ECOLE
POLYTECHNIQUE
DE BRUXELLES

ULB

UNIVERSITÉ LIBRE DE BRUXELLES

Cooperative Relative Localization of a Robot Swarm

based solely on UWB distance estimation

Thesis submitted for the award of the degree of
Master of Science in Computer Science and Engineering

Quentin Roels

Director
Professor Mauro Birattari

Advisors
David Garzòn Ramos
Miquel Kegeleirs

Service
IRIDIA

Academic year
2023 - 2024

ECOLE
POLYTECHNIQUE
DE BRUXELLES

DOCUMENT TO BE INCLUDED IN THE MASTER THESIS

CONSULTATION OF THE MASTER THESIS

I, the undersigned

NAME Roels

FIRST NAME Quentin

MASTER PROGRAM Master of Science in Computer Science and Engineering

MASTER THESIS TITLE Cooperative Relative Localization of a Robot Swarm based solely on UWB distance estimation

AUTHORIZE*

~~**REFUSE***~~

(IN CASE OF REFUSAL, THIS DOCUMENT MUST BE COUNTERSIGNED BY THE MASTER THESIS SUPERVISOR)

(Delete as appropriate)*

Consultation of this master thesis by users of the libraries of the Université libre de Bruxelles.

If consultation is authorised, the undersigned hereby grants the Université libre de Bruxelles a free and non-exclusive licence, for the duration of the legal term of protection of the work, to reproduce and communicate to the public the above-mentioned work, on graphic or electronic media, in order to enable users of the libraries of the ULB and other institutions to consult it within the framework of inter-library loan.

Brussels, 30/05/2024

Signature of the student

QRoels

Signature of the supervisor
(only if consultation is refused)

Abstract

This Master's Thesis was submitted by Quentin Roels in the academic year 2023-2024 for the award of the degree of Master of Science in Computer Science and Engineering. The thesis consists of the *cooperative relative localization of a robot swarm based solely on ultra-wideband (UWB) distance estimation*.

Enabling complex swarm behaviors, such as efficient path planning, assembly, dispersal, or morphogenesis, requires each agent to know its relative position with respect to other agents. Since most research uses the fusion of numerous sensors or an external infrastructure to enable localization within a swarm, this thesis aimed to determine to what extent it is possible to perform cooperative relative localization using only distance information measured with ultra-wideband radio communication modules.

On the one hand, an ultra-wideband communication protocol was designed to enable distance measurement and communication within the swarm. On the other hand, different localization algorithms, known as multidimensional scaling (MDS) and particle filters (PF), were implemented to use the distance data generated. A comparative study of localization algorithms was carried out to determine the best candidate. While the MDS algorithm was more successful in reducing agent positioning errors, it lacked stability for continuous localization, resulting in poor orientation estimation. The particle filter proved more stable and improved orientation estimation but reduced the quality of position estimation for large swarms. However, both methods failed to consistently estimate the agent's orientation, preventing these location estimates from being used for complex behaviors.

In conclusion, MDS and the particle filter produced low-quality results for relative localization using distance measurements alone. On the contrary, displacement data added in the particle filter improved the overall quality, showing the benefits of sensor fusion in cooperative relative localization. Later research on relative localization should avoid using distance alone as it does not allow for the consistent estimation of a good-quality direction. Furthermore, the implemented system based solely on distance measurements was shown to perform better in landmarked situations, which limits its application to swarms of robots trying to avoid dependence on external structures.

Keywords: swarm robotics, cooperative relative localization, ultra-wideband, multidimensional scaling, particle filter.

Acknowledgments

This research has been partially supported by the Turku Intelligent Embedded and Robotic Systems (TIERS) group at the University of Turku in Finland. Particularly, I wish to thank Prof. Tomi Westerlund for providing a set of UWB modules for conducting experiments. Also, I wish to thank Dr. Jorge Peña Queralta and Mrs. Paola Torrico Morón for the assistance provided in setting up the modules.

I would like to thank Professor Mauro Birattari, my thesis director, and IRIDIA, the artificial intelligence research laboratory of the Université Libre de Bruxelles, for authorizing my research and providing a solid foundation for my work through the Mercator project and all the work underlying the creation of the experimental setup.

I would like to thank Miquel Kegeleirs for his guidance and advice throughout the year, Jeanne Szpirer for her help setting up the robots, and Guillermo Legarda Herranz for his work on the tracking system. I would also like to express my deep gratitude and respect to David Garzón Ramos, who has followed me for the last two years at IRIDIA, taught me the ways of research and kept me motivated in all circumstances.

I would like to thank the friends who helped me during this research: Gaspard for the brain-stormings and Saïd, who helped me prepare more robots and set up the experiments.

Thank you to my family, the BEP, and all the incredible people who shaped me over the past years. I will remember all of you.

Thank you, Salomé, for your neverending support, which has been crucial to every aspect of my life for the last four years.

Table of contents

Chapter 1

Introduction

Swarm robotics aims to enable a large group of agents to cooperate to accomplish tasks impossible to perform alone [1]. Whether for regrouping at the same place, covering the largest possible area, or forming patterns, the spatial organization of agents is one of the most important and studied behaviors in swarm robotics [2, 3]. While many behaviors can be realized by following local rules [4, 5], enabling more complex behaviors requires a high-level understanding of the swarm's topology. To that extent, we introduce collective relative localization, defined by Schranz et al. in [6] as "allowing the robots in the swarm to find their position and orientation relative to each other via establishment of a local coordinate system throughout the swarm". Therefore, relative localization is invaluable data for complex spatial organization, as it enables planning agents' next steps based on a descriptive map of the current circumstances.

Studies perform relative localization in different ways depending on the application. Outdoor applications such as UAVs (unmanned aerial vehicles) generally allow GPS-assisted localization. However, edge applications such as missions indoors, underground, or in other GPS-denied environments require another implementation. Those systems typically use numerous sensors and complex mathematical filters based on control theory to achieve sensor data fusion and ensure qualitative localization. Another implementation often discussed in the literature is the addition of an external infrastructure, leveraging nodes with known positions called landmarks to position the agents in the reference frame. Based on sensor fusion, the former requires numerous sensors, which increases the cost of the robotic platform but avoids the need for an infrastructure. Based on landmarks, the latter reduces the number of sensors required but limits localization to prepared environments. Other studies have proposed using some of the agents of the swarm as mobile landmarks, which reduces the number of sensors while removing the need for an external infrastructure. However, this solution involves modifying the agents' movement to apply localization.

Those observations raise the question of whether it is possible to enable real-time relative localization using distance alone, without altering the agents' movement and without an external infrastructure—a question that this thesis will try to answer.

1.1 Context

In [1], Dorigo et al. define swarm robotics as the design of a group of robots that do not require any external infrastructure or form of centralized control to operate. Also mentioned by Navaro and Matia in [7] and by Bonabeau et al. in [8], swarm intelligence principles are the fertile soil in which swarm robotics takes root, trying to reproduce the complex behavior achieved by communities of insects through local rules.

Swarm intelligence is generally described by the following set of properties [1, 9, 10] :

- Group of individuals, often numerous.
- Homogeneous, i.e. a restricted set of individual types.
- Self-organized, i.e. organizing through local communication or changes in their environment.
- Collaborative, i.e. the task must be impossible for a robot to perform alone.
- Distributed, i.e. only access to local information and sensing, and no centralized control.

Using the taxonomy defined by Iocchi et al. in [3] and shown in Figure 1.1, a swarm of robots is cooperative, aware, coordinated, and distributed. It means that all members of the swarm operate together to perform the same task, are aware of the existence of the other agents of the swarm, take into account the actions taken by other agents to create coherence, and are autonomous in their decision-making process.

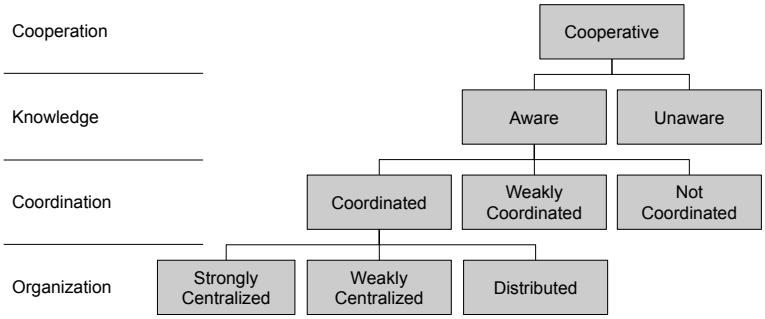


Figure 1.1: Taxonomy of mutli-robot systems defined by Iocchi et al. in [3]

Schranz et al. [6] have studied the variety of behaviors given to robot swarms over the years and their different real-life applications. Figure 1.2 shows the landscape of behaviors they described, which gives a comprehensive overview of the topic and allows to situate the current thesis work within swarm robotics. As described in the previous section, the subject of this thesis addresses two of the behaviors defined in Figure 1.2. The underlying behavior studied in this work is cooperative localization, classified as a navigation behavior, while the thesis aims to enable the various behaviors classified under spatial organization.

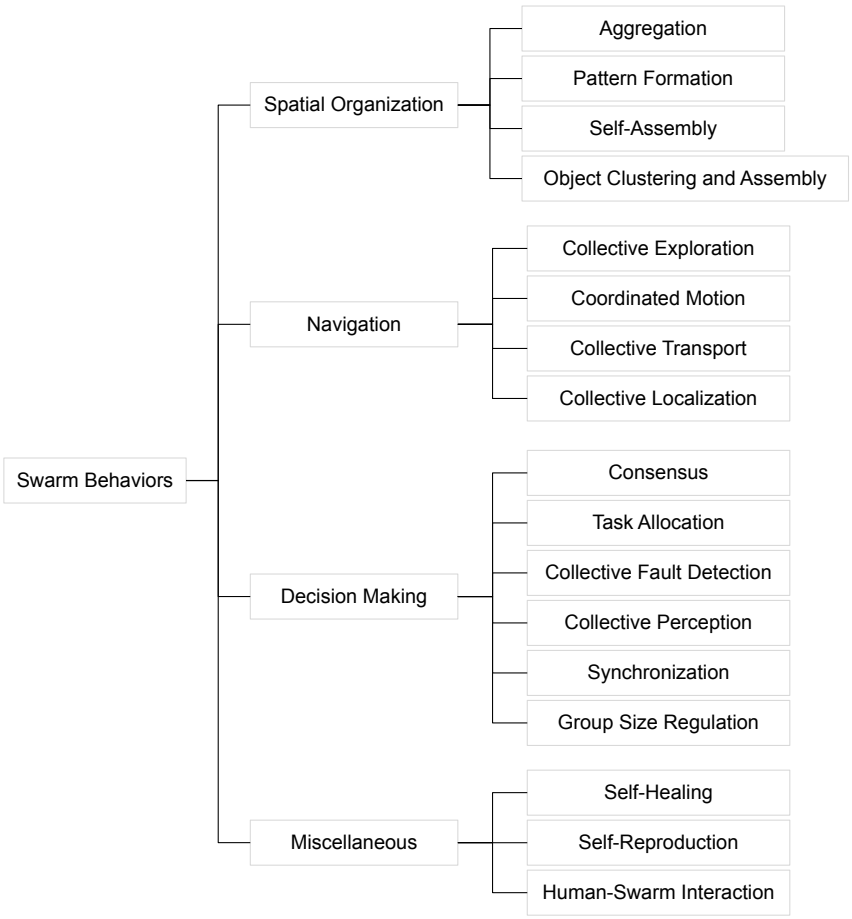


Figure 1.2: Taxonomy of swarm behaviors defined by Brambilla et al. in [11] and later updated by Schranz et al. in [6]

1.2 Contribution

The main objective of the thesis is to enable behaviors based on the relative localization of the agent in the swarm. Typically, the aim is to reach a specific position in the swarm to obtain, aggregation/repulsion behaviors and other actions, such as following a given agent. To this extent, agents must have a high-level understanding of their relative position in the swarm, which induces the research topic of this thesis: cooperative relative localization, i.e., getting agents in the swarm to cooperate so that they localize each other relative to themselves. The contribution includes implementing a communication and telemetry protocol using UWB modules and a localization algorithm using the distances communicated by the swarm via the protocol.

The topic of this thesis aligns with the research of IRIDIA, the robotics and artificial intelligence lab of the Université Libre de Bruxelles (ULB). Generating aggregation and repulsion would allow Mercator [12], their last robotic platform to have all behaviors of IRIDIA automatic off-line design tool, AutoMoDe [13]. This would allow them to finish implementing all e-puck behaviors they previously had on their current robots and conclude their study of the portability of automatic behaviors from one robotic platform to another.

1.3 Structure

This chapter introduces the research topic by explaining the basic concepts of swarm robotics and explaining the contribution of the thesis, the following chapters are organized as follows: Chapter 2 provides an overview of the research landscape in which the thesis is embedded. Chapter 3 describes the material used in the thesis. The 4th chapter summarizes the various implementations of the thesis, namely the communication protocol of the robotic platform and the algorithms used to enable relative localization. Next, chapter 5 deals with the experimental setup, protocols and measurements used to assess the quality of the communication protocol and algorithms implemented. Then, chapter 6 discusses the results of these same experiments. Finally, the thesis concludes with discussions and further work in chapter 7 and a conclusion in chapter 8.

1.4 Symbol and Abbreviations

Abbreviation	**Meaning**	**Reference**
ROS	Robot Operating System	3.3
rosbag	ROS service recording messages sent in the topics over time	5
BLE	Bluetooth Low Energy	2.1
SoL	Speed of Light	4.1.2
MDS	Multidimensional Scaling	4.3.2
init	initialization of MDS with previous estimation of localization.	4.3.2
offset	distance measurements corrected with error model.	6.1
certainty	weighted MDS to give less importance to old measurements.	4.3.2
PF	Particle Filter	4.3.3
init	starting up the filter with good relative positions.	6.3
offset	distance measurements corrected with error model.	6.1
certainty	removing old values for the update phase	6.3
	Direction estimates	4.3.4
distances	Direction estimate using evolution of pairwise distances	4.3.4
displacement	Direction estimate using displacement in the position estimates	4.3.4
particles	Direction estimate using the mean value of the PF particles	4.3.4

Chapter 2

Literature Review

The idea of relative localization appeared in distributed sensor networks long before its implementation in indoor localization. For example, in 1978 [14], it was already mentioned that knowledge of relative positions can be determined by measuring pairwise distances and is crucial for interpreting the data collected by the network. Therefore, whether to optimize the communication path or improve distributed sensor measurements, most of the research on relative localization comes from wireless sensors and communication networks, to locate the different network nodes [15].

Collective localization is defined by Schranz et al. as "allowing the robots in the swarm to find their position and orientation relative to each other via the establishment of a local coordinate system throughout the swarm" [6]. Generally, satellites enable the use of Global Navigation Satellites Systems (GNSS) to obtain the position of a robot in a global reference frame. GNSS, like GPS and many other positioning systems, are often supported by a compass to determine the robot's current direction to the reference frame [16], therefore completing the requirements for localization of the agents. However, current research is mainly trying to avoid those exact technologies [17, 18], the main reasons being :

- GNSS precision is about 5 meters [19], unusable for small-scale swarms [20].
- GNSS denied environments (prevents indoor, underground, and other missions)
- GNSS is too expensive and energy-insensitive for simple robots [21]

These reasons have led cooperative localization research to focus on GNSS-free solutions, on the assumption that the main applications of the swarm are in GNSS-free environments, mostly indoors, such as in warehouses [18, 22, 23]. In this context, the question arises about which technology to use for indoor localization.

2.1 Indoor Localization

Indoor localization is not a new topic; Werb et al. discussed the design of such a system in 1998 [24]. Numerous studies have addressed the various technologies that enable cooperative localization and their use for distance measurement and sometimes bearing measurement [25, 26, 27]. Some studies, particularly that of Chen et al. in [25], have

shown that UWB technology is a good solution, even more so in a swarm of robots where scalability is a vital requirement.

UWB technology has been used for decades as a means of communication [28] and has recently gained interest in the field of localization [29, 30] as it enables communication and distance estimation thanks to its large bandwidth, allowing more accurate Time of Flight (ToF) distance estimation than other media, such as Wi-Fi or BLE technologies with less bandwidth [18].

Citing Siva et al. in [18], "The vast majority of UWB localization systems are implemented in an environment where multiple UWB transceivers act as anchors while the robot to be localized carries another one referred to as the tag". Nowadays, the anchor-tag protocol is implemented in most off-the-shelf solutions available on the market [31]. The main reason is that these solutions are often dedicated to applications such as indoor tracking of moving objects [32], where the anchor-tag localization system is very effective [33].

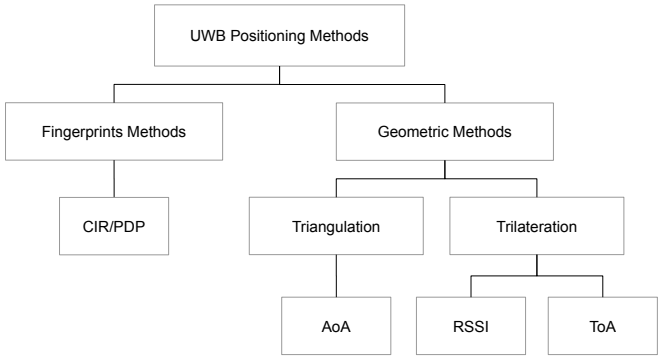


Figure 2.1: UWB Positioning Methods classification taken from Mazhar et al. in [30], Geometric Methods are divided between Triangulation and Trilateration after [34, 35, 36]

Many localization techniques are using Ultra-Wideband, the classical ones are listed in Figure 2.1.

The first category concerns fingerprinting methods:

- PDP, or Power Delay Profile describes how signal power varies as a function of different delays, providing a better understanding of multipath propagation in radio environments.
- CIR, or Channel Impulse Response, reflects how a communication channel distorts and delays signals.

In fingerprint-based localization, PDP and CIR are used to create a database of signal characteristics at different locations. The PDP captures variations in signal strength over time, while the CIR details how the environment distorts signals. By matching the PDP and CIR observed at an unknown location with the database, the system can estimate the location based on the closest match, enabling precise localization indoors or in complex

environments. Fingerprinting methods are more complex as they commonly require a database consisting of previous measurements at several known positions [29] on which to apply machine learning algorithms [37], leading to geometric methods being studied more often.

The second category concerns triangulation techniques:

- AoA, or Angle-of-Arrival is a triangulation technique that exploits the angle at which the signal arrives at the antenna, using a set of sensors that typically measure the phase difference for a given received signal.

In triangulation-based localization, the angles from known locations to the target are used to define the target's position at the intersection points of the defined lines. Triangulation methods are believed to be less suitable than trilateration methods for position estimation [38], which is why AoA is often used in conjunction with another method such as ToA [39, 40, 41]. However, in some cases, AoA can outperform other techniques, but it requires special systems made up of UWB antenna arrays [42, 43, 44], which complicates the implementation of such systems.

The last category concerns trilateration techniques:

- RSSI, or Received Signal Strength Indicator
- ToA/ToF, or Time of Arrival/Flight

Trilateration-based localization aims to find pairwise distances between network nodes to locate and position them relative to known landmarks. There are many ways of calculating pairwise distances between UWB sensors [45], but only the most common are discussed here. Concerning RSSI, Siva et al. mention that "RSSI measurements are not reliable, because they suffer from large errors induced by multipath effects as well as a distinct lack of stability even when the environment is static" [18], which makes it difficult to use indoors (corroborated by Dong et al. [46]).

For the above reasons, the most common techniques for UWB localization are the ToF-based ones, namely TDoA and RToF [47]

- TDoA (Time Difference of Arrival) consists of a tag broadcasting a message periodically; the anchors receiving the message are tightly synchronized to determine the exact time of arrival of the message. The information is then centralized to compute the most likely position of the tag.
- RToF (Round Trip Time of Flight), also known as Two-Way-Ranging (TWR), involves a tag sending a message to each anchor point independently; the round-trip time is used to estimate the distance using the transmission speed. The tag then uses the distances to the known anchor points to calculate its position relative to them.

Table 2.1 summarizes the advantages and disadvantages of the two methods according to Ridolfi et al. in [47]. TDoA and RToF have complementary strengths and weaknesses, but the architecture required is the most differentiating factor. TDoA requires robust

	Advantages	Disadvantages
TDoA	• Tags must send only one packet per position estimate, decreasing energy consumption. • The server has more computational power than the infrastructure/mobile nodes, resulting in faster position estimation. • All anchors in the range can be used for positioning so we obtain more accurate results.	• Anchors must be synchronized accurately, which causes synchronization beacons overhead. • The calculated positions are unknown to the tag, so the estimated position must be communicated via a second network. • Data aggregation or collaborative localization is impossible since the mobile tags are only transmitting and not receiving.
RToF	• Location could be computed by the mobile nodes. • Anchors do not have to be synchronized.	• More messages are required for localization. • More complex implementation, e.g., selecting the best anchors set, roaming management. • More energy consumption.

Table 2.1: Advantages and Disadvantages of TDoA and RToF, taken from [47]

synchronization and does not provide distance information to mobile nodes, requiring a centralized server to retrieve measurements from all nodes and perform the location calculation. Such a strict architecture runs counter to the expectations of swarm robotics [1, 11], and explains the appeal of RToF for application to UWB relative localization. It is worth noting that power consumption is not an issue in our case, even when the aim is to equip sensors in swarm robots with limited power; UWB technology is a significant improvement on Bluetooth and Wi-Fi [48].

However, implementing RToF requires a communication protocol between anchors and tags so that the round-trip message can be sent back, which increases airtime and channel utilization compared with TDoA [49], therefore increasing interference. As the number of tags increases, the channel utilization increases, too, to reduce packet losses in that setting; however, some channel access schemes can be implemented.

2.2 Channel Access Schemes

There exist many schemes such as ALOHA (Random Access), CSMA (Carrier Sense Multiple Access), TDMA (Time-Division Multiple Access), FDMA (Frequency-Division Multiple Access) or CDMA (Code-Division Multiple Access) [50, 51]. For implementation reasons, the most discussed ones are ALOHA, which is straightforward; TDMA, which requires MAC-level implementation of the communication modules; and CDMA, which requires a deeper knowledge of devices spreading codes. Even if CDMA can offer great performances [52], it is often disregarded in research on applied topics for simplicity.

Typically, the ALOHA scheme, which resends the packet after a certain time until it is received, is the simplest and is more beneficial than doing nothing since the chances of the packet being received are increased by resending it. With ALOHA, however, scalability is a problem: if the number of nodes in the network increases, packets that are resent will always end up interfering with others and so will never be received. Another option is CSMA, which senses the channel before starting the communication to avoid causing interference by sending it on top of another ongoing communication. However, sensing the channel is a complex task.

Arguing that ALOHA and CSMA are not suitable for heavy-load networks, one might want to find a solution elsewhere. In [53], Cao et al. propose a solution using the TDMA scheme, which they argue fits well the purpose of relative localization even for more significant sizes of swarm than what ALOHA can support. ALOHA is effectively chosen for its straightforward implementation in low-load applications. But as soon as the load increases, choosing another channel access scheme becomes more effective, and TDMA is used most of the time [53, 54, 55].

2.3 Cooperative Localization

Let us now delve into cooperative localization applied to robotics. As mentioned previously, most research on cooperative localization comes from the study of Wireless Sensor Networks, which inevitably causes a bias toward situations with a robust infrastructure of non-moving nodes [56]. Those nodes with known locations or *landmarks* can then be used as reference points in the localization, which significantly simplifies the problem of relative localization, being for cases with non-moving nodes only or hybrid cases such as Mobile Sensor Networks composed of moving and non-moving nodes [32]. However, the infrastructure hypothesis considerably reduces the interest of robot swarm studies since their primary goal, as described in [11], is to work together to avoid relying on an external infrastructure and centralized computation.

2.3.1 Alternating Landmarks

A typical solution to the need for infrastructure is the Alternating Landmark Network (ALN) proposed by Kurazume et al. in 1994 [57]. The idea is described as a division of the robots into two teams alternating between moving toward their target position and stopping to act as landmarks for the other team. This method offers the advantages of landmarks for relative localization, showing excellent results [58, 59], but requires robot coordination using synchronization. Furthermore, it forces the agents to adopt a behavior that might interfere with the mission. However, this method has proven itself in cases where uncertainty due to noise in the measurement is too significant, offering stability by relying on non-moving nodes.

The most common solution for relative localization in dynamic systems is to use sensor fusion through recursive Bayesian frameworks such as Kalman filters and particle filters. The main idea of both filters is to predict a state vector, such as the positions of the robots in the swarm, and leverage measurements made with sensors to lead the filter toward a good approximation of the actual state through an iterative correction process between each prediction. Furthermore, the filters typically account for the noise of the

various measurements made by the sensors, which explains the interest of these filters in robotics, where systems are often noisy.

On the one hand, Kalman filters use the system's dynamics to estimate the state's evolution. It is, therefore, necessary to derive the dependencies between the motion of the system and the observations made on it. The Kalman filter requires a linear system. However, there are non-linear variants, such as the extended Kalman filter (EKF) and the unscented Kalman filter (UKF), which are sub-optimal due to their inherent approximation. The EKF uses analytical linearization, i.e. a first-order truncation of the Taylor approximation of the system dynamics, which requires the system dynamics to be differentiable to return to a linear system. While the EKF gives good results for slightly non-linear systems, it becomes difficult to use for strongly non-linear systems. UKF is better suited in this case, as it has the same computational complexity as EKF and is derivative-free, allowing for non-differentiable systems (discontinuities, ...) [60]. UKF is based on the fact that "*it is easier to approximate a probability distribution than it is to approximate an arbitrary non-linear function or transformation*" [61], meaning that it will use Unscented Transformation (UT), based on deterministic sampling, to approximate the output distribution of the non-linear function instead of trying to approximate the non-linear function itself.

On the other hand, Particle filters (PF) are sequential Monte Carlo methods based on the simplification that the system acts as a Markov process. The goal is then to estimate the posterior distribution of the state using the observation made on the dynamic system through the sensors [62]. The main steps are predicting the state through a weighted average of the particles drawn from the distribution, correcting the posterior distribution using the observations, and resampling the distribution to have new particles [63]. While much research is done on particle filters and Kalman filters, it is commonly accepted that particle filters bring better results but are more computationally expensive than Kalman filters [64]. Therefore, when applied to small robots, localization is often based on Kalman filters.

2.3.2 Kalman Filters

In [65], Araki et al. compare filters for merging range measurement and odometry information to perform relative localization of up to 5 moving robots. The authors propose a version of UKF, EKF, and PF that leverages synchronization between the robots to compute the steps asynchronously—allowing for a more straightforward real-life implementation of the filters. They show that PF is more computationally expensive than EKF and UKF and that UKF performs better than the other filters for a more significant number of agents, i.e. for a highly non-linear system. As EKF is not robust against large initial error [66], Chakraborty et al. [67] propose a N-vehicle cooperative relative localization estimator using range-only measurements among vehicles in the presence of large initial uncertainty. The Multi-Hypothesis Extended Kalman Filter (MHEKF) described is lightweight compared to Particle filters, performs better than EKF when initial error is high, and can be used for many robots.

However, as the number of robots increases, the state to be estimated by the various filters increases in size, which tends to generate an overhead in computation. To solve the problem of EKF computational overload in multi-robot systems, Roumeliotis et al.

[68] propose a distributed version of the Kalman filter in which each agent computes the part of the Kalman filter associated with its displacement and shares its result with the other robots so that they can compute the global state. Unfortunately, for a robot to calculate the global state, it must receive factors from all the other agents, which requires full connectivity and communication between each filter update phase.

Different distributed non-linear filters have been proposed, such as the Interleaved Extended Kalman Filter (IEKF). In [69], Panzieri et al. describe an implementation of IEKF that overcomes the need for a fixed number of agents in the system. By locally computing the fusion of host sensors and only exchanging sensor measurements and the corresponding covariance matrix with robots within their range, each agent improves its position instead of the position of other agents in the swarm, decoupling the Kalman filter and cross-correlation terms. This result tends to give non-uniform accuracy but allows localization even when some agents lose connection with the team. However, the filter loses accuracy when not applied with landmarks, and the authors correct this by adding agents playing the role of alternative landmarks.

The basic assumption of the distributed filter is that all agents compute their position through the same frame, which we might want to avoid. The literature also describes relative localization methods based on communication between two agents alone. Many techniques return to what ter Horst defines as *tangolation* in [54], using the displacement of the two agents between two time steps and the measurement of the distance before and after the time step, we can recreate the relative movement of the two agents [70, 71, 72]. As a result, the robots only need to communicate their odometry data and measure the distance between them, which does not require common calculation techniques. However, it requires synchronization to use the correct measurements and it often requires a common frame, which typically means adding a compass to the system. Furthermore, this relative localization technique was designed for just two robots and is not scalable.

2.3.3 Particle Filters

Despite the computation overhead avoided by using Kalman filters instead of Particle filters and their intensive distribution sampling, the various Kalman filters are not always applicable. Observability is the primary concern when describing a system dynamic to apply a Kalman filter because it quantifies whether the state can be estimated through the measurements made to the system [73]. For that reason, many studies perform observability analysis of various systems with different combinations of sensors; for example, Sharma et al. derive the observability of bearing-only cooperative localization in [74], while Burchett et al. describe the observability of range-only cooperative localization in [75]. Burchett et al. show that a system based on odometry and range measurement for moving and non-moving nodes is weakly-observable. Araki et al. show in [65] that observability is kept once the landmarks are removed. However, they end up mentioning that their main experiment required the robots to move in groups, resulting in an alternating landmarks system. When observability is low, ambiguities appear that the Kalman filter does not handle well, leading to degraded results. In such cases, particle filters can be helpful [54].

In addition, the particle filter, being non-parametric, can be applied more generally than Kalman filters, which assume that the system is Gaussian [76]. Building on the latter argument, Liu et al. [77] propose improving dead reckoning navigation by fusing inertial information from an IMU sensor with distances measured by UWB transceivers. Sensor fusion is achieved using a particle filter, which gives excellent results in a landmark-free situation. Similarly, Li et al. [78] propose a two-stage relative localization algorithm to account for two main problems: state initialization and particle filter distribution for multi-robot systems.

This work aims to remove the need for a sensor other than distance measurement to perform relative localization. This approach can also be found in [79] where Zhou et al. argue that an additional inertial sensor requires sampling synchronization and increases the cost and size of the system. Another typical way of proceeding would be to add velocity to the state to be estimated by the filter, but this has two negative aspects. Firstly, the system's observability is reduced since the state to be estimated is larger, but no additional measurements are added. Secondly, increasing the size of the state may cause the filter to suffer from the "curse of dimensionality". In their work, Zhou et al. focus on two new approaches to estimating robot velocity as a function of the previous state estimate, one based on particle filters and the other on Chan-Taylor. Unfortunately, both approaches rely heavily on a high-quality position estimation made possible using landmarks. On the other hand, Liu et al. [80] propose a particle filter that merges information from RSS sensors and UWB range estimation. Since no information on agent motion is known, they represent the motion model as a random walk.

2.4 Thesis Work

Since most research on the topic of cooperative relative localization requires either modifying the movement of agents to facilitate the localization process or retrieving information from many other sensors, which increases the system's dependence on sometimes expensive sensors, the question arises as to what can be achieved using only pairwise distance measurements. This research aims to provide a basis for distance-based localization, either by showing that several sensors are indeed needed to achieve sound localization or by showing that current systems do not need as many sensors as they are equipped with.

Chapter 3

Material

This section describes the hardware and software used in the different experiments of the thesis to ensure reproducibility.

3.1 DWM1001-DEV

The DWM1001-DEV is a development board manufactured by Qorvo that enables easy implementation and prototyping using the DW1000 ultra-wideband transceiver. The base components for this thesis usage of the board are (see Figure 3.1) :

- The DWM1001 module, based on the DW1000 UWB transceiver, adds a microcontroller (Nordic Semiconductor nRF52832), a Bluetooth antenna, and a motion sensor.
- The USB connection for reprogramming, debugging, and power supply.
- The LEDs, buttons, and J-Link On-Board Debugger for quick debugging.

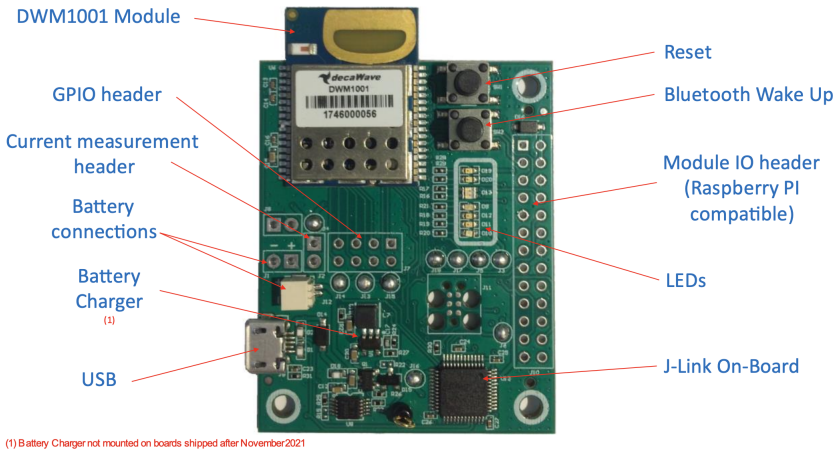


Figure 3.1: Double-Sided Two-Way-Ranging Protocol

A hands-on explanation of the development pipeline for the DWM1001-DEV board can be found in the documentation on Qorvo's website. However, the development board is primarily made for plug-and-play evaluation of the features and performance of the DWM1001 transceiver module. It comes with a proprietary library and API for an anchor-tag localization scheme.

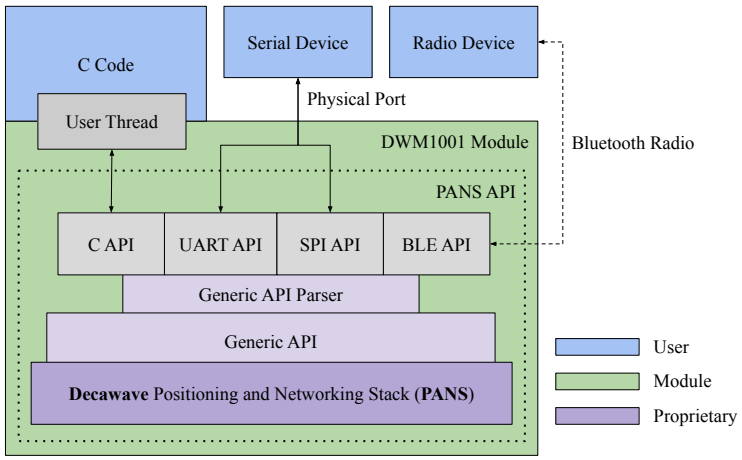


Figure 3.2: Decawave Positioning and Networking Stack (PANS) API Usage

The API usage is shown in Figure 3.2. The user can interact with the API via the C code flashed onto the module (on top of the proprietary firmware), through SPI/UART via a wired connection to the module, or through the Bluetooth antenna of the DWM1001 module.

However, the Positioning and Networking Stack (PANS) is a finished proprietary product designed and fine-tuned for the sole purpose of anchor-tag localization in the way intended by the manufacturer. The following limitations can be noted:

- **Data Transfer**: the PANS library does not allow for data transfer between the tags and the anchors; the only data transfer happening is between tags and particular local central nodes (gateways) that are unnecessary for the localization process. Furthermore, the up-stream and down-stream information that can be exchanged is limited; the usual application is for over-the-air modules firmware updates.
- **Underlying Protocol**: the modules are communicating through a Time-Division Multiple Access (TDMA) scheme, which requires synchronization and using the antenna at precise moments. This prevents adding a UWB communication layer on top of the module firmware to compensate for the data transfer problem just described.
- **Distance Measurement**: the distances measured by the module are limited for efficiency. A maximum of four distances are measured at each localization step (between the tag and four different anchors), and the anchors to which the distance is estimated cannot be chosen manually; they are chosen by the black-box localization algorithm.
- **Proprietary Library**: the PANS library is a pre-compiled proprietary firmware that cannot be modified as it is not open-source. This final limitation makes it impossible to modify the firmware to compensate for the above issues it raises.

Those particular points have led to prototyping a limited communication protocol that allows for range measurements and data communication using the user C code interface and calls to the low-level API of the DW1000 transceiver, which prevents the PANS firmware from running at the same time on the modules.

3.2 ARGoS

"**What is ARGoS?** ARGoS is a multi-physics robot simulator. It can simulate large-scale swarms of robots of any kind efficiently. You can customize ARGoS easily by adding new plug-ins" (from ARGoS website) and was created by Pinciroli et al. [81].

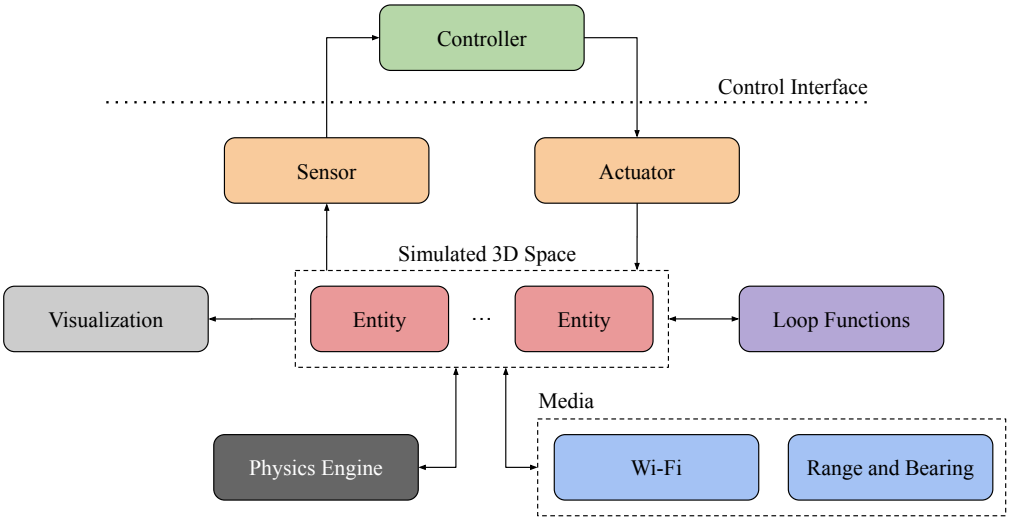


Figure 3.3: ARGoS Modular Architecture, taken from [81]

The simplified architecture of ARGoS is shown in Figure 3.3. The simulation is based on different entities, typically the simulated robots. Each entity is defined by its physics, the media it uses, its actuators/sensors and a controller. The controller is a user-created program that receives data from the sensors and applies modifications to the related entity through its controllable components, called actuators. For example, receiving data from the proximity sensors, the controller decides on the direction and modifies the wheel actuator values accordingly to make the agent rotate. Loop functions are global controllers allowing for additional computing on top of the simulation.

ARGoS was used to prototype the localization software. Range estimation and communication were simulated using the range and bearing sensor and actuator plug-ins. The range information of the sensor simulated the UWB modules' distance estimation, and the actuator's ability to transmit information between agents' controllers simulated the communication protocol. To approximate the noise of the physical situation in such a system, a noise like that observed with actual modules was added to the distance measurements (see noise profile determination later on), and a high percentage of drop rate was used to mimic interference in the communication procedure.

3.3 ROS

"The Robot Operating System (ROS) is a set of software libraries and tools that help you build robot applications. From drivers to state-of-the-art algorithms, and with powerful developer tools, ROS has what you need for your next robotics project. And it's all open source." [82]. ROS was used for communication between the various localization processes in the robot, both in simulation and real experiments. The ROS Noetic distro was used since the robots were running Ubuntu 20.04.

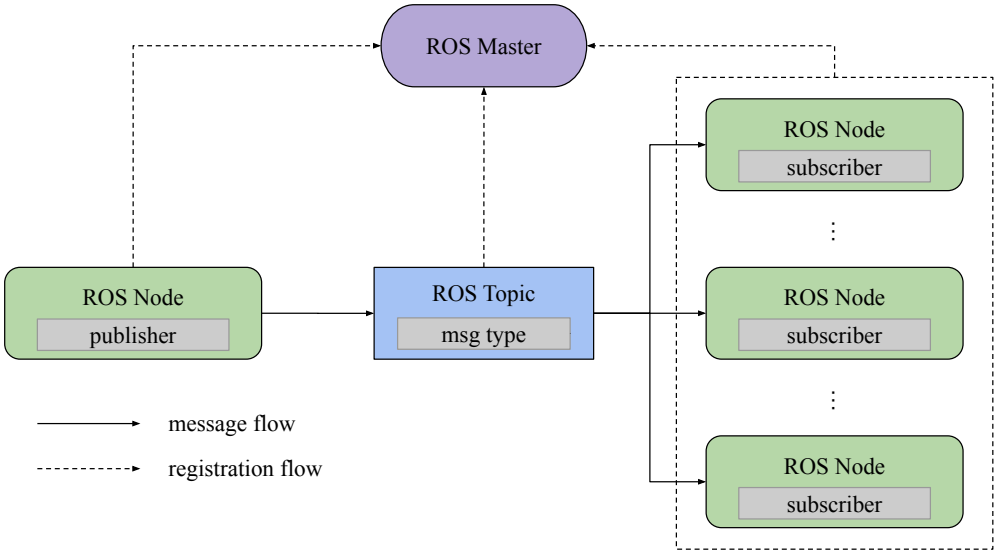


Figure 3.4: Typical ROS communication architecture

As shown in Figure 3.4, ROS is based on nodes that will either publish or subscribe to *topics*. For the system to be aware of the current connections, all nodes and topics are registered to a common centralized master (for ROS1). Topics are described by a name and only accept messages of a given type, determined at creation. ROS also has a recording function that allows for the storage of messages sent on different topics over a period of time. It was used to record experiment data and be able to rerun them as many times as required.

3.4 Mercator

The robots used for the experiments outside simulation are the robots of the Mercator project of IRIDIA at the ULB. Based on the Sphero RVR robot (see Figure 3.5 (a)), it has been augmented with many sensors, such as proximity sensors, a LiDAR, a camera, UWB modules and RaspberryPi's for computation (see Figure 3.5 (b)).

(a) Base Robot (Sphero RVR) (b) Final Robot (Mercator)

Figure 3.5: Mercator

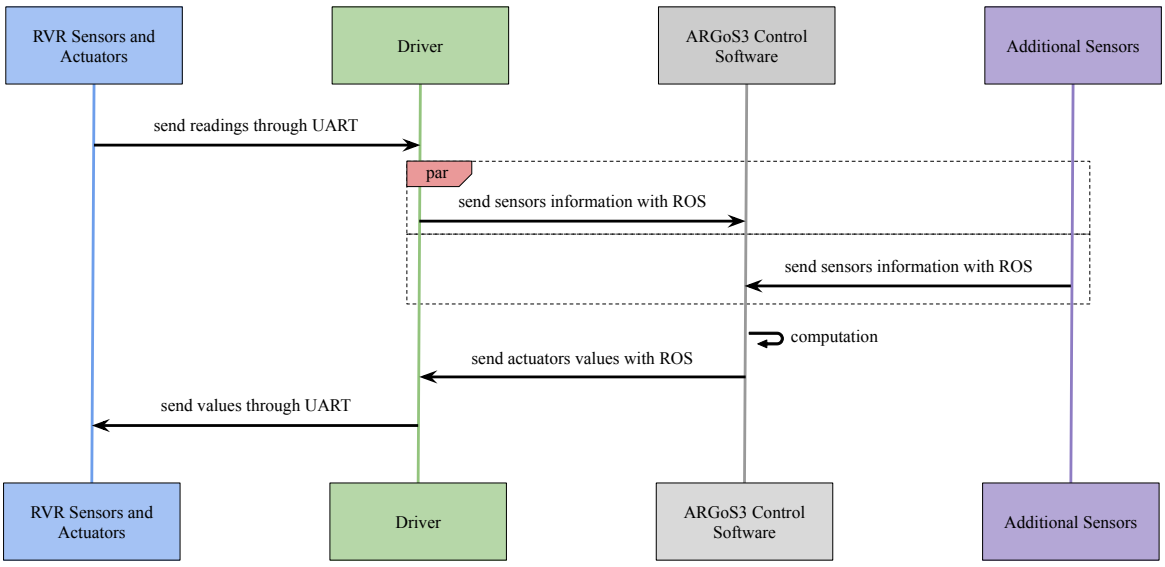


Figure 3.6: Marcator complete architecture (after Kegeleirs et al. in [12])

The internal architecture and data flow in Mercator is shown in Figure 3.6. A driver reads data from the Sphero RVR's internal sensors and publishes the readings on different topics. These can then be used by the controller, within the ARGOS3 control software. Once the calculation is complete in the controller, the actuator values are sent back to the RVR controller via the driver. In addition, external sensors can be added to the robot, as is the case with LiDAR, cameras, and UWB modules. These sensors must publish data in ROS topics to fit the Mercator architecture. A handmade driver is typically required to interface the external sensor readings to ROS.

Chapter 4

Method

4.1 Ranging Process

This section examines the design choices in implementing the ranging process, including low-level implementations such as the firmware for ultra-wideband modules and the communication scheme.

4.1.1 Channel Access Scheme

As described in Section 3.1, the firmware shipped with the DWM1001-DEV modules implements a polished TDMA scheme. Such a scheme divides the communication in time frames during which each tag (one at a time) is allowed to initiate single-sided two-way ranging with the close-by anchors. However, the proprietary nature of the firmware makes it impossible to modify it to meet the requirements of this thesis. To avoid the need to implement TDMA from scratch using the DW1000 (as it has already been done by Cao et al. in [53] and by ter Horst in [54]), this thesis focuses on prototyping an ALOHA channel of sufficient quality and building on top of it, with the promise that any results found for larger swarms would be greatly improved by better implementation of the channel.

4.1.2 SS-TWR vs. DS-TWR

While the choice between RToF and other distance estimation techniques has been justified in detail in the literature review, its operation has yet to be mentioned. Round Time of Flight (RToF), Round-Trip Time (RTT), or Two-Way Ranging (TWR) describe the same concept and are separated into two typical protocols for peer-to-peer measurement of distance:

- Single-Sided Two-Way Ranging (SS-TWR)
- Double-Sided Two-Way Ranging (DS-TWR)

SS-TWR and DS-TWR are common trade-offs between a simpler protocol that allows for computing the distance only on one side and a more complex protocol that allows for computing the distance on both sides but causes more interference due to more messages being sent.

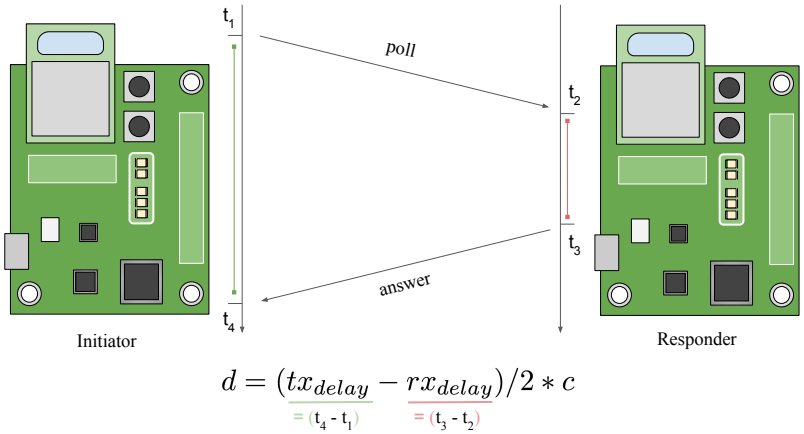


Figure 4.1: Single-Sided Two-Way-Ranging Protocol

First, the concept of TWR is easily grasped through the explanation of the Single-Sided method shown in Figure 4.1. The module willing to estimate the distance separating it from another module initiates the communication protocol; it is called the initiator. It sends a polling message to the other module and saves the timestamp (t_1) at which it sent the message. Then, the module that has been polled, called the responder, will answer the poll with a message containing the timestamp at which it received the polling message (t_2) and the moment at which it sent it back (t_3). Finally, when receiving the answer from the responder, it saves the timestamp (t_4). At this point, the initiator has everything it needs to compute the distance estimate through RToF.

Following the equation in Figure 4.1, the initiator effectively estimates the time that both messages have spent flying in the protocol ($tx_{delay} - rx_{delay}$), which it divides by two to have the estimated time of one message going from one module to the other, and multiplies the result by the speed of light (SoL) c to find the distance through the equation $d = v.t$, where d is the distance, v is the speed, and t is the time. *Note: SoL being a good approximation of the propagation speed of messages sent by UWB transceivers.*

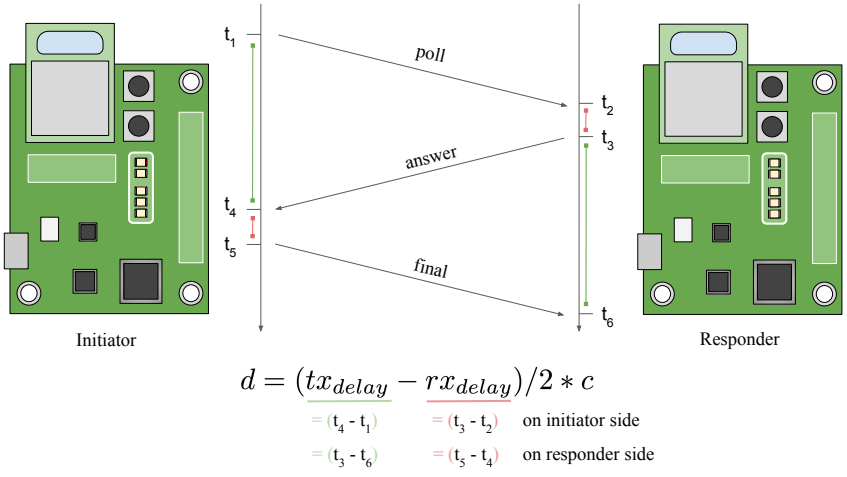


Figure 4.2: Double-Sided Two-Way-Ranging Protocol

In comparison, the Double-Sided TWR protocol is shown in Figure 4.2. The idea is to send a final message from the initiator to the responder so that the responder has everything it needs to do the same computation as the initiator. Therefore, it allows both ends of the communication channel to have access to the distance estimation at the cost of one more message sent through the channel.

Having a two-way communication protocol in a swarm of robots is not usual. Often, agents in a swarm announce more than they discuss during communication, usually by broadcasting messages, while the others listen for messages when they need to. This offers a simple channel with high message drop rates and interference but without the need for acknowledgment messages in the protocol. In this case, however, the ranging process requires two-way communication, which, added to the ALOHA scheme, will not help reduce interference and the drop rate. From this point of view, it seems complicated to justify a final acknowledgment that would also increase the number of frames in flight and, consequently, the amount of interference. The main benefit would be a better distribution of distance knowledge but at the cost of an additional message per protocol iteration (a 50% increase in the number of messages sent during a distance estimation phase) and a shorter availability for the agent responding to another agent's query. The comparison of the two protocols would be more interesting in a TDMA context, where each robot is allocated a moment in time for distance estimation with its neighbors. Indeed, responders' availability is not a problem when they respond to the only module currently polling for distances. This thesis builds on single-sided two-way ranging.

4.1.3 1-Module vs. 2-Module Operation

The RToF ranging process requires two nodes with different behaviors: the initiator, polling for distances, and the responder, answering the polls to allow distance estimation. These behaviors being different, here are possible ways to implement the protocol on independent agents:

- **1-Module, Heterogeneous**: One module per robot, having either one behavior or the other (exclusive "or"). Heterogeneous firmware between the different agents of the swarm, some acting as initiators, others acting as responders.
- **1-Module, Homogeneous**: One module per robot, switching between the two behaviors. Homogeneous, all agents have the same firmware on their communication module.
- **2-Module, Homogeneous**: Two modules per robot, each having one of the two behaviors (each having a different one). Homogeneous, all agents have the same modules, with the same protocol.

1-Module, Heterogeneous

The main problem with this implementation is the Two-Way Ranging (TWR) design. Whether for unilateral or bilateral TWR, the initiator is the basis of the ranging protocol. The interaction does not occur without the initiator, preventing responder modules from estimating their relative distance from other responders. Conversely, the responder's behavior is mandatory when responding to a poll from an initiator, prohibiting two

initiators from running the ranging process together. Even in the best configuration, an agent could only communicate with one-half of the swarm.

1-Module, Homogeneous

Switching from one behavior to the other is the least costly option, as it keeps the number of equipped communication modules as low as possible. However, it has an inherent cost-efficiency trade-off. When one agent switches to the initiator state, it becomes impossible for other robots to initiate communication with the agent. Conversely, when switching to the responder behavior, the only way for the agent to access the range estimate would be to implement the Double-Sided TWR protocol, which allows both sides to estimate the distance with a final message coming back to the responder.

Another difference between the two behaviors is situated in the timeout while waiting to receive a message. While the initiator is supposed to wait only a short time for the answer before it can be considered lost, the responder on its side cannot timeout too often while waiting for an incoming message as it would be missing frames. While reasonable if the goal is to offer simple communication in the swarm, the context-switching protocol causes many communication losses, making robots unavailable for answering polls half the time. In the case of relative localization, the application greatly benefits from a more efficient channel, being by implementing a TDMA communication scheme or, in this case, avoiding the 1-module context switching operation.

2-Module, Homogeneous

This can be seen as putting an anchor on each agent in addition to the tag it would carry if it was localized with landmarks. Each module is efficient in its goal, either listening to the channel (for the responder) or polling for distance updates (for the initiator). It is, therefore, the best option compared to the objective of the thesis, which led to its implementation. However, it comes at the expense of a second module for each robot and having many modules initiating the ranging process at each time step. This can cause interference, making the ALOHA scheme insufficient for greater sizes of swarms.

4.2 Communication Protocol

To enable relative localization using distance measurements alone, each agent must have access to more information than its relative distance from the other agents. Intuitively, applying the ranging process alone does not allow for having access to all the information required to localize the swarm as it only provides distances between the initiating agent and other agents, pairwise distance of other agents are required. To this extent, a communication protocol must also be implemented to share relative distances within the swarm, thus enabling *Cooperative Relative Localization*, defined as localization enabled by cooperation between nodes. Cooperation is typically achieved whether by modifying the movement of agents, sharing sensor data between them, or combining both.

Communication can be achieved through Wi-Fi, Bluetooth, Infra-Red, Ultra-Wideband, and other technologies. However, as UWB is alreadu used to communicate distance estimation messages, it can be leveraged to integrate data transmission into the existing communication protocol. Figure 4.3 and 4.4 show the proposed communication protocol.

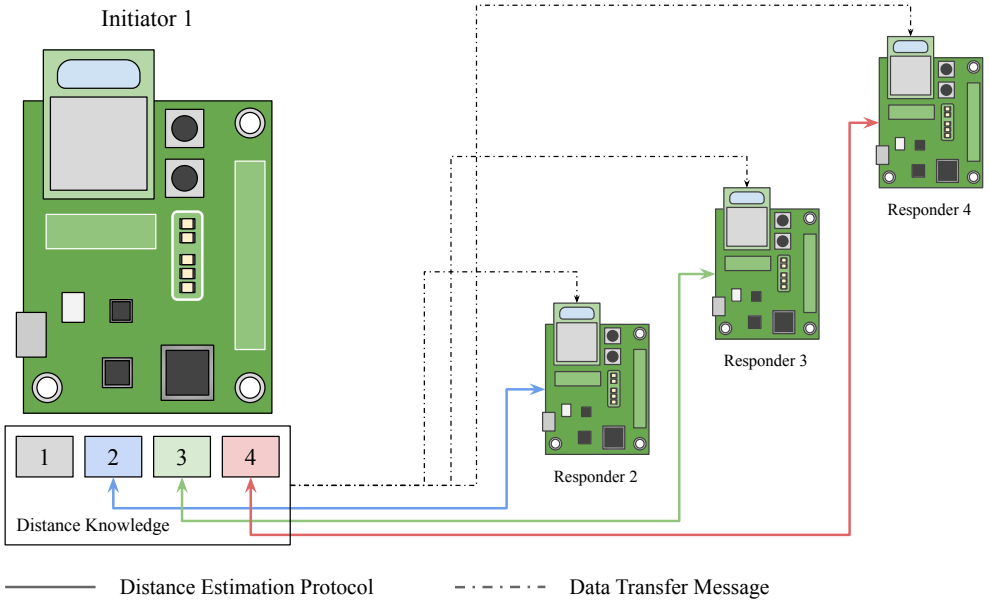


Figure 4.3: Initiator side of the communication protocol (from agent 1)

The first part, described above, is the distance estimation phase; the initiator iteratively requests an acknowledgment from each responder module in the swarm to calculate the distance estimate. The second part is the data transfer phase; the initiator will broadcast a message containing its knowledge of distances, which any currently listening responder will receive.

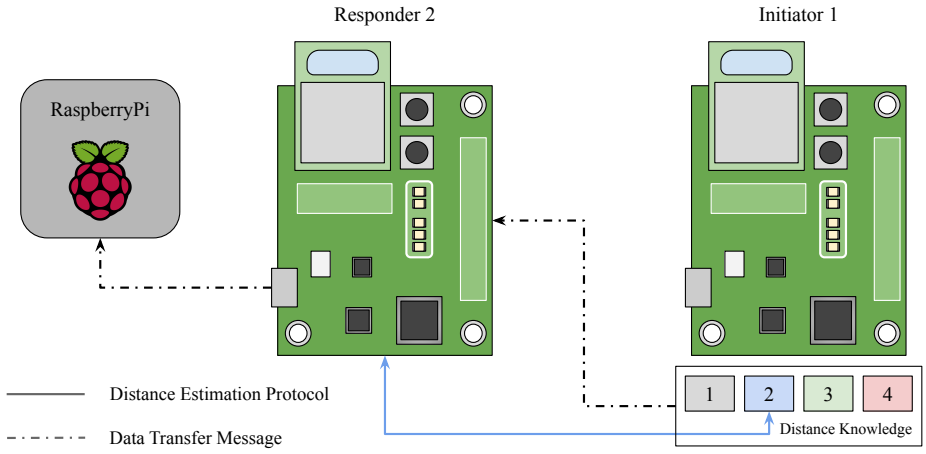


Figure 4.4: Responder side of the communication protocol (from agent 2)

On the other side of the communication channel, the responder will respond to incoming poll messages from the different initiators in the swarm and sometimes receive data messages containing the known distance of a given agent. When receiving a data message, the responder will forward it to a RaspberryPi through the serial port. This implementation generalizes to messages containing any type of data; it only requires communicating the data to add to the message to the initiator via the RaspberryPi.

4.3 Localization

4.3.1 Trilateration

Trilateration is an essential concept in relative localization, as it is one of the most widely used techniques in landmarked environments. However, its application to landmark-free contexts at the early stages of this thesis work has proved unreliable. Therefore, it will only be used to give an intuition about the ambiguity of relative distance-based localization in such contexts.

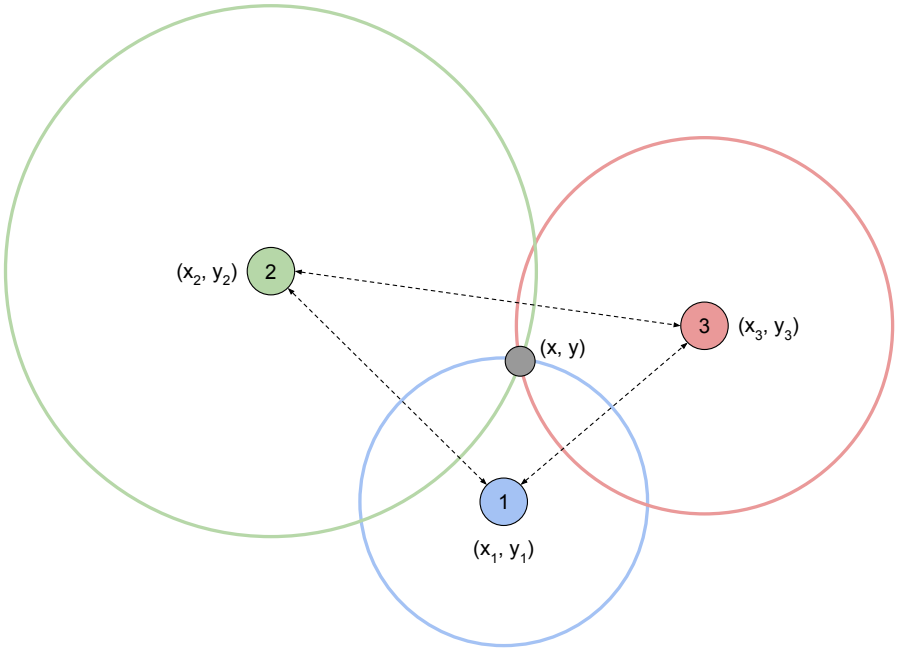
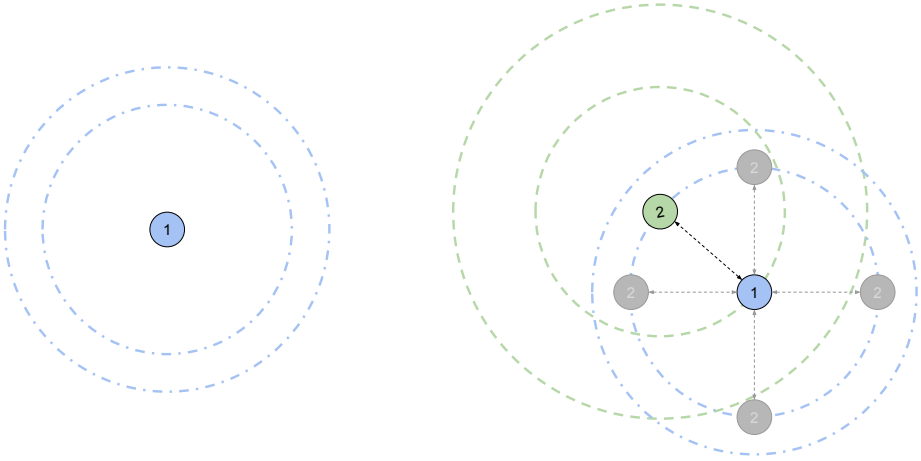


Figure 4.5: Trilateration of a moving object (x, y) using non-moving nodes (nodes 1, 2 and 3)

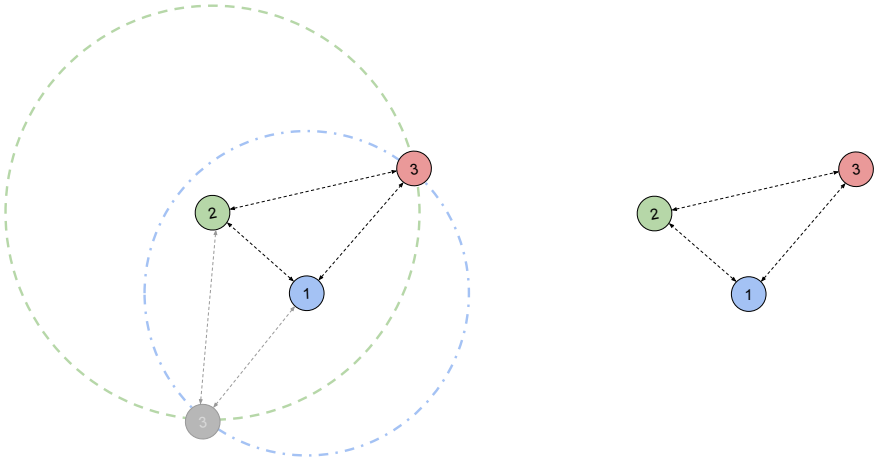
As shown in Figure 4.5, using reference points with known positions, called anchors, the distances between the node of interest, commonly called the tag, and the different anchors allow for finding the common intersection in their relative coordinates frame to position the tag. Trilateration differs from triangulation by nature. While trilateration uses distances to constrain a node's placement possibilities, triangulation uses angles from landmarks to tags. Therefore, it requires methods that measure the angle of arrival of the localization message rather than the measure of the distance between the nodes.

As described above, this thesis aims to remove the need for landmarks in the localization process to enable the relative localization of moving robots without external infrastructure and without constraining their movement. In most cases, similar studies choose between the different possibilities of positioning the first three nodes, as represented in Figure 4.6, which allows placing all other nodes relative to the first three. A typical choice (not represented here) is to put the first node on the origin, the second node on the x-axis at the right distance of the first one, and the third node in the positive y-axis values.



(a) Starting Knowledge

(b) Placement of Second Node



(c) Placement of Third Node

(d) Final Result

Figure 4.6: Relative trilateration without known-position landmarks

However, some ambiguities appear due to the relative-distance localization performed by trilateration without reference points :

- Intuition of the rotation ambiguity is shown in Figure 4.6 (b): from the distances between nodes 1 and 2, it is impossible to recover the second node's position. It only describes a circular locus around node 1.
- Intuition of the flip ambiguity is shown in Figure 4.6 (c): from the distances between node 1-3 and node 2-3, it is impossible to position the third node with total certainty.

Unfortunately, detecting and correcting these ambiguities is impossible based on the static knowledge of the situation currently described. Trilateration, therefore, requires a minimum of three reference points to avoid ambiguity. Furthermore, better performances can be achieved with four or more landmarks, and as the network grows and spreads, it becomes interesting to have more landmarks well distributed, enabling the localization process to further increase the quality of the results.

Rotations are not possibly detectable in the context of this thesis. As no infrastructure and no sensors, such as compass, allows for placing nodes in a more global frame of reference, rotation detection and correction are not possible. Therefore, we are concentrating solely on what can be learned without the knowledge of rotation.

Limitations

This section justifies the reasons for abandoning the trilateration method. As shown above, trilateration techniques are typically applied in a landmarked situation, where known position nodes are used to fix all moving nodes in a common reference frame. When trying to implement them for relative localization between moving nodes, each node can only be placed based on other nodes, which requires deciding which nodes to place first and iteratively deciding in which order to place nodes based on the relative measurements to nodes already placed. This raises several important points that limit the application of trilateration techniques in the relative localization of a moving network.

Iterative processes on uncertain measurements showed increasing error as more nodes were placed, even more in cases with noise on the measurements. Some heuristics based on the certainty of the measurements (considering measurement age and previous measurements) were implemented without any success. Techniques based on least squares and other noise mitigation processes were studied without more success. Even though trilateration showed good results in landmarked environments, implementation made for the thesis greatly suffered from the uncertainty brought by localization based on moving nodes and was never successful.

To avoid the need for iterative decision heuristics and the error it can bring, some solutions using mathematical systems based on triangular inequality relaxations were tested. However, each added measurement represented another constraint in the mathematical system, which significantly increased the computation overhead the more was known about the system. While relaxations typically improve the run-time of such algorithms, no practical middle ground was found between estimation quality and computation overhead. This is mainly due to both the computation time and the errors in the measurement requiring relaxation of the problem.

In conclusion, trilateration techniques could have shown better results in the context of moving networks with an external structure on which to base the localization of the moving nodes. In a landmarked context, each new measurement from a landmark to the moving node improves the localization quality, and four nodes are enough to obtain a centimeter precision. Hence, the main application of DMW1001-DEV modules is using an anchor-tag structure to obtain the advertised accuracy in positioning the tags in the network. However, in the setting without landmarks, the above-mentioned failed attempts did not show promising results, which led to the hypothesis that trilateration was not fitted for the work. Therefore, the implementations discussed in the next section are based on the observed limitations of trilateration. The main take away of this section is the intuition behind the different ambiguities that arise from relative localization based solely on range measurements.

4.3.2 Multidimensional Scaling

Multidimensional scaling (MDS) is a standard solution for relative localization. Based on the previous conclusion on trilateration techniques, optimization procedures could be the solution for placing nodes in two-dimensional space with the minimum error on all nodes.

Practically speaking, Multidimensional Scaling is an iterative procedure that aims to optimize the following loss function $\sigma(X)$ called the stress, i.e., the weighted mean square error of the relative distances of each placed node. Intuitively, if the distance between node i and node j is not respected, it increases the stress, decreasing the results quality.

$$\sigma(X) = \sum_{i<j\leq n} w_{ij}(d_{ij}(X) - \delta_{ij})^2, \tag{4.1}$$

Where $w_{ij} \geq 0$ is a weight given to the pairwise distance from node i to node j and can serve as a certainty factor, $d_{ij}(X)$ is the Euclidean distance between the current estimated positions of the nodes i and j, δ_{ij} is the actual distance between the nodes i and j, that is estimated through pairwise range measurements using RToF.

There are different ways to optimize this loss function. In this thesis, the SMACOF implementation found in the *scikit-learn* Python package was used, slightly modified to accept weights after de Leeuw et al. in [83]. SMACOF, or "Scaling by MAjorizing a COmplicated Function", is an algorithm that leverages bounds on the different components of the loss function to simplify the optimization process.

Starting with the classical loss function $\sigma(X)$, we can extend it to find:

$$\sigma(X) = \sum_{i<j} w_{ij}\delta_{ij}^2 + \sum_{i<j} w_{ij}d_{ij}^2(X) - 2\sum_{i<j} w_{ij}\delta_{ij}d_{ij}(X), \tag{4.2}$$

Where $\sigma(X) = \sum_{i<j} w_{ij}\delta_{ij}^2$ is a constant, $\sum_{i<j} w_{ij}d_{ij}^2(X)$ is quadratic in X, and can be rewritten for better computability, and $\sum_{i<j} w_{ij}\delta_{ij}d_{ij}(X)$ can be minorized by a linear function, which allows for a majorization of the stress $\sigma(X)$. Using the upper bound, the loss function becomes easier to optimize.

To mitigate the fact that the distance matrix will not be updated at each iteration (latency in the communication), a certainty variant of MDS was implemented based on the fact that we can add weight to each measurement. The weights are based on the time since the measurement was taken and decrease by a factor of 0.99 at each iteration.

Unfortunately, MDS is not suited for real-time localization by design as it aims to optimize a stress function that is only constrained by the distances between the nodes and not the past positions of the nodes. It is impossible to make total sense of the displacement observed in the position estimate over time. However, the optimization algorithm allows for the initial state to be decided. Previous estimations can be input as the initial value for optimization, allowing for the algorithm to search its local minimum at a position that has physically more sense than flickering solutions of the base MDS algorithm with random initialization. This observation led to the implementation of the initialized MDS variant and the particle filter discussed in the next section.

4.3.3 Particle Filter

The particle filter is a Bayesian filter, which consists, after random initialization of the particles, in three steps repeated until the end of the computation [84] :

1. **Initialize** the state of the particle filter. The initialization step randomly draws a first list of N states $\mathbf{x} = (x, y, \theta)$ that we call particles. They contain a position (x, y) and an orientation (θ) information. The initialization variant of the particle filter consists of giving starting relative positions as information, therefore sampling the starting particles around the given coordinates.

2. **Predict** the particle movement using the system's dynamic model. The prediction step consists of making the particles move depending on the state they represent by following the movement equation of the system. Logically, the better the movement law describes the system, the better the results.

   In our case, random walk is considered due to the lack of information on the system's dynamic. The random walk mainly consists of making the particles move linearly in a random direction. A constant linear speed of 30cm/s is considered to match the RVR's speed, and the rotation speed is described as null but with an error of 2π, which effectively causes the particles to choose a new direction randomly at each step.

3. **Update** the particle distribution using real-life measurements through the following formula :

   $$\overline{\mathbb{P}(\mathbf{x})} = \|\mathcal{L} \cdot \mathbb{P}(\mathbf{x})\|, \tag{4.3}$$

   Where $\mathcal{L}$ is the likelihood, $\mathbb{P}(\mathbf{x})$ is the prior distribution of $\mathbf{x}$. The norm is used to normalize the posterior probability distribution of $\mathbf{x}$, i.e. $\overline{\mathbb{P}(\mathbf{x})}$. This is a realization of the Bayes theorem (Equation (4.4))

   $$P(\mathbf{x}|\mathbf{z}) = \frac{P(\mathbf{z}|\mathbf{x})P(\mathbf{x})}{P(\mathbf{z})}, \tag{4.4}$$

   Where $\mathbf{z}$ represents the measurements of the real system, and $\mathbf{x}$ is the estimated position of the agent. Intuitively, the estimated quality of the particles when seeing the measurement $\mathbf{z}$ (i.e. the posterior probability $P(\mathbf{x}|\mathbf{z})$) is proportional to the probability of the agent being in position $\mathbf{x}$ (i.e. $P(\mathbf{x})$, which is estimated by the previous iteration) multiplied by the likelihood of obtaining measurement $\mathbf{Z}$ when the agent is placed in position $\mathbf{x}$ (i.e. $P(\mathbf{z}|\mathbf{x})$), up to a normalizing factor $P(\mathbf{z})$.

   In the particle filter case, the certainty variant removes the uncertain measurements before applying the update. Therefore, old measurements do not impact the estimated probability distribution.

4. **Resample** the particles. The resampling step redraws new particles to match the current probability distribution estimate better if too many particles have a bad fitness.

The particle filter was implemented by modifying the filter proposed in [84]. The main difference is that the dependence on landmarks had to be removed and replaced by pairwise measurements of other agents. To do so, from the point of view of any agent, each other agent is simulated by another particle filter, which allows for decoupling the different estimations. The other particle filters are used to estimate the position from which to consider the measured pairwise distances. However, this may cause computational overhead and will not allow large-scale implementations.

4.3.4 Direction

Localization is the fusion of both position and orientation estimation. While robotic platforms would typically be equipped with compasses to put their current direction into the reference frame of Earth's magnetic field, the thesis focuses on what can be achieved by pairwise distance alone. This section describes three methods imagined to estimate the direction of the agents based on the positioning algorithm previously discussed and pairwise distance measurement over time.

Distances Evolution Estimation

Based on distance alone localization, the first implementation of direction estimation was designed using the evolution of the distance matrix. The basic idea is that the relative evolution of distances could give a rough estimation of the agent's direction relative to the other agents of the swarm.

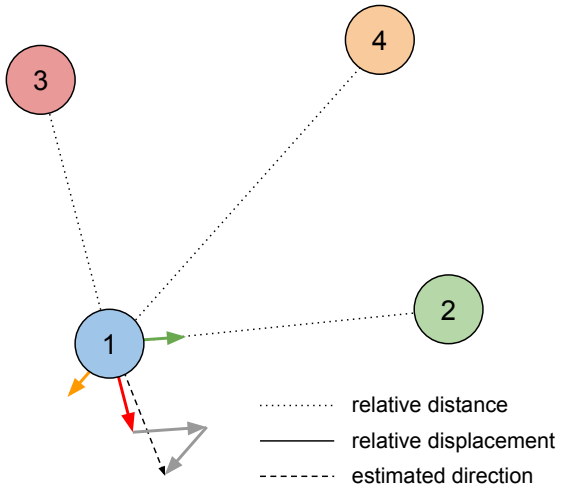


Figure 4.7: Direction Estimation through pairwise distance evolution

Figure 4.7 showcases the main concept of direction estimation with distance evolution. The different vectors composing the distance estimation are aligned to the vectors connecting the other nodes to the first node, with the direction chosen depending on the sign of the distance evolution (positive when moving away from each other). A direction angle can then be computed relative to the estimated positions using the position estimate generated by the previously described positioning algorithms.

While it worked well in a landmarked situation in simulation, the quality decreased too much in landmark-free environments to be usable. The main reason was that all the agents moving in seemingly random directions caused the estimation to generate small direction vectors that could be in any direction depending on the noise of the measurements. This observation led to the implementation of a distance history, which allows to take distances from earlier times and, therefore, increase the vector size with greater distance evolution.

Displacement Estimation

Another investigated option is to use the previous position estimations as information for the direction estimation. The idea is to use the positions over time to recover the displacement vectors. In this case, the historic variant of the direction estimation works as follows: the larger the history, the more of the previous positions are considered while aggregating the results with a moving average. The moving average is a time-variant of the classical average that gives more importance to recent values.

However, for this to work, the position estimation must be stable, meaning that the localization $\mathbf{x}(t)$, should make sense compared to the previous estimation $\mathbf{x}(t-1)$, by following some dynamic law explaining the movement between the two time steps. This will typically not be the case if the method is not designed for it. For example, a trilateration heuristic, which always puts the first node at the origin and the second on the positive x-axis, will generate cases where the first node has moved while the estimated position has not changed over time, causing other ambiguities.

Usually, it would suffice to have landmarks to overcome this problem; fixed nodes would allow for rotating and translating the estimate to match the previous position of those same nodes, but the aim is to have no landmarks, which adds the previously discussed flip and rotation ambiguities. Therefore, the best candidates for this method are the initialized MDS, which starts from a predefined previous position, and the particle filter, which attempts to simulate the movement of agents over time. Nevertheless, we can expect undesirable behavior from both methods since the initialized MDS updates the position to minimize stress without trying to make sense of the movement, and the particle filter is mainly probabilistic, which could also lead to unexpected results.

Particle Filter Estimation

As described in the section on the particle filter, each particle also has its angle (relative to the simulation axis) as part of the state to predict agent displacement. We can, therefore, use this information to aggregate an estimate of each agent's current direction, in the same way as the filter aggregates particles' positions :

$$\mu = \frac{1}{n} \sum_{i=1}^{n} w_i \mathbf{x}_i, \tag{4.5}$$

where n is the number of particles, w_i is the weight of each particle and $\mathbf{x}_i$ is the particle state (x, y, θ). Which gives the estimate μ of the position and heading of the agent. However, the system's dynamic being estimated through random walk will not help the particle filter make sense of the particles' displacement. It would, therefore, be a better estimate after adding sensor fusion with odometry data.

4.4 Simulation Parameters

Since a large part of the thesis is based on UWB communication, the simulation results must be cautiously separated from a real-life situation, as simulated communication protocols are often far from the reality of communication and all its physical constraints. While good simulation results may not be transposed to reality, bad simulation results will undoubtedly lead to bad results. Therefore, it is interesting to discuss the weaknesses and, only to some extent, the strengths of each of the methods and variants described above in the simulation. To try and get closer to reality, the following parameters of the simulation can be tweaked: the message *drop rate* of the simulated communication modules, the range *measurements noise* of the simulated sensor, and the simulation *iteration rate*.

The above-discussed noise model is applied to the simulation except for the computed offset. Manually adding the offset we computed in the previous section would make no sense as it would exactly match the algorithm variants considering that model. Therefore, only the computed standard deviation is added as noise to the measurements, and the study of the simulation results will not make use of the offset variants of each algorithm.

Further analysis of the system is required to determine the correct drop rate and iteration rate. Let us first derive the meaning of the drop rate; the probability of receiving a message from at least one of the three other agents at a given timestep is computed as follows:

$$P(m \geq 1) = 3 \cdot p_d^2 \cdot (1 - p_d) + 3 \cdot p_d \cdot (1 - p_d)^2 + (1 - p_d)^3, \tag{4.6}$$

Where m is the number of messages received at a given simulation step, p_d is the probability of a frame being dropped (effectively, the drop rate). The probability is computed as the sum of the probability of each resolution that matches the goal (i.e. receiving all three, the three possible combinations of receiving two of the three, and the three possible combinations of receiving one of the three). Which is more convenient to compute the other way around:

$$P(m \geq 1) = 1 - P(m < 1) = 1 - P(m = 0) = 1 - p_d^3. \tag{4.7}$$

This can easily be transferred to a case with n robots:

$$P_n(m \geq 1) = 1 - p_d^n. \tag{4.8}$$

This probability can effectively be estimated by counting the number of messages received by an agent over time, allowing for deciding on the optimal drop rate for the simulation :

$$\text{message\_per\_second} = (1 - p_d^3) \cdot \text{iteration\_rate} \tag{4.9}$$

To ensure that the correct number of messages is transferred over one second, the simulation will randomly choose one message per iteration and discard all the others.

4.5 Summarized Architecture

A RaspberryPi is mounted on each robot and connected to the two modules to calculate the position of each agent. It will receive information on the distance estimations of the modules and use this to calculate the localization. Figure 4.8 shows the final architecture mounted on each swarm robot.

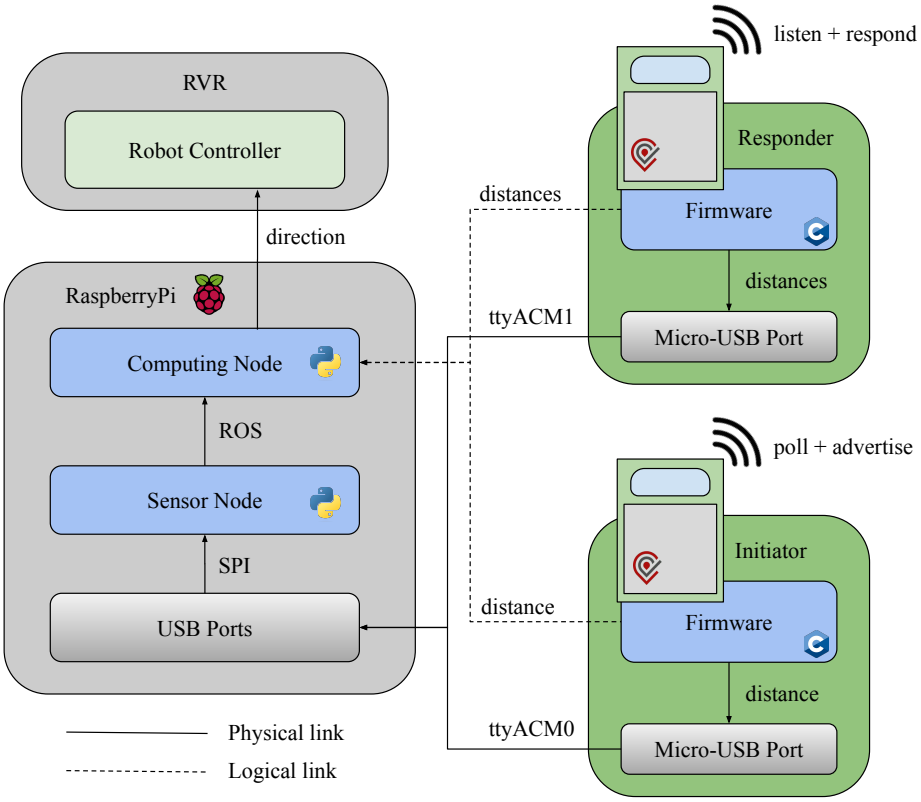


Figure 4.8: Interface between DWM1001-DEV Modules and the Robot Controller

First, we have the operation of each UWB module, which was explained in the previous section. Then, we have the calculation part, where, as mentioned above, the UWB modules share their knowledge with the RaspberryPi through the serial port. The RaspberryPi hosts two Python processes, one to read data from the serial port and the other to calculate agent positions in real-time, both communicating over ROS topics. Finally, it would be possible to communicate information to the robot controller if we wanted the calculated location to have an impact, typically to enable morphogenesis behaviors such as aggregation or repulsion.

Chapter 5

Experiments

5.1 Experimental Setup

This section explains the configuration used to carry out the various experiments in the physical and simulated contexts.

5.1.1 Physical Setup

For the experiments outside of the simulation, the previously introduced Mercator robots were equipped with an initiator/responder pair of UWB modules. The agents evolved in a dodecagonal arena with a diameter of 2.5 meters.

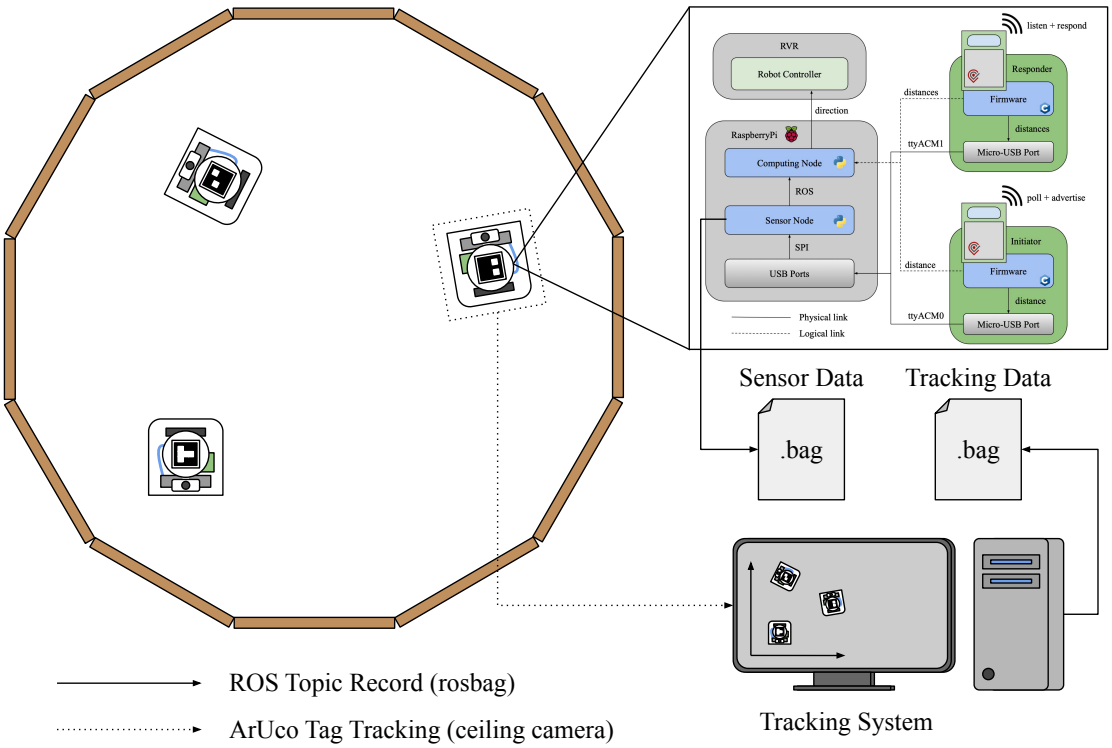


Figure 5.1: Experimental setup data flow, three agents placed in the arena, the tracking system knowledge and internal architecture of agents (described in Section 4.5).

As depicted in Figure 5.1, during experiments, the actual positions were recovered using a video tracking system based on the recognition of ArUco tags. Each sensor's measurement during the experiment was recorded using rosbag to re-simulate the experiment as many times as needed. Before each experiment, the tracking system and the RaspberryPi clocks were synchronized through their internet connection. Recordings from rosbag have timestamps based on the system clock for every recorded message, which allows for easy synchronization of the data of the different bags while replaying the experiment.

5.1.2 Simulation Setup

The simulations were carried out using ARGoS3, presented in section 3.2. As the main objective of the simulation was to test the system with an increasing number of agents, the arena was chosen as a square of about 4 meters on a side, deliberately larger to accommodate more robots. The foot-bots plugin was used to simulate the agents, implemented with approximately the same number of proximity sensors as Mercator to enable random walking in the arena. The range and bearing sensor/actuator plugin was used to simulate the UWB modules. While bearing measurements were dropped, distance measurements were used to generate the swarm distance estimate. The range and bearing actuator was used to transmit information between agents and, therefore, simulate the data communication of the physical system.

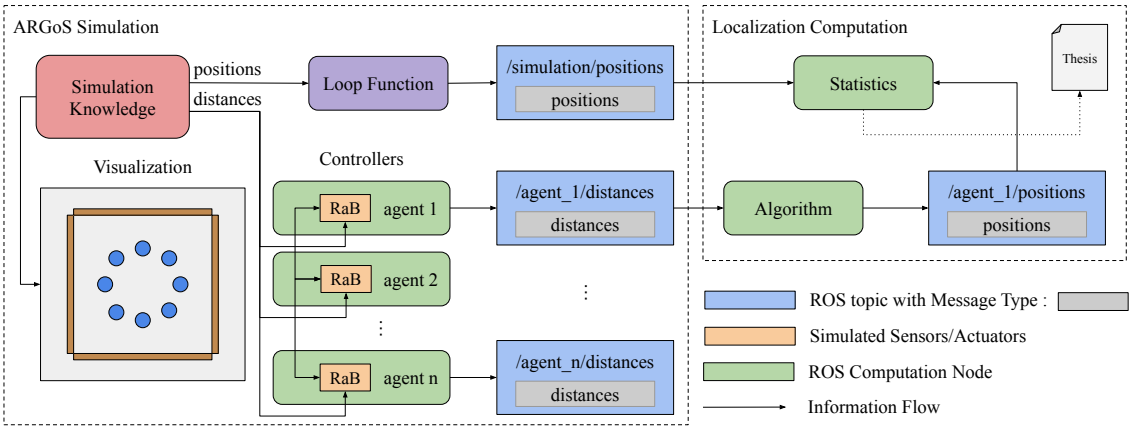


Figure 5.2: Simulation setup architecture scheme, showing the data flow between the different components of the simulation.

On the one hand, the tracking system was simulated using a *loop-function*; a controller with access to all the simulation information, running in the background during the experiments in ARGoS. It mainly consisted of publishing each simulated robot's exact position and orientation on the related ROS topic. On the other hand, the control software of the robots simulated a random walk and published sensor data to each linked ROS topic. Finally, the localization computation was implemented using ROS nodes reading data from the various ROS topics in which the loop-function and robot controller were publishing. Localization algorithms were applied in real-time on top of the simulation to reproduce some of the communication delays introduced by ROS. (see Figure 5.2)

5.2 Experiments

Two main types of experiments were carried out as part of the thesis: static experiments, in which UWB communication modules were placed at known positions to study the system without the randomness involved in module movement, and dynamic experiments, in which communication modules were placed on moving agents to study the system with movement.

5.2.1 Static Experiments

The study of dynamic systems involving several agents is complex. Therefore, static experiments were carried out to simplify the analysis and generate a basic intuition of the system's underlying rules. The main interest of these experiments was to derive a noise model for range measurements and to characterize the communication channel when using the implemented communication protocol.

Noise Model Experiment

The experiment aimed to characterize the noise on the measurement made with the UWB modules using two-way ranging. The derivation of the noise model has two main interests. On the one hand, studying the estimate's accuracy could lead to important information, such as a significant bias in the measurement, which could be corrected manually in the localization process. On the other hand, the derived noise model enables us to understand which parameters to give to the simulation to bring it closer to the reality of the ranging process.

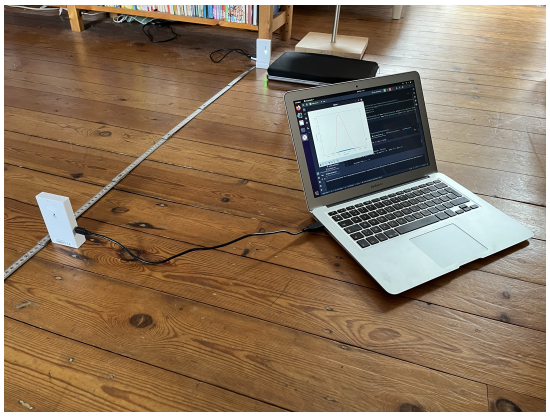


Figure 5.3: Experimental setup for analyzing the static distribution of the RToF estimated distances.

As shown in Figure 6.1, two UWB modules (one initiator and one responder), placed facing each other, were used to measure distance. A folding meter was used to measure the actual distance between the two modules. The distance measurements generated at the initiator were saved over 30 seconds using a USB cable connected to a computer, recording the data shared by the communication module. Finally, the experiment was carried out 20 times, starting with 10 centimeters separating the two modules up to 2 meters apart, with steps of 10 centimeters. The experiment's output is a distribution of the range measured for different known distances, which allows the study of the accuracy of RToF measurements with UWB modules compared to reality.

Channel Characterization Experiment

The main benefit of channel characterization is that it enables a comparison between the implemented communication protocol and future implementations such as TDMA. A positive impact of knowing the channel's communication frequency is that it enables the proper simulation parameters to be chosen, bringing the simulation closer to the reality of physical communication, i.e., interference or other physical phenomena that are difficult to model in simulation.

For the characterization of the communication channel, four pairs of UWB modules were placed in a random configuration, each pair composed of one initiator module and one responder module tied against each other. For each experiment, modules were placed in another configuration, and for the duration of the experiment (60 seconds), the number of messages received at the initiator and responder modules were saved independently.

- Message count of initiator: amount of successful distance estimation
- Message count of responder: amount of successful data communication

This allows for computing the frequency of distance estimations and data communication at a given swarm node.

Data Convergence Experiment

The main objective of this experiment was to gain insight into the speed at which the system converges, as well as the speed of data flow. The goal was also to study the behavior of the different localization algorithms and variants in a static setting.

Figure 5.4: Experimental setup for static convergence analysis

As shown in Figure 5.4, the module pairs were placed in a known configuration (in a square shape with edges of 1.6 meters) to allow for comparing the quality of the position estimation with regard to the actual configuration. One pair was connected to a computer to recover and save the data it received throughout the experiment. The three other pairs were connected to the same multi-plug to be switched on simultaneously. Each experiment batch started with all modules switched off except for the modules connected to the computer, which were reset manually before starting each experiment. Directly after starting the experiment on the computer (record of data over time), the other

modules were started. Therefore, a delay of about one second can be expected due to the human intervention required, which should not be considered when discussing the results. The experiment was repeated five times for 60 seconds to enable the averaging of results over different batches of data and to avoid outliers leading to erroneous conclusions. The quality of each algorithm's position estimation was calculated using the difference with the known positions in which the modules are placed.

5.2.2 Dynamic Experiments

Building on the results of the static experiments, experiments in a dynamic context were conducted to study the system in a realistic setting. On the one hand, dynamic experiments were conducted with an increasing number of moving agents. On the other hand, experiments were conducted in simulation to study the scalability of the system and its dependence on noise, which are difficult to test in a real setting, as only a limited number of robots are available. The noise due to increasing interference is not easily generated in reality without increasing the number of modules and, therefore, the cost of the installation. All dynamical experiments are 60-second long runs of the agents moving around the arena; the localization computation is restricted to 10 times per second.

Physical Setting

Two experiments were conducted using the setup described above to assess the overall quality of the localization protocol in different settings. One experiment was dedicated to trying the algorithms in a setting close to reality, with all the agents moving. The other experiment was conducted with one agent moving while the others were at rest to act as landmarks. For all the experiments in the physical setting, the experimental protocol consisted of the following:

1. Booting up the robots, subsequently starting the communication modules.
2. Launching the tracking system and starting the tracking data recording.
3. Starting the sensor data recording in the RaspberryPi of each agent.
4. Putting the required agents in motion for a given duration.

Simulation Setting

While the simulation was mainly used for rapidly implementing the different algorithms and creating proofs-of-concept, it constituted an important tool to test the proposed implementations in settings that are difficult to recreate in reality. To that extent, different experiments were carried out with an increasing number of agents. The idea was to show whether the system could adapt to a large number of agents in the perfect simulation environment and assess the deterioration in localization quality as the number of agents increased.

The experimental protocol consisted of, for each number of agents considered, running all the components of the simulation setup described above, i.e., the ARGoS experiment, generating data and publishing them on given ROS topics in real-time, and the localization pipeline, computing localization from the generated data and comparing them with the agents' actual positions.

5.3 Quality Assessment

This section describes the different measures used to evaluate relative localization quality and compare the implementations described in the previous chapters. They are divided into two categories: position quality, which focuses on how algorithms position agents in a 2D plane relative to each other, and direction quality, which focuses on the quality of the direction estimate of the agent.

Preliminary note: as mentioned in Section 4.3.1 on trilateration, algorithms are prone to rotation and flip ambiguities when trying to obtain a relative localization with distance alone and without landmarks. Although solutions have been proposed to detect these ambiguities by taking advantage of node motion [85], they generally require retrieving all distance measurements at the new node position in a single time step, which works well in simulation but is virtually impossible in a real environment with the underlying communication constraints. This is why, before applying the above measures, an optimal rotation, inversion (flip) and translation transformation, found with the Kabsch-Umeyama algorithm, is applied to the points with the actual positions as reference points. Inversion can occur when the points are rotated; to detect this, we can calculate the determinant of the rotation matrix; if the determinant is negative, an inversion has occurred. Consequently, when calculating statistics, it is possible to detect whether the localization algorithm generates an inverted version of the real positions.

5.3.1 Position Quality

Position quality is estimated by calculating the mean square error of estimated versus actual positions. This means that the metric is lower when all error estimates are small and higher when the errors on individual estimates are significant. The metric gives the best results when most of the robots are well placed, while poor results are due to one or more robots being completely misplaced.

5.3.2 Direction Quality

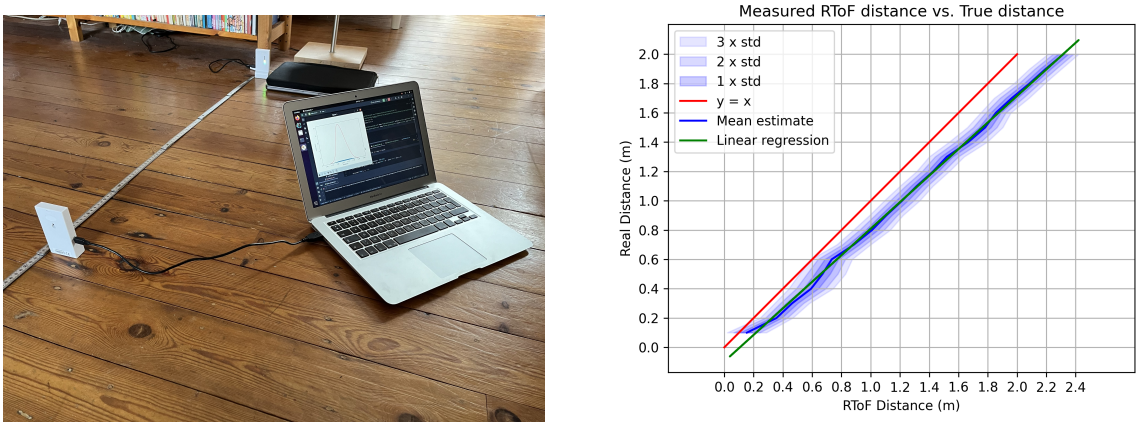
Agent direction knowledge is essential for enabling morphogenesis behaviors. The most crucial direction estimate is the one of the agent currently calculating relative localization before performing its next step. As a result, the metric for evaluating the quality of the heading is the error on the angle estimate of a given agent i when calculated using its knowledge. The error is the difference between the actual angle and the estimated angle, modulo 2π and shifted to obtain an error of $\pm\pi$ (either too far left or too far right, relative to the actual heading). Then, to avoid going from π to $-\pi$ between two time steps while only changing the estimate from 2 degrees, the absolute value of the error is considered, resulting in an undirected error between 0 and π. This metric will ensure that random guessing of the direction will not average the error to 0° but to 90°. It will allow to differentiate better methods that increase the orientation estimation quality from methods that create a random estimate averaged to 0° of error.

Chapter 6

Results

6.1 Noise Model

A static distribution experiment was conducted to study the error in the measurement estimation made with RToF (Round Time of Flight) using the DWM1001-DEV modules and the communication protocol depicted in Section 4.2.



(a) Experimental Set-up

(b) Results

Figure 6.1: Study of the static distribution of RToF measurements compared to the real distance

The results obtained at short distances show that the RToF estimate systematically overestimates the distance by around 20 centimeters. A linear regression was applied to the data, and although it showed the linearity of the range estimate, a slope of 0.906 was found, leading to the hypothesis that the error is 10 centimeters higher as the distance to the responder module increases by 1 meter. While this might not be the case for measurements of higher magnitudes, all experiments were conducted in spaces where no measurement could exceed 3 meters, which removes the need to model the error for greater distances.

Since the resulting noise model has a convenient form, an *offset* variant has been implemented for each algorithm. It uses the derived noise model to transform the measurements before localization computation, expecting it to describe the actual distances better.

6.2 Channel Characterization

An experiment was conducted to characterize the communication channel, counting the messages received from ranging and communication separately. The results are averaged for four experiments, each in a different position configuration.

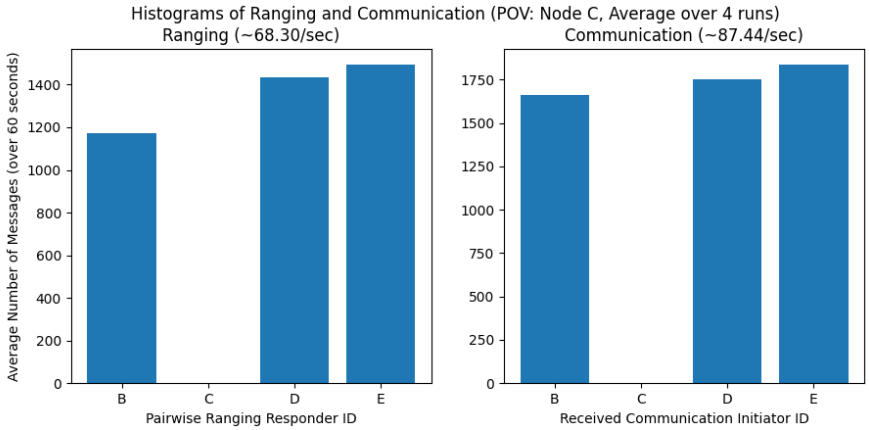


Figure 6.2: Study of the communication frequency averaged over different static settings, frequency separated by communication type and by agents communicating with the studied node C

The experiment results in Figure 6.2 show about 70 messages per second for the ranging process and 90 messages per second for the data communication process. Communications seem to be well distributed between the different agents, except for node C, which does not communicate with itself through UWB. However, fewer messages were received from node B, which shows that some other factors not present in the simulation can change the communication properties. Some examples are the closeness to the agent, the multi-path effect, some obstruction between the nodes, or a random synchronization between two nodes hindering communication with other nodes.

Based on the obtained frequency of communication, the following parameters were used for the simulation :

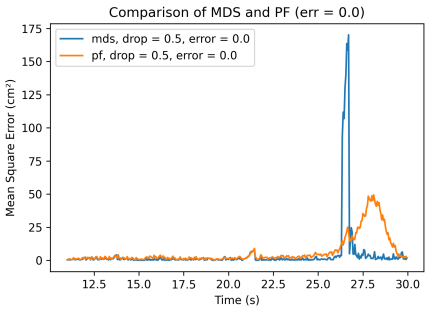
- iteration rate: set to 100 to allow for up to 100 successful messages per second.
- drop rate: set to 0.5 under the equation (4.9) :

$$(1 - 0.5^3) \cdot 100 = 87,5 \tag{6.1}$$

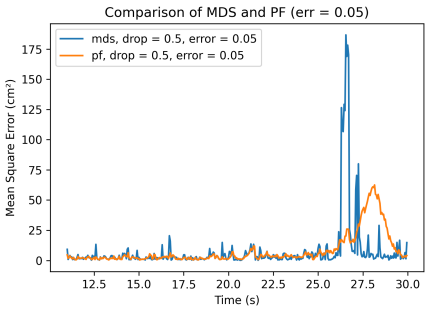
Under those parameters, the simulation has the same amount of successful transmissions per second as the experiment, allowing for approximately the same data transmission rate. To match the number of new measurements made per second, an 80% probability of dropping the new measurement coming from the message transmission was added. It causes the number of new measurements per second to be $87.5 \cdot 0.8 = 70$, which gets the simulation to approximately the same ranging frequency as the experiment.

6.2.1 Parameter Assessment

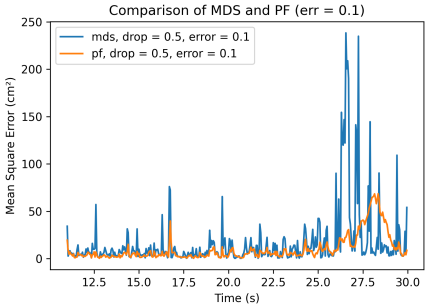
This section compares the base results for dynamic experiments between real settings and simulations. Figure 6.3 summarizes the results for different values of noise standard deviation in the simulation.



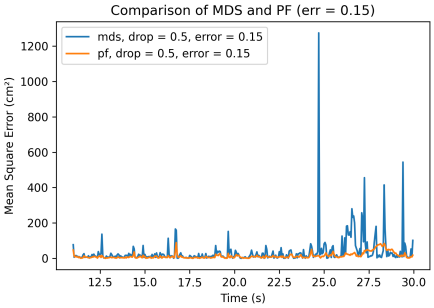
(a) MDS vs. PF, noise standard deviation of 0cm



(b) MDS vs. PF, noise standard deviation of 5cm



(c) MDS vs. PF, noise standard deviation of 10cm



(d) MDS vs. PF, noise standard deviation of 15cm

Figure 6.3: Study of the effect of noise standard deviation on the localization quality using MDS and Particle Filter in simulation

Later results leveraging the real implementation (see Figure 6.4) tend to corroborate the previously defined noise model and dynamic behavior. The results show that the error on the localization is approximately the same (error bound between 0 and $100cm^2$) as for the simulation case using an error standard deviation of 10 to 15 centimeters.

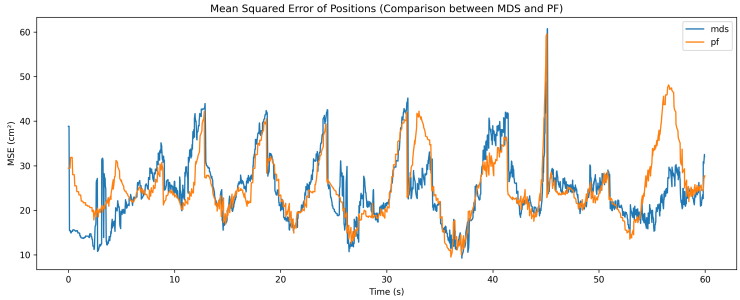


Figure 6.4: Study of the localization quality between MDS and Particle Filter, comparison between their mean square error over time.

However, the attenuation of particle filter errors is not observed anymore. The main

reasons for this difference could be, firstly, that the noise model is not perfect and, secondly, that the particle filter parameters are over-adjusted for the simulation. Various parameters were therefore tested to improve the particle filter results but without success. Although the results improved, they only got closer to the MDS results, with the same error curves in some experimental cases (shown in Figure 6.5).

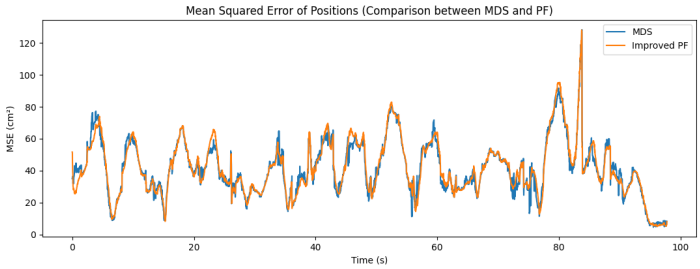


Figure 6.5: Study of the localization quality between MDS and Particle Filter with improved parameters, comparison between their mean square error over time.

This last result highlights a limitation in estimation quality, which, given the total opposition in their implementation, does not depend on the algorithms but rather on the information available and its quality. Suppose the quality of the estimate was limited by the data available. In that case, it may be due either to the speed of transmission (out-of-date measurements) or to the quality of the measurements (excessively noisy measurements on the tracking system or the UWB modules).

This discrepancy does not appear in the simulation, but well in the results of the experimental setup; the main causes of error in the experimental setup compared to the simulation should be the following:

- Errors in the positions computed by the tracking, mainly due to the loss of ArUco tags trace at some point, causing translation of the agents to undesired positions.
- The position of the UWB modules next to the ArUco tags, which generates an offset of 5 to 10 centimeters between the actual center of the robot (as seen by the tracking system) and that which the localization algorithm with perfect range measurements would estimate.
- The measurement bias, which, as described in Section 6.1, increases with distance and can reach 20 to 30 centimeters for measurements over 2 meters.
- The system dynamics, which means that information always arrives later than it was measured since agents move at speeds of up to $50cm/s$.
- Finally, data propagation and computation time mean that the estimate is time-shifted compared to the agent's actual positions.

The tracking system data were manually shifted in time to check the impact of the last source of error. The results in Figure 6.6 show that it takes some time for the system data to reach the localization algorithm, resulting in a delay of around 0.5 seconds in the position estimate. However, this error alone cannot explain the discrepancy between the simulation and experimental configuration.

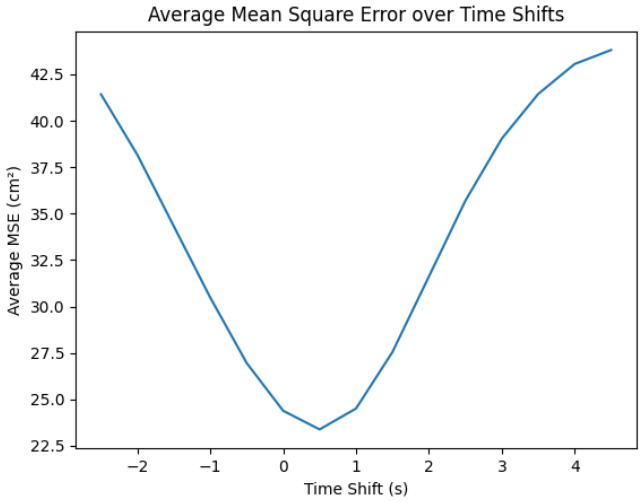


Figure 6.6: Average mean square error of the position estimate using the particle filter algorithm for different time shifts of the real positions (estimated by the tracking system)

For other sources of error, such as the rate at which the swarm transmits information, the certainty variants should improve resistance to missing data compared with current results; for sources of error, such as measurement noise, the offset variants should improve overall error. If none of the variants radically improves the results, the cause of the error may be something else.

The last error source that must be investigated is the tracking system error. Since the tracking system is the ground truth used to compute statistics on the localization quality, errors should be noticed to avoid false conclusions on the results. Figure 6.7 shows a correlation between error peaks in location quality estimation and tracking system speed outliers. While not all error peaks are due to the tracking system (see the 55-second timestamp), there is also a strong correlation between the most significant tracking system outliers and most error peaks, showing that tracking system errors are not negligible. Results would, therefore, be less noisy with an improved tracking system.

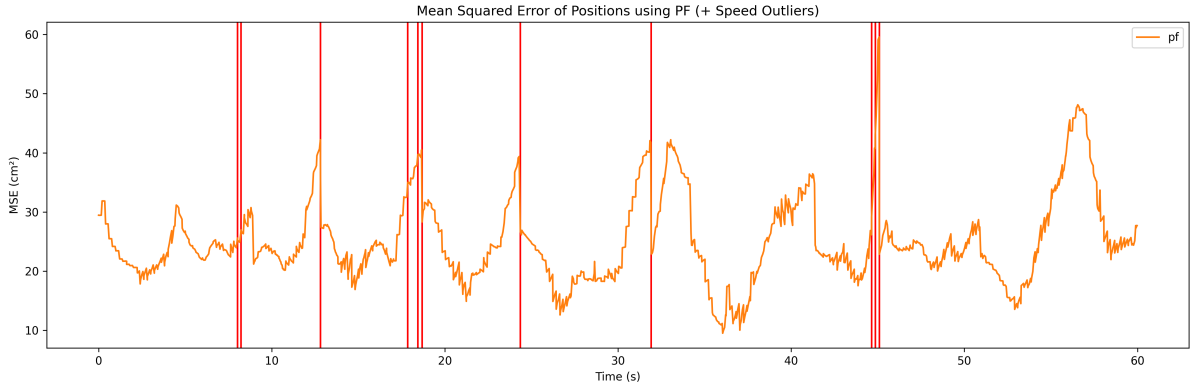


Figure 6.7: Showcase of correlation between speed outliers of the tracking system and the computed error on localization (speed outliers represented as vertical lines).

6.3 Convergence Analysis

Multidimensional Scaling

The results are averaged over five experiments using the following variants of MDS:

- *init*: initialization of MDS with the previous localization estimation.
- *offset*: distance measurements corrected using the derived error model.
- *certainty*: weighted variant of MDS to give less importance to old measurements.

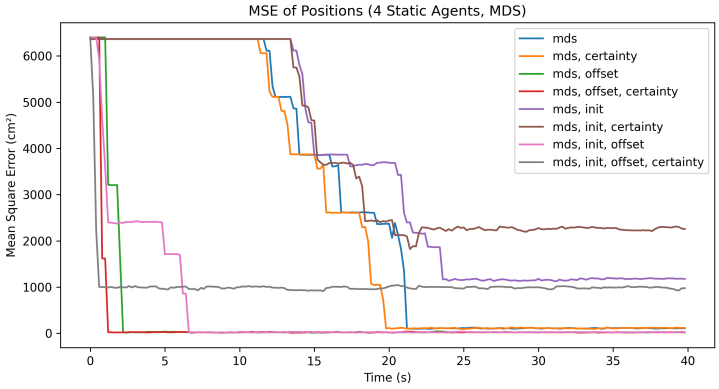
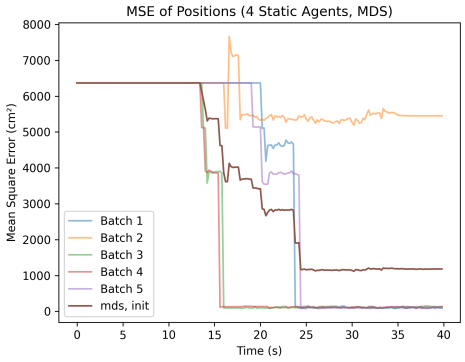
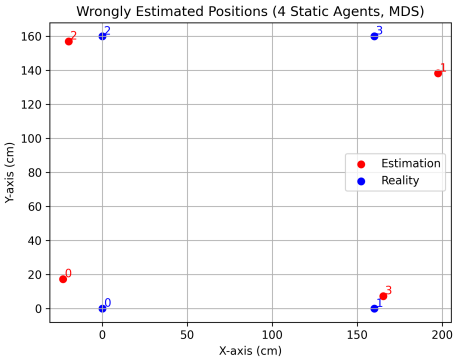


Figure 6.8: Evolution of the Mean Square Error over time for all tested variants of MDS

Figure 6.8 shows that some models converge in terms of mean square error while others do not. Those that do not converge all come from the *init* variant of MDS. As shown in Figure 6.9 (a), the non-convergence behavior is because some batches converge while others remain at a high and never decreasing error level. Figure 6.9 (b) illustrates a typical mis-estimated position and shows that the initialized MDS cannot escape a local minimum of its objective function.



(a) MDS with initialization variant, batch decomposition



(b) Resulting localization at last time-step for non-converging batch

Figure 6.9: Final positions estimation for a non-converging batch of MDS with initialization variant

Furthermore, from Figure 6.8, we see that the offset variant has caused two improvements in the behavior: a significant speed-up in the convergence speed and a lower mean-square error at convergence. Lower convergence mean-square error is easily explained by the measures being closer to reality for the convergence speed. However, it is not trivial as to why it causes such a big difference. It might be caused either by the agents being close together (about 1.6 meters from each other), which causes errors to impact the estimation quality significantly or by the distances being closer from reality, allowing MDS to generate better estimates faster. Finally, even though the certainty variant seems to generate faster convergence, the difference is not enough to conclude.

Particle Filter

Let us now look at the results of the Particle Filter. Here is a quick reminder of the particle filter variants implemented:

- *init*: starting the filter with good relative positions (but not the right distances).
- *offset*: same as previously, updated distance values with derived error model.
- *certainty*: removing old values from the measurement in the update phase.

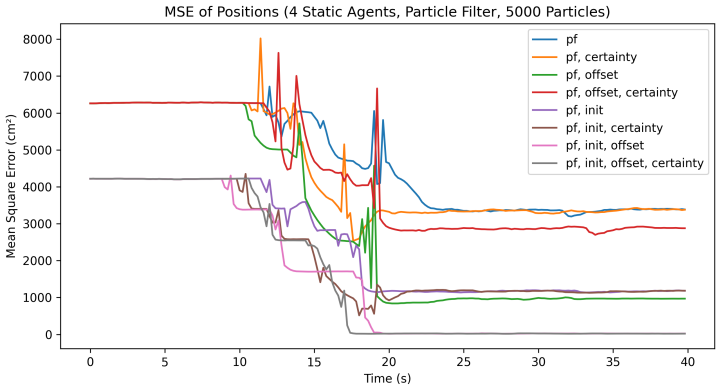


Figure 6.10: Evolution of the Mean Square Error over time for all tested variants of the Particle Filter

It can be seen in Figure 6.10 that the particle filter converges less often than MDS; this is mainly due to the nature of the solution, which tries to simulate the movement of agents, forcing a logical transition between each time step. Consequently, if the particle filter finds itself in an ambiguous position, such as a local minimum, it will not be able to return to a correct estimate, even more so if the number of measurements it can rely on to update the probabilities is reduced, as it is the case here with relative localization based on distance alone. However, giving a good first approximation to the filter allows for converging most of the time, as it happened with the initialization variant proposed here.

6.4 Mobile Setting Analysis

This section analyses the localization quality for the different algorithms and variants implemented. All the results in this section come from experiments done in the physical setting with all agents moving.

6.4.1 Position Estimation Quality

The various algorithms implemented were run on the data of the dynamic experiments, with all the agents moving to study the impact of each variant on localization quality.

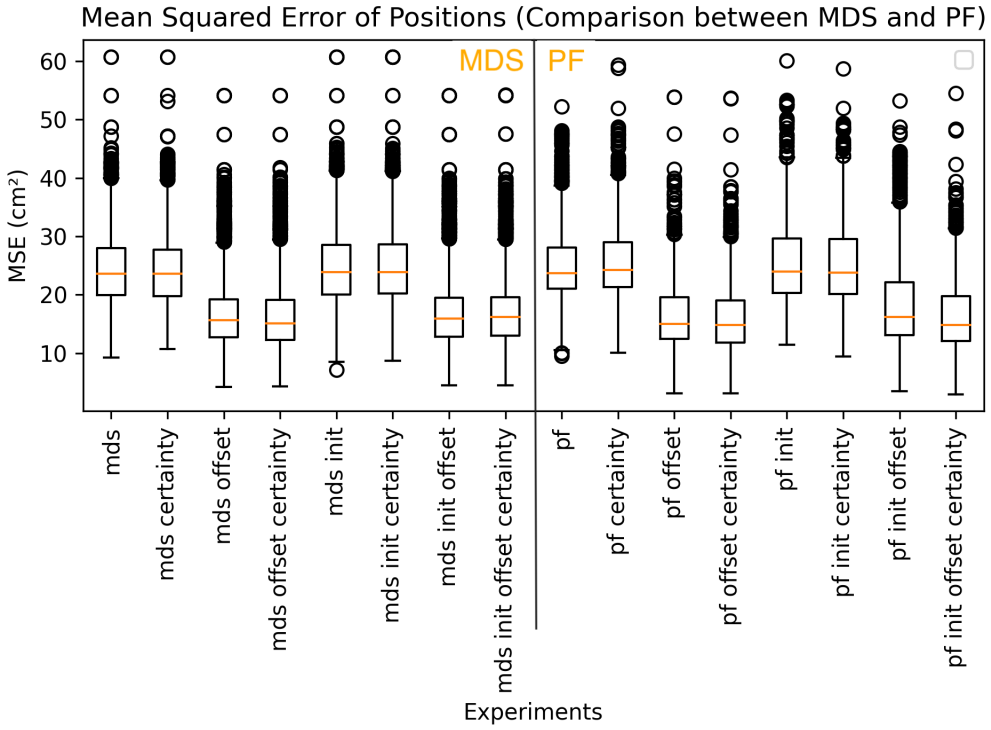


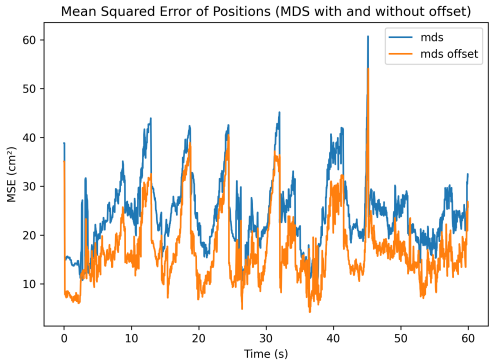
Figure 6.11: Boxplot of the positioning mean square error over a 60-second run with all robots moving. Comparison between the two algorithms and their variants based on the localization error.

The results in Figure 6.11 show that the offset variant is the only variant with a significant impact on localization quality, showcasing the interest of the previously derived noise model. All the results in Figure 6.12 corroborate the previous conclusion since, over time, the quality remains the same for the algorithm with and without variant except for the offset variant, which consistently decreases the observed mean square error. However, compared with the static convergence study, all the algorithms converge, which means that the system dynamics allow enough randomness to prevent the algorithms from getting stuck in local minima.

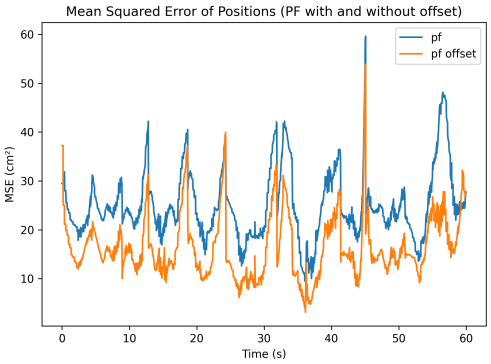
Furthermore, initialization variants have a positive impact at the beginning of the experiments, but they have the same results as the algorithm without the variant. Finally, the certainty variant having no impact on the quality of the estimate either means that the channel is not the limitation of the localization quality or that the certainty factor is

not decreasing fast enough for outdated measurements. In both cases, it causes all values to have about the same certainty, changing nothing to the algorithm's results.

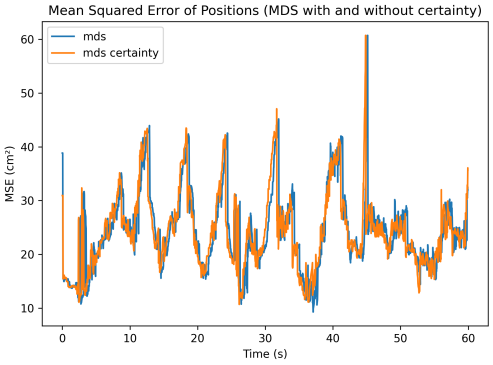
However, certainty decreases by a factor of 0.99 at a frequency of about $80Hz$, causing it to be equal to 50% after one second. Furthermore, certainty only increases when a new measurement is made, and the average certainty of the values over an experiment is about 95%, which tends to show that the error is not a channel limitation.



(a) MDS, normal vs. offset



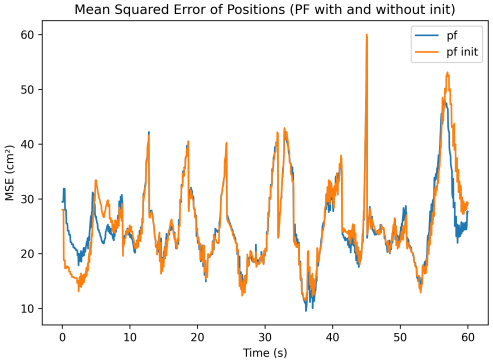
(b) PF, normal vs. offset



(c) MDS, normal vs. certainty

(d) PF, normal vs. certainty

(e) MDS, normal vs. init



(f) PF, normal vs. init

Figure 6.12: Comparison of the localization results for MDS and PF when using the offset, certainty and init variants.

6.4.2 Direction Estimation Quality

This section discusses the direction estimation through the different methods described in Section 4.3.4. All the results in this section come from experiments done in the physical setting with all agents moving.

Preliminary Method Comparison

Figure 6.14 showcases the three direction estimation mechanisms results for MDS and PF. The historic parameter has a different impact on each method:

- Displacement: historic saves the previously computed positions and averages the displacements from position to position with a decreasing weight as the position estimate is older.
- Distances: historic saves the distances measured over time, and uses the difference from the first and last distance estimation of the historic to compute the distance evolution.
- Particles: historic saves the total weighted average particle of each iteration and computes the average of direction with a decreasing weight as the particle is older.

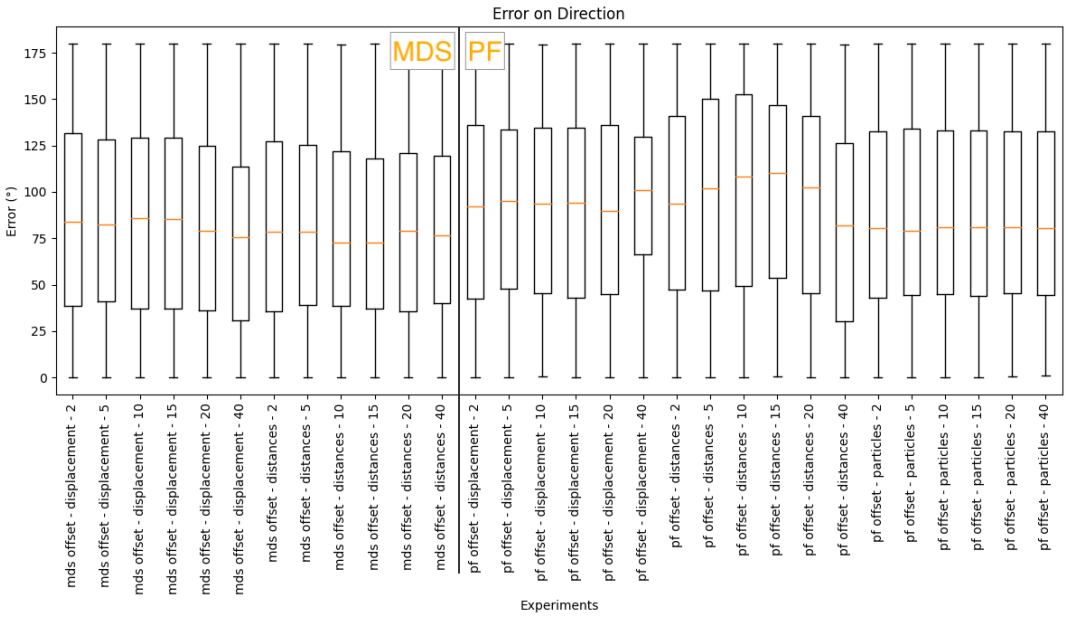
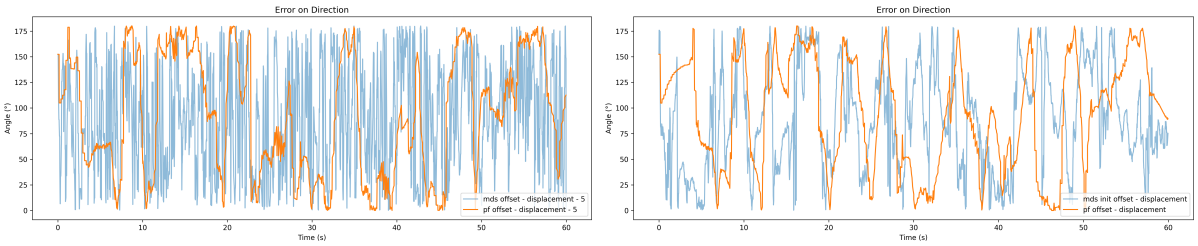


Figure 6.13: Boxplot of the error on the estimation of direction using the three estimators on both MDS and PF ("particles" estimator not relevant for MDS) and for different sizes of historic

The boxplot does not showcase great results for the different direction estimation methods. Even though some methods decrease the variance and others tend to decrease the mean error, on average, the error level is around 80 to 100°, which is the expected result of a random guess since the metric used is bound between 0 and 180°. This shows that the overall quality of the direction estimate is low. However, the direction estimates fluctuating a lot over the whole experiment could be studied in more detail to compare the quality of the different methods.

Direction Estimation Stability

As shown in Figure 6.14, even though both estimations have about the same average quality, the particle filter version is way more stable in its estimate, which is better than the seemingly random behavior of MDS.



(a) PF vs. Classical MDS

(b) PF vs. Initialized MDS

Figure 6.14: Comparison of the error on the direction using the displacement estimation between PF and MDS with and without the initialized variant

As mentioned, this is mainly due to the difference in construction between MDS and PF. MDS does not make sense of the previous estimation, which causes new position estimation to appear completely random compared to previous estimation. The initialized version of MDS was introduced for that reason and proved to have the intended effect in Figure 6.14.

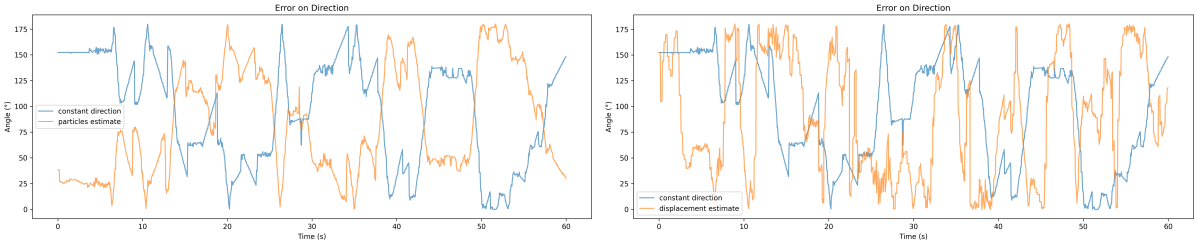
However, the results based on MDS are still less stable than those of PF-based estimations for both the *displacement* and *distances* methods. Overall, the *distances* based estimation showed unstable results for any historic size. The following section will, therefore, only focus on comparing the *displacement* and *particles* based methods using the particle filter and the *displacement* method using MDS, which are the only methods that showed preliminary results.

Method Comparison

Instability in the direction can come from different factors depending on the method. For example, the *particles* estimate instability comes mainly from the fact that the particles do not have a dynamic model to rely on, therefore basing itself on a random walk, causing the directions to be less stable. It also means that the method should greatly benefit from sensor fusion with odometry data. Then, the *distances* estimate showed promising results in simulation with one agent moving in a landmarked setting but no consistency over situations with every agent moving randomly. Since the historic size did not change anything to the results, the cause must be the randomness brought by the displacement of all agents, which leads to the conclusion that it is not suited for the purpose. Finally, the *displacement* estimate stability directly depends on the stability of the positioning method.

However, we can see in Figure 6.15 (a) that the best results for orientation, given by the *particles* estimation, match the quality of the constant direction. This should mainly be caused by the random direction given to the particles to ensure a good position estimation even without information on the system's dynamic. Indeed, averaging angles evenly spread between $-\pi$ and π will always give 0 and, therefore, no information on

the agent direction. For this estimation to work, it should have a consequent part of the particles oriented in approximately the right direction. However, it would require the particle filter to receive additional information to remove the need for the random walk implementation.



(a) Particles direction vs. constant direction (b) Displacement direction vs. constant direction

Figure 6.15: Comparison between implemented methods of direction estimation and continuously estimating the same direction.

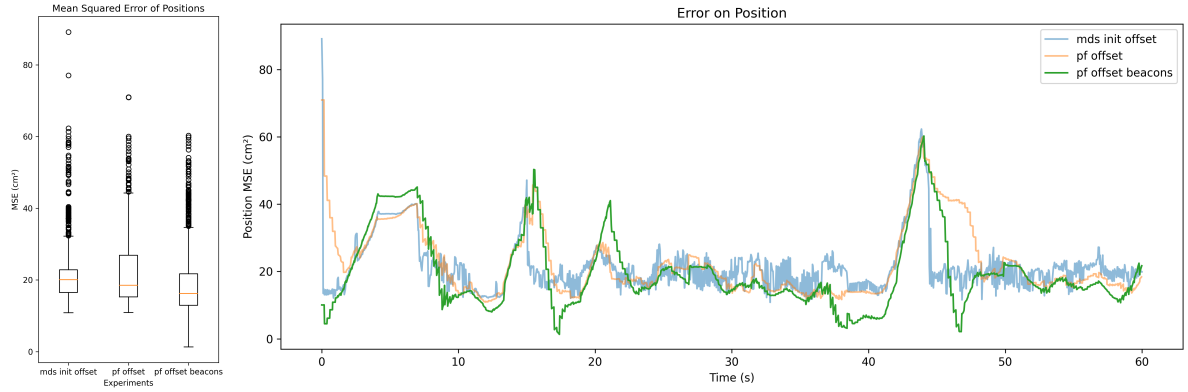
Figure 6.15 (b) shows the result of the *displacement* method. While it does not improve the overall quality of the estimation (average to about 50% of the time under 90° of error), it has a behavior that does not correspond to the constant estimation, which shows that a more stable estimation should allow for improving the result. However, more stability will not be achieved unless landmarks or sensor fusion are added to the particle filter. In conclusion, the orientation estimation methods implemented are not qualitative and will not allow consistent results. Furthermore, from all the implemented methods, *displacement* is the best option for distance alone orientation estimation, and *particles* should be tried with a better-known dynamic when adding odometry data.

6.5 Landmarked Setting Analysis

This section summarizes the experiment results with landmarks, i.e., one agent moving and all the others at rest. It is deliberately kept short to support the improvements that an environment with reference points brings to localization.

6.5.1 Position Estimation Quality

As shown in Figure 6.16, the position estimate is approximately the same. However, some minor improvements can be observed, as the particle filter is able to decrease the estimation error even more once in a while when implemented with landmarks. Once again, some error can be observed, showcasing problems in the quality of the data but not a problem with the implementation itself.



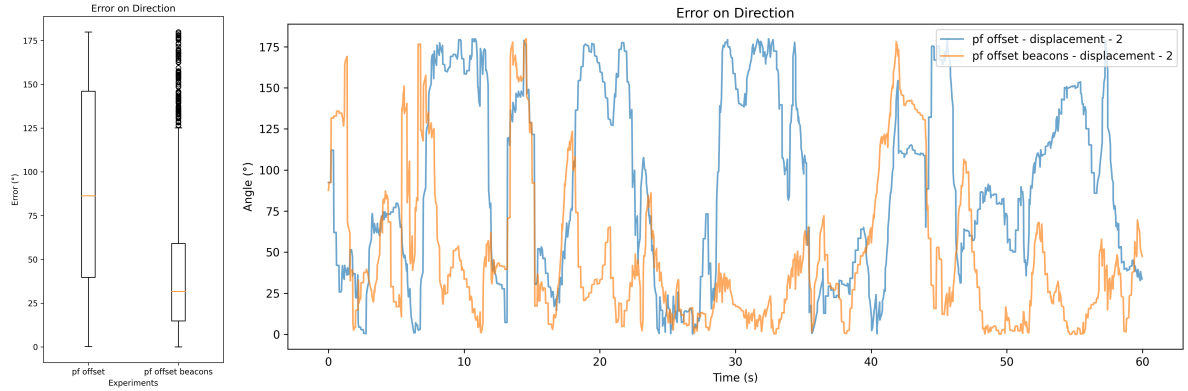
(a) Boxplot of position error

(b) Position error over time

Figure 6.16: Comparison of position error between MDS with initialized variant and PF with and without considering non-moving agents as fixed points in localization algorithms

6.5.2 Direction Estimation Quality

Landmarked settings are often used to increase the stability of distance localization in literature; therefore, they can be expected to have a good impact on the direction estimate.



(a) Boxplot of direction error

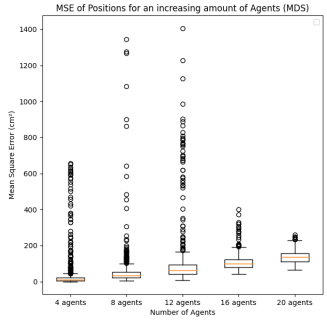
(b) Direction error over time

Figure 6.17: Quality comparison of direction estimation with displacement method between MDS with initialized variant and PF with and without considering non-moving agents as fixed points in localization algorithms

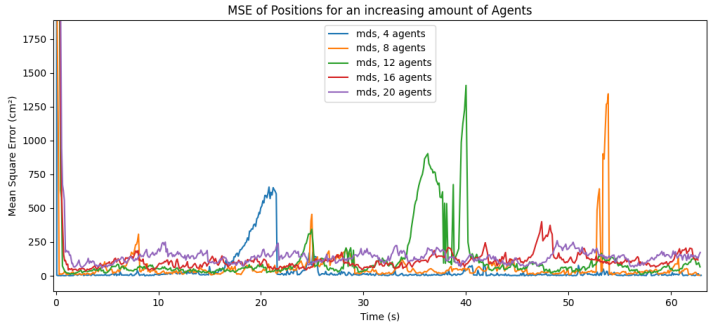
As shown in Figure 6.17, while it does not become perfect, the quality of the direction estimation significantly improves when using landmarks. Once again, displacement is the only method whose quality has increased, corroborating that its main requirement is the stability of the position estimate. Unfortunately, the distances direction estimation does not improve with the landmark situation in the real context, showing that displacement estimation is the only promising method of the three presented to achieve consistent direction estimates for distance-only settings.

6.6 System Scalability

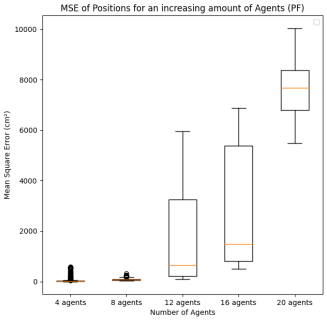
Based on the simulation, experiments were carried out for 4, 8, 12, 16, and 20 robots. Equation 4.8 was used to derive the drop rate for each number of agents to match the channel characterization. The previously derived noise model and communication parameters were therefore used, which is the main limitation of this analysis since interference and message loss can be expected to increase considerably for an increased number of agents. This problem is indeed caused by the ALOHA-based communication protocol. It cannot be expected to work this well when used with so many agents in real life. However, the results could prove that even with an idealized channel, specific problems do occur.



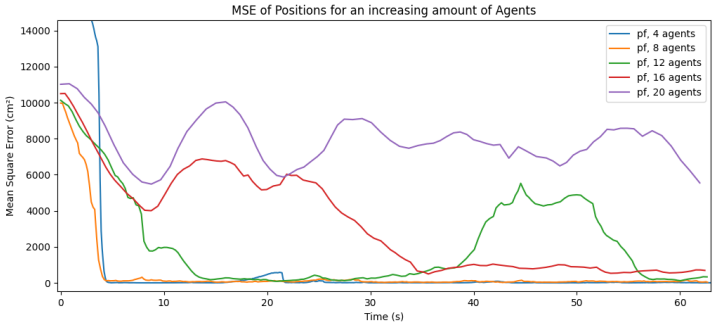
(a) Boxplot of MDS results



(b) MDS results over time for different swarm scale



(c) Boxplot of PF results



(d) PF results over time for different swarm scales

Figure 6.18: Comparison of the localization quality over time using MDS and Particle Filter for an increasing number of agents

Figure 6.18 shows MDS and PF results for different numbers of agents. Firstly, when the algorithms reach convergence, the error is, on average, greater for larger swarms. In fact, a linear decrease in quality is observed for MDS while an exponential decrease is observed for PF. Secondly, we find that whenever the algorithms have an error peak, they take longer to smooth out for simulations with a larger number of agents. On the MDS side, the error peaks in the 12-agent setting are larger and more significant than for the 4- and 8-agent settings. On the PF side, it takes around 20 seconds to smooth out an error peak with 16 agents, whereas it takes around 15 seconds for the 12-agent setting. Finally, the particle filter algorithm does not converge for a larger swarm. Indeed, we find that, while MDS retains a low error, PF is unable to reduce the localization error over time and retains large errors in the 20-agent configuration.

Chapter 7

Discussions

7.1 Ultra-Wideband Protocol

The ALOHA protocol implemented proved to be very efficient for the swarm size considered during the channel characterization. ALOHA is a good starting point for prototyping cooperative localization algorithms without focusing on telecommunications implementations. The main limitation of the protocol is its scalability, which limits the applicability of the designed protocol to larger swarms. However, it did not show to be the most important limitation for the scalability of the proposed implementation. Consequently, studies based on more than 5 to 10 agents will require a more scalable protocol. One solution of choice seems to be TDMA, which has been successfully applied in many papers dealing with the current topic [53, 54].

7.2 Localization Algorithms

Both positioning algorithms produced great results. Although some error peaks were observed, the methods implemented gave the same error on position estimations. Given that their implementation is completely different, this observation led to the hypothesis that they always minimized the error according to the data available and its quality. In general, MDS showed better error reduction capabilities than the particle filter. However, as MDS is not based on a dynamic model of the system, it proved less stable than the particle filter. As a result, orientation estimations (which were highly dependent on the stability of the positioning algorithm) worked better with the particle filter than with MDS.

None of the three methods devised for estimating the relative orientation of the agent in the swarm yielded results that could be put to further use. Distances-based estimation was unable to cope with the randomness of a situation in which too many agents were moving. Particle direction estimation proved too random, which can be explained by the fact that the particle filter is based on the random walking of agents, while it is designed to base itself on odometry knowledge. Displacement estimation failed to deliver the expected results, as it proved too dependent on the stability of the underlying positioning method. As expected from the previous results, this final technique performed better with the particle filter thanks to the stability it brings to the position estimate.

Overall, the results presented in the previous chapter are mixed. While the positioning algorithms produced good results, none of the techniques implemented produced convincing estimates of the agent's orientation. Since localization aims to recover both the agent's positioning and directional information, we can conclude that this objective has not been achieved. Given the efforts made to implement variants to correct the drawbacks of each method and the few results obtained, we can conclude that relative localization based solely on pairwise distance is unreliable in a fully mobile environment.

7.3 Distance-based Localization

Performing relative localization based on pairwise distances leads to ambiguities. These ambiguities are the main obstacle to distance-only localization, as they prevent the creation of a stable reference frame in which to position agents. In the literature, these ambiguities are generally resolved by merging data from other sensors to increase the number of observations made on the system or by fixing nodes to reduce the number of unknown variables. The first solution increases the system's observability and reduces the number of ambiguities, while the second uses fixation to generate a stable environment without ambiguities.

Application of the designed system to a swarm of robots is reduced to missions where agents can act as beacons for other agents, creating immobile nodes on which the localization algorithm can rely to resolve ambiguities. Consequently, distance alone is better suited to applications with external infrastructure, such as the localization of moving objects in warehouses. In conclusion, basing the localization system on self-contained robotic platforms equipped with a larger number of sensors would make more sense in the context of swarm robotics, as this does not restrict the agents' movements, enabling broader applications. This observation is corroborated by the many excellent results obtained in the literature for sensor fusion localization.

7.4 Later Work

The communication and ranging protocol designed is efficient for small-scale swarms. Furthermore, later work could extend the communication protocol to send more than just ranging measurements. Besides, it provides an additional sensor for the Mercator robotic platform, enabling pairwise distance measurements. The designed UWB architecture increases the agents' knowledge of the system and allows for the implementation of sensor fusion with odometry data retrieved by the robot. The next logical step is to implement this sensor fusion to obtain a relative localization in the swarm and enable the complex spatial organization behavior described in the introduction.

The particle filter gave good results when positioning agents using only knowledge of distance. Furthermore, sensor fusion is easily achieved by extending the update phase of the proposed particle filter, enabling future implementations to be started based on this thesis work. However, the scalability study showed that the multi-filter approach will only apply to small swarms. Further work with many robots will require an improved localization algorithm by implementing distributed filters or less computationally expensive options, such as different Kalman filters.

Chapter 8

Conclusion

To the author's knowledge, no paper discussing the feasibility of relative localization using solely pairwise distances in a network of moving nodes exists. Based on that observation, this work aimed to study possible options of localization algorithms that could leverage knowledge of distances to construct relative positions and orientations of the agents in a swarm.

The first part of the thesis was devoted to designing a communication and ranging protocol using Qorvo's DWM1001-DEV UWB modules. The second part focused on implementing two real-time relative positioning algorithms, MDS and a particle filter, based solely on distance measurements from the designed protocol. The third part discussed different methods for estimating the relative direction of the robots. Combining position and orientation would generate a complete localization system and enable spatial organization behaviors in the swarm using distance measurements alone.

On the one hand, the ALOHA-based ultra-wideband protocol design showed high-quality results for small-scale swarms, allowing imagined localization algorithms to be implemented based on an efficient communication channel. On the other hand, localizing the agents based on distances alone proved difficult. While the implementation achieved good results in positioning the agents relative to each other, the proposed methods for determining orientations proved to be suboptimal, limiting the use of the derived algorithms for localization. However, those results allow consistent computation of high-level knowledge about the swarm. For example, one can expect to compute metrics such as the area, the perimeter, the global shape of the swarm, and many other metrics comprising clusters and concentration of agents around a given agent. The main limitation is the estimated orientation of the robots. Therefore, systems based on this thesis implementation will not enable complex spatial organization behavior as they will not allow the agents to perceive the direction they are heading towards.

Consequently, the main finding of this thesis is the impossibility of distance-only localization to resolve the underlying ambiguities brought by the lack of knowledge of the system. The thesis subsequently showed the interest of the different solutions proposed throughout the literature, such as sensor fusion to increase the system observability or landmarks to resolve the ambiguities. Later work wanting to enable a swarm of self-contained robots to collaborate toward relative localization should focus on implementing sensor fusion. However, applications in known environments would greatly benefit from non-moving reference points to increase the localization quality.

Bibliography

[1] M. Dorigo, M. Birattari, and M. Brambilla, "Swarm robotics," *Scholarpedia*, vol. 9, no. 1, pp. 1462–1463, 2014.

[2] A. Hénard, J. Rivière, E. Peillard, S. Kubicki, and G. Coppin, "A unifying method-based classification of robot swarm spatial self-organisation behaviours," *Adaptive Behavior*, vol. 31, no. 6, pp. 577–599, 2023.

[3] M. Hannebauer, J. Wendler, E. Pagello, L. Iocchi, D. Nardi, and M. Salerno, "Reactivity and deliberation: a survey on multi-robot systems," in *Balancing Reactivity and Social Deliberation in Multi-Agent Systems: From RoboCup to Real-World Applications*, pp. 9–32, Springer, 2001.

[4] A. E. Turgut, H. Çelikkanat, F. Gökçe, and E. Şahin, "Self-organized flocking in mobile robot swarms," *Swarm Intelligence*, vol. 2, pp. 97–120, 2008.

[5] A. Chatty, I. Kallel, P. Gaussier, and A. M. Alimi, "Emergent complex behaviors for swarm robotic systems by local rules," in *2011 IEEE Workshop on Robotic Intelligence in Informationally Structured Space*, pp. 69–76, IEEE, 2011.

[6] M. Schranz, M. Umlauft, M. Sende, and W. Elmenreich, "Swarm robotic behaviors and current applications," *Frontiers in Robotics and AI*, vol. 7, p. 36, 2020.

[7] I. Navarro and F. Matía, "An introduction to swarm robotics," *Isrn robotics*, vol. 2013, pp. 1–10, 2013.

[8] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm intelligence: from natural to artificial systems*. Oxford university press, 1999.

[9] G. Beni, "From swarm intelligence to swarm robotics," in *International Workshop on Swarm Robotics*, pp. 1–9, Springer, 2004.

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

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

[12] M. Kegeleirs, R. Todesco, D. G. Ramos, G. L. Herranz, and M. Birattari, "Mercator: hardware and software architecture for experiments in swarm slam," *IRIDIA, Université libre de Bruxelles, Brussels, Belgium, Tech. Rep. TR/IRIDIA/2022-012*, 2022.

[13] M. Kegeleirs, D. G. Ramos, K. Hasselmann, L. Garattoni, G. Francesca, and M. Birattari, "Transferability in the automatic off-line design of robot swarms: From sim-to-real to embodiment and design-method transfer across different platforms," *IEEE Robotics and Automation Letters*, vol. 9, no. 3, pp. 2758–2765, 2024.

[14] Y. Yemini, "The positioning problem—a draft of an intermediate summary," in *Proceedings of the Conference on Distributed Sensor Networks*, pp. 137–145, sn, 1978.

[15] C. Gao, G. Zhao, and H. Fourati, *Cooperative localization and navigation: theory, research, and Practice*. CRC Press, 2019.

[16] E. M. H. Zahugi, A. M. Shabani, and T. V. Prasad, "Libot: Design of a low cost mobile robot for outdoor swarm robotics," in *2012 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*, pp. 342–347, 2012.

[17] H. Akcan, V. Kriakov, H. Brönnimann, and A. Delis, "Managing cohort movement of mobile sensors via gps-free and compass-free node localization," *Journal of Parallel and Distributed Computing*, vol. 70, no. 7, pp. 743–757, 2010.

[18] J. Siva and C. Poellabauer, *Robot and Drone Localization in GPS-Denied Areas*, pp. 597–631. Cham: Springer International Publishing, 2019.

[19] N. National Coordination Office for Space-Based Positioning and Timing, "Global positioning system accuracy," 2022. `https://www.gps.gov/systems/gps/performance/accuracy/` [Accessed: 31st of March 2024].

[20] H. Widyantara, M. Afandi, R. Akseptori, and U. Umar, "Navigation and formation of swarm robotics with local positioning system," *JURNAL NASIONAL TEKNIK ELEKTRO*, 11 2022.

[21] N. Patwari, *Localization, Cooperative*, pp. 616–622. Boston, MA: Springer US, 2008.

[22] K. Guo, Z. Qiu, W. Meng, L. Xie, and R. Teo, "Ultra-wideband based cooperative relative localization algorithm and experiments for multiple unmanned aerial vehicles in gps denied environments," *International Journal of Micro Air Vehicles*, vol. 9, no. 3, pp. 169–186, 2017.

[23] J. P. Queralta, C. M. Almansa, F. Schiano, D. Floreano, and T. Westerlund, "Uwb-based system for uav localization in gnss-denied environments: Characterization and dataset," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4521–4528, IEEE, 2020.

[24] J. Werb and C. Lanzl, "Designing a positioning system for finding things and people indoors," *IEEE Spectrum*, vol. 35, no. 9, pp. 71–78, 1998.

[25] S. Chen, D. Yin, and Y. Niu, "A survey of robot swarms' relative localization method," *Sensors*, vol. 22, no. 12, p. 4424, 2022.

[26] A. Yassin, Y. Nasser, M. Awad, A. Al-Dubai, R. Liu, C. Yuen, R. Raulefs, and E. Aboutanios, "Recent advances in indoor localization: A survey on theoretical approaches and applications," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 2, pp. 1327–1346, 2016.

[27] G. Deak, K. Curran, and J. Condell, "A survey of active and passive indoor localisation systems," *Computer Communications*, vol. 35, no. 16, pp. 1939–1954, 2012.

[28] M.-G. Di Benedetto, "Uwb communication systems: a comprehensive overview," 2006.

[29] F. Mazhar, M. G. Khan, and B. Sällberg, "Precise indoor positioning using uwb: A review of methods, algorithms and implementations," *Wireless Personal Communications*, vol. 97, no. 3, pp. 4467–4491, 2017.

[30] A. Alarifi, A. Al-Salman, M. Alsaleh, A. Alnafessah, S. Al-Hadhrami, M. A. Al-Ammar, and H. S. Al-Khalifa, "Ultra wideband indoor positioning technologies: Analysis and recent advances," *Sensors*, vol. 16, no. 5, p. 707, 2016.

[31] A. R. J. Ruiz and F. S. Granja, "Comparing ubisense, bespoon, and decawave uwb location systems: Indoor performance analysis," *IEEE Transactions on instrumentation and Measurement*, vol. 66, no. 8, pp. 2106–2117, 2017.

[32] M. Delamare, R. Boutteau, X. Savatier, and N. Iriart, "Static and dynamic evaluation of an uwb localization system for industrial applications," *Sci*, vol. 2, no. 2, p. 23, 2020.

[33] L. Zwirello, T. Schipper, M. Harter, and T. Zwick, "UWB localization system for indoor applications: Concept, realization and analysis," *Journal of Electrical and Computer Engineering*, vol. 2012, pp. 1–11, 2012.

[34] T. Kim Geok, K. Zar Aung, M. Sandar Aung, M. Thu Soe, A. Abdaziz, C. Pao Liew, F. Hossain, C. P. Tso, and W. H. Yong, "Review of indoor positioning: Radio wave technology," *Applied Sciences*, vol. 11, no. 1, p. 279, 2020.

[35] T. Dag and T. Arsan, "Received signal strength based least squares lateration algorithm for indoor localization," *Computers & Electrical Engineering*, vol. 66, pp. 114–126, 2018.

[36] R. Zandian, *Ultra-wideband based indoor localization of mobile nodes in ToA and TDoA configurations*. PhD thesis, Dissertation, Bielefeld, Universität Bielefeld, 2018, 2019.

[37] L. Polak, S. Rozum, M. Slanina, T. Bravenec, T. Fryza, and A. Pikrakis, "Received signal strength fingerprinting-based indoor location estimation employing machine learning," *Sensors*, vol. 21, no. 13, p. 4605, 2021.

[38] S. Gezici and H. V. Poor, "Position estimation via ultra-wide-band signals," *Proceedings of the IEEE*, vol. 97, no. 2, pp. 386–403, 2009.

[39] L. Taponecco, A. D'Amico, and U. Mengali, "Joint TOA and AOA estimation for UWB localization applications," *IEEE Transactions on Wireless Communications*, vol. 10, no. 7, pp. 2207–2217, 2011.

[40] B. Hanssens, D. Plets, E. Tanghe, C. Oestges, D. P. Gaillot, M. Liénard, L. Martens, and W. Joseph, "An indoor localization technique based on ultra-wideband AoD/AoA/ToA estimation," in *2016 IEEE International Symposium on Antennas and Propagation (APSURSI)*, pp. 1445–1446, IEEE, 2016.

[41] F. Shang, B. Champagne, and I. Psaromiligkos, "Joint TOA/AOA estimation of IR-UWB signals in the presence of multiuser interference," in *2014 IEEE 15th international workshop on signal processing advances in wireless communications (SPAWC)*, pp. 504–508, IEEE, 2014.

[42] M. Heydariaan, H. Dabirian, and O. Gnawali, "Anguloc: Concurrent angle of arrival estimation for indoor localization with uwb radios," in *2020 16th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, pp. 112–119, IEEE, 2020.

[43] N. Smaoui, M. Heydariaan, and O. Gnawail, "Single-antenna AoA estimation with UWB radios," in *2021 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–7, IEEE, 2021.

[44] J. Xu, M. Ma, and C. L. Law, "AOA cooperative position localization," in *IEEE GLOBECOM 2008-2008 IEEE Global Telecommunications Conference*, pp. 1–5, IEEE, 2008.

[45] M. I. Hossain, "Distance measurements using ultra wideband," 2012.

[46] Q. Dong and W. Dargie, "Evaluation of the reliability of RSSI for indoor localization," in *2012 International Conference on Wireless Communications in Underground and Confined Areas*, pp. 1–6, IEEE, 2012.

[47] M. Ridolfi, S. Van de Velde, H. Steendam, and E. De Poorter, "Analysis of the scalability of UWB indoor localization solutions for high user densities," *Sensors*, vol. 18, no. 6, p. 1875, 2018.

[48] M. S. I. M. Zin and M. Hope, "A review of UWB MAC protocols," in *2010 sixth advanced international conference on telecommunications*, pp. 526–534, IEEE, 2010.

[49] T. Laadung, S. Ulp, M. M. Alam, and Y. Le Moullec, "Novel active-passive two-way ranging protocols for UWB positioning systems," *IEEE Sensors Journal*, vol. 22, no. 6, pp. 5223–5237, 2021.

[50] J. Zhang, P. V. Orlik, Z. Sahinoglu, A. F. Molisch, and P. Kinney, "UWB systems for wireless sensor networks," *Proceedings of the IEEE*, vol. 97, no. 2, pp. 313–331, 2009.

[51] W. Dargie and C. Poellabauer, *Fundamentals of wireless sensor networks: theory and practice*. John Wiley & Sons, 2010.

[52] P. Poudereux, E. García, J. J. García, J. Ureña, *et al.*, "Performance comparison of a TDMA-and CDMA-based UWB local positioning system," in *International Conference on Indoor Positioning and Indoor Navigation*, pp. 1–9, IEEE, 2013.

[53] Y. Cao, C. Chen, D. St-Onge, and G. Beltrame, "Distributed TDMA for mobile UWB network localization," *IEEE Internet of Things Journal*, vol. 8, no. 17, pp. 13449–13464, 2021.

[54] T. ter Horst, "Ultra-wideband communication and relative localisation for swarming robots," 2019.

[55] Y. Deng, D. Kim, Z. Li, and Y.-J. Choi, "Implementing distributed TDMA using relative distance in vehicular networks," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 7, pp. 7295–7305, 2020.

[56] K. Langendoen and N. Reijers, "Distributed localization in wireless sensor networks: a quantitative comparison," *Computer networks*, vol. 43, no. 4, pp. 499–518, 2003.

[57] R. Kurazume, S. Nagata, and S. Hirose, "Cooperative positioning with multiple robots," in *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, pp. 1250–1257, IEEE, 1994.

[58] R. Kurazume and S. Hirose, "An experimental study of a cooperative positioning system," *Autonomous Robots*, vol. 8, pp. 43–52, 2000.

[59] T. Matsuda, T. Maki, Y. Sato, and T. Sakamaki, "Experimental evaluation of accuracy and efficiency of alternating landmark navigation by multiple auvs," *IEEE Journal of Oceanic Engineering*, vol. 43, no. 2, pp. 288–310, 2018.

[60] H. M. Menegaz, J. Y. Ishihara, G. A. Borges, and A. N. Vargas, "A systematization of the unscented kalman filter theory," *IEEE Transactions on automatic control*, vol. 60, no. 10, pp. 2583–2598, 2015.

[61] S. J. Julier and J. K. Uhlmann, "Unscented filtering and nonlinear estimation," *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401–422, 2004.

[62] A. Doucet, N. De Freitas, N. J. Gordon, *et al.*, *Sequential Monte Carlo methods in practice*, vol. 1. Springer, 2001.

[63] S. Thrun, "Particle filters in robotics.," in *UAI*, vol. 2, pp. 511–518, Citeseer, 2002.

[64] Z. Xue and H. Schwartz, "A comparison of several nonlinear filters for mobile robot pose estimation," in *2013 IEEE International Conference on Mechatronics and Automation*, pp. 1087–1094, IEEE, 2013.

[65] B. Araki, I. Gilitschenski, T. Ogata, A. Wallar, W. Schwarting, Z. Choudhury, S. Karaman, and D. Rus, "Range-based cooperative localization with nonlinear observability analysis," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pp. 1864–1870, IEEE, 2019.

[66] A. Chakraborty, K. Brink, R. Sharma, and L. Sahawneh, "Relative pose estimation using range-only measurements with large initial uncertainty," in *2018 Annual American Control Conference (ACC)*, pp. 5055–5061, IEEE, 2018.

[67] A. Chakraborty, K. M. Brink, and R. Sharma, "Cooperative relative localization using range measurements without a priori information," *Ieee Access*, vol. 8, pp. 205669–205684, 2020.

[68] S. I. Roumeliotis and G. A. Bekey, "Distributed multi-robot localization," in *Distributed Autonomous Robotic Systems 4*, pp. 179–188, Springer, 2000.

[69] S. Panzieri, F. Pascucci, and R. Setola, "Multirobot localisation using interlaced extended kalman filter," in *2006 IEEE/RSJ international conference on intelligent robots and systems*, pp. 2816–2821, IEEE, 2006.

[70] X. S. Zhou and S. I. Roumeliotis, "Robot-to-robot relative pose estimation from range measurements," *IEEE Transactions on Robotics*, vol. 24, no. 6, pp. 1379–1393, 2008.

[71] N. Trawny and S. I. Roumeliotis, "On the global optimum of planar, range-based robot-to-robot relative pose estimation," in *2010 IEEE International Conference on Robotics and Automation*, pp. 3200–3206, IEEE, 2010.

[72] J. Strader, Y. Gu, J. N. Gross, M. De Petrillo, and J. Hardy, "Cooperative relative localization for moving uavs with single link range measurements," in *2016 IEEE/ION Position, Location and Navigation Symposium (PLANS)*, pp. 336–343, IEEE, 2016.

[73] B. Southallzy, B. Buxtony, and J. Marchant, "Controllability and observability: Tools for kalman filter design," in *British Machine Vision Conference*, vol. 98, pp. 164–173, 1998.

[74] R. Sharma, R. W. Beard, C. N. Taylor, and S. Quebe, "Graph-based observability analysis of bearing-only cooperative localization," *IEEE Transactions on Robotics*, vol. 28, no. 2, pp. 522–529, 2011.

[75] B. T. Burchett, "Unscented kalman filters for range-only cooperative localization of swarms of munitions in three-dimensional flight," *Aerospace Science and Technology*, vol. 85, pp. 259–269, 2019.

[76] D. Fox, S. Thrun, W. Burgard, and F. Dellaert, "Particle filters for mobile robot localization," in *Sequential Monte Carlo methods in practice*, pp. 401–428, Springer, 2001.

[77] R. Liu, C. Yuen, T.-N. Do, D. Jiao, X. Liu, and U.-X. Tan, "Cooperative relative positioning of mobile users by fusing imu inertial and uwb ranging information," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5623–5629, IEEE, 2017.

[78] M. Li, Z. Chang, Z. Zhong, and Y. Gao, "Relative localization in multi-robot systems based on dead reckoning and uwb ranging," in *2020 IEEE 23rd International Conference on Information Fusion (FUSION)*, pp. 1–7, IEEE, 2020.

[79] N. Zhou, L. Lau, R. Bai, and T. Moore, "Novel prior position determination approaches in particle filter for ultra wideband (uwb)-based indoor positioning," *NAVIGATION: Journal of the Institute of Navigation*, vol. 68, no. 2, pp. 277–292, 2021.

[80] R. Liu, C. Yuen, T.-N. Do, W. Guo, X. Liu, and U.-X. Tan, "Relative positioning by fusing signal strength and range information in a probabilistic framework," in *2016 IEEE Globecom Workshops (GC Wkshps)*, pp. 1–6, IEEE, 2016.

[81] 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.

[82] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng, "ROS: an open-source Robot Operating System," in *Proc. of the IEEE Intl. Conf. on Robotics and Automation (ICRA) Workshop on Open Source Robotics*, (Kobe, Japan), may 2009.

[83] J. De Leeuw and P. Mair, "Multidimensional scaling using majorization: Smacof in r," 2011.

[84] R. Labbe, "Kalman and bayesian filters in python," *Chap*, vol. 7, no. 246, p. 4, 2014.

[85] W. Li, B. Jelfs, A. Kealy, X. Wang, and B. Moran, "Cooperative localization using distance measurements for mobile nodes," *Sensors*, vol. 21, no. 4, p. 1507, 2021.