. The general idea is that the base station emits a 0 hop message periodically, and whenever a node receives such a message (either from the base station or from another neighbour), it re-emits the message with an incremented count. Note that a node will only emit the message based on the smallest hop count it received; these counts are stored in the previously mentioned hash table. This percolation process from the station to the nodes creates a hop count gradient. An example of such gradient can be found in Fig. 3.2. Assuming a message can only be forwarded to an immediate neighbour

within a given time step, the required time to build the routing table in a connected graph is bounded by $\Omega(1)$ and $\mathcal{O}(|A|)$ as it takes at most $|A|$ steps to send a message from the base station to the furthest robot in a pathological topology (a line) and 1 step if the network is fully connected. However, in more realistic scenarios such as tree networks or scale-free topologies, building the table can be expected to take on average $\mathcal{O}(log|A|)$ steps because of the network's depth. We implement message forwarding through a gossip algorithm, which allows local broadcasting between robots.

Algorithm 2: Building/Updating the Routing Table

$routing\_table \longleftarrow listen\_neighbor\_hop\_count()$

if $id = 0$ **then**
  $min\_hops \longleftarrow 0$
else
  $min\_hops \longleftarrow min(routing\_table)$
end

$broadcast(min\_hops + 1)$

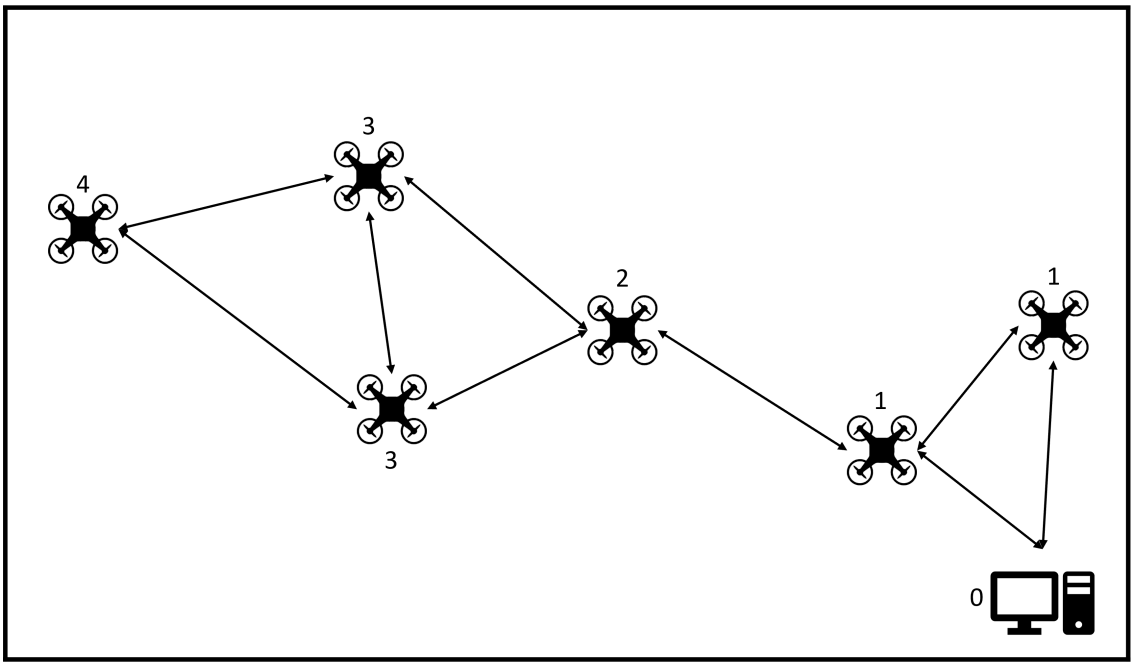

Figure 3.2 An example of minimal hop count towards the base station (bottom right) values held by robots after building the routing table.

Potential-Based Percolation

Our risk-aware storage system draws inspiration from SwarmMesh [35], in that each node periodically assigns itself a potential ϕ_i based on its fitness to store data, given by:

$$\phi_i = \begin{cases} \frac{1}{\alpha h_i + \beta r(\boldsymbol{x}_i)} & \text{if } m_i > 0 \\ 0 & \text{otherwise} \end{cases} \tag{3.3}$$

where m_i is the memory available on node i, r_i is the risk associated with the current node's location (stored in the distributed belief map) and h_i refers to the minimum hop count required to reach the base station from i as specified in the routing table. Parameters α and β are respectively the routing weights and risk weights, which allow adapting the policy based on the relative importance of the routing time and the environmental risk with respect to each other. Similarly to [35], a node which becomes unfit to store data will evict such data by moving it into its routing queue. The condition for "unfitness" is simply:

$$T\phi_i < \max_{j \in \mathcal{N}} \phi_j \tag{3.4}$$

where $\mathcal{N}$ and T are the set of i's neighbours and the fitness threshold, respectively. The latter's purpose is to ensure that data is transferred only when the neighbours' max fitness is significantly higher than ϕ_i to avoid instability and overhead which would result from frequent and inefficient transfers. This fitness policy causes data to naturally percolate along the edges towards the base station because nodes with a higher potential are both closer to it and located in safer areas. When necessary, data is evicted using a Least Recently Used (LRU) policy and transmitted to the fittest neighbour.

3.4 Simulations

3.4.1 Experimental Setup

We ran extensive simulations in a physics-based simulator, ARGoS [85] with models of Khepera IV [101] robots to eliminate the effect of potential hardware issues on the conceptual validity of our system and also to verify that it scales well to large swarm sizes. We executed 30 simulation runs with 100 robots for each type of experiment to reach results with low uncertainty. We used a 20m x 20m arena and a communication radius $R = 3$m. Three radiation sources are randomly distributed in the environment around the origin. The base station is located in a corner of the arena and its storage capacity is assumed to be infinite.

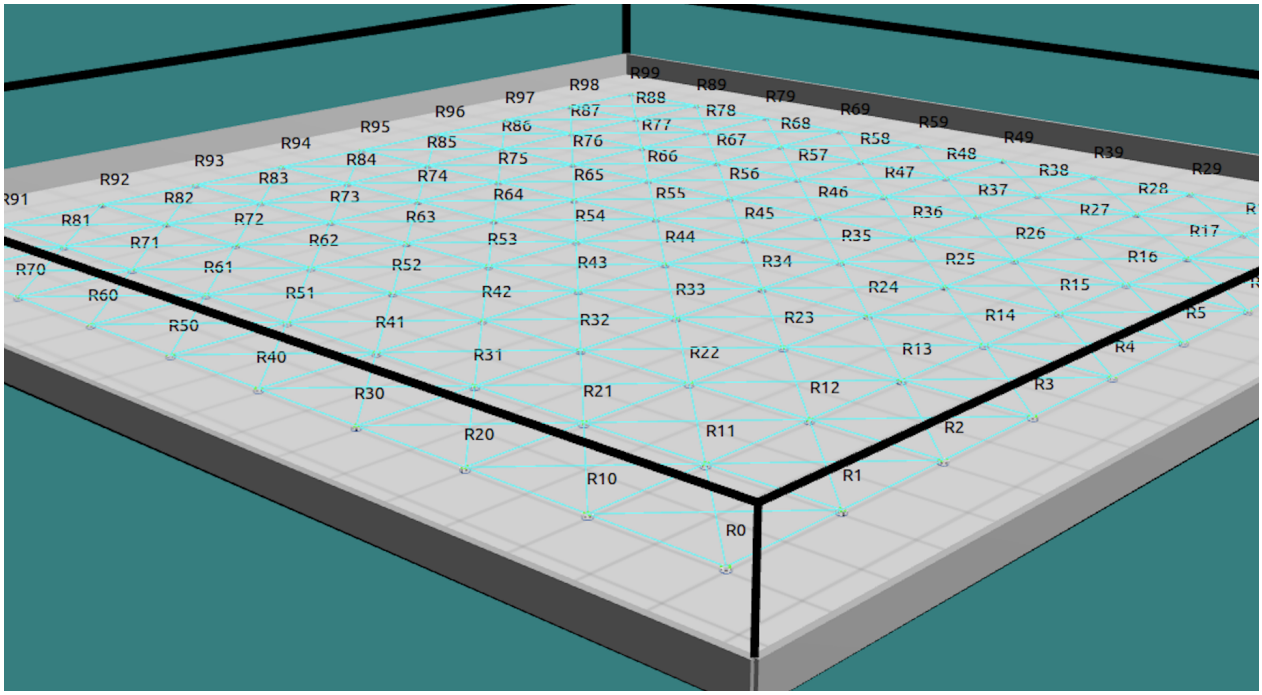

Figure 3.3 400m^2 environment in the ARGoS simulator with 100 KheperaIV robots distributed in a grid-like pattern.

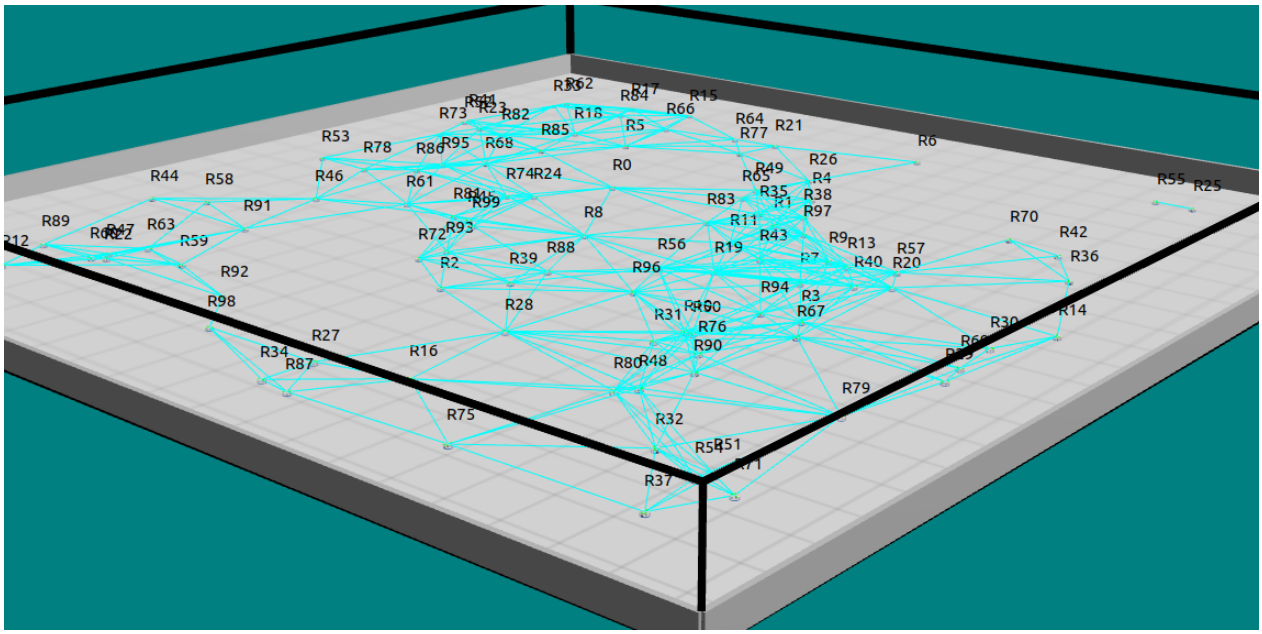

Figure 3.4 400m^2 environment in the ARGoS simulator with 100 KheperaIV robots distributed in a scale-free pattern.

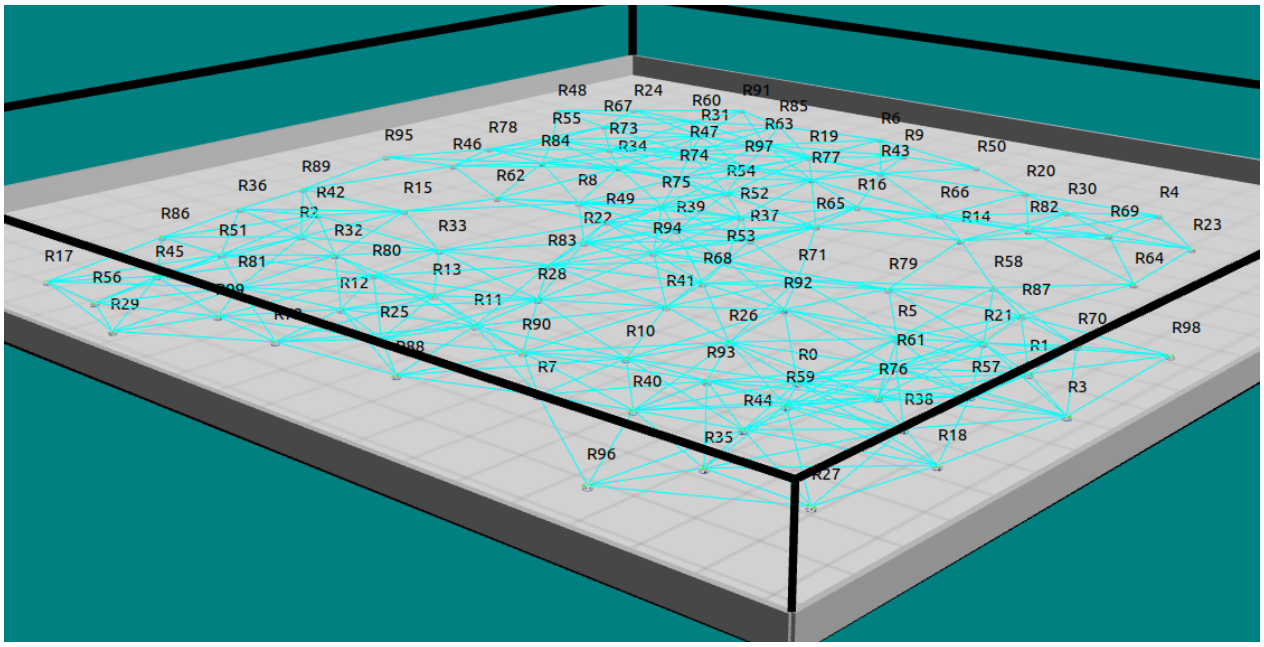

Figure 3.5 400m^2 environment in the ARGoS simulator with 100 KheperaIV robots in a formation obtained through Lennard-Jones potential interactions.

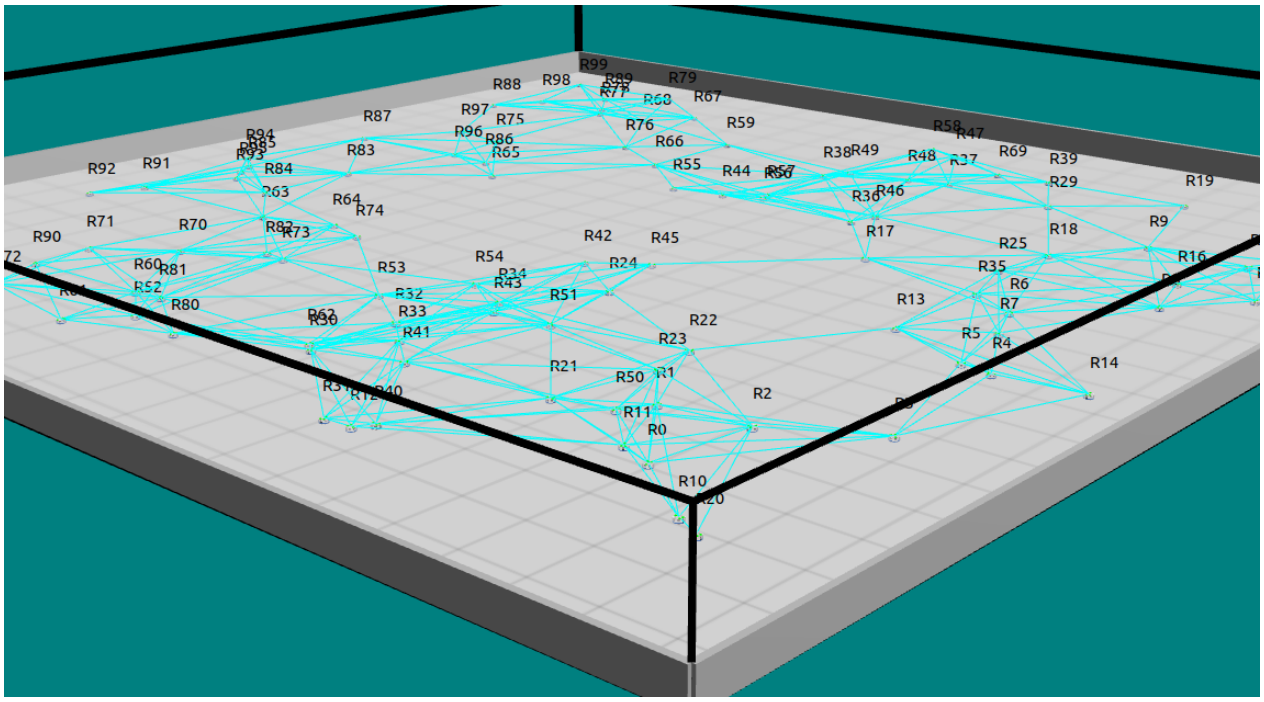

Figure 3.6 400m^2 environment in the ARGoS simulator with 100 KheperaIV robots in a formation obtained through random walk motions.

In order to replicate realistic operation scenarios, we chose to artificially introduce bandwidth limitations. In all experiments, robots can only exchange up to 10 data items at every time step. For the same reason, our simulated robots have a limited storage capacity of 50 data items of at most 50 bytes each, as the data are simple items such as small tables or floating-point numbers. This gives a total storage capacity of 2500kb per robot. Because the robots can only exchange up to 20% of their stored data at a given time step, it amounts to a bandwidth of 0.5kb/s. Data is generated by each robot at fixed out of phase intervals.

To evaluate the performance of our system in different scenarios, we tested it with static topologies: a grid-like formation, and a scale-free network; as well as with dynamic topologies: a formation obtained through Lennard-Jones potential interactions and a formation evolving from random walk motions. Testing with static topologies allows us to verify applicability with fixed wireless sensor networks relevant to IoT applications, while experiments with dynamic topologies are more relevant for mobile robotics applications.

To setup RASS, we used values of $\alpha = 10$ and $beta = 1$ in Eq. 3.3 because risk measurement values are normalized between 0 and 1 while hop-count values are typically upper-bounded to 10 (for 100 robots). Our first benchmark algorithm is to use a fitness policy based purely on hop count, i.e. if required, data is sent only to neighbours closer to the base station. Our second comparison baseline is to store data in a virtual stigmergy (a CRDT) [33]. Using the virtual stigmergy practically ensures that no data can be lost due to corruption because it is fully replicated across the system. The first metric we used in our performance evaluation is the reliability R, expressed as:

$$R = \frac{n_g - n_l}{n_g} \tag{3.5}$$

where n_g and n_l are respectively the amount of data generated and lost at every time step. The second metric is the average data transfer speed, measured as the delay between the creation of a given datum and its arrival by percolation to the base station. We excluded results of this metric for the virtual stigmergy, as the stigmergy cannot include the concept of a base station (since all nodes are peers), and stigmergy propagation speeds are detailed in [33]. The third metric is the evolution of the system's total storage capacity over time, i.e. the amount of data stored by the agents and the base station combined.

3.4.2 Results

The results obtained in the 30 simulation runs for the static topologies (grid-like and scale-free) as well as for the dynamic topologies (Lennard-Jones potential and random walk) are

presented in groups for each metric (reliability, transmission speed and total storage capacity) in the following pages.

Results show that RASS outperforms the hop-count algorithm in terms of reliability. For the grid topology experiments, Fig. 3.7 shows that RASS was more than 1.5 times as reliable as the closest benchmark. For the experiments using the scale-free topology (Fig. 3.8, the results were much closer, but RASS performed the best. The reliability obtained for the Lennard-Jones (Fig. 3.9) and the random topologies (Fig. 3.10 were similar, as they were on average 1.4 times better than the best competitor. Because of the risk awareness component included in its fitness policy as detailed in Eq. 3.3, robots do not always route the data through the shortest path to the base station. RASS avoids the dangerous storage nodes of the system when routing data which explains the higher reliability levels displayed in Fig. 3.7, Fig. 3.8, Fig. 3.9 and Fig. 3.10 . This is why, on average, RASS takes 54.89% more time to route the data to the base station when compared to the hop-count algorithm. The shortest path might not always be the safest one; RASS will take an alternate route if the risk associated with the shortest one is too high. This happens whenever the risk element in Eq. 3.3 is high enough to offset a possibly low value in the hop count term. On the other hand, the hop-count algorithm always takes the shortest path towards the base station regardless of the risk associated with it. This leads to a higher number of data losses due to corruption and an overall lower reliability. The clear outlier for all topologies is the virtual stigmergy, achieving very poor reliability figures. The reasons for this are explained further because they are also reflected by another metric.

The hop count algorithm can yield faster transfer speeds for all topologies. As explained in Sec. 3.4.1, transfer speed analysis for the virtual stigmergy was excluded. In the grid topology (Fig. 3.11), the mean transmission time for the hop count algorithm half smaller than RASS's. The smaller amount of messages shown in the graph for the hop count method is explained by its poor reliability, meaning fewer messages reached their destination. Fig. 3.12, showing results for the scale-free topology, is where the results are closer: the distributions are almost identical, except for a small tail end in RASS's case Fig. The transfer speed distributions for the Lennard-Jones and the random topologies are presented in 3.13 and Fig. 3.14 respectively. They are quite similar: RASS has slower transfer speeds, but its curve is in both cases has significant overlap with the one from the hop count algorithm.

The other studied metric was the evolution of the total storage capacity of the system over time. As for the other metrics, the gap between RASS,the hop count algorithm and the virtual stigmergy was most pronounced in the grid topology results presented in Fig. 3.15. In this situation, RASS managed to store twice as much data as any of the benchmarks.

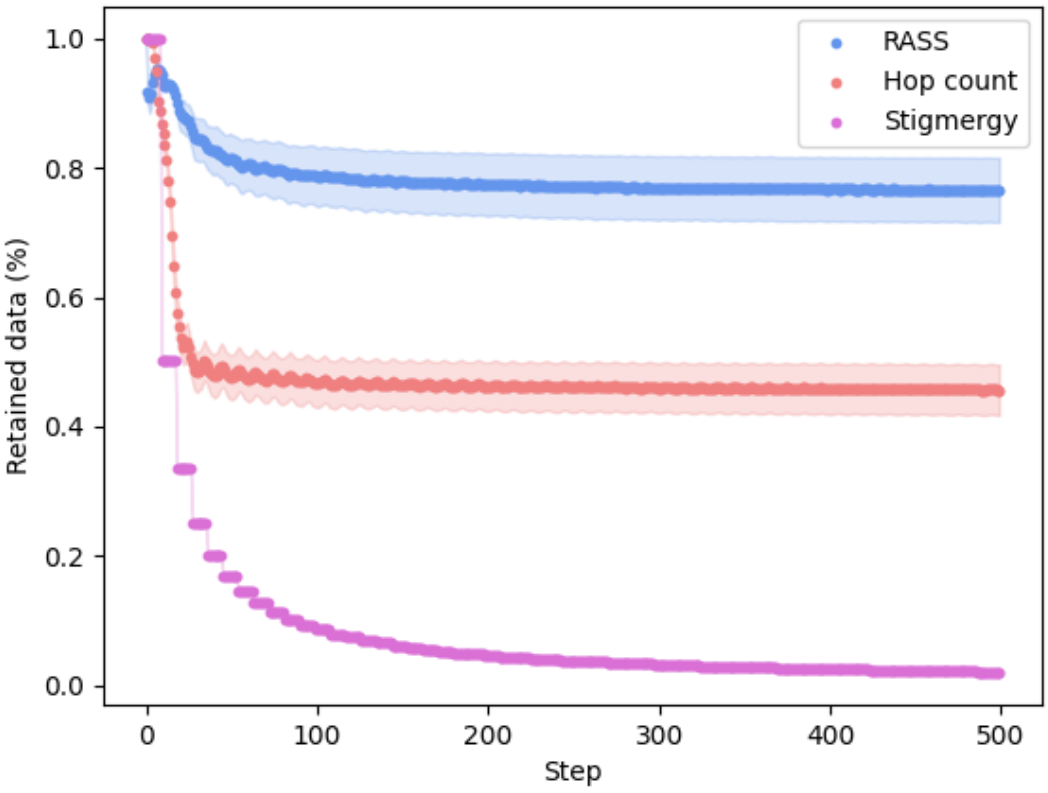

Figure 3.7 RASS reliability (grid topology)

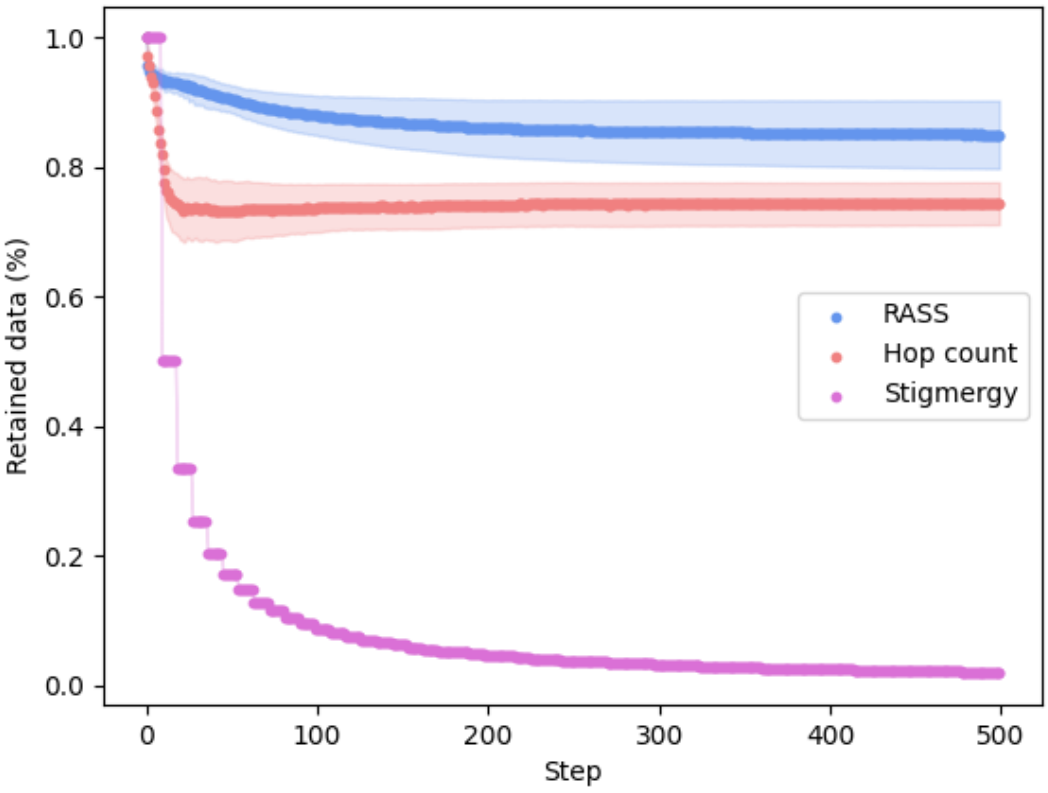

Figure 3.8 RASS reliability (scale-free topology)

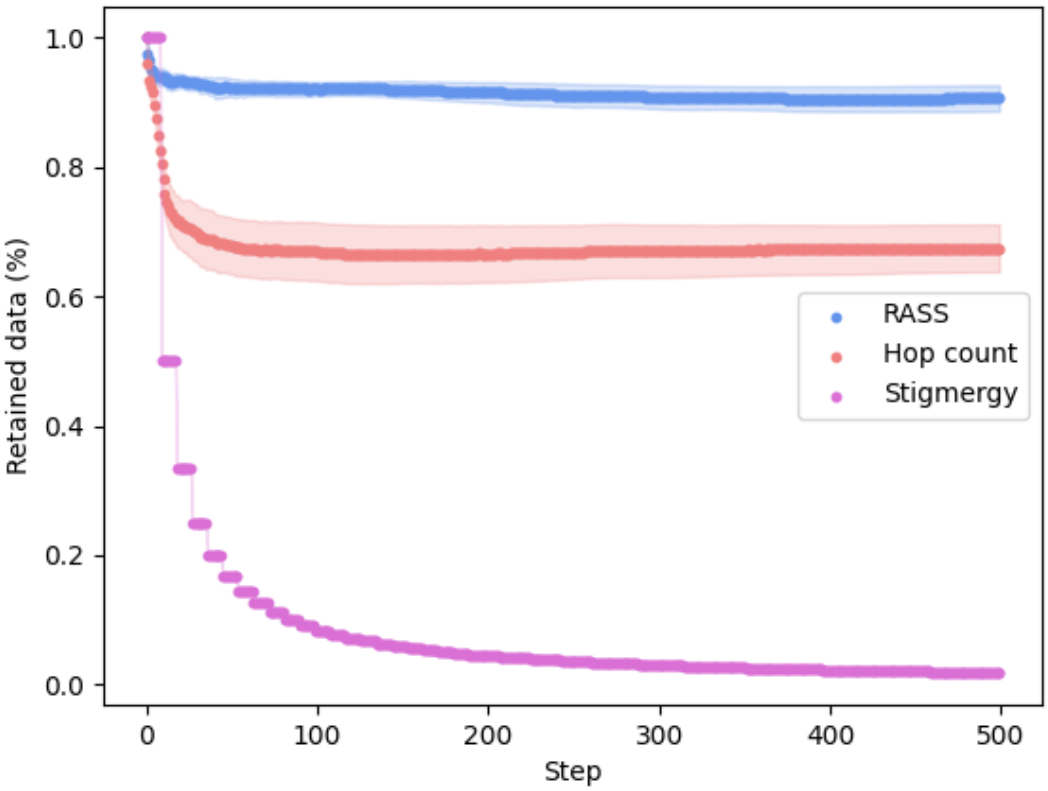

Figure 3.9 RASS reliability (Lennard-Jones topology)

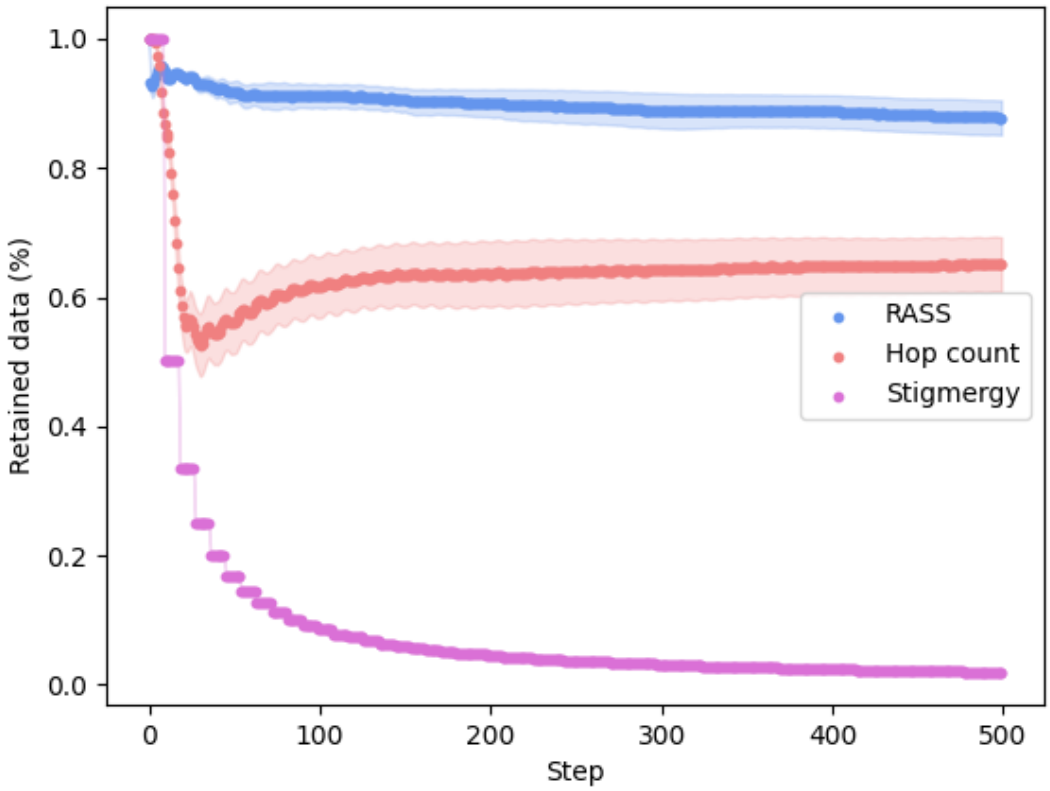

Figure 3.10 RASS reliability (random topology)

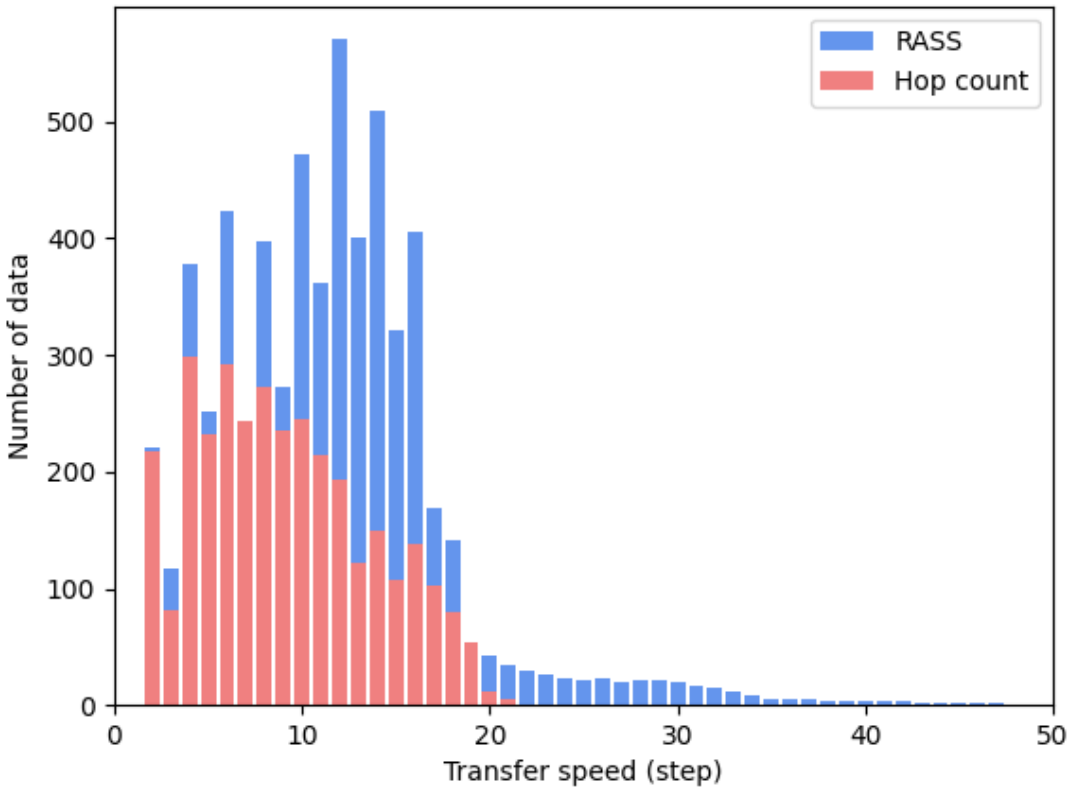

Figure 3.11 RASS transmission speed (grid topology)

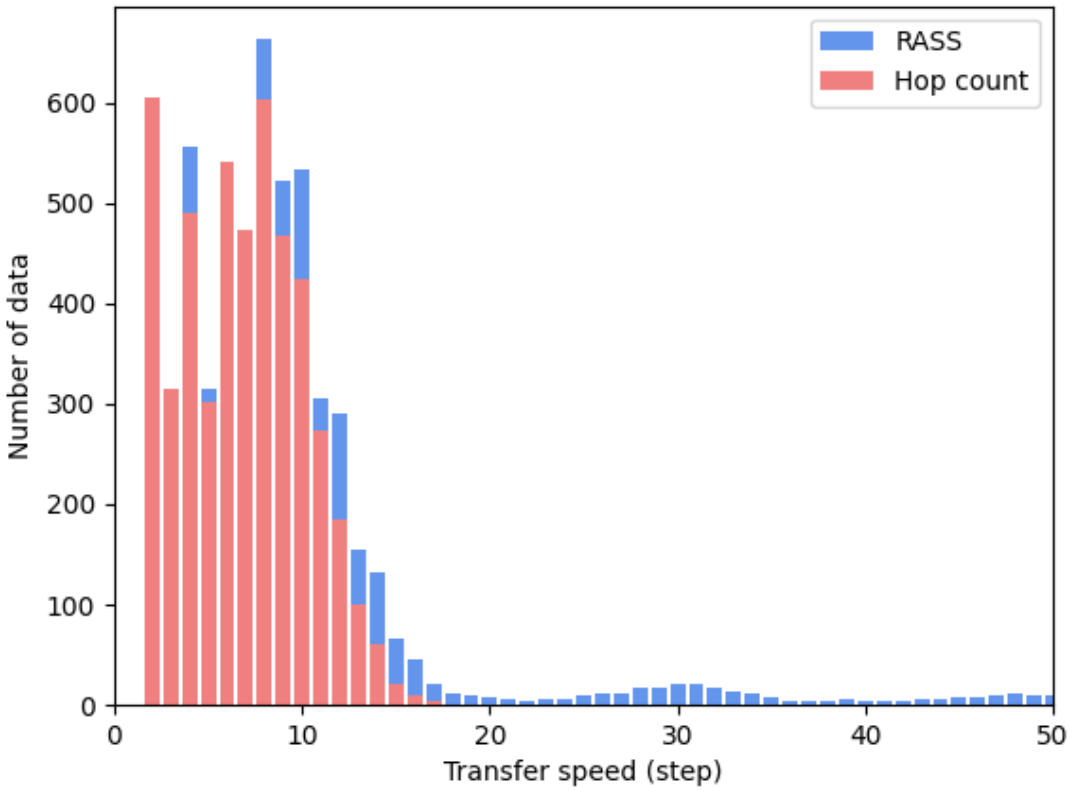

Figure 3.12 RASS transmission speed (scale-free topology)

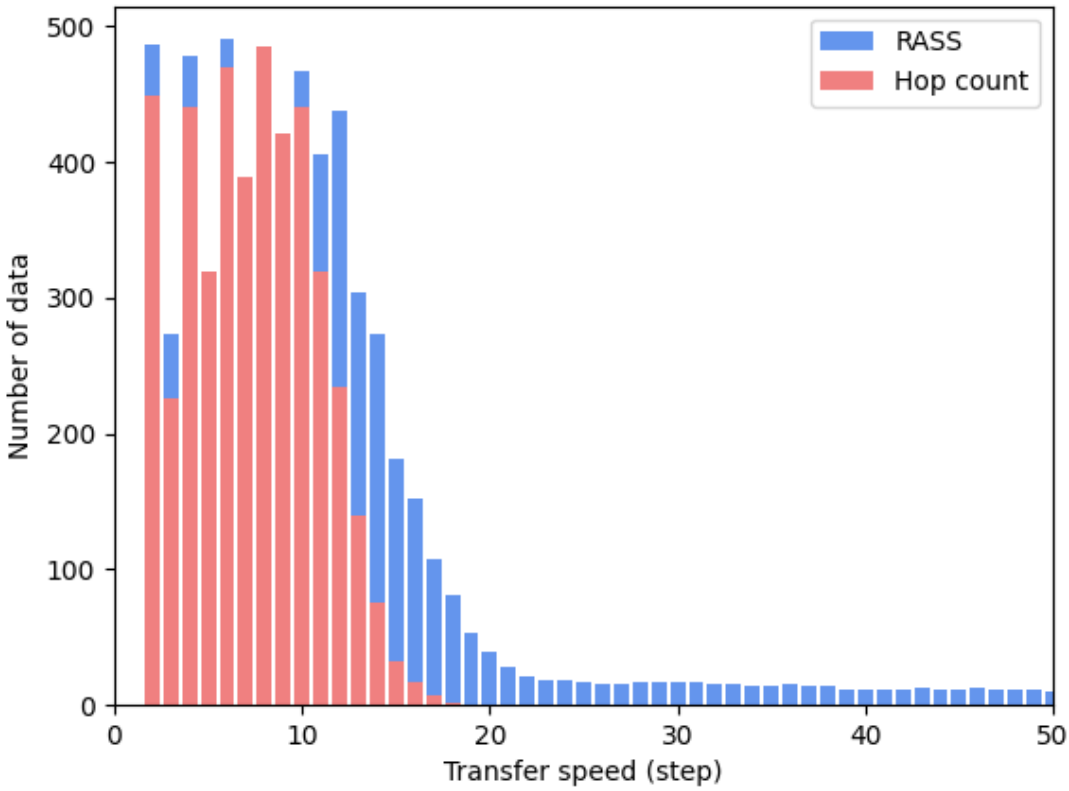

Figure 3.13 RASS transmission speed (Lennard-Jones topology)

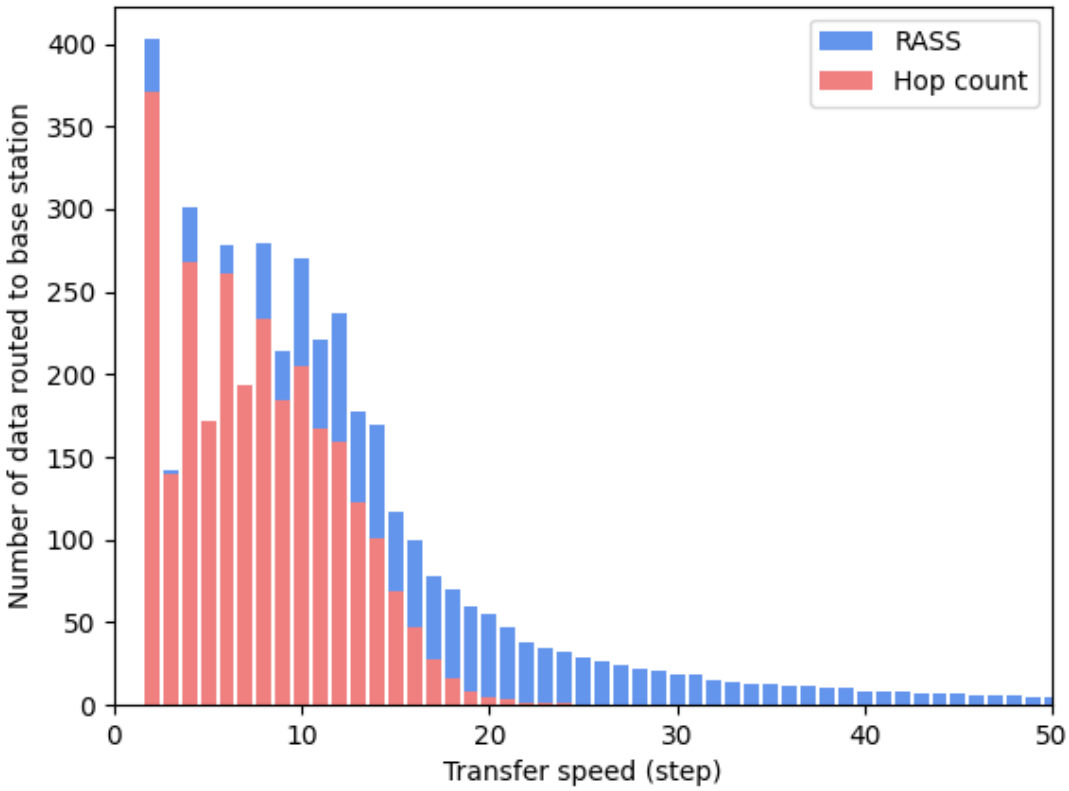

Figure 3.14 RASS transmission speed (random topology)

For the scale-free topology, the results were much closer as shown in Fig. 3.16, but the risk-aware solution still performed the best. For the Lennard-Jones and random topologies, RASS achieved a 45.2% and 60% improvement in system-wide storage capacity over the next best compared method.

Table 3.1 Average transfer speed and average individual memory usage with different topologies.

Topology	*Algorithm*	*Transfer speed (hops)*	*Memory used (%)*
Grid-like	RASS	11.45	1.35
	Hop-Count	9.11	0.61
	Stigmergy	N.A.	100.00
Scale Free	RASS	11.44	1.95
	Hop-Count	6.85	0.50
	Stigmergy	N.A.	100.00
Lennard-Jones	RASS	12.51	1.69
	Hop-Count	7.32	0.51
	Stigmergy	N.A.	100.00
Random search	RASS	12.68	1.67
	Hop-Count	7.76	0.57
	Stigmergy	N.A.	100.00

Looking at a summary of these results can help gain further understanding, and table 3.1 is useful to this end. For the virtual stigmergy, most of the data losses can be attributed to the storage having reached its maximum capacity. Indeed, because of the fully replicated nature of the stigmergy, the memory of the agents is quickly saturated. Table 3.1 shows that on average, across the 500 steps of the simulations runs, the virtual stigmergy has 100% of every individual robot memory used. This means that the nodes of the system are simply full and cannot store data anymore. In comparison, RASS uses between 1% and 2% of the local memories of the nodes, and hop-count is even lower at values around 0.5%. This full redundancy prevents losing data from corruptions. However, it entails a very inefficient use of the memory of the robots and ultimately leads to data losses due to insufficient memory capacity. The result is an unchanging storage capacity over time and poor reliability as shown in Fig. 3.7 for the virtual stigmergy strategy. The low values of local memory usage shown in Table 3.1 for RASS and hop-count were obtained because the topologies used to test the algorithms were usually well connected in accordance with the connected graph assumption we made in 3.3.2. For the most part, multiple routes were connecting the nodes to the base station and as a result, the collected data was routed towards the base station instead of being kept locally. Such a result implies that the system, by maintaining a low individual storage occupancy, allows the robots to adapt to situations in which they would

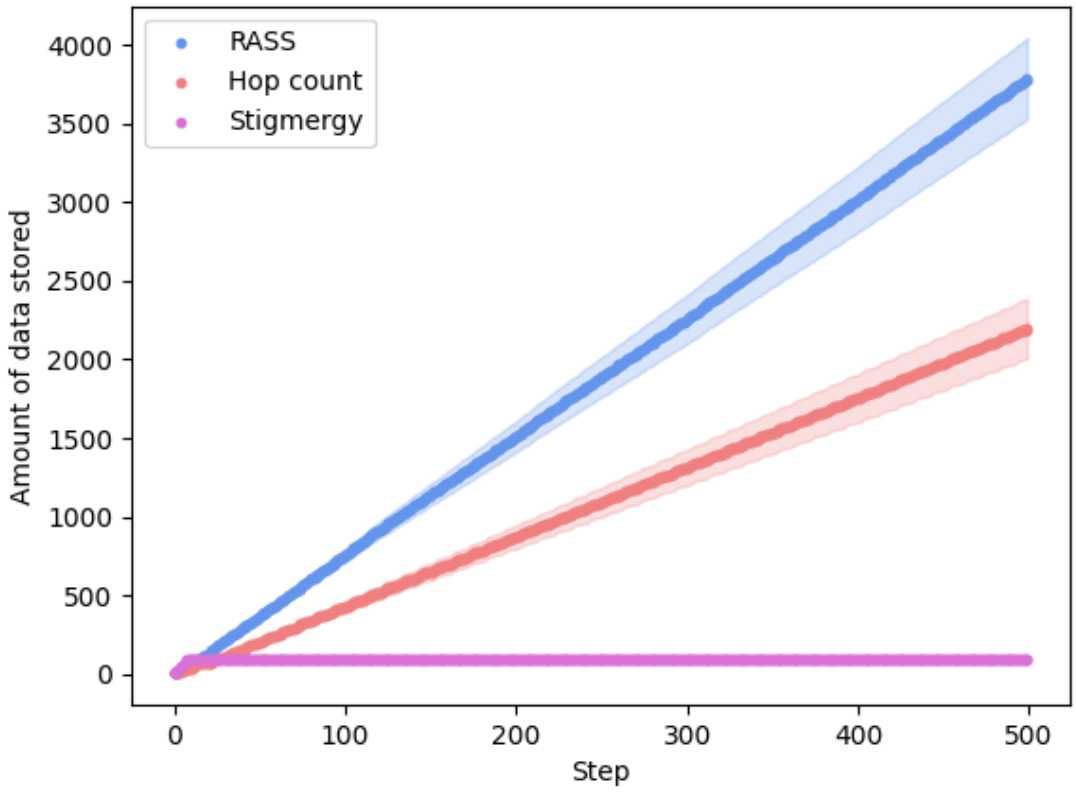

Figure 3.15 RASS storage capacity (grid topology)

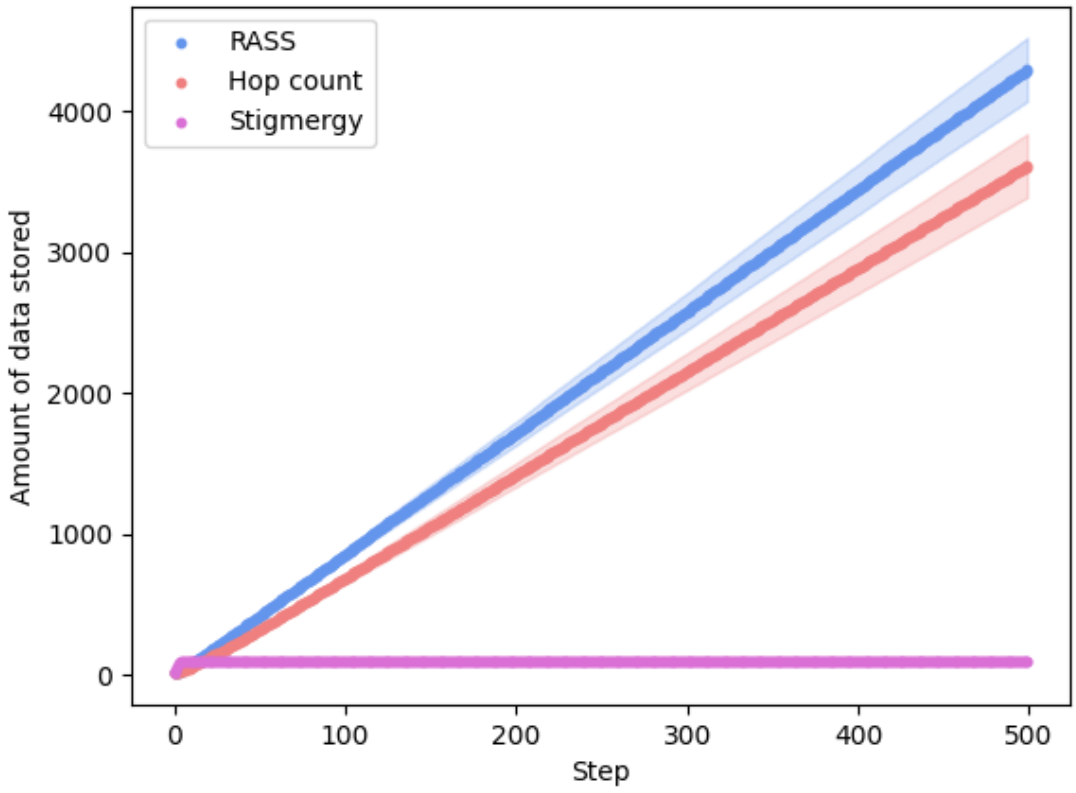

Figure 3.16 RASS storage capacity (scale-free topology)

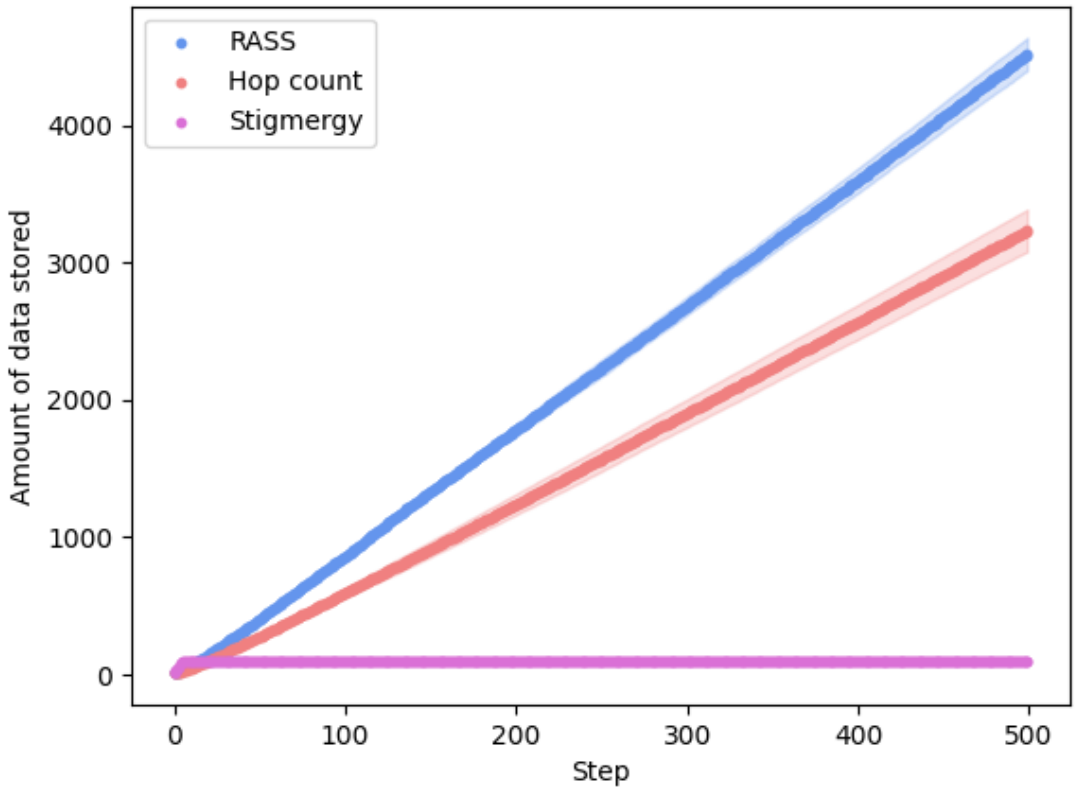

Figure 3.17 RASS storage capacity (Lennard-Jones topology)

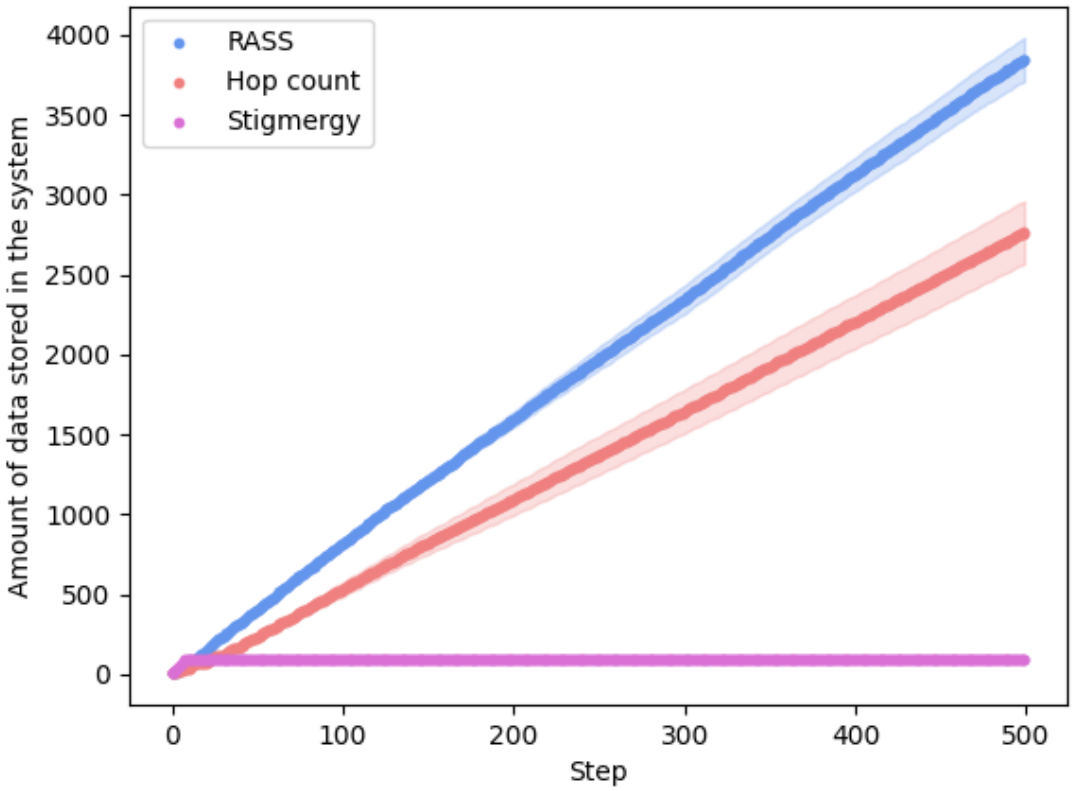

Figure 3.18 RASS storage capacity (random topology)

be temporarily stranded: if their storage were to be mostly full, they would not be able to generate new data without promptly losing it. This storage buffer thus allows them to continue their data collection while they are temporarily disconnected from the rest of the swarm.

3.5 Physical experiments

3.5.1 Experimental setup

We evaluated RASS' performance with the same metrics on physical robots to confirm the real-world applicability of our system. We used 5 small drones, the CogniFlies [102], in a controlled indoor environment. These drones use standard Raspberry Pi Zeros as their main computer, meaning they have relatively low capabilities, and are therefore well suited to verify our algorithm. We designed a static topology with one drone acting as a base station and the 4 others acting as agents. The radiation source (red cone) in Fig. 3.19 is positioned to make one of the paths more dangerous, allowing us to verify if data is routed in the longer but safer path. The communication range is set to 1.5m, which is a realistic but restrictive value for drones equipped with WiFi capabilities. The topology is illustrated in Fig. 3.20. To control the topology, we kept the drones in fixed positions. To assess RASS' performance, we compared it with a hop-count algorithm over 3 runs.

3.5.2 Results

The reliability results of the physical experiments conducted on the drones are presented in Fig. 3.21. They show that RASS outperforms the hop-count algorithm in terms of reliability by a factor of almost 2. In other words, RASS lost much less data due to corruption issues. This lead to overall greater swarm storage. The results obtained for the total storage capacity of the swarm are shown in 3.22. They are somewhat less convincing, because they have a higher variance. This could be mitigated by running more simulations. However, the trend is clear: even by accounting for large variance, RASS outperforms the benchmark. In a realistic interpretation of the data, RASS achieved a total storage capacity of nearly double that of the hop count algorithm, which is coherent with the results obtained for the reliability. Even if the topology used to assess the performance of our algorithm was simple and the number of agents in the system was limited, the physical experiments confirm the real-world applicability of RASS. Using only local interactions, it was able to choose safer paths for the data to be routed through which resulted in fewer data corruptions.

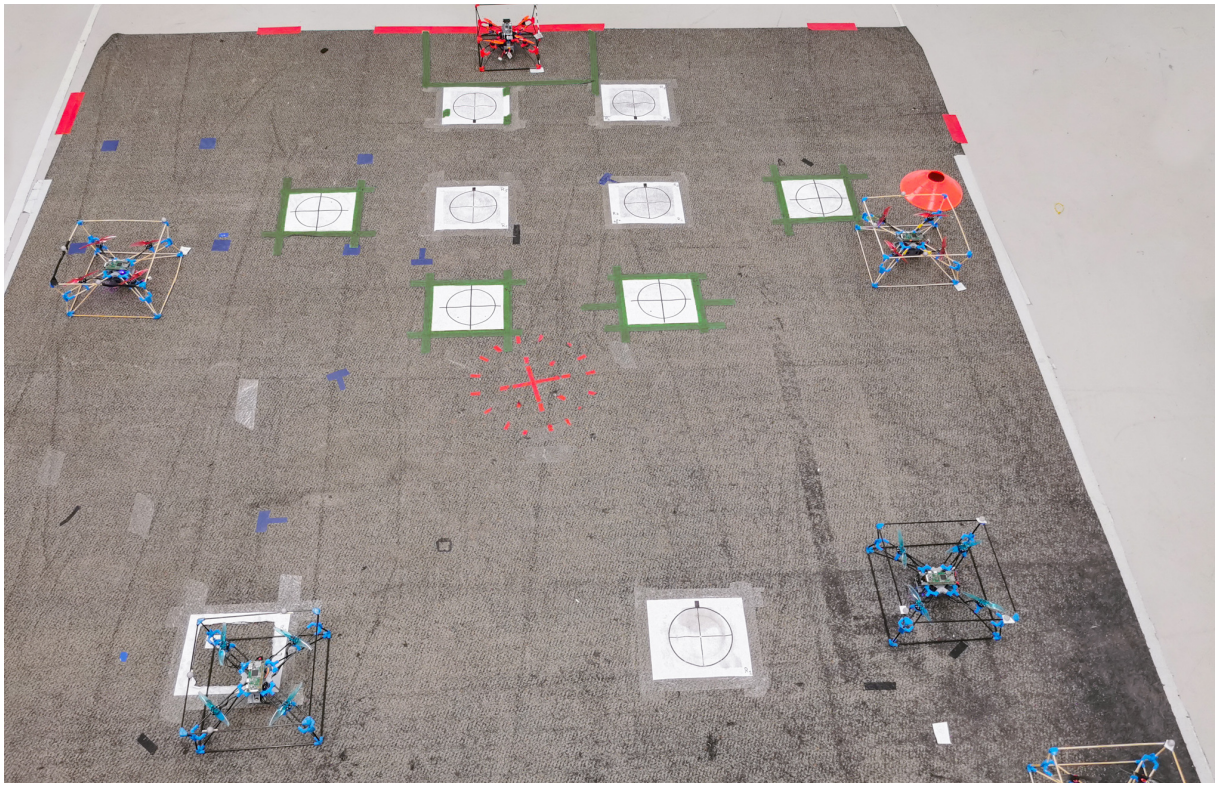

Figure 3.19 Physical disposition of the 3x3m environment with 5 drones and a radiation source (red cone).

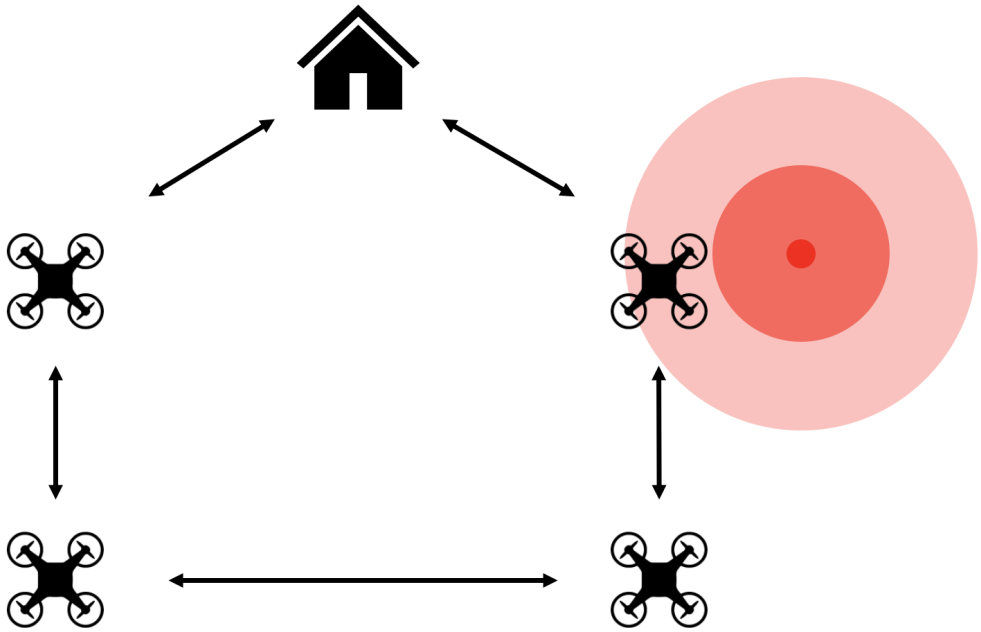

Figure 3.20 A simplified graph description of the physical testing environment. The decaying radiation source is represented by the concentric red circles. The base station, at the top of the figure, is also a drone.

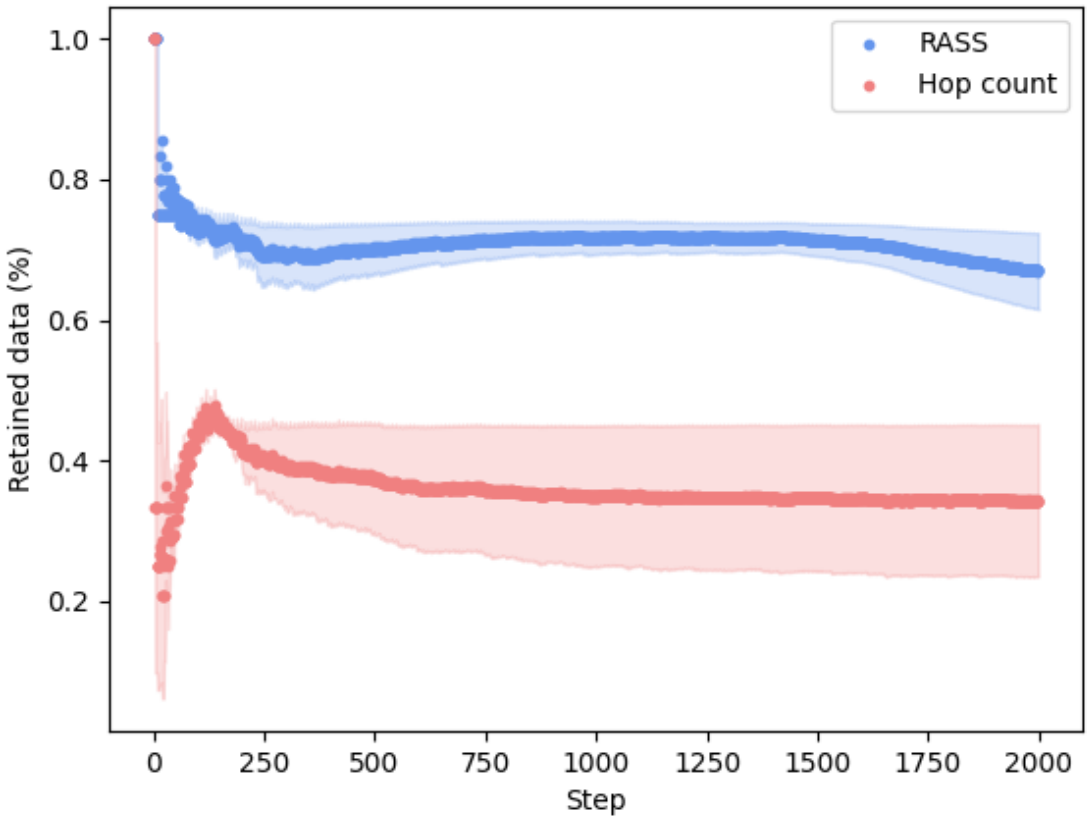

Figure 3.21 Evolution of reliability over time on the the physical experiments

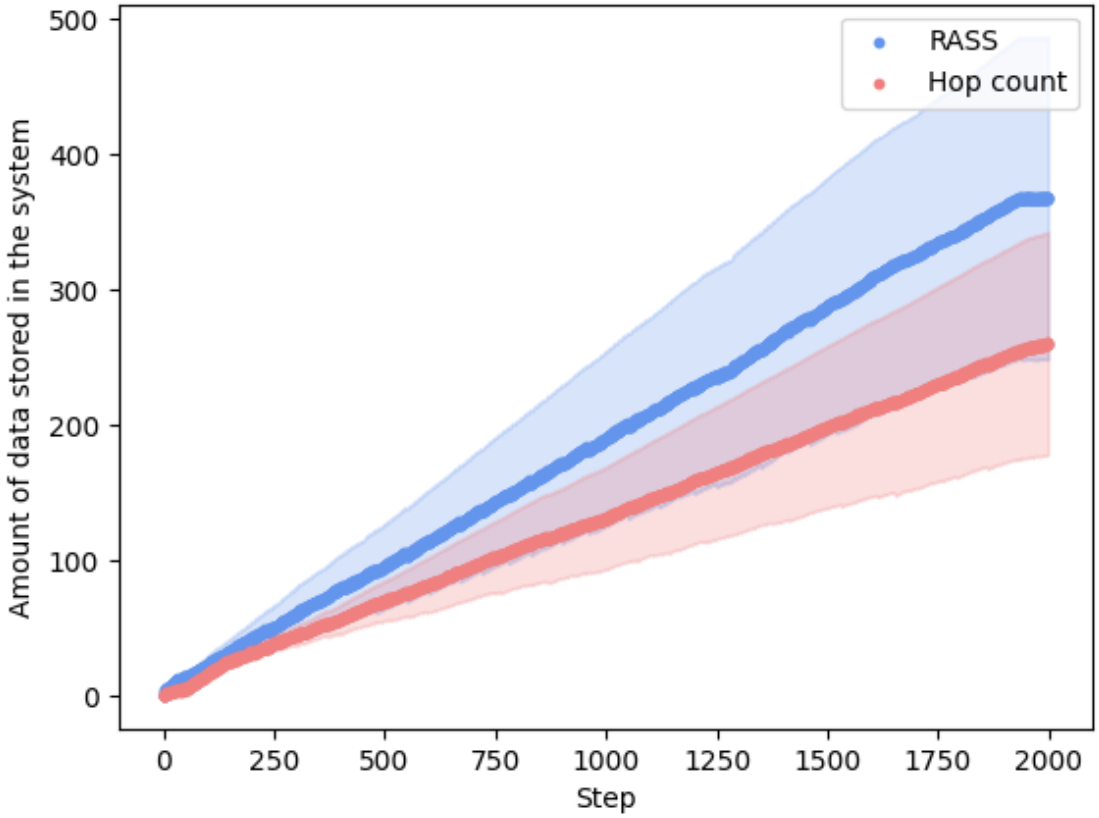

Figure 3.22 Evolution of storage over time on the the physical experiments

3.6 Conclusion

We presented RASS, a Risk-Aware Swarm Storage system in which a swarm of robots can collectively store data on strategically chosen members. This choice is made without central coordination and is purely based on local information shared between the robots. This information is simply composed of risk measurements and topological distance from a robot to a base station, and is used to determine a robot's fitness to store data as well as to establish the most reliable and fast route towards the base station.

We show in our experiments that RASS largely outperforms a hop-count based solution as well as a virtual stigmergy in terms of reliability and total swarm storage capacity, while only being slightly slower in terms of percolation speed compared to the hop-count-based algorithm. RASS showed good scalability in physics-based experiments as it repeatedly performed well with a large number of robots. It also performed well in experiments on physical robots.

A possible improvement for the system could be to use a Kalman Filter [58] to eliminate outliers in radiation measurement which can arise due to sensor imprecision. Also, while RASS considered the risk to which robots are subjected, it would be interesting to study the impact of how risk affects the communication links between them. More generally, an interesting direction for future work could be to conduct experiments in more diverse scenarios, for example in search and rescue applications where image storage and processing is required, therefore increasing the system's workload.

CHAPTER 4 RISK-AWARE EXPLORATION

In this chapter, the article in which DORA was first described [94] is summarized. DORA was created in the context of this thesis and is closely related to the theme of risk awareness in swarm robotics. To avoid redundancy and to make this chapter more concise, the related works and background necessary to understand this system are presented in Ch. 2. Figures in this chapter are taken from the article with permission from the authors.

Because exploration of unknown environments is an important challenge in the field of robotics, we wanted to show the benefits of taking risk awareness into account when developing solutions for this kind of challenge. We based our exploration strategy on distributed belief maps. This way, swarm efficiency is leveraged because robots collaborate by sharing useful information. The key idea behind DORA is to minimize risk exposure while maximizing exploration coverage, leading to more efficient exploration because of fewer risk-related failures.

4.1 Introduction

Unknown environment exploration by robots has applications in numerous fields. For example, search-and-rescue missions [103] and space missions [104] can both benefit from using robots, as they often operate in environments inaccessible to humans. This inaccessibility can stem from multiple causes, such as the remote nature of these places (space, other planets, oceanic depths), the impossibility for humans to reach them (caverns with too small openings) or their dangerous nature (radioactive zones, forest fires, flooded areas). For all of these, there is necessarily an associated risk factor, which means that robots sent to explore them will necessarily be exposed to some danger increasing the risk of failures. Robot swarms mitigate the effects of individual failures [18–20] and improve overall terrain coverage performance [92]. However, even in robot teams, failures have negative effects on overall performance, which is why we sought to reduce them as much as possible through DORA.

The motivation for creating a purely decentralized algorithm is to avoid the pitfalls related to centralized systems, such as bottlenecks and single points of failures. The first can be related to low capacity robots, or to noisy environments affecting communication efficiency or causing memory corruption. The second is particularly important in the risky scenarios where individual malfunctions are more likely and could cause system-wide failures. Consequently, relying on locally shared information and onboard computation is necessary.

As we found no existing decentralized risk-aware exploration algorithm, we sought to create one.

4.2 System Model

The motivation for including risk awareness in a distributed exploration algorithm can be seen in Figs. 4.2 and 4.3, where failures lead to decreasing performance over time. Conversely, in Figs. 4.4 to 4.7, avoiding dangerous areas reduces failures and maintains exploration efficiency. In order to design our system, we modelled the environment as a 2D grid ($E \subset \mathbb{Z}^2$) in which agents $a_i \in A$ carry out their task. Several design considerations shaped DORA. They are summarized here, and more details can be found in the referenced paper.

First, risk and failure probability had to be formally modelled. We chose to represent risk as point radiation sources $s_j \in S$ with an exponentially decaying intensity I_j, but any other type of danger could have been used with our system. The equations representing the radiation perceived by a robot a_i failing due to radiation in cell $\boldsymbol{x}_i$ can be combined as:

$$r(\boldsymbol{x}_i) = b + \sum_{\boldsymbol{s}_j \in S} \frac{I_j}{1 + \lambda \rho_j^2} \tag{4.1}$$

Where ρ_j is the distance between a_i and s_j, λ is a decay constant and b is Gaussian noise. The probability of failure necessarily increases where $r(\boldsymbol{x}_i)$ is higher.

Second, we defined the objective of exploration as gaining information by visiting cells from E. Logically, recently visited cells are unlikely to yield any information gain. Therefore, priority is given to unvisited and less recently visited cells. Doing so requires saving the last time of exploration of $\boldsymbol{x}_i$ in a scalar field $\epsilon(\boldsymbol{x}_i) = t_\epsilon$. The probability of finding useful information decreases exponentially for values of $\epsilon(\boldsymbol{x}_i)$ which are closer to the current time step t.

Third, as way of storing these values, we use a CRDT: the virtual stigmergy from [33]. This allows robots to exchange information whenever possible (thereby achieving a loose eventual consistency) without relying on a central communication hub. Both $r(\boldsymbol{x}_i)$ and $\epsilon(\boldsymbol{x}_i)$ are stored in distributed belief maps. These are updated at every time step.

Fourth, we describe a position-based control law for the robots which minimize risk and maximize information gain. These optimizations are described by local gradients of each scalar field in a Moore neighborhood (see Fig. 4.1) around a_i.

The risk gradient $\boldsymbol{\nabla}_{r;i}$ is given by:

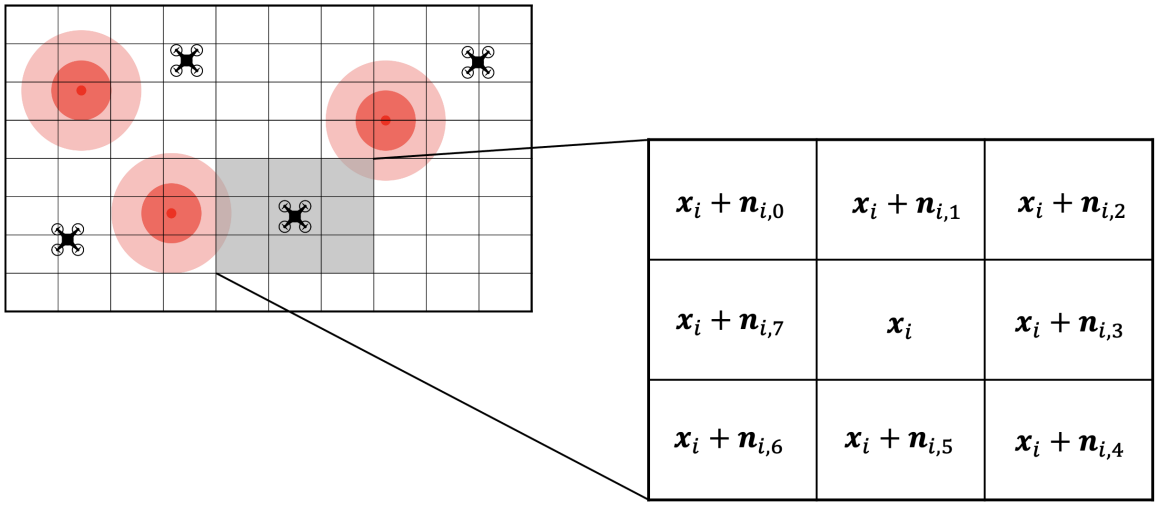

Figure 4.1 $\boldsymbol{x}_i$'s neighborhood. $\boldsymbol{n}_{i,0} = (-1, 1)$ is neighbor 0's offset from $\boldsymbol{x}_i$.

$$\boldsymbol{\nabla}_{r;i} = \sum_{\boldsymbol{n}_j \in \nu} \hat{\boldsymbol{n}}_{i,j} \cdot (r(\boldsymbol{x}_i) - r(\boldsymbol{n}_{i,j})) \tag{4.2}$$

where $\hat{\boldsymbol{n}}$ is the unit form of $\boldsymbol{n}$. The exploration gradient $\boldsymbol{\nabla}_{\epsilon;i}$ is calculated in the same manner. We combine these to obtain the control law:

$$\boldsymbol{x}_i^{t+1} = \boldsymbol{x}_i^t + (\alpha \boldsymbol{\nabla}_{r;i} + \beta \boldsymbol{\nabla}_{\epsilon;i} + \gamma \boldsymbol{o}_i) \tag{4.3}$$

where α, β, γ are parameters related to risk avoidance, exploration gain and obstacle avoidance control. The latter is added from robustness and inspired from [105].

Finally, to ensure scalability, we made sure DORA's computational and communication costs remained low. They are represented by $C(A, \nu, E)$ and $D(A, \nu, E)$ where ν is the neighborhood and they are both bounded by $\Theta(|\nu|)$ because of the nature of the algorithm and of the virtual stigmergy.

4.3 Experiments

We performed experiments both in simulations and on physical robots. In both cases, for consistency, we used KheperaIV [101] robots, which are relatively small and equipped with infrared sensors required for obstacle avoidance. It should be noted that they are capable of wireless networking through the 802.11b/g WiFi protocol, making swarm communication

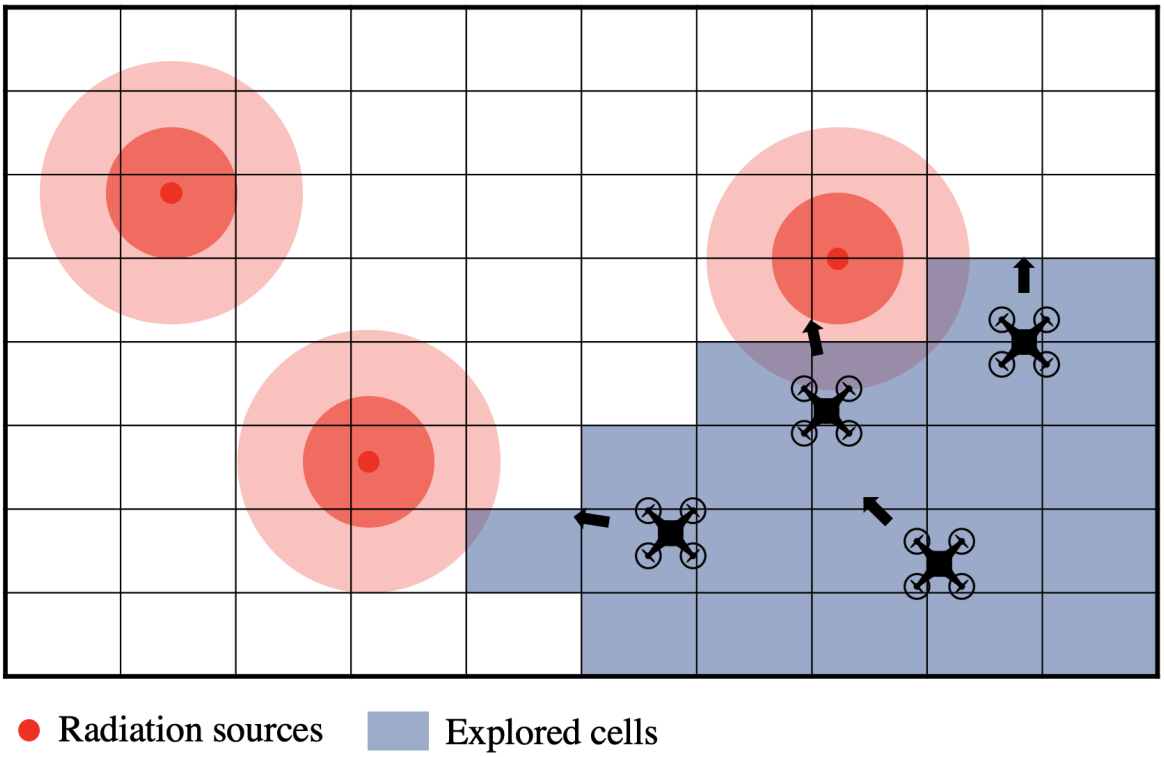

Figure 4.2 Robots start exploring but are unable to sense environmental radiation. The only driving force of the algorithm is exploring new cells.

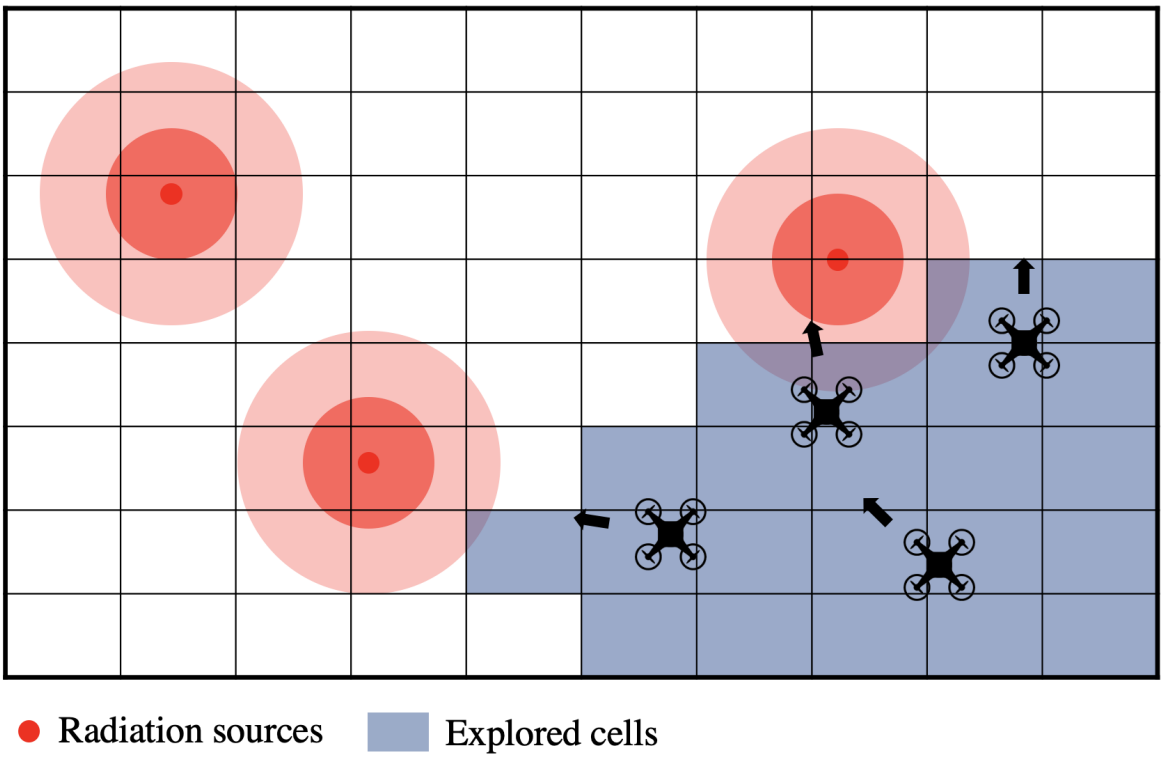

Figure 4.3 Robots fail because they do not discriminate between safe and dangerous cells. Exploration is carried out by fewer robots. Exploration efficiency drastically decreases and large areas of the environment remain uncovered.

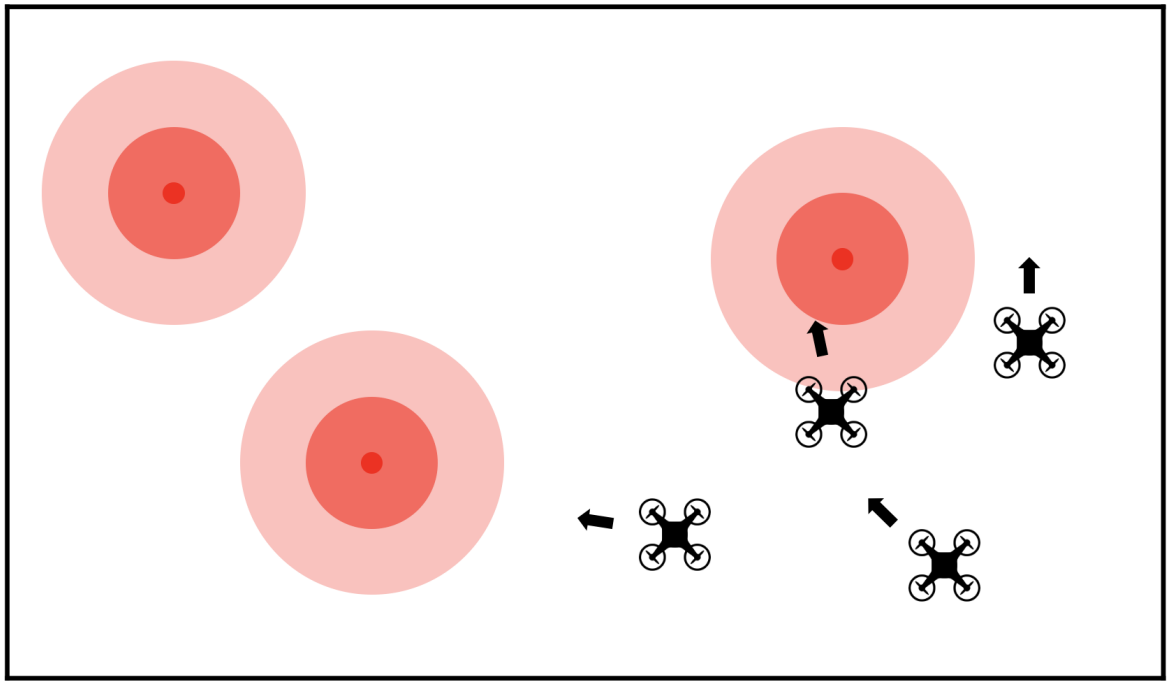

Figure 4.4 Robots start exploring a hazardous environment.

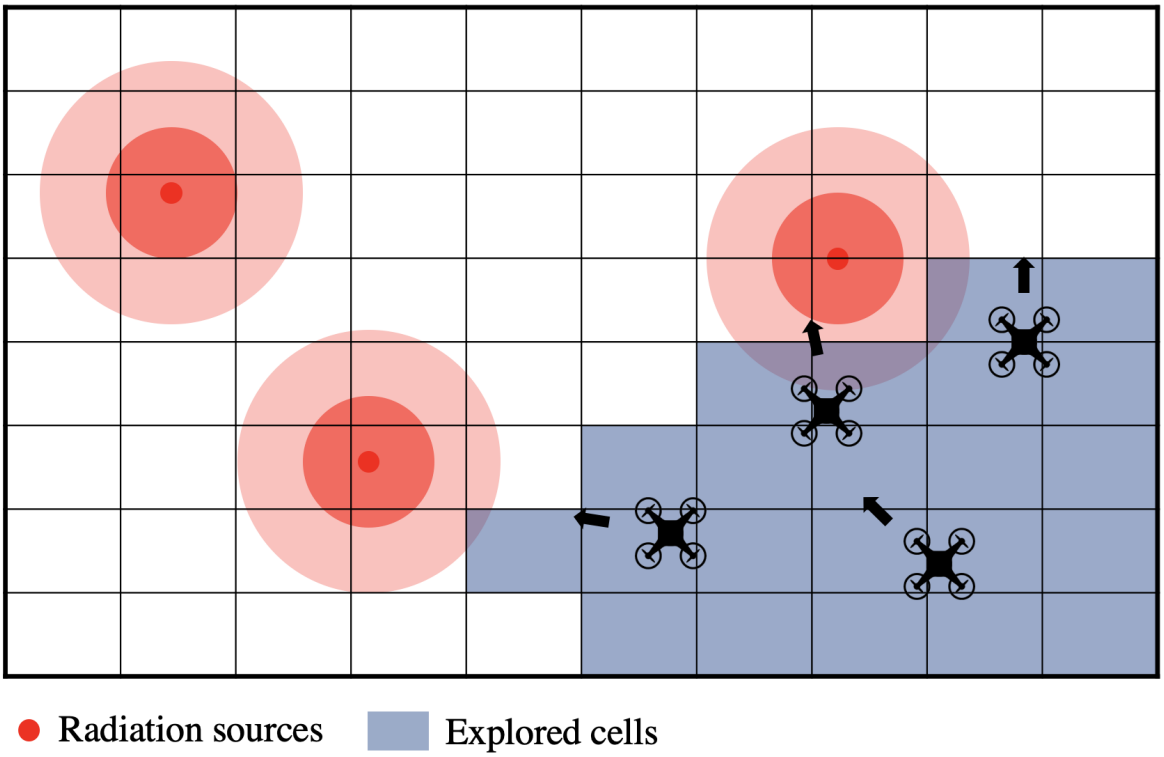

Figure 4.5 A grid is formed. When a new cell from this grid is explored, the sensed radiation is used to update the DBM.

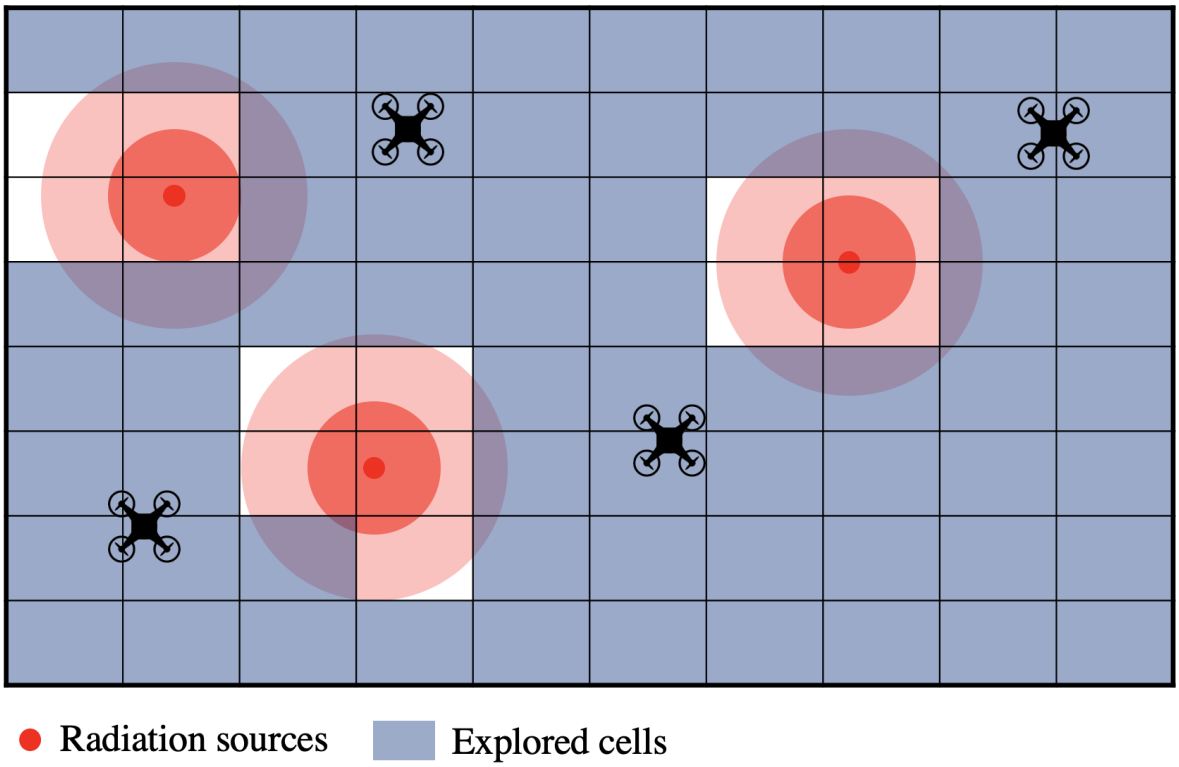

Figure 4.6 The cells have been mostly covered by the robots.

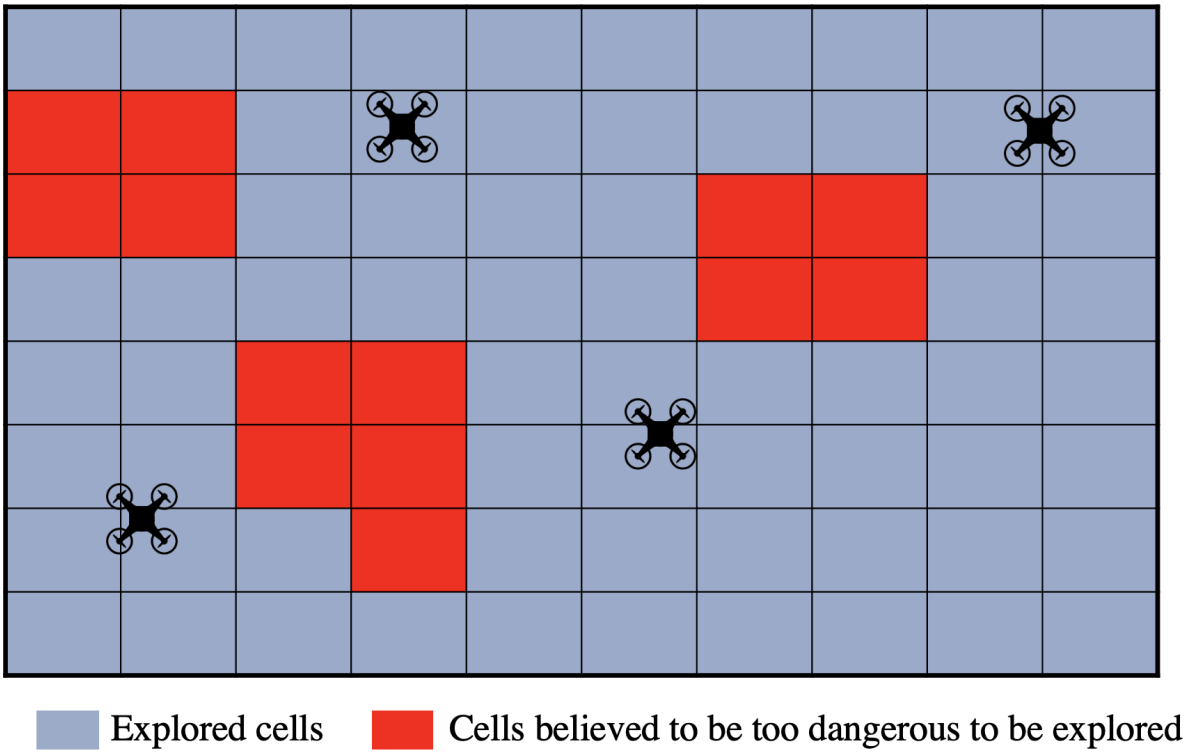

Figure 4.7 Only cells believed to be too dangerous remain unexplored.

possible. We compared our algorithm with two baselines: FBE and a random walk algorithm. The metrics we used to evaluate our system's performance were the number of active (not failed) robots over time, the total number of cells explored over time, and the bandwidth usage. Failures were triggered randomly based on (emulated) perceived radiation from (4.1). Detailed motivation for parameter and metric choices can be found in the article.

The first step was to test DORA by doing simulations in ARGoS. To verify the effect of swarm size on scalability, we ran our virtual experiments with varying swarm sizes ($N = \{10, 15, 20\}$) deployed randomly in a 20x20m environment with randomly positioned radiation sources and obstacles. To be thorough, we performed 50 simulation runs with 300 time steps each for each algorithm.

The second step was the physical experiments. We ran them on 5 robots with fewer time steps (200) because of equipment and time constraints respectively. Robot positioning was obtained through motion tracking performed with OptiTrack Motive [106]. The environment consisted of a 2x2m arena split into a 10x10 cell grid.

4.4 Results

The following shows the average results obtained in the simulation runs as well as those from physical experiment runs.

Figs. 4.8 to 4.11 show that DORA attains similar terrain coverage to FBE. Moreover, the performance gap between the two reduces as N increases, showing good scalability from DORA. Both algorithms outperform the random walk one by far. Interestingly, DORA covered more cells than FBE in physical experiments. This is probably because the small size of the arena presented a pathological case for FBE.

Where DORA truly shows its worth is in Figs. Figs. 4.12 to 4.15, in which the number of active robots over time for each value of N is presented. Indeed, in every experiment scenario, DORA experienced far fewer robot failures than both benchmark algorithms. As for terrain coverage, DORA's performance improves with respect to the other algorithms as the number of robots involved increases. This further shows our algorithm's scalability.

Intuition for how DORA results from Figs.4.8 to 4.11 and Figs. 4.12 to 4.15 are related can be gained by observing Figs. 4.17 to 4.18. These three figures are examples of radiation belief maps of the 20x20m environment for each exploration algorithm of one specific simulation. Blank cells are unvisited areas, red stars are the point radiation sources and grey squares are the randomly generated obstacles. Whereas FBE visited the areas around the radiation sources (as can be seen by the cells with a high radiation level), DORA avoided them. This

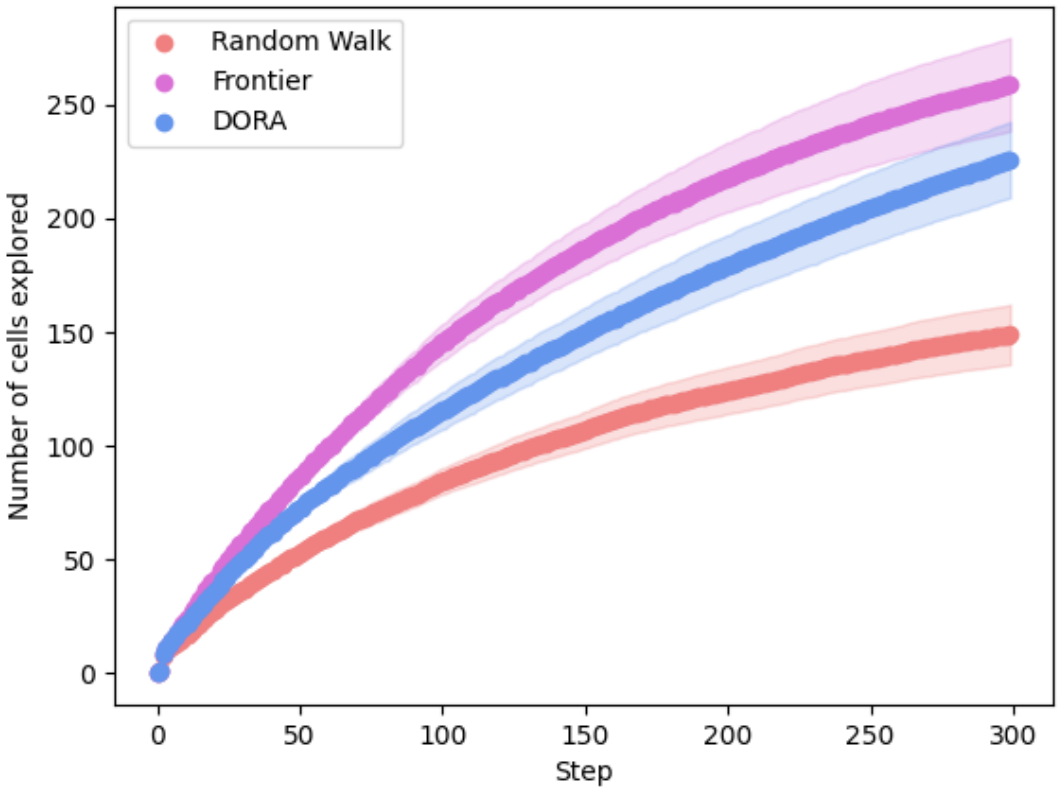

Figure 4.8 DORA cell exploration performance (N=10 robots)

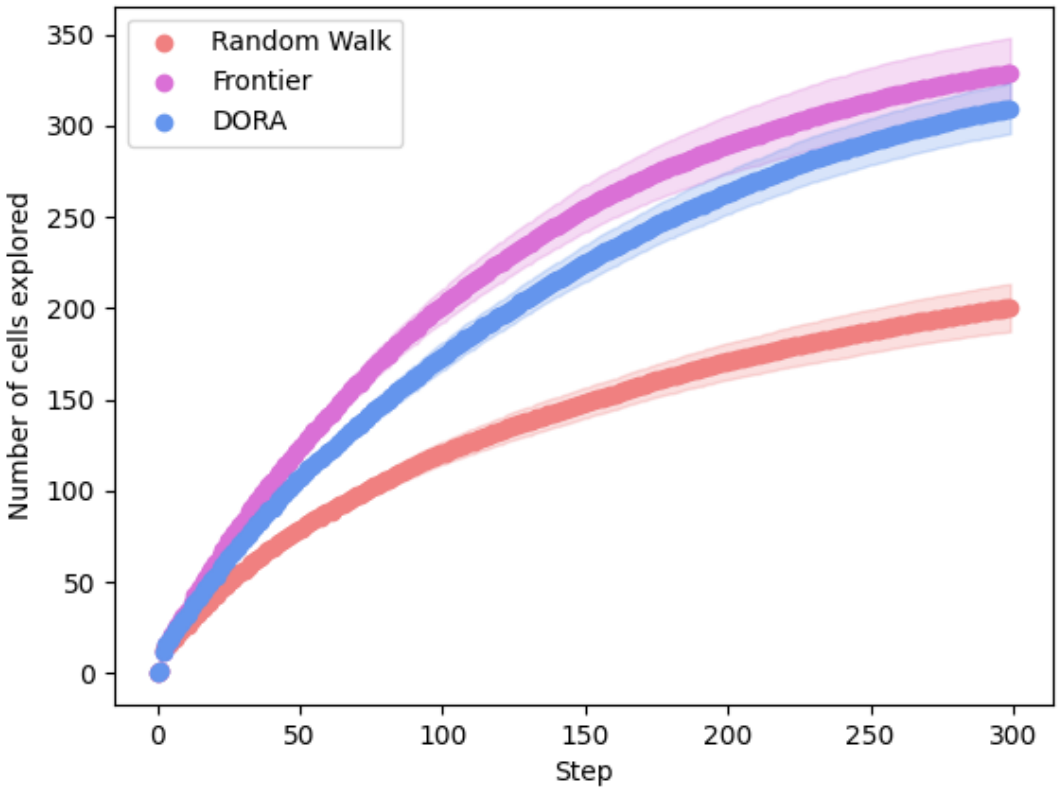

Figure 4.9 DORA cell exploration performance (N=15 robots)

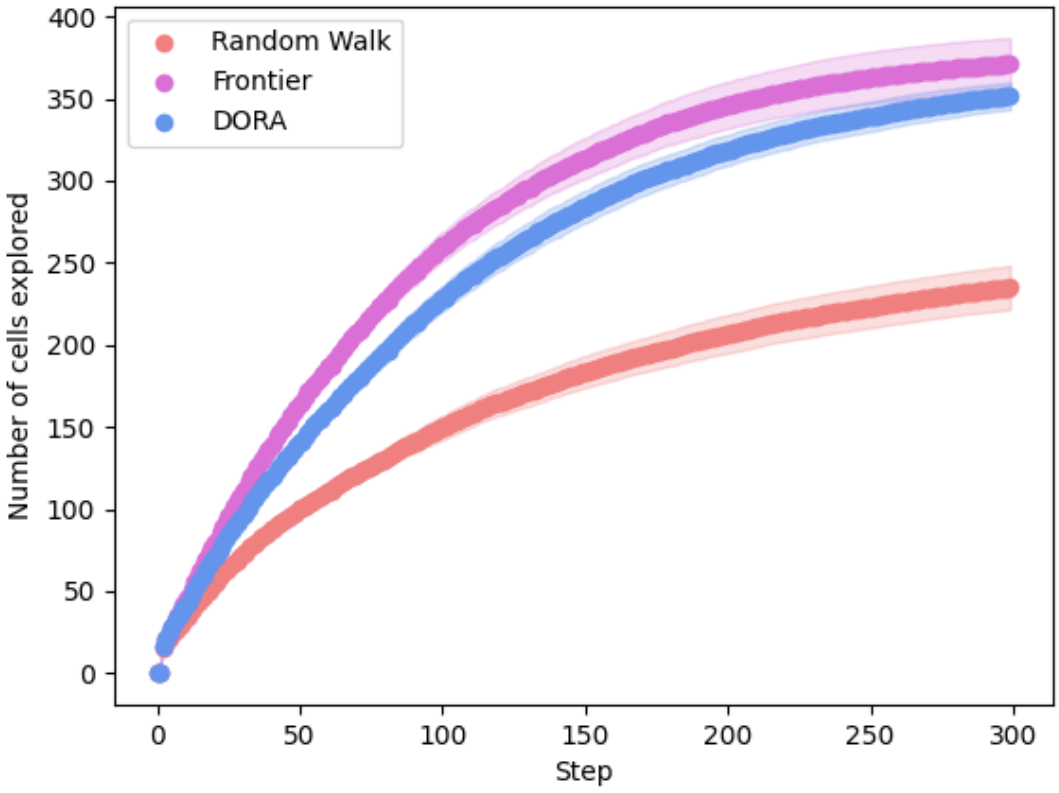

Figure 4.10 DORA cell exploration performance (N=20 robots)

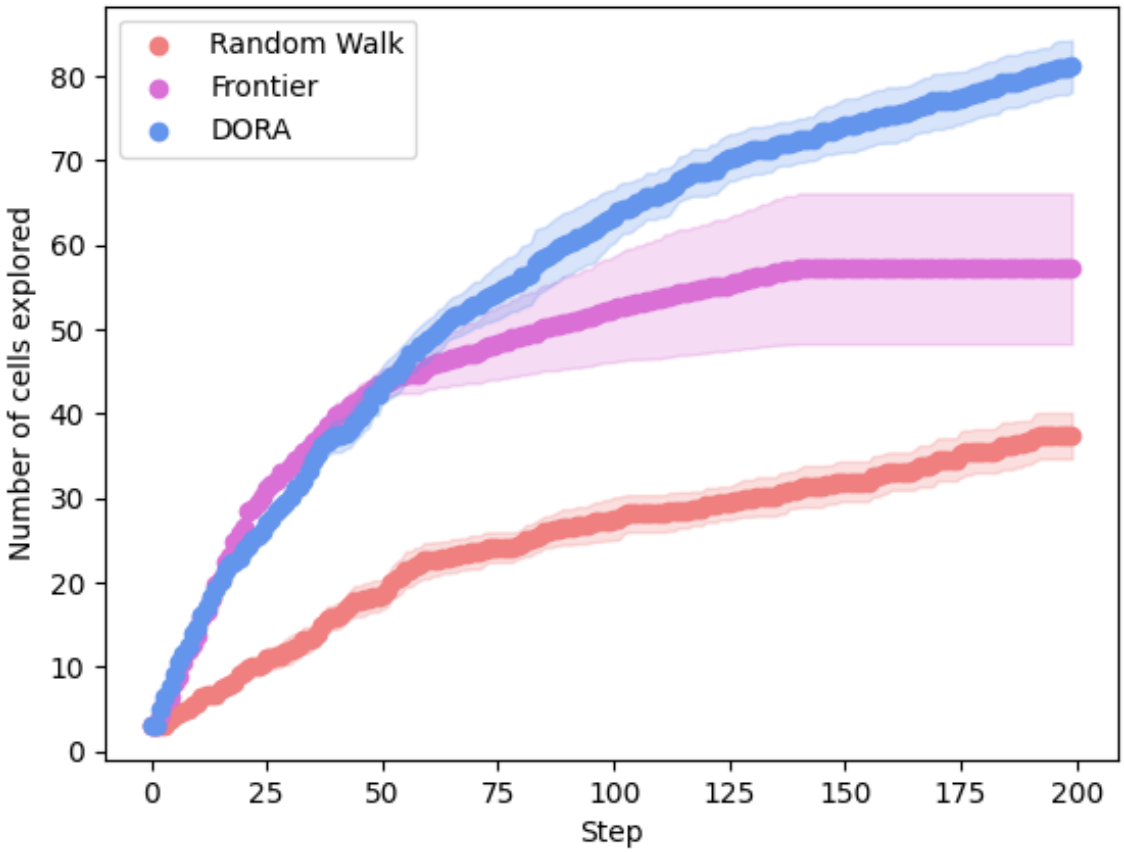

Figure 4.11 DORA cell exploration performance (N=5 physical robots)

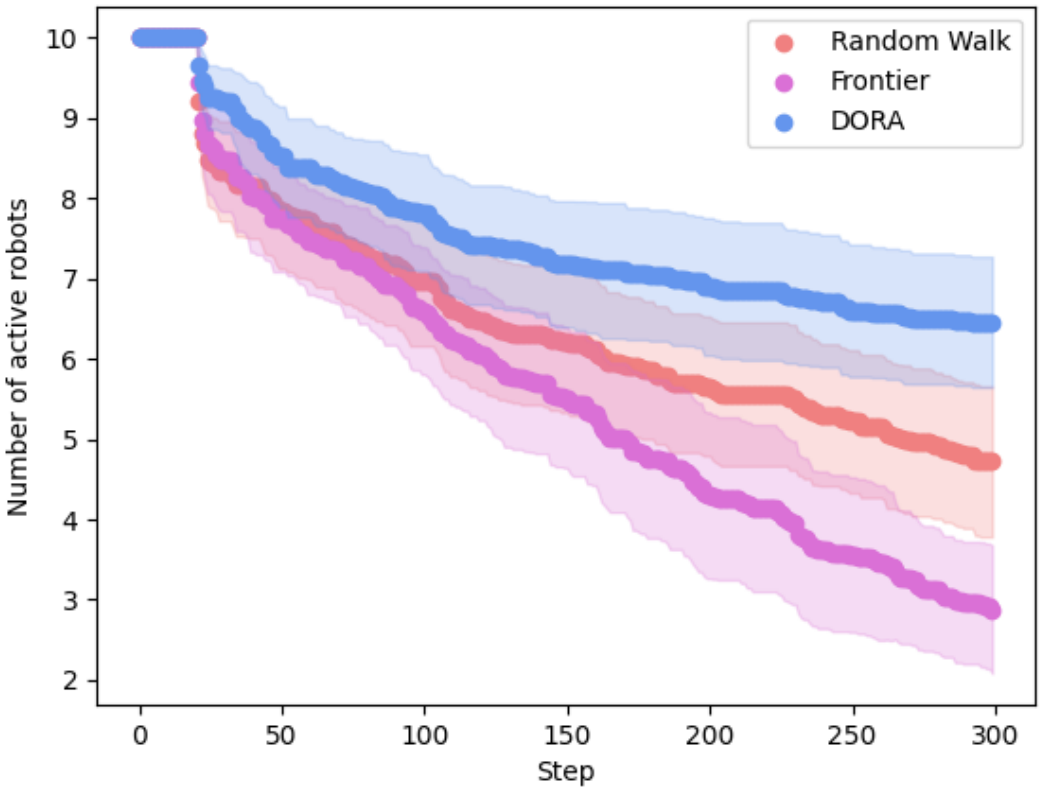

Figure 4.12 DORA survival performance (N=10 robots)

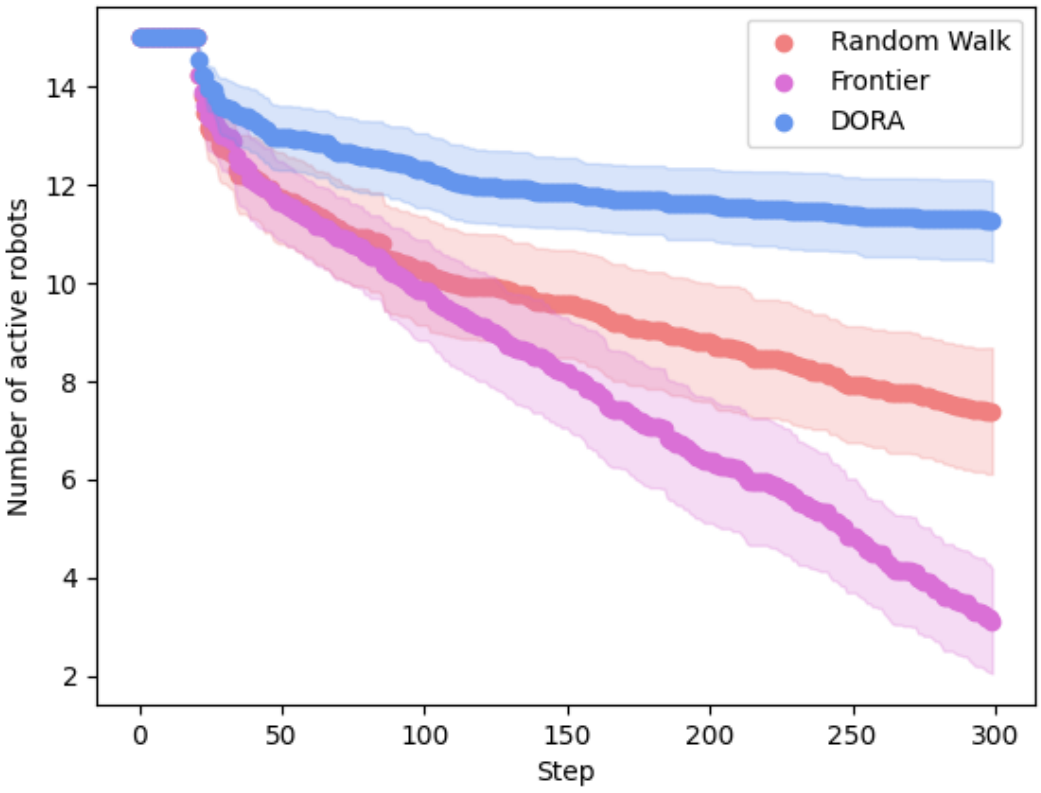

Figure 4.13 DORA survival performance (N=15 robots)

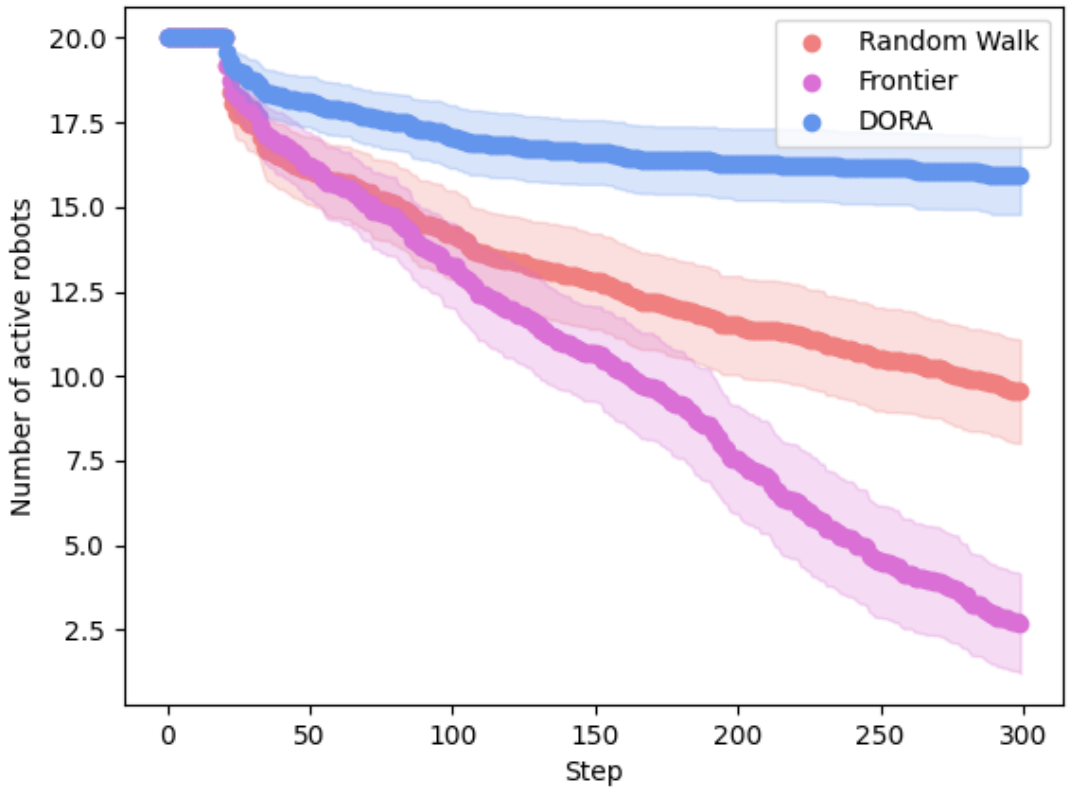

Figure 4.14 DORA survival performance (N=20 robots)

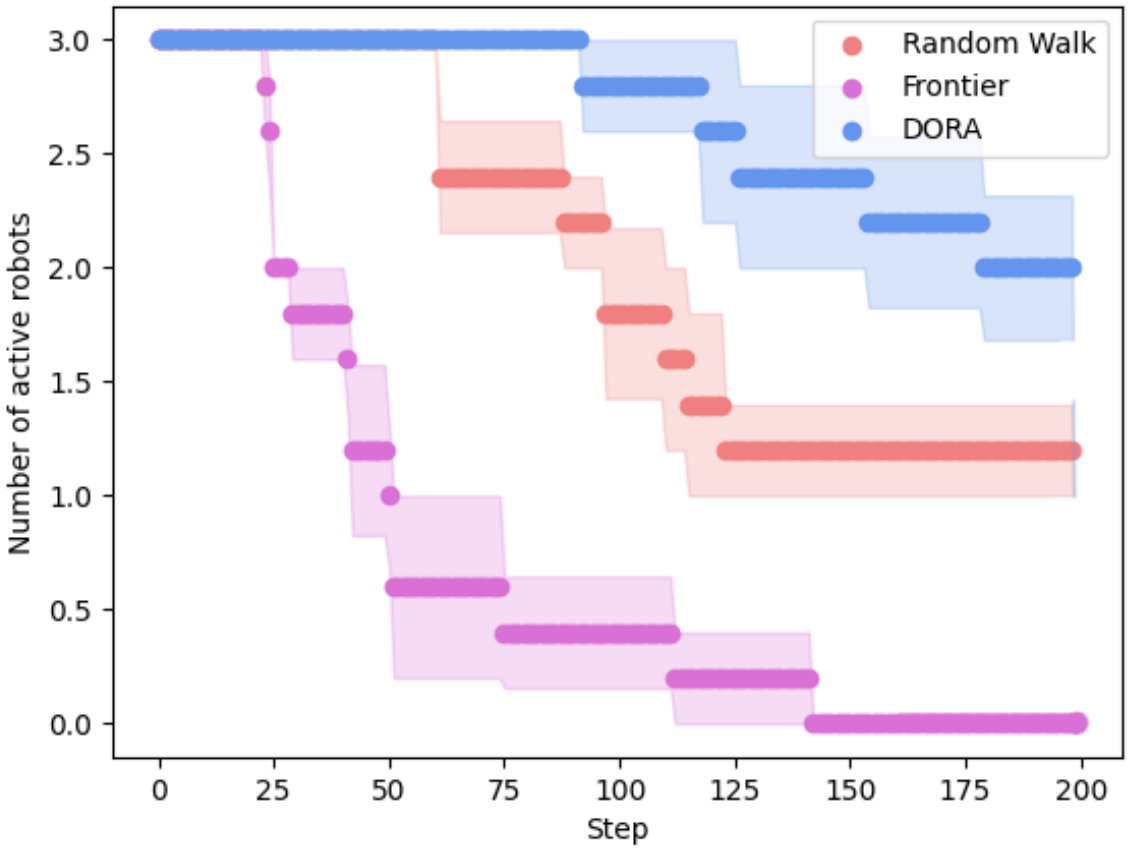

Figure 4.15 DORA survival performance (N=5 physical robots)

results in a slightly lower exploration coverage, but in a much lower failure rate.

In Fig. 4.19, we can see the amount of data exchanged by robots in the simulations conducted with DORA and FBE. The takeaway is that DORA is outperformed by FBE for this metric, and that communication costs are aligned with the theoretical values from 4.2. This difference in bandwidth usage is explained by the fact that our implementation of FBE uses one stigmergy, while DORA uses two. This metric was not calculated for the random walk algorithm as it does not require any coordination nor communication. It was not measured in physical experiments.

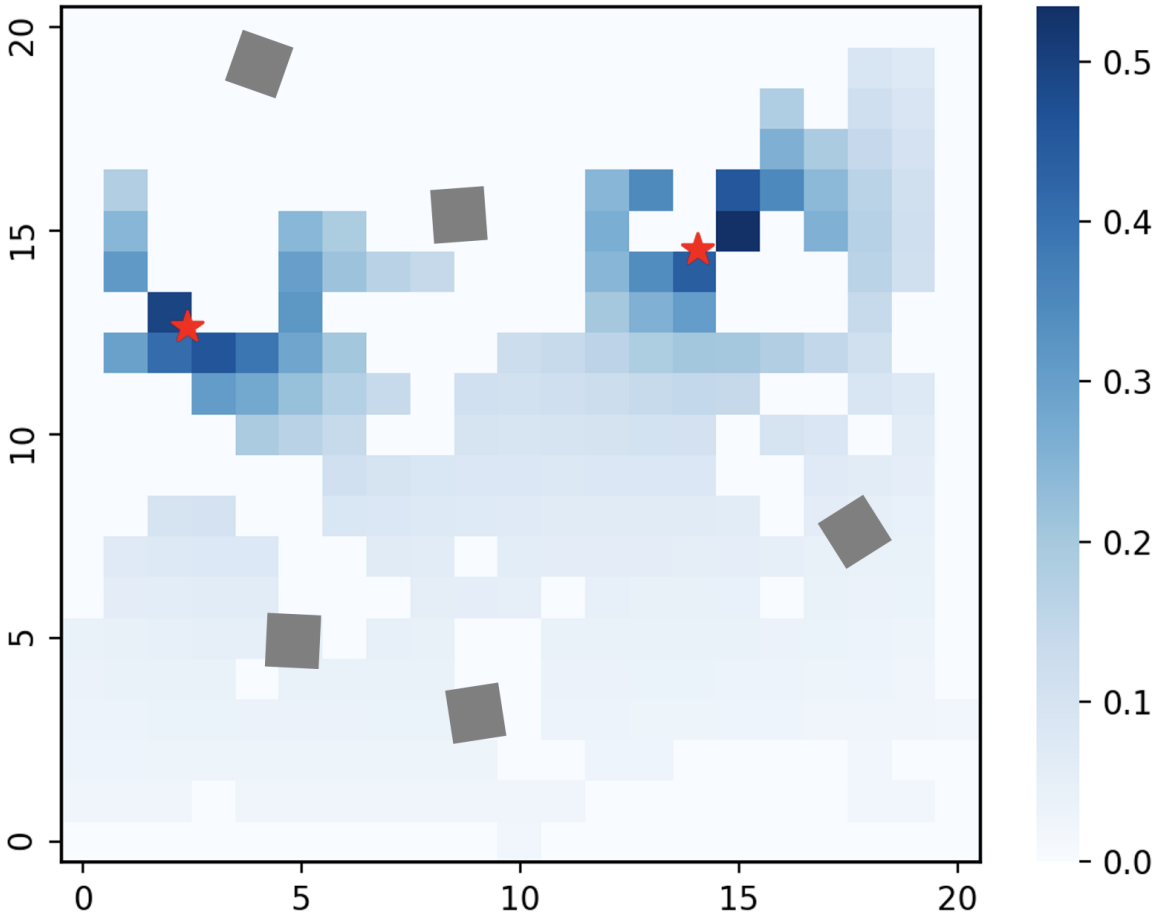

Figure 4.16 Random walk heatmap

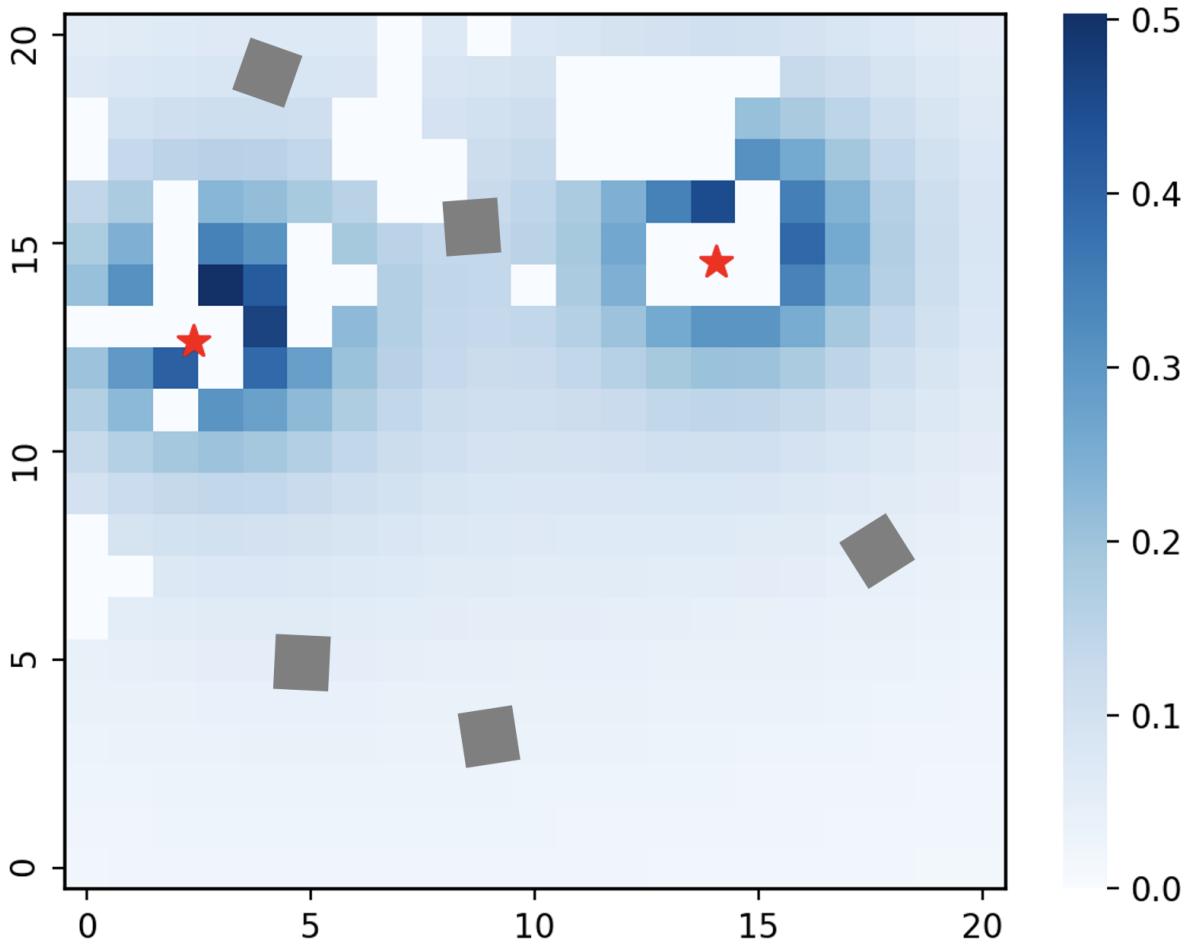

Figure 4.17 FBE heatmap

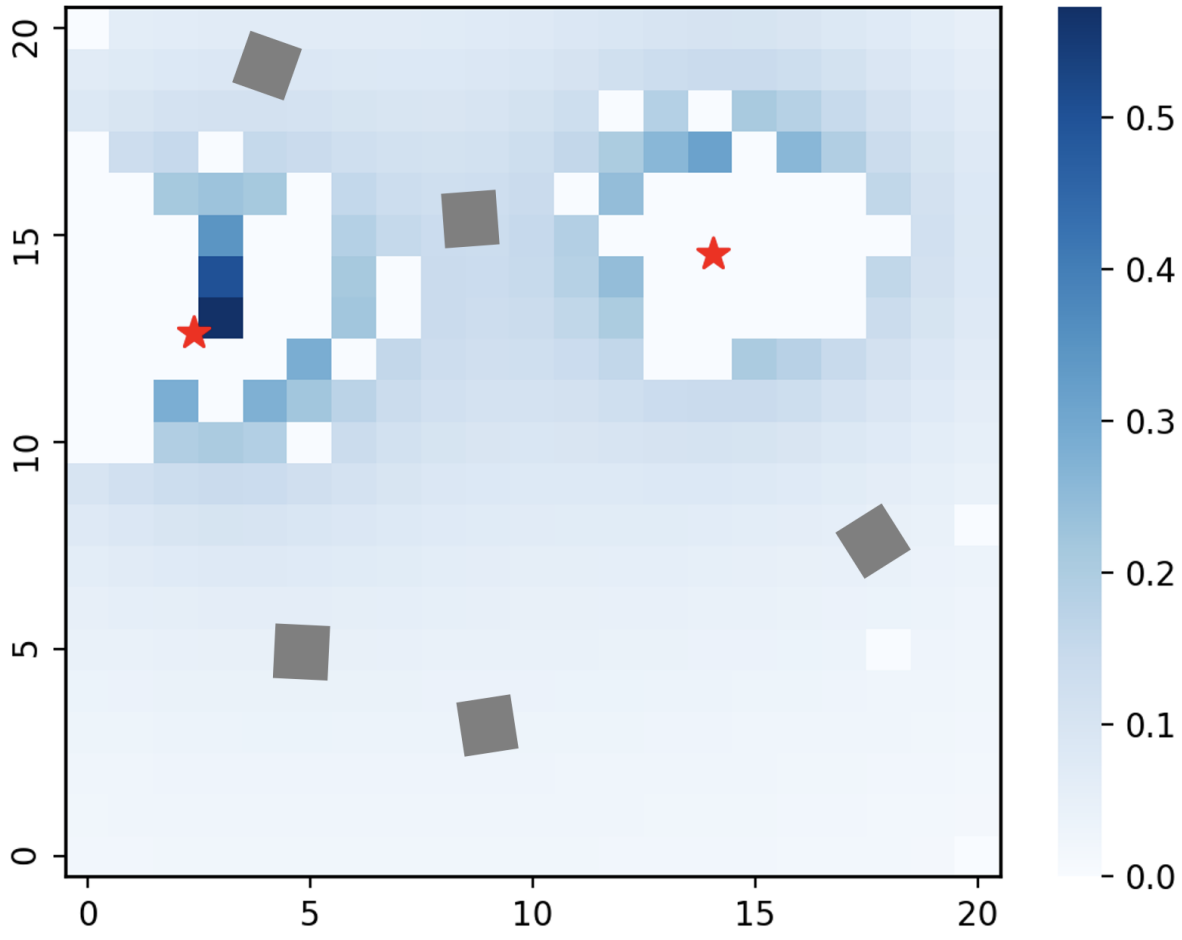

Figure 4.18 DORA heatmap

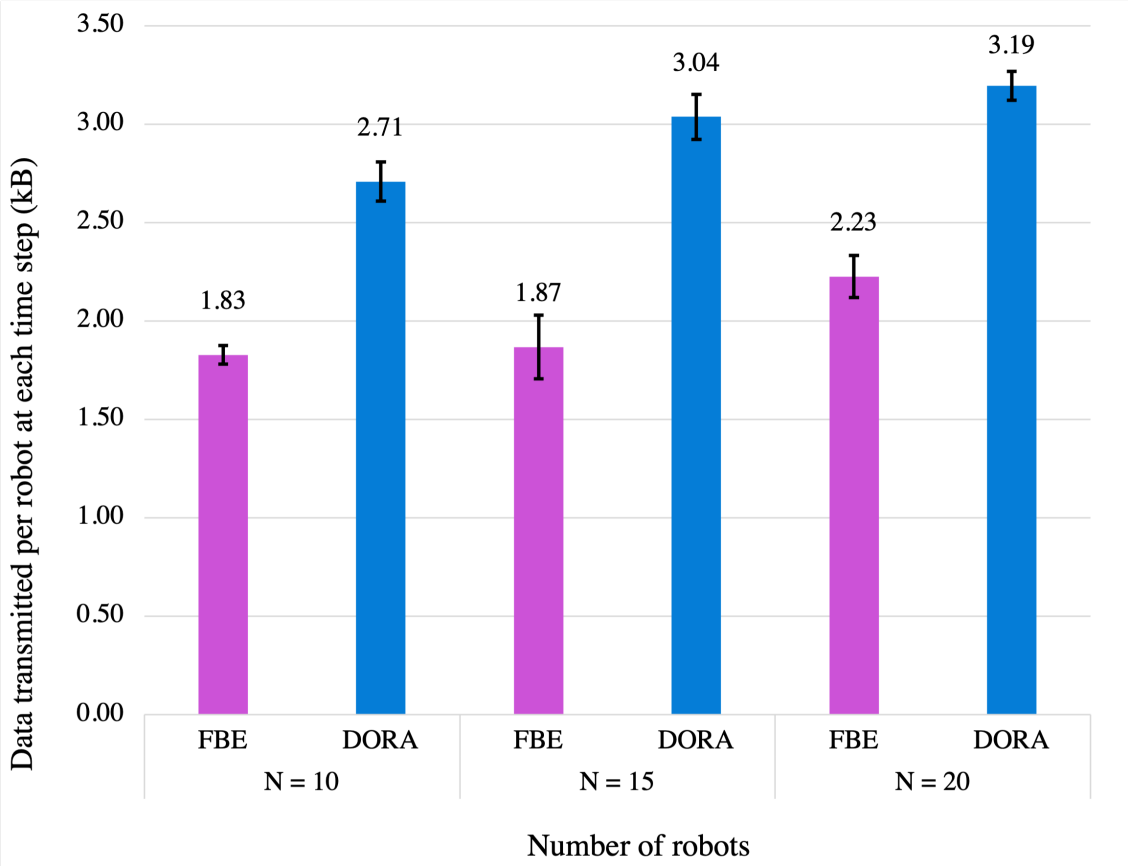

Figure 4.19 Communication costs for DORA and FBE

4.5 Conclusion

DORA Explorer is a lightweight risk-aware exploration algorithm. It minimizes the risk to which robots expose themselves to allow them to maximize the amount of explored terrain. By leveraging DBMs it outperforms non-coordinated solutions. Indeed, it succeeds in considerably reducing the likeliness of robot failures while keeping similar ground coverage performance compared to other solutions proposed in the literature. DORA also showed good scalability thanks to its low communication costs. It also showed applicability to real world scenarios through experiments with physical robots.

In future work, it could be interesting to modify DORA to allow time-varying risk-tolerance thresholds. Additionally, this work assumed that the risk associated with the environment can be measured by the robots' sensors, which is not always the case. To account for this issue, risk belief maps could be constructed using the previous failures of the agents to infer the presence of danger.

CHAPTER 5 CONCLUSION

The objective of the research conducted for this thesis was to design risk-aware algorithms to improve the resiliency of swarm robotics systems. RASS and DORA are both meant to be used as part of other systems, and thus to be stepping stones towards trustworthy industrial applications.

5.1 Summary of Works

Through a thorough examination of the state-of-the art in the field of swarm robotics, this work established the problems that needed to be solved by the design of novel risk-aware systems. Some, like decentralized storage efficiency and swarm exploration performance, were definitely application-specific. The rest were more generic and included failure resilience, scalability and adaptability to limited resources. In all cases, these objectives were met and confirmed by the results obtained through virtual and physical experiments. Knowing these objectives were accomplished the impact of this work in the field of swarm robotics should be highlighted. The main contribution is that because the systems are computationally simple, efficient and aligned with swarm robotics paradigms, they can be easily used as a part of the foundation of complex systems. For example, a distributed surveillance platform could use DORA to cover its necessary ground and RASS to safely store its recorded feed. By providing these risk-aware methods, this thesis can help to abstract away some of work from swarm application developers. Furthermore, this thesis has shown that risk-aware methodologies are not only compatible and capable of integration with existing state-of-the-art methods of the domain, but can also improve these systems by making them more resilient.

5.2 Limitations

Because of the robot-simulation gap problem, further experiments would be beneficial. Indeed, physical scalability, in both RASS and DORA, has not been extensively tested. This is due to a limited number of robots available for each experiment. In RASS's case, we could only use 5 CogniFlies; for DORA, we only had access to 5 KheperaIV. Thus, conducting experiments with more robots could increase confidence in the results obtained for both systems. The effects of varying parameter values in DORA were not studied; experiments performed in this regard could lead to performance improvements. The experiments could have benefited from the use of more varied benchmark algorithms. The effects of commu-

nication issues on routing and exploration performance could have been studied in further experiments with [107]. Another limitation of this work is that topologies with more than one connected component were not considered. Yet, unless systems are designed to prevent it, these situations might arise in swarm robotics applications were subgroups are likely to wander and get disconnected from the rest of the swarm. Studying the effect of these disconnections could provide more insight. These limitations will be addressed in upcoming improvements on the articles.

5.3 Future Research

Adapting solutions to heterogeneous swarms with different resistance to risk could prove an interesting challenge and would show the adaptability of both RASS and DORA to more diverse scenarios in which swarms could retain their group performance advantage derived from heterogeneity [108]. Testing in more challenging environments (rugged terrain, terrains with hidden lines of sights, dynamic risk conditions, etc.). Further improvements could be integrated to the system, such as improvements to security by using techniques like sinkhole detection proposed by [100] in the context of routing. Going forward, the most interesting path for RASS and DORA would be to integrate them with larger systems to see what benefits they can bring to complex swarm robotics applications.

REFERENCES

[1] M. A. Lones, "Metaheuristics in nature-inspired algorithms," in *Proceedings of the Companion Publication of the 2014 Annual Conference on Genetic and Evolutionary Computation*, 2014, pp. 1419–1422. [Online]. Available: https://doi.org/10.1145/2598394.2609841

[2] M. A. Lewis and G. A. Bekey, "The behavioral self-organization of nanorobots using local rules." in *IROS*, 1992, pp. 1333–1338. [Online]. Available: http://dx.doi.org/10.1109/IROS.1992.594558

[3] M. M. al Rifaie and A. Aber, "Identifying metastasis in bone scans with stochastic diffusion search," in *2012 International Symposium on Information Technologies in Medicine and Education*, vol. 1. IEEE, 2012, pp. 519–523. [Online]. Available: https://doi.org/10.1109/ITiME.2012.6291355

[4] D. Martens, B. Baesens, and T. Fawcett, "Editorial survey: swarm intelligence for data mining," *Machine Learning*, vol. 82, no. 1, pp. 1–42, 2011. [Online]. Available: https://doi.org/10.1007/s10994-010-5216-5

[5] A. Prorok, M. Malencia, L. Carlone, G. S. Sukhatme, B. M. Sadler, and V. Kumar, "Beyond robustness: A taxonomy of approaches towards resilient multi-robot systems," *arXiv preprint arXiv:2109.12343*, 2021. [Online]. Available: https://arxiv.org/abs/2109.12343

[6] M. Dorigo, G. Theraulaz, and V. Trianni, "Swarm robotics: Past, present, and future [point of view]," *Proceedings of the IEEE*, vol. 109, no. 7, pp. 1152–1165, 2021. [Online]. Available: https://hal.archives-ouvertes.fr/hal-03362874/

[7] A. Colorni, M. Dorigo, V. Maniezzo *et al.*, "Distributed optimization by ant colonies," in *Proceedings of the first European conference on artificial life*, vol. 142. Paris, France, 1991, pp. 134–142. [Online]. Available: https://www-public.imtbs-tsp.eu/~gibson/Teaching/Teaching-ReadingMaterial/ColorniDorigoManiezzo91.pdf

[8] D. T. Pham, A. Ghanbarzadeh, E. Koç, S. Otri, S. Rahim, and M. Zaidi, "The bees algorithm—a novel tool for complex optimisation problems," in *Intelligent production machines and systems*. Elsevier, 2006, pp. 454–459. [Online]. Available: https://doi.org/10.1016/B978-008045157-2/50081-X

[9] D. Karaboga, "Artificial bee colony algorithm," *scholarpedia*, vol. 5, no. 3, p. 6915, 2010. [Online]. Available: http://dx.doi.org/10.4249/scholarpedia.6915

[10] H. Hamann and H. Wörn, "A framework of space–time continuous models for algorithm design in swarm robotics," *Swarm Intelligence*, vol. 2, no. 2, pp. 209–239, 2008. [Online]. Available: https://doi.org/10.1007/s11721-008-0015-3

[11] T. Vicsek and A. Zafeiris, "Collective motion," *Physics reports*, vol. 517, no. 3-4, pp. 71–140, 2012. [Online]. Available: https://doi.org/10.1016/j.physrep.2012.03.004

[12] M. Coppola, J. Guo, E. Gill, and G. C. de Croon, "Provable self-organizing pattern formation by a swarm of robots with limited knowledge," *Swarm Intelligence*, vol. 13, no. 1, pp. 59–94, 2019. [Online]. Available: https://doi.org/10.1007/s11721-019-00163-0

[13] C. Zhang, P. Dhungel, D. Wu, and K. W. Ross, "Unraveling the bittorrent ecosystem," *IEEE transactions on parallel and distributed systems*, vol. 22, no. 7, pp. 1164–1177, 2010. [Online]. Available: https://doi.org/10.1109/TPDS.2010.123

[14] R. Brooks, "A robust layered control system for a mobile robot," *IEEE journal on robotics and automation*, vol. 2, no. 1, pp. 14–23, 1986. [Online]. Available: https://doi.org/10.1109/JRA.1986.1087032

[15] K. Lim and M. Eslami, "Robust adaptive controller designs for robot manipulator systems," *IEEE Journal on robotics and automation*, vol. 3, no. 1, pp. 54–66, 1987. [Online]. Available: https://doi.org/10.1109/JRA.1987.1087070

[16] J.-J. E. Slotine, "The robust control of robot manipulators," *The International Journal of Robotics Research*, vol. 4, no. 2, pp. 49–64, 1985. [Online]. Available: https://doi.org/10.1177/027836498500400205

[17] J.-J. E. Slotine, W. Li *et al.*, *Applied nonlinear control*. Prentice hall Englewood Cliffs, NJ, 1991, vol. 199, no. 1.

[18] R. K. Ramachandran, J. A. Preiss, and G. S. Sukhatme, "Resilience by reconfiguration: Exploiting heterogeneity in robot teams," *arXiv preprint arXiv:1903.04856*, 2019. [Online]. Available: https://arxiv.org/abs/1903.04856

[19] R. Wehbe and R. K. Williams, "Probabilistic resilience of dynamic multi-robot systems," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1777–1784, 2021. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9357930

[20] A. F. Winfield and J. Nembrini, "Safety in numbers: fault-tolerance in robot swarms," *International Journal of Modelling, Identification and Control*, vol. 1, no. 1, pp. 30–37, 2006. [Online]. Available: https://www.inderscienceonline.com/doi/abs/10.1504/IJMIC.2006.008645

[21] M. Rubenstein, C. Ahler, and R. Nagpal, "Kilobot: A low cost scalable robot system for collective behaviors," in *2012 IEEE international conference on robotics and automation*. IEEE, 2012, pp. 3293–3298. [Online]. Available: https://doi.org/10.1109/ICRA.2012.6224638

[22] M. Rubenstein, A. Cornejo, and R. Nagpal, "Programmable self-assembly in a thousand-robot swarm," *Science*, vol. 345, no. 6198, pp. 795–799, 2014. [Online]. Available: https://doi.org/10.1126/science.1254295

[23] N. Fontbonne, O. Dauchot, and N. Bredeche, "Distributed on-line learning in swarm robotics with limited communication bandwidth," in *2020 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2020, pp. 1–8. [Online]. Available: https://doi.org/10.1109/CEC48606.2020.9185697

[24] G.-Z. Yang, J. Bellingham, P. E. Dupont, P. Fischer, L. Floridi, R. Full, N. Jacobstein, V. Kumar, M. McNutt, R. Merrifield *et al.*, "The grand challenges of science robotics," *Science robotics*, vol. 3, no. 14, p. eaar7650, 2018. [Online]. Available: https://doi.org/10.1126/scirobotics.aar7650

[25] N. Jakobi, P. Husbands, and I. Harvey, "Noise and the reality gap: The use of simulation in evolutionary robotics," in *European Conference on Artificial Life*. Springer, 1995, pp. 704–720. [Online]. Available: https://link.springer.com/chapter/10.1007/3-540-59496-5_337

[26] G. Francesca and M. Birattari, "Automatic design of robot swarms: achievements and challenges," *Frontiers in Robotics and AI*, vol. 3, p. 29, 2016. [Online]. Available: https://doi.org/10.3389/frobt.2016.00029

[27] M. Dorigo, G. Theraulaz, and V. Trianni, "Reflections on the future of swarm robotics," *Science Robotics*, vol. 5, no. 49, 2020. [Online]. Available: https://crca.cbi-toulouse.fr/wp-content/uploads/2020/12/121.pdf

[28] A. Renzaglia, L. Doitsidis, A. Martinelli, and E. B. Kosmatopoulos, "Multi-robot three-dimensional coverage of unknown areas," *The International Journal*

of Robotics Research, vol. 31, no. 6, pp. 738–752, 2012. [Online]. Available: https://doi.org/10.1177%2F0278364912439332

[29] K. N. McGuire, C. D. Wagter, K. Tuyls, H. J. Kappen, and G. C. H. E. de Croon, "Minimal navigation solution for a swarm of tiny flying robots to explore an unknown environment," *Science Robotics*, vol. 4, no. 35, Oct. 2019.

[30] D. T. Davis, T. H. Chung, M. R. Clement, and M. A. Day, "Consensus-based data sharing for large-scale aerial swarm coordination in lossy communications environments," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 3801–3808. [Online]. Available: https://doi.org/10.1109/IROS.2016.7759559

[31] A. Dutta, A. Ghosh, S. Sisley, and O. P. Kreidl, "Efficient communication in large multi-robot networks," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 6647–6653. [Online]. Available: https://doi.org/10.1109/ICRA40945.2020.9196672

[32] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization," *IEEE computational intelligence magazine*, vol. 1, no. 4, pp. 28–39, 2006. [Online]. Available: https://doi.org/10.1109/MCI.2006.329691

[33] C. Pinciroli, A. Lee-Brown, and G. Beltrame, "A tuple space for data sharing in robot swarms," in *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS)*, 2016, pp. 287–294. [Online]. Available: https://carlo.pinciroli.net/pdf/Pinciroli:BICT2015.pdf

[34] W. Vogels, "Eventually consistent," *Communications of the ACM*, vol. 52, no. 1, pp. 40–44, 2009. [Online]. Available: https://doi.org/10.1145/1435417.1435432

[35] N. Majcherczyk and C. Pinciroli, "Swarmmesh: A distributed data structure for cooperative multi-robot applications," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 4059–4065. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9197403

[36] N. Majcherczyk, D. J. Nallathambi, T. Antonelli, and C. Pinciroli, "Distributed data storage and fusion for collective perception in resource-limited mobile robot swarms," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5549–5556, 2021. [Online]. Available: https://doi.org/10.1109/LRA.2021.3076962

[37] V. S. Varadharajan, D. St-Onge, B. Adams, and G. Beltrame, "Soul: data sharing for robot swarms," *Autonomous Robots*, vol. 44, no. 3, pp. 377–394, 2020. [Online]. Available: https://doi.org/10.1007/s10514-019-09855-2

[38] F. Amigoni, J. Banfi, and N. Basilico, "Multirobot exploration of communication-restricted environments: A survey," *IEEE Intelligent Systems*, vol. 32, no. 6, pp. 48–57, 2017. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8267592

[39] W. Wu, Y. Chen, X. Zhang, X. Shi, L. Cong, B. Deng, and X. Li, "Ldht: Locality-aware distributed hash tables," in *2008 International Conference on Information Networking*. IEEE, 2008, pp. 1–5. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/4472811

[40] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, and S. Shenker, "Ght: a geographic hash table for data-centric storage," in *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, 2002, pp. 78–87. [Online]. Available: https://doi.org/10.1145/570738.570750

[41] J. P. Ahulló, P. G. Lopez, M. S. Artigas, and A. F. G. Skarmeta, "Supporting geographical queries onto dhts," in *2008 33rd IEEE Conference on Local Computer Networks (LCN)*. IEEE, 2008, pp. 435–442. [Online]. Available: https://doi.org/10.1109/LCN.2008.4664201

[42] F. Araújo, L. Rodrigues, J. Kaiser, C. Liu, and C. Mitidieri, "Chr: a distributed hash table for wireless ad hoc networks," in *25th IEEE international conference on distributed computing systems workshops*. IEEE, 2005, pp. 407–413. [Online]. Available: https://doi.org/10.1109/ICDCSW.2005.48

[43] E. W. Dijkstra *et al.*, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[44] R. Draves, J. Padhye, and B. Zill, "Comparison of routing metrics for static multi-hop wireless networks," *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 4, pp. 133–144, 2004. [Online]. Available: https://doi.org/10.1145/1030194.1015483

[45] T. Watteyne, K. Pister, D. Barthel, M. Dohler, and I. Auge-Blum, "Implementation of gradient routing in wireless sensor networks," in *GLOBECOM 2009-2009 IEEE Global Telecommunications Conference*. IEEE, 2009, pp. 1–6. [Online]. Available: https://doi.org/10.1109/GLOCOM.2009.5425543

[46] J. Kuruvila, A. Nayak, and I. Stojmenovic, "Hop count optimal position-based packet routing algorithms for ad hoc wireless networks with a realistic physical layer," *IEEE Journal on selected areas in communications*, vol. 23, no. 6, pp. 1267–1275, 2005. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/1435519

[47] X. Zhang, Z.-H. Qian, Y.-Q. Guo, and X. Wang, "An efficient hop count routing protocol for wireless ad hoc networks," *International Journal of automation and computing*, vol. 11, no. 1, pp. 93–99, 2014. [Online]. Available: https://link.springer.com/content/pdf/10.1007/s11633-014-0770-0.pdf

[48] F. Al-Salti, N. Alzeidi, K. Day, and A. Touzene, "An efficient and reliable grid-based routing protocol for uwsns by exploiting minimum hop count," *Computer Networks*, vol. 162, p. 106869, 2019. [Online]. Available: https://doi.org/10.1016/j.comnet.2019.106869

[49] K. Li, C. E. Torres, K. Thomas, L. F. Rossi, and C.-C. Shen, "Slime mold inspired routing protocols for wireless sensor networks," *Swarm Intelligence*, vol. 5, no. 3, pp. 183–223, 2011. [Online]. Available: https://doi.org/10.1007/s11721-011-0063-y

[50] N. Jiang, Y. Cheng, J. Zhou, T. Zhou, W. Xu, and D. Xu, "Toward biology-inspired solutions for routing problems of wireless sensor networks with mobile sink," *Soft Computing*, vol. 22, no. 23, pp. 7847–7855, 2018. [Online]. Available: https://doi.org/10.1007/s00500-018-3506-1

[51] A. Jiang and L. Zheng, "An effective hybrid routing algorithm in wsn: Ant colony optimization in combination with hop count minimization," *sensors*, vol. 18, no. 4, p. 1020, 2018. [Online]. Available: https://doi.org/10.3390/s18041020

[52] W.-H. Liao, Y. Kao, and C.-M. Fan, "Data aggregation in wireless sensor networks using ant colony algorithm," *Journal of Network and Computer Applications*, vol. 31, no. 4, pp. 387–401, 2008. [Online]. Available: https://doi.org/10.1016/j.jnca.2008.02.006

[53] E. Tolstaya, L. Butler, D. Mox, J. Paulos, V. Kumar, and A. Ribeiro, "Learning connectivity for data distribution in robot teams," *arXiv preprint arXiv:2103.05091*, 2021. [Online]. Available: https://arxiv.org/abs/2103.05091

[54] D. Ganesan, B. Krishnamachari, A. Woo, D. Culler, D. Estrin, and S. Wicker, "An empirical study of epidemic algorithms in large scale multihop wireless networks," Technical Report IRB-TR-02-003, Intel Research, Tech. Rep., 2002.

[55] J. W. Hui and D. Culler, "The dynamic behavior of a data dissemination protocol for network programming at scale," in *Proceedings of the 2nd international conference on Embedded networked sensor systems*, 2004, pp. 81–94. [Online]. Available: https://doi.org/10.1145/1031495.1031506

[56] G. Dhand and S. Tyagi, "Data aggregation techniques in wsn: Survey," *Procedia Computer Science*, vol. 92, pp. 378–384, 2016. [Online]. Available: https://doi.org/10.1016/j.procs.2016.07.393

[57] S. I. Roumeliotis, G. S. Sukhatme, and G. A. Bekey, "Sensor fault detection and identification in a mobile robot," in *Proceedings. 1998 IEEE/RSJ International Conference on Intelligent Robots and Systems. Innovations in Theory, Practice and Applications (Cat. No. 98CH36190)*, vol. 3. IEEE, 1998, pp. 1383–1388. [Online]. Available: https://doi.org/10.1109/IROS.1998.724781

[58] R. E. Kalman, "A new approach to linear filtering and prediction problems," 1960. [Online]. Available: https://doi.org/10.1115/1.3662552

[59] U. Lerner, R. Parr, D. Koller, G. Biswas *et al.*, "Bayesian fault detection and diagnosis in dynamic systems," in *Aaai/iaai*, 2000, pp. 531–537. [Online]. Available: https://www.aaai.org/Papers/AAAI/2000/AAAI00-081.pdf

[60] P. Li and V. Kadirkamanathan, "Particle filtering based likelihood ratio approach to fault diagnosis in nonlinear stochastic systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 31, no. 3, pp. 337–343, 2001. [Online]. Available: https://doi.org/10.1109/5326.971661

[61] A. L. Christensen, R. OGrady, and M. Dorigo, "From fireflies to fault-tolerant swarms of robots," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 4, pp. 754–766, 2009. [Online]. Available: https://doi.org/10.1109/TEVC.2009.2017516

[62] D. Tarapore, A. L. Christensen, and J. Timmis, "Generic, scalable and decentralized fault detection for robot swarms," *PloS one*, vol. 12, no. 8, p. e0182058, 2017. [Online]. Available: https://doi.org/10.1371/journal.pone.0182058

[63] X. Xiao, J. Dufek, and R. R. Murphy, "Robot risk-awareness by formal risk reasoning and planning," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2856–2863, 2020. [Online]. Available: https://doi.org/10.1109/LRA.2020.2974434

[64] L. De Filippis, G. Guglieri, and F. Quagliotti, "A minimum risk approach for path planning of uavs," *Journal of Intelligent & Robotic Systems*, vol. 61, no. 1, pp. 203–219, 2011. [Online]. Available: https://doi.org/10.1007/s10846-010-9493-9

[65] E. R. Hunt, G. Jenkinson, M. Wilsher, C. P. Dettmann, and S. Hauert, "Spider: a bioinspired swarm algorithm for adaptive risk-taking," in *Artificial Life Conference Proceedings*. MIT Press One Rogers Street, Cambridge, MA 02142-1209 USA journals-info@ mit ..., 2020, pp. 44–51. [Online]. Available: https://doi.org/10.1162/isal_a_00279

[66] A. Soltani and T. Fernando, "A fuzzy based multi-objective path planning of construction sites," *Automation in construction*, vol. 13, no. 6, pp. 717–734, 2004. [Online]. Available: https://doi.org/10.1016/j.autcon.2004.04.012

[67] M. Ono and B. C. Williams, "An efficient motion planning algorithm for stochastic dynamic systems with constraints on probability of failure." in *AAAI*, 2008, pp. 1376–1382. [Online]. Available: http://hdl.handle.net/1721.1/40803

[68] M. P. Vitus and C. J. Tomlin, "On feedback design and risk allocation in chance constrained control," in *2011 50th IEEE Conference on Decision and Control and European Control Conference*. IEEE, 2011, pp. 734–739. [Online]. Available: https://doi.org/10.1109/CDC.2011.6160721

[69] E. R. Hunt, S. Jones, and S. Hauert, "Testing the limits of pheromone stigmergy in high-density robot swarms," *Royal Society open science*, vol. 6, no. 11, p. 190225, 2019. [Online]. Available: https://doi.org/10.1098/rsos.190225

[70] A. Undurti and J. P. How, "An online algorithm for constrained pomdps," in *2010 IEEE International Conference on Robotics and Automation*. IEEE, 2010, pp. 3966–3973. [Online]. Available: https://doi.org/10.1109/ROBOT.2010.5509743

[71] S. Thiébaux, B. Williams *et al.*, "Rao*: An algorithm for chance-constrained pomdp's," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 30, no. 1, 2016. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/10423

[72] W. Luo and K. Sycara, "Voronoi-based coverage control with connectivity maintenance for robotic sensor networks," in *2019 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*. IEEE, 2019, pp. 148–154. [Online]. Available: https://ieeexplore.ieee.org/document/8901078

[73] M. Santos, S. Mayya, G. Notomista, and M. Egerstedt, "Decentralized minimum-energy coverage control for time-varying density functions," in *2019 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*. IEEE, 2019, pp. 155–161. [Online]. Available: https://ieeexplore.ieee.org/document/8901076

[74] X. Xu and Y. Diaz-Mercado, "Multi-robot control using coverage over time-varying domains," in *2019 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*. IEEE, 2019, pp. 179–181. [Online]. Available: https://ieeexplore.ieee.org/document/8901067

[75] B. Yamauchi, "Frontier-based exploration using multiple robots," in *Proceedings of the second international conference on Autonomous agents*, 1998, pp. 47–53. [Online]. Available: https://dl.acm.org/doi/abs/10.1145/280765.280773

[76] Y. Wang, A. Liang, and H. Guan, "Frontier-based multi-robot map exploration using particle swarm optimization," in *2011 IEEE symposium on Swarm intelligence*. IEEE, 2011, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/5952584

[77] A. Topiwala, P. Inani, and A. Kathpal, "Frontier based exploration for autonomous robot," *arXiv preprint arXiv:1806.03581*, 2018. [Online]. Available: https://arxiv.org/abs/1806.03581

[78] P. Dames, M. Schwager, V. Kumar, and D. Rus, "A decentralized control policy for adaptive information gathering in hazardous environments," in *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*. IEEE, 2012, pp. 2807–2813. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6426239

[79] M. Schwager, P. Dames, D. Rus, and V. Kumar, "A Multi-robot Control Policy for Information Gathering in the Presence of Unknown Hazards," in *Robotics Research : The 15th International Symposium ISRR*, ser. Springer Tracts in Advanced Robotics, H. I. Christensen and O. Khatib, Eds. Cham: Springer International Publishing, 2017, pp. 455–472. [Online]. Available: https://doi.org/10.1007/978-3-319-29363-9_26

[80] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng *et al.*, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, no. 3.2. Kobe, Japan, 2009, p. 5.

[81] D. Pianini, M. Viroli, and J. Beal, "Protelis: Practical aggregate programming," in *Proceedings of the 30th Annual ACM Symposium on Applied Computing*, 2015, pp. 1846–1853. [Online]. Available: https://doi.org/10.1145/2695664.2695913

[82] L. Mottola, M. Moretta, K. Whitehouse, and C. Ghezzi, "Team-level programming of drone sensor networks," in *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems*, 2014, pp. 177–190. [Online]. Available: https://doi.org/10.1145/2668332.2668353

[83] C. Pinciroli and G. Beltrame, "Buzz: A programming language for robot swarms," *IEEE Software*, vol. 33, no. 4, pp. 97–100, 2016. [Online]. Available: https://doi.org/10.1109/MS.2016.95

[84] R. Ghosh, C. Hsieh, S. Misailovic, and S. Mitra, "Koord: a language for programming and verifying distributed robotics application," *Proceedings of the ACM on Programming Languages*, vol. 4, no. OOPSLA, pp. 1–30, 2020. [Online]. Available: https://dl.acm.org/doi/abs/10.1145/3428300

[85] C. Pinciroli, V. Trianni, R. O'Grady, G. Pini, A. Brutschy, M. Brambilla, N. Mathews, E. Ferrante, G. Di Caro, F. Ducatelle, M. Birattari, L. M. Gambardella, and M. Dorigo, "ARGoS: a modular, parallel, multi-engine simulator for multi-robot systems," *Swarm Intelligence*, vol. 6, no. 4, pp. 271–295, 2012.

[86] J.-S. Gutmann and K. Konolige, "Incremental mapping of large cyclic environments," in *Proceedings 1999 IEEE International Symposium on Computational Intelligence in Robotics and Automation. CIRA'99 (Cat. No. 99EX375)*. IEEE, 1999, pp. 318–325. [Online]. Available: https://doi.org/10.1109/CIRA.1999.810068

[87] D. Hahnel, D. Schulz, and W. Burgard, "Map building with mobile robots in populated environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 1. IEEE, 2002, pp. 496–501. [Online]. Available: https://doi.org/10.1109/IRDS.2002.1041439

[88] C. Stachniss and W. Burgard, "Mapping and exploration with mobile robots using coverage maps," in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*, vol. 1. Las Vegas, Nevada, USA: IEEE, 2003, pp. 467–472. [Online]. Available: https://doi.org/10.1109/IROS.2003.1250673

[89] F. Kobayashi, S. Sakai, and F. Kojima, "Sharing of exploring information using belief measure for multi robot exploration," in *2002 IEEE World Congress on Computational Intelligence. 2002 IEEE International Conference on Fuzzy Systems. FUZZ-IEEE'02. Proceedings (Cat. No.02CH37291)*, vol. 2, May 2002, pp. 1544–1549 vol.2. [Online]. Available: https://doi.org/10.1109/FUZZ.2002.1006736

[90] ——, "Determination of exploration target based on belief measure in multi-robot exploration," in *Proceedings 2003 IEEE International Symposium on Computational Intelligence in Robotics and Automation. Computational Intelligence in Robotics and Automation for the New Millennium (Cat. No.03EX694)*, vol. 3, Jul. 2003, pp. 1545–1550 vol.3. [Online]. Available: https://doi.org/10.1109/CIRA.2003.1222227

[91] V. Indelman, "Cooperative multi-robot belief space planning for autonomous navigation in unknown environments," *Autonomous Robots*, vol. 42, no. 2, pp. 353–373, Feb. 2018. [Online]. Available: https://doi.org/10.1007/s10514-017-9620-6

[92] W. Burgard, M. Moors, C. Stachniss, and F. E. Schneider, "Coordinated multi-robot exploration," *IEEE Transactions on robotics*, vol. 21, no. 3, pp. 376–386, 2005. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/1435481

[93] G. Kantor, S. Singh, R. Peterson, D. Rus, A. Das, V. Kumar, G. Pereira, and J. Spletzer, "Distributed search and rescue with robot and sensor teams," in *Field and Service Robotics*. Springer, 2003, pp. 529–538. [Online]. Available: https://doi.org/10.1007/10991459_51

[94] D. Vielfaure, S. Arseneault, P.-Y. Lajoie, and G. Beltrame, "Dora: Distributed online risk-aware explorer," 2021. [Online]. Available: https://arxiv.org/abs/2109.14551

[95] R. Sharp and M. Decreton, "Radiation tolerance of components and materials in nuclear robot applications," *Reliability Engineering & System Safety*, vol. 53, no. 3, pp. 291–299, 1996. [Online]. Available: https://doi.org/10.1016/S0951-8320(96)00054-3

[96] G. C. Messenger and M. S. Ash, *The effects of radiation on electronic systems*. Van Nostrand Reinhold Co Inc., 1986. [Online]. Available: https://inis.iaea.org/search/search.aspx?orig_q=RN:18091073

[97] V. S. Varadharajan, D. St-Onge, B. Adams, and G. Beltrame, "Swarm relays: Distributed self-healing ground-and-air connectivity chains," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5347–5354, 2020. [Online]. Available: https://doi.org/10.1109/LRA.2020.3006793

[98] "Gateways - batman-adv - open mesh." [Online]. Available: https://www.open-mesh.org/projects/batman-adv/wiki/Gateways

[99] D. Johnson, N. S. Ntlatlapa, and C. Aichele, "Simple pragmatic approach to mesh routing using batman," in *2nd IFIP International Symposium on Wireless*

Communications and Information Technology in Developing Countries, CSIR, 2008. [Online]. Available: http://hdl.handle.net/10204/3035

[100] M. I. Abdullah, M. M. Rahman, M. C. Roy *et al.*, "Detecting sinkhole attacks in wireless sensor network using hop count," *IJ Computer Network and Information Security*, vol. 3, pp. 50–56, 2015. [Online]. Available: http://dx.doi.org/10.5815/ijcnis.2015.03.07

[101] K-Team, "Khepera IV," 2021. [Online]. Available: https://www.k-team.com/khepera-iv

[102] R. de Azambuja, H. Fouad, and G. Beltrame, "A flexible exoskeleton for collision resilience," *arXiv preprint arXiv:2107.11090*, 2021. [Online]. Available: https://arxiv.org/abs/2107.11090

[103] A. Matos, A. Martins, A. Dias, B. Ferreira, J. M. Almeida, H. Ferreira, G. Amaral, A. Figueiredo, R. Almeida, and F. Silva, "Multiple robot operations for maritime search and rescue in eurathlon 2015 competition," in *OCEANS 2016-Shanghai*. IEEE, 2016, pp. 1–7. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7485707

[104] T. Fong and I. Nourbakhsh, "Interaction challenges in human-robot space exploration," *Interactions*, vol. 12, no. 2, pp. 42–45, 2005. [Online]. Available: https://dl.acm.org/doi/fullHtml/10.1145/1052438.1052462

[105] M. Shahriari, I. Švogor, D. St-Onge, and G. Beltrame, "Lightweight collision avoidance for resource-constrained robots," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1–9. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8593841

[106] OptiTrack, "Motive - Optical motion capture software," 2021. [Online]. Available: https://optitrack.com/software/motive/

[107] M. Selden, J. Zhou, F. Campos, N. Lambert, D. Drew, and K. S. Pister, "Botnet: A simulator for studying the effects of accurate communication models on multi-agent and swarm control," *arXiv preprint arXiv:2108.13606*, 2021. [Online]. Available: https://arxiv.org/abs/2108.13606

[108] E. Ferrante, A. E. Turgut, E. Duéñez-Guzmán, M. Dorigo, and T. Wenseleers, "Evolution of self-organized task specialization in robot swarms," *PLoS computational biology*, vol. 11, no. 8, p. e1004273, 2015. [Online]. Available: https://doi.org/10.1371/journal.pcbi.1004996