NTNU
Norwegian University of
Science and Technology

Robust Potential Field-Based Communication & Control Architecture for Robotic Swarms

Using the nRF52 System-on-Chip and a mesh network topology

Henrik Malvik Halvorsen

Master of Science in Cybernetics and Robotics
Submission date: June 2018
Supervisor: Jan Tommy Gravdahl, ITK

Norwegian University of Science and Technology
Department of Engineering Cybernetics

Abstract

Which technologies and algorithmic design are suitable for both the control strategy and the communication protocol to ensure robustness within a robotic swarm?

The aim of this study is to answer this question. The study is performed by obtaining an understanding of the field of swarm robotics and proposing a robotic swarm system with robustness in mind. The robotic swarm system contains possible solutions for the control strategy, the communication protocol, hardware design and an in-room localization method. The innovative part of this swarm system is the idea of joining communication and control in a cooperative manner, based on previous studies on different robotic swarms. The communication protocol was designed to establish a mesh topology using the Thread protocol, MQTT-SN and ROMANOs. ROMANOs is a new application overlay protocol based on ROMANO, and is first introduced in this project. The control strategy introduces a potential-field based PID-controller, designed for its efficiency and practical approach. The cooperation between communication and control is the fact that the estimation of signal strength values will directly affect the potential fields of the control strategy. The control strategy will then initiate control actions to maintain the communication network. The hardware design platform consists of two printed circuit boards that houses all necessary electronics to realize the swarm algorithm, and a 3D-printed cylindrical chassis. The in-room localization method is based on ranging measurements from laser sensors, signal strength estimates and heading computations based on magnetometer data. A listening device consisting of a Node.JS MQTT module is created in order to acquire performance data from the swarm which is stored in an SQL database and further analysed in MATLAB.

The proposed swarm application yields a mostly favorable, but mixed result. The joint control- and communication algorithm is successful in finding a local minima within the network and maintaining said network. However, some parts of the algorithm are taxing and could be further refined. The communication protocol is only able to maintain a messaging frequency of up to 5 Hz for a six-member swarm, and the processor is only able to run the proposed swarm algorithm at a maximum of 50 Hz. What measures that should be taken to mitigate this and future work for this swarm application will be proposed. Nevertheless could the proposed swarm application work as an important stepping-stone for a true robust swarm application in the future.

Keywords : robotic swarms, potential-field based control, thread mesh network, MQTT-SN, ROMANO, laser distance measurements, magnetometer, joint control- and communication algorithm, PCB design, algorithm design, robust swarm application

Sammendrag

Hvilke teknologier og algoritmedesign er passende for både kontrollstrategien og kommunikasjonsprotokollen for å forsikre robusthet i en robotsverm?

Målet med dette studiet er å svare på dette spørsmålet. Studiet er utført med å danne en forståelse for forskingsfeltet som omhandler robotsvermer, og å foreslå et svermsystem med robusthet som hovedfokuspunkt. Svermsystemet inneholder mulige løsninger for kontrollstrategien, kommunikasjonsprotkollen, maskinvaredesign og innendørs lokasjonsmetoder. Den innovative parten i dette svermsystemet er idéen om å kombinere kommunikasjon og kontroll i et samarbeidende system, basert på tidligere studium på tidligere svermsystem. Kommunikasjonsprotokollen var designet for å etablere en mesh topologi ved å bruke Thread protokollen, MQTT-SN og ROMANOs. ROMANOs er en ny applikasjons protokoll basert på ROMANO, og er først introdusert i dette prosjektet. Kontrollstrategien introduserer en PID-kontroller som baserer seg på potensielle felt, designet for dens effektivitet og praktiske tilnærmingsmetode. Samarbeidet mellom kommunikasjon og kontroll baserer seg på det faktum at estimeringene av signalstyrken will påvirke kontrollstrategiens potensielle felt direkte. Kontrollstrategien vil deretter sette i gang et motorpådrag som vil prøve å ivareta kommunikasjonsnettverket. Maskinvareplattformen består av to kretskort som inneholder alle nødvendige komponenter for å realisere svermalgoritmen, og et 3D-printet sylindrisk deksel. Lokasjonsmetoden baserer seg på avstandsmålinger fra lasersensorer, estimeringer av signal styrke og retningsmålinger basert på magnetometer data. En lytterenhet er konstruert for å samle ytelsesdata, denne enheten samler dataene i en SQL-database som senere kan bli analysert i MATLAB. Lytterenheten baserer seg på en Node.JS MQTT modul.

Den foreslåtte svermalgoritmen har et blandet, men i all hovedsak lovende resultat. Den kombinerte kontroll- og kommunikasjonsalgoritmen lykkes i å finne et lokalt minimum innenfor nettverkets grenser, og i å opprettholde nettverket. Likevel er det noen deler av algoritmen som er nok så krevende og krever raffinering. Kommunikasjonsprotokollen har bare mulighet til å opprettholde en sendefrekvens på opp til 5 Hz for en sverm med seks medlemmer, og prosessoren selv kneler ved kjørefrekvenser over 50 Hz. Hvilke tiltak som må gjøres for å redusere disse begrensningene, samt videre arbeid for denne svermalgoritmen will be diskutert. Det blir konkludert med at dette svermsystemet, til tross for dens begrensninger, har et større potensial til å fungere som et rammeverk for en sann robust robotsverm i fremtiden.

Nøkkelord : robotic swarms, potential-field based control, thread mesh network, MQTT-SN, ROMANO, laser distance measurements, magnetometer, joint control- and communication algorithm, PCB design, algorithm design, robust swarm application

Prerequisites and resources

This master thesis, *Robust Potential Field-Based Communication & Control Architecture for Robotic Swarms* is given in cooperation with the Norwegian University of Science and Technology, NTNU, and Nordic Semiconductor. The project gives a new outlook on the field of swarm robotics, and little work is done beforehand. A part of this project is to decide what research question should be pursued based on the problem description.

Nordic Semiconductor contributes with giving access to its laboratory facilities and necessary development equipment. Full training to use the equipment is also provided, along with funding to create a swarm network and up to seven robotic devices at reasonable cost. Office space, computer and software with relevant licensing was also provided. Guidance on hardware design, software solutions, the different processor solutions and 3D-printing was provided by the applications group at Nordic Semiconductor.

The literature studies are based online through platforms such as Oria and arXiv, but also trough course material and with the use of the school library. The physical swarm platform consists of two printed circuit boards, a 3D-printed chassis and the firmware running on a nRF52-series microchip. The printed circuit boards are created through the use of Altium Designer, a tool for creating embedded systems. The printed circuit boards themselves are fabricated externally, but soldered by hand with the use of a stencil and an infrared heating chamber. Before the printed circuit boards were orded were some review done by Nordic Semiconductor in order to guarantee that they would work without revision. The antenna is measured through an RFX2 antenna gain measurer. The 3D-printed chassis is first designed in Onshape, and then printed with a Mojo Idea Series 3D printer. The firmware is developed in the Atom text editor and then compiled in the Segger Embedded Studios IDE. The firmware builds partly upon the Thread 11 Software Development Kit by Nordic Semiconductor. Additional external devices use ready-made firmware by Nordic Semiconductor, this applies to the Thread Border Router / MQTT-SN solution. A test platform for acquiring performance data from the swarm is created, based on Node.JS, mySQL and MATLAB.

Acknowledgements

This thesis marks the beginning of my life long dream.

I believe that everyone who knows me also recognizes my love for computer science, gaming and especially robotics. I still remember fondly my first encounter with a Nintendo 64, and the love for technology that sparked soon thereafter. It didn't take long before I set my eyes on Cybernetics and Robotics, which proved itself to be a long and bumpy road. I can only thank all of my friends, family and teachers for making it this far.

I would like to start by giving thanks to my two supervisors, Jan Tommy Gravdahl and Ketil Erichsen. You are probably not aware, but both of you have had a great influence in my life and future. To you, Tommy, who informed me about the opportunity to take a Bachelor's degree in order to apply for a master's program in Cybernetics. Without you would I probably not be aware of my options. And to you, Ketil, who helped me through both my Bachelor's and Master's thesis, and also helped me make a future career. I couldn't look more forward to work with you as your colleague. I thank you both.

To all of my friends, this has truly been a journey! Thank you for waiting. I could never even begin to possibly imagine all the love and support you have given me over these past years. If only I could mention you all! I truly mean it when I say that you have made me feel accomplished and capable. Thank you for lending me your strength, I will never forget it.

To my mother, father, sister and grandmother; thank you for teaching me about the world, and your much needed help in troubled times. I love you all.

I would like to give a special thanks to TT, for your support and expertise in linguistics. To Nicole, Simen and Johann for sticking with me on my journeys and shenanigans, and to my partners in crime during my studies and start of career, Martin and Jan Tore.

And most importantly of all, to Idun. I truly mean it when I say that this wouldn't be possible without you. Thank you for being there from the start. This one is for you.

Contents

List of Figures

1 Introduction

1.1 Background

Swarm technology is without a doubt an active area of research within the robotic field [1, 2, 3, 4]. The idea of a robotic swarm is that many simple robots work together in order to execute increasingly complicated tasks, giving the swarm near-infinite use cases. The robotic swarm problem could be defined as how one implements a robotic swarm for any given task, whilst a solution is a complete implementation of a robotic swarm in said task. Such a solution could open up for new technologies, and could be implemented in dealing with tasks such as search-and-rescue [5], the deployment of an *ad hoc* wireless network [6], and surveillance [4]. No matter the discussion will robustness clearly be a very important aspect for any robotic swarm. In a worst-case scenario could a robotic swarm lose several members after performing a task, either because a node moves out of range or because the terrain constrains the cluster of robots capability to communicate with each other. In a search-and-rescue scenario could it be critical if a robotic swarm lost its capability to perform a predefined task.

An interesting approach within robotic swarms, although not a formal one, is to divide the robotic swarm problem into five subproblems. These subproblems are control, hardware design, communication, swarm intelligence and in-room localization, see figure 1.1. This Divide-and-Conquer mindset is a common method in algorithm theory [7, p. 65]. The idea is that each subproblem is solved, and then the solutions of the subproblems are combined into the solution for the original problem. It should be noted that it is more reasonable to solve each subproblem in parallell instead of separately. For example would some of the hardware be determined by what communication protocol that is decided to been used, i.e. the antenna. The swarm problem itself is broad, and with this approach it is easier to discuss certain parts of robotic swarms individually.

When a robotic swarm performs any task, more often than not will this action require some sort of movement. Now, in order for a robot to move within a space, it needs to understand on a certain level how to perform this movement. Which actuators that should move and how much power they will apply over a time domain is usually governed by *control theory*. By introducing a feedback to each robot of how they are positioned within a space, in conjunction with the desired position of the robot, one can perform necessary computations to arrive at a signal output that should set the robot in the correct position. An interesting problem is the strategy for this control. The control strategy could be linear of some sorts [8, 9, p. 21, p. 282], non-linear [10, 11, p. 469-505, p. 289], or one could introduce predictive control [12], to mention some common strategies. Each control strategy are different and have their own advantages and disadvantages. Its a problem in an of itself to determine which strategy is best suited for the swarm task. However, if a control strategy is poorly implemented one could reach a sub-optimal system, which is either underdamped or overdamp, or even unstable. Many control strategies are governed by the laws of physics and proper modeling strategies are important for the system to compute the right control action [13]. If a control strategy is poorly implemented, each member of the robotic swarm could perform unwanted motions which could be catastrophic for the swarm. One should also note that for this case a good control strategy heavily relies on some sort of localization feedback, and the control action itself relies on proper hardware to perform necessary actuation.

Hardware design for robotic swarms poses interesting problems. The hardware decides what a single robot can and can't do, and in turn affects the abilities of the whole swarm. Naturally, a human can't walk properly without legs, or speak without functioning vocal chords. This applies for the domain of robotics too, a robot needs all the necessary peripherals to perform certain actions, and a sensory system would most like also be needed to provide some sort of feedback to the robot. Surely, a processing unit would be needed to govern the entire system, and a power system to boot. All of these choices would probably lead to some sort of PCB design, with its own governing firmware. This is the problem of designing embedded hardware [14]. These problems are of course up for discussion, because the specifications of a robot are naturally not set in stone. For what exactly are the hardware needed for a member of a swarm? Surely this is determined by the specific task the robotic swarm is supposed to perform, and what is required by the solutions to the other sub-problems discussed earlier. It is important that this embedded system is robust. Short circuits, loss of data, unwanted resets of the processing unit, faulty actuators and sensory systems or even a badly tuned antenna will jeopardize the entire swarm system.

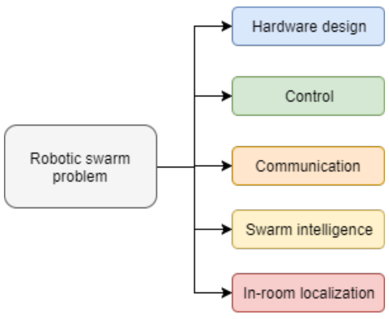


Figure 1.1: The division of the swarm problem into five lesser problems.

In order to implement position data as a feedback for the control strategy, one needs some sort of infrastructure in order to localize its position. Localization in particular is not something that is unique for robotic swarms. It has been on several occasions been discussed in great detail. A common approach for localization is through computer vision [15, p. 42], although limiting the space the swarm can work within. Another approach is to use a sonar system, creating an echolocating system similar to bats. One could also use infrared sensors in order to keep track of its position relative in the swarm [1, p. 797]. Another approach for relative positioning, - bridging the gap between localization and intelligence -, is to use the Min-max and Particle Swarm Optimization algorithm in order to compute its relative position based on pre-defined anchors [16]. These are only some of many solutions existing within the world of localization. It is thoroughly researched already, but necessary for a robotic swarm. If the problem of localization was not solved, one would not be able to implement position feedback for the control system.

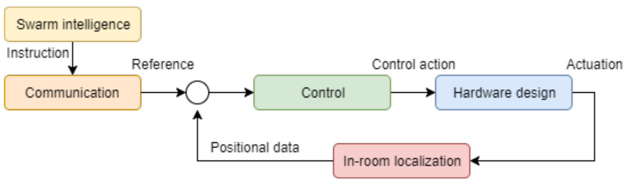


Figure 1.2: How the different problems can work together in a system from an intelligent instruction to a physical actuation in the swarm problem .

Additionally is communication in the digital realm possibly one of the most interesting technologies in this era. Without digital communication would not have the Internet. For a robotic swarm, communication decides how the different members should interact with each other, and also how the entire swarm could communicate with an external user or client. Intuitively, it is crucial for every member of the robot to be able to communicate with the rest of the swarm. If not, the member would struggle with being an active part within the swarm; for example, it would be difficult to understand when and where to move. On the issue of communication rises many problems, such as what medium should the swarm use for communication, what kind of protocols should govern the communication and what data should be sent. Not surprisingly is wireless communication a popular choice for swarm communication [6, 3, 4, 17, 1, 2]. However, for wireless communication, - especially with an increasingly large swarm -, arises the problems of noise. Wireless radio protocols such as ZigBee, Thread and Bluetooth is constrained by bandwith [18, 19, 20], and when a lot of data is sent in the same space, it might increase package loss [2, 5, p. 4,p. 112].

Lastly, its the matter of artificial intelligence. This is an interesting topic for all sorts of robotics, and robotic swarms are no exception. An intelligent swarm could make decisions on how it should move, which robot should move where, what should be done in order to recover lost nodes and of course, learning. A robotic swarm equipped with the ability to learn about the environment on-line, to *reason* on uncertainties and make both simple and complex decisions would of course be powerful. However, to even design a knowledge base and a language representation to obtain such a learning environment is complex, and is its own field of study entirely [21]. Additionally, a learning environment would require a good amount of computational resources, further stressing the processing units on-board the robots. However, an intelligent swarm would lead to a more robust system because of its potential path finding and decision making during the loss of members within the swarm.

Although intelligence is an interesting study, it is not believed to be the most important aspect within robust robotic swarms. As long as one predefines every task and command for the robotic swarm, one could rule out the need to research swarm intelligence for now. What is interesting to research further is how the problem of swarm control and swarm communications can be solved and tied-in together in order to achieve robustness. If either control and communication is non-robust, one would possibly lose a node during certain tasks damaging the longevity of a swarm.

1.2 Project description

This project is given in cooperation by the Norwegian University of Science and Technology and Nordic Semiconductor ASA. The given assignment task are as follows:

> *Nordic Semiconductor wants to research how our newest series of System-on-Chip, the nRF52-series, could be used for a swarm application. The nRF52-series brings total flexibility with low-power solutions that could be used for robotic devices in a restrictive environment. This project should focus on both the control and communication aspects of a robotic swarm with robustness in mind. First, a robust communication algorithm for communicating to and within a varying cluster size of nodes will be researched and discussed. This algorithm could be implemented as a layer on top of an already existing communication protocol; such as Bluetooth Mesh, ZigBee and Thread. Secondly, a robust control algorithm should be proposed which governs a node such that it does not execute a motion that would result in a loss of connection.*
> ***Key points for the assignment**:*
>
> - *Perform a thorough literature study in order to map out the most important parts of the robotic swarm.*
> - *Propose a control and communication algorithm for a robotic swarm with robustness in mind. A node should be able to seamlessly join the robotic swarm when desired. Also upon loss of a node, measures should be taken in order to reacquire said node.*
> - *Weigh different communication protocols against each other and argument for which would suit best in a swarm setting. Here should different network topologies be discussed, such as Point-to-Point, Broadcast and Mesh.*
> - *Weigh different control solutions against each other, and propose a control law that would contribute to the most robust solution in a swarm setting. Keywords here are Proportional-Integral-Derivative controller, Linear-Quadratic-Controller and Linear-Quadratic-Gaussian control, although not restrictive.*
> - *As a proof-of-concept should the control and communication algorithm run on the nRF52 Series System on Chip. Here both dual- and single-chip solutions are desirable. A system architecture should be introduced running the entire system in a swarm setting, essentially working as a test platform.*
>
> *As the project is broad and rather new is it up to the student to decide which parts of the project should be the focus points, given the time scope of the project.*

This project will pursue the following question;

Which technologies and algorithmic design are suitable for both the control strategy and the communication protocol to ensure robustness within a robotic swarm?

The goal of this project is to arrive at a robust platform for future swarm applications. An answer to this question could possibly lead- to a framework which can be built upon to give a future solution to the entire robotic swarm problem. A system architecture running the choices for control and communication will be realized in a practical setting.

1.3 Important previous studies

It is important to note that this project is a continuation of previous course work done by Halvorsen [22], the same author of this project. The previous work was done in the course program *TTK 4552 Engineering Cybernetics Specialization Project*, and its purpose was to do a feasibility study for the current master's thesis. The course itself was a program at NTNU worth 7.5 points out of a possible 30, resulting in an estimated time scale of 180 hours. Most of this feasibility study revolved around obtaining an understanding of the field of swarm robotics through a literature study. Additionally were some design choices made that have influenced the work done in this work, such as the general strategy for communication and control. However, as the previous course work done is not published, - and therefore not available to the reader -, was it decided that his thesis should be an independent report. This means that some work done in the previous project will be presented here, and the sections this relates to are sections 1.1, 2.1, 2.2, 3 and 4.1.2. It was seen as more efficient to include the previous work rather than use time to write the same presentations and discussions all over again. It must be emphasized that these sections are *based* on the previous work on different degrees. Most work has been further developed and changed as this project progressed, based on new findings. Some parts of the theory has been left out as it were no longer seen as relevant, and other parts were further elaborated. Nevertheless will the adoption of the previous work result in this project to be standalone.

An interesting approach for robotic swarms is to use them for deployment of a wireless mesh network (WMN), discussed by Hatori et al [6]. This is commonly known as the *sensor cover problem*, where a WMN can provide a temporary communication network in the aftermath of a major disaster. A mesh network, which is a network where every member of the network communicate with each other, is shown in this study to contain promise in the problem of network coverage [6, p. 440]. The proposed method uses roughly estimated positional data obtained by the Radio Signal Strength Indicator (RSSI) value, which is a property of several communication protocols indicating signal strength in-between nodes. Hatori et al proposes an algorithm which handles simple decision making within the swarms, adaptive direct control and a movement function in the case of a collision with an obstacle such as a wall. The algorithm does not require any sort of localization except from the rough data given by RSSI values. The algorithm defines members of the swarm to be anchors and other members to be repulsed by the anchor's RSSI data. When the RSSI signal has been lowered to an acceptable value will the member halt its movement, and through this method will the network increase its coverage. This method of using RSSI-data shows promise in the way of robustness.

As stated by Ghosh *et al.* [2] is the Robot Operating System (ROS) a common and effective method for control of robots and acquiring relevant data [23]. However, this Operating System requires extensive hardware with high computational power, similar to a computer. Ghosh *et al.* proposes the Robotic Overlay coMmunicAtioN prOtocol, ROMANO, a lightweight overlay for communication protocols on the application level. On a practical level does this algorithm define what data is to be sent in-between robots, and what the message formats are. ROMANO works as an abstraction layer on top of the MQTT-SN protocol [24], and shows great promise for its robustness [2, p. 5]. The MQTT-SN protocol is a a protocol that can work with both Bluetooth, ZigBee and Thread as an overlay protocol, and it has been shown to be both fast and reliable for robotic applications [25]. It is believed that ROMANO is currently the only application overlay protocol for robotic swarms proposed. However, ROMANO have only implemented simple instructions; from sensor data to movement commands as *forwards* and *move right*. There is no functionality to couple the ROMANO directly with a controller and coordinate system as of yet. ROMANO and MQTT-SN will be further discussed in section 2.1.3.

In a study done by Dunbar and Esposito in 2005 is a Potential Field Controller (PFC) for a robotic swarm proposed [15]. It is believed to be one of the first time a PFC controller is proposed for the swarm problem. The idea is to create artificial potential fields that will either repulse or attract a robot to its destination. In the proposed potential function is the power of the radio transmitter and inter-robot collision accounted for [15, p. 23]. It is stated that the PFC increases connectivity by around 200-300%, but is demanding excessive computational power resulting in an 25-45% increase in time to reach any goal. However, this algorithm does not take into account modern technologies such as increased computational power, mesh-networks and dynamic RSSI values. It is becoming apparent that by coupling communication and control together could possibly lead to increased robustness. A modern approach to the PFC, proposed in 2017 by Rasekhipour *et al.* [26], is to use the PFC and an Model Predictive Controller (MPC) in conjunction with each other for path planning [12]. MPC is an optimal controller which computes the optimal control action for each iteration and a moving horizon into the future based on the system dynamics [12, p. 39]. A PFC does not contain system dynamics, and this approach is interesting in which a PFC is implemented in cascade with another controller in order to enhance its functionality. This is shown to have promising results for the sake of robustness [26, p. 1265].

During the case studies done in this report it started to dawn an interesting idea in how to combine communication and control for the sake of robustness. This idea was to combine a PFC with RSSI values to let the communication topology directly affect the potential fields used for control. It was believed that this idea was an original one; however during the literature study it was discovered that this idea was proposed as early as 2011 by Penders *et al.* [5]. Here there is proposed a system that uses PFC to follow a fire fighter who walks through an unknown environment. The robots will follow the fire fighter and map out the area, but will not move too far out because of a repulsive potential field directly controlled by the RSSI values. Although this study shows great promise is the report itself not that extensive. Exactly how each part of the algorithm and swarm system works is not fully accounted for, and the system itself is not replicable. The ideas and thoughts went into this project should surely be built upon, but the implementation lacks necessary documentation to be doing anything worth while with. However, the idea of using RSSI with PFC is shown to have promise, even in an unknown environment [5, p. 112].

1.4 Approach

A swarm system that houses solutions for both control, communication, hardware design and in-room localization is researched. It is believed that all these parts are of equal importance to arrive at a true robust swarm application platform. This platform could be further built upon by introducing an intelligence or methods for complex decision-making, but for now will all types of decision making be disregarded.

Robotic swarm applications are a relatively new area of research in constant development. It was therefore necessary to perform literature studies in order to map this research field before progressing further. The literature studies are based online through platforms such as Oria and arXiv, but also trough course material and with the use of the school library. The literature study lays the foundation for the swarm algorithm, - and the entire swarm system -, presented in this project.

Additionally is a physical platform designed to run the swarm algorithm. The purpose of this physical platform is to run performance tests on real-world scenarios. This would make it possible to create a mapping of the proposed swarm applications performance and possible revisions for the future. The physical platform consists of two printed circuit boards, a 3D-printed chassis and the firmware running on the nRF52-series microchip. The printed circuit boards is created through the use of Altium Designer [27], a tool for creating embedded systems. The printed circuit boards themselves are fabricated externally, but soldered by hand with the use of a stencil and an infrared heating chamber. The antenna is measured through an RFX2 antenna gain measurer [28]. The 3D-printed chassis is first designed in Onshape [29], and then printed with a Mojo Idea Series 3D printer. The firmware is developed in the Atom text editor [30] and then compiled in the Segger Embedded Studios IDE [31]. The firmware builds partly upon the Thread 11 Software Development Kit by Nordic Semiconductor [32]. Additional external devices uses ready-made firmware by Nordic Semiconductor, this applies to the Thread Border Router/ MQTT-SN solution.

Further are scenarios, expected behaviour and different tests defined. A test platform for acquiring performance data from the swarm is created, based on Node.JS [33], mySQL and MATLAB [34]. The swarm systems performance, shortcomings, and what research remains to be done in the future is discussed based on the data acquired from the test platform. A mapping of the swarm systems robustness is discussed before one arrives at a conclusion.

1.5 Outline

This report will first set a theoretical background that will be used for further development of the swarm algorithm. The theoretical background is divided into three parts. First, in section 2.1 will different methods and protocols for digital communication be presented. Digital communication is a vast field, and the focus points are those that are relevant for the swarm application. Different control strategies that may be applicable to the swarm system are presented in section 2.2, and different sensory systems are discussed in section 2.3. It is important to note that all theory that has influenced this project is presented here.

The solutions for control and communication are presented in section 3, as they make up the swarm algorithm. Similarly are the solutions for hardware design and in-room localization presented in section 3, along with the realization of the aforementioned algorithm. This makes up the swarm system architecture. The theoretical foundation laid earlier is used to discuss all design choices for said solutions.

The test platform and the definition of different tests are explained in section 5. All results acquired from the data acquisition are presented and considered in section 6. Potential shortcomings and future work will also be presented here. A final conclusion based on the previous work and the test results are presented in section 6.

1.6 Contributions

This project contributes to the following innovations within the field of swarm robotics:

- Complete joint control- and communication algorithm
  - Thread based mesh network with MQTT-SN and ROMANOs at the application layer.
  - Complete definition of the ROMANOs overlay communication protocol.
  - Moving average CRC filtration for swarm applications.
  - Potential-field based PID control strategy.
  - Definition of signal strength, obstacle avoidance and heading fields.
  - Pseudo-code abstraction and illustration of the algorithm methods and behaviour.

- Swarm system architecture with hardware platform and realization of swarm algorithm.
  - Main board PCB with the nRF52840 SoC and all electrical components to fulfill the specifications introduced from the algorithm design. With added flexibility for potential revisions of the algorithm.
  - Secondary board PCB with laser sensors for range measurements. Houses a connection interface for mounting on the main board.
  - Robot chassis design and 3D-model that houses main- and secondary PCB's, battery and two micro metal DC motors.
  - Suggestion of a swarm hierarchy for long range communication, consisting of the swarm robot devices, a joint Thread Border Router/MQTT-SN Gateway, and an external MQTT broker.
- Test platform and listening device for the swarm application for data acquisition using Node.JS, mySQL and MATLAB.
- Illustrations and datasets of the different parts of the swarm architecture, mapping the systems complete robustness.

1.7 Limitations

The swarm application consists of many different parts, ranging from mathematics, algorithm design, soldering, programming, 3D-design and so on. Several parts of this project required the use of skills that were new and never tried out before as a student. A big limitation for this project was therefore to acquire a new set of skills and more knowledge. This meant that some parts of the project would use more time and resources than initially planned for. As a consequence were a there a few short-cuts to be had, and there was little room for revision. The project itself consists of a total of 30 points, meaning that it should be worked on full-time during a semester. With a complete week set to 40 working hours, one arrives at the following:

$$40hours \cdot 20weeks = 800hours$$

One could say that 800 hours is a relatively large amount of time. Yet this time is again subdivided to different parts, from planning, research, design, construction, testing, analysis and writing. It is important to use the given time efficiently.

Additionally are robotic swarm applications vast, complex, and ever-changing. It is difficult and infeasible to map out the whole research field over a single year. Some of the research discussed in this report is brand-new, and was published in the start of this project [2], but changed the approach of this project dramatically. It is to be expected that the same could happen for the future. Previous knowledge is also one limitation, although not that restrictive. Swarm applications is not greatly discussed in the cybernetics program, so some time resources this semester was used to acquire new knowledge about a previously unknown subject.

2 Theoretical background

2.1 Communication networks

In this section will there be discussed different properties of digital communication. This topic is complex and can be discussed extensively, which does not fit the scope of this thesis. For this project is it sufficient to address the main aspects of communication that contributes to a systems robustness. First, in section 2.1.1 will there be discussed what tools one can use to indicate the received signal strength of a device within a communication network. Secondly, in section 2.1.2 will three important communication networks be introduced, Zigbee, Bluetooth and Thread; whilst delving deeper into the inner workings of the Thread topology in subsection 2.1.2.1. In section 2.1.3, will the upper most layer of communication protocols be discussed by introducing COAP and MQTT; with an elaboration on MQTT-SN in section 2.1.3.1. Lastly, in section 2.1.4, will the application overlay protocol ROMANO be looked into, which utilizes MQTT-SN.

2.1.1 Received signal strength indication

One property of digital communication is to have an indication of a device's signal strength within a network. This is often adopted into a scalar value named *Received Signal Strength Indication (RSSI)*. RSSI describes the total signal power received from a given message in an arbitrary number, and is usually expressed in a logarithmic scale such as dBm [35]. The scale of the values is highly dependent of the physical antenna; and is therefore relative to the receiving device. The RSSI values usually resides between $0dBm$ and $-100dBm$; the higher the value, the stronger the signal. It is however stated in the specification for the nRF52-series that the valid accuracy range for the RSSI values is between $-20dBm$ and $-90dBm$, with a resolution of $1dBm$ [36, p. 289]. It should be noted that for a rapidly changing environment would the RSSI-value be subject to noise, as the signal strength diminishes if there is an object between the publishing and receiving device; RSSI is therefore an *indication* of the signal strength.

RSSI does not paint the entire picture. Although a device has a great signal strength value will it not guarantee the *quality* of the transmitted data. Accidental changes to raw data might occur regardless of signal strength. This change can be discovered through a cyclic redundancy check (CRC); a method for error detection. CRC is a *checksum* algorithm [14, 183]; the receiver computes a value based on the received data and checks whether or not if the value indicates an error on said data. The CRC algorithm bases its calculation on the modulo operation, and uses the following [37]:

- The received data, *D*, interpreted as a binary stream. Often called the divident.
- The generator polynomial, *P*, statically defined by the algorithm. Often called the divisor.

The CRC value is then given by:

$$CRC_{Value} = D \ mod \ P \tag{1}$$

Clearly, the generator polynomial determines the outcome of the CRC-value. The polynomial uses the received data,- perceived as a bitstream -, as coefficients. As the complexity of the polynomial rises, the higher the chance for detecting an data error. However, this comes at a efficiency cost. Different CRC-algorithms uses different generator polynomials; the peripheral radio in the NRF52-series defines the following [36, p. 261]:

$$G(x) = x^{16} + x^{12} + x^5 + 1 \tag{2}$$

This is commonly known as the CRC-16 ITU-T algorithm; it uses the value of bit 16, 12 and 5 from the received data as coefficients. It should be noted that CRC is a method for detecting an error, but not a method for acting upon it. Additional programming must be present to address such an error detection. Commonly will a CRC value of 0 indicate that no data is corrupted, but this must be defined on both communication ends.

2.1.2 Communication protocols

As previously stated are one of the requirements to use the nRF52-series in this project. This adds some constrains to what communication protocol that can be used, as the physical layer, the antenna, is already defined. The nRF52-series is primarily created with classic Bluetooth in mind, however it supports Bluetooth Mesh and radio protocols which conforms to the IEEE 802.15.4 radio standard, such as Zigbee and Thread. The documentation for Bluetooth Mesh, ZigBee and Thread are extensive, and are covered in their respective documentations [38, 19, 18]. However, some of the main selling points of each protocol will be discussed in order to establish an overview. This will make it possible to compare these protocols in the future. A comparison chart is given in figure 2.1, specifications like data rate and range are presented there, rather than *on the fly* in the text.

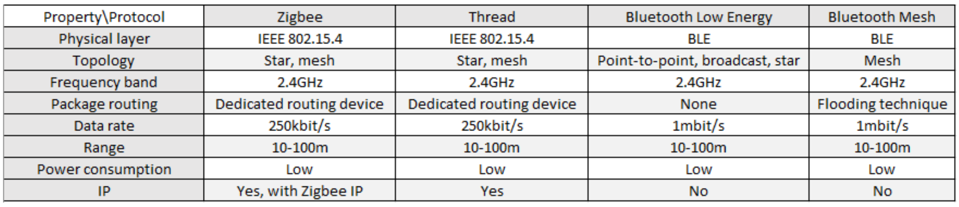
The Zigbee protocol has seen a lot of use in home appliances. It is marketed as being low-power, low-cost and a low-complexity networking solution for the Internet of Things. It is maintained by the Zigbee Alliance, and they offer three specifications that adheres to the Zigbee protocol; Zigbee PRO, Zigbee RF4CE and Zigbee IP. Zigbee PRO is the default specification and builds upon the IEEE 802.15.4 standard by adding application, network and security layers. The Zigbee RF4CE specification is mainly designed for two-way device-to-device control applications and is therefore less complex than Zigbee PRO and uses less memory. Lastly, the Zigbee IP specification is an open standards-based IPv6 specification for wireless sensor networks. It enables low-power devices to participate natively in a network with other IPv6 devices. From here on out, will the term Zigbee be used to discuss the Zigbee PRO specification [18].

The Zigbee specification, as previously stated, builds upon the IEEE 802.15.4 standard. This standard defines the physical layer and the link layer and are then built upon by Zigbee. The IEEE 802.15.4 standards physical layer manages the physical RF transceiver and defines functions for channel selection and energy and signal management. It operates in three unlicensed frequency bands:

- 868.0 - 868.6 MHz: Europe. One channel.
- 902.0 - 928.0 MHz: North America. Up to thirty channels.
- 2400.0 - 2483.5 MHz: Worldwide use. Up to sixteen channels.

The link layer introduces the media access control (MAC) protocol. This protocol takes care of addressing of destination stations, error protection and the access control of the physical medium. Zigbee enhances the functionality of the IEEE 802.15.4 standard by building a layered infrastructure upon this standard. Zigbee consists of discovery and pairing mechanisms, and incorporates power saving functionality and support for battery-less devices. Security is done through the AES-128 scheme [19, p. 4].

A Zigbee network has either a star, mesh or a tree topology and are composed of tree device types; the Zigbee Coordinator, Zigbee Router and Zigbee End Device. The Zigbee coordinator works as the central hub that is responsible for setting up the network. The network can be extended through the Zigbee Routers, whilst the Zigbee End Device usually consists of the control and sensory devices. Every Zigbee End Device is either connected directly to the Zigbee Coordinator or to exactly one Zigbee Router. The network can also be extended by a Zigbee Coordinator/Gateway in order to communicate across networks, such as the internet.



Property\Protocol	Zigbee	Thread	Bluetooth Low Energy	Bluetooth Mesh
Physical layer	IEEE 802.15.4	IEEE 802.15.4	BLE	BLE
Topology	Star, mesh	Star, mesh	Point-to-point, broadcast, star	Mesh
Frequency band	2.4GHz	2.4GHz	2.4GHz	2.4GHz
Package routing	Dedicated routing device	Dedicated routing device	None	Flooding technique
Data rate	250kbit/s	250kbit/s	1mbit/s	1mbit/s
Range	10-100m	10-100m	10-100m	10-100m
Power consumption	Low	Low	Low	Low
IP	Yes, with Zigbee IP	Yes	No	No

Figure 2.1: Comparison chart over the different communication protocols.

Another contender for the home network is Thread, first introduced in 2014 [19]. It builds upon the IEEE802.15.4 standard like Zigbee. It is maintained by the Thread Group, and they claim that the protocol is best suited for low-power and low-cost wireless device-to-device communication. The Thread protocol is specifically made for home applications that wishes to use IP-based networking, making the devices able to communicate with other networks over the internet. It uses UDP at the transport layer. In order for IPv6 packets to be sent over this standard uses Thread *IPv6 over Low-Power Wireless Personal Area Networks*, 6LoWPAN [19, p. 7]. 6LoWPAN defines encapsulation and header compression mechanisms that make the header data from IPv6 as lightweight as possible, making it suited for small devices and simple networks.

Thread consists of two topologies; star and mesh. When only two devices are connected to the network will a star topology be established. When another device connects to said network it will automatically develop to a mesh topology. A Thread network consists of four device types; Border Routers, Routers, Router-eligible End Devices (REED) and Sleepy End Devices (SED). A Border Router work as a central hub and sets up and manages the network, there may even be several Border Routers in a Thread network, working as gateways to other networks or extending a networks capabilities. Routers simply extends the network, forwarding messages in-between border routers and end-devices whilst also providing security services for devices trying to join the network. REEDs are end devices which have the capability to become Routers, but are not needed to inherit that role at the moment because of the current topology. SED's are the host devices, which communicate only with its parent Router. Thread is a self-healing network, meaning that if a end device does not find its parent Router it will establish a connection with another Router.

Both Zigbee and Thread needs a router for sending packets within the network. This is not the case for Bluetooth Low Energy (BLE). BLE is also known as Bluetooth Smart or Bluetooth 4.2 (or now Bluetooth 5.0 with the newest specification), from here on out will the standard be referred to as BLE [20]. BLE is a low-power wireless technology that is used to communicate between devices. Currently there exists three different topologies within the BLE-specification, point-to-point, broadcast and mesh. The BLE technology defines its own protocol stack, from the physical layer all up to the application layer. The physical layer consists of an RF transceiver that operates in the 2402 MHz - 2480 MHz band. BLE defines 40 channels within this band, where three channels are dedicated for advertisement. Since the 2.4 GHz ISM band is open, it is subject to a lot of traffic across several devices and protocols. Bluetooth uses a technique called *frequency hopping*; where the transmitting and receiving devices rapidly switch between the 40 frequency channels in order to minimize interference. The sequence is pseudorandom and known to both devices. BLE is also trying to be as low power as possible,- marketing as "ultra low power" -, by specifying that all devices should enter sleep cycles as much as possible.

Bluetooth Mesh is its own specification that builds upon BLE [38]. Its specification was released in mid 2017, and it expands the capabilities of traditional Bluetooth in order to create a mesh topology. Bluetooth Mesh is created with already established Bluetooth devices in mind; since Bluetooth Mesh is dependent and builds upon BLE, it natively supports all previously BLE-ready devices. Bluetooth Mesh uses a publish/subscribe messaging system. Devices can send a message to different addresses that other devices subscribe to. Every device that subscribes to this address will receive the message and be able to process it. Package routing is done by a technique called *flooding*. Some devices in the network are designated relays, and whenever a message is obtained by a relay will it broadcast it to the rest of the network. A message can and probably will be relayed several times, defines as *hops*. All messages includes a packet-field known as Time To Live, TTL. This values sets the number a message can hop, with a maximum of 127 times. Every device also consists of a cache so it can determine if has sent a message before or not. If it has, it will not process the message any further. Bluetooth Mesh also has a function to designate low power nodes that work in conjunction with designated devices called *friends*. These friends has a larger power capacity than the low power nodes, and it will store messages relayed to the low power nodes; only delivering them when the low power nodes asks for it. This removes the need for a low power node to enable its radio with a higher frequency than necessary in order to listen for possible messages.

2.1.2.1 Further into the thread protocol

One of the most noticeable design choices of the Thread topology is the *No Single Point of Failure* functionality [39, p. 27]. The idea is that the communication network will take recovery action when a point in the network fails to communicate. Initially, a device will on start up search for nearby devices. If it doesn't find a network to connect to, it will designate its role as a Leader. Note that this is independent of the device being either a Border Router or a Router. According to the Thread specification has the Leader the following responsibility [39, p. 27]:

> *A router or Border Router can assume a Leader role for certain functions in the Thread Network. This Leader is required to make decisions within the Thread Network Partition. For example, the Leader assigns router addresses and allows new router requests. The Leader is elected and if the Leader fails, another router or Border Router becomes the Leader.*

As previously stated are SED's dependent of a parent Router in order to communicate with the rest of the network. If this Router fails to respond will the SED search for another Router to choose as its parent. This might trigger an REED to change its role into a router; but the SED might fail to find an eligible router. The SED will then continue its search indefinitely. The main design philosophy is that the self-healing functionality is automatic from a users point of view. The transition between roles should not be visible to a user and does not affects a units behaviour on an application level.

When a device wants to join the network is it required to identify an already trusted device of the network, and communicate with said device in a point-to-point fashion [39, p. 29]. This is a part of a security measure to mitigate the number of rogue devices within a network. Additionally is the joining device not able to connect to the network without providing a security key during the joining process; communication is afterwards secured with a different network key. When neighboring devices establishes a communication link between each other can the units utilize Thread's *Neighbor Detection* functionality. The devices forms an estimate of their two-way link quality; as radio links can be asymmetric and be more reliable in one direction than the other [39, p. 59]. If the link quality is poor will the devices decide to not form a link in-between themselves, but rather use the mesh network for indirect two-way communication. Each device joining the network will receive a 2 byte short address, with the first byte being the router ID and the second byte being the child ID. This makes it possible to locate where a device exists in the network based on its address alone.

As mentioned in section 2.1.2 builds the Thread protocol upon the IEEE 802.15.4 standard at the physical layer, using 6LoWPAN, IPv6 and UDP. Additionally it adds commissioning protocols and security layers. However, Thread does not contribute with any protocols at the application layer; resulting in added ambiguity. It will be up to the network designer to add a suitable application protocol tailored for the task at hand. It should be noted that the IEEE 802.15.4 standard limits the size of the packets by 127 bytes at the physical layer to limit energy consumption [40, p. 4]. The application protocol should therefore have a small overhead in order to ensure compatibility with the physical standard.

2.1.3 The application layer - CoAP, MQTT and MQTT-SN

Among the many protocols that exist within the application layer are there mainly three who claim that their main purpose are for lightweight applications; CoAP, MQTT and MQTT-SN. By lightweight applications it is implied a system that does not require as much computational power and memory and is therefore suited for systems that inherits such constrains.

The Constrained Application Protocol (CoAP) is specified in RFC 7252 [41]. CoAP was standardized in 2014 and was co-developed by Internet Engineering Task Force and ARM. It is designed for web applications and is compatible with the Internet of Things network model, which enables a network using CoAP to communicate with other networks over the internet. CoAP depends on the Representational State Transfer (REST) architecture, similar to the Hyper Text Transfer Protocol [42]. The goal of CoAP is that it works as a subset of HTTP that works for constrained machine-to-machine applications. The CoAP protocol is based on the exchange of messages over UDP between endpoints. CoAP defines four types of messages: Confirmable, Non-confirmable, Acknowledgement and Reset. CoAP introduces a short fixed-length header for its data packages; with only 4 bytes which may be followed by compact binary options and then the payload. This gives the total message header a small header size; since the ensured minimum packet size of UDP is 8 bytes.

CoAP operates by *request and response*, which is similar to HTTP. A message sent from a client that arrives to a server will send back a response based on the message type. If the message is of the non-confirmable type, there is no need for said response [41, p. 23]. This message type is naturally more lightweight since it does not require a response from the server. If the message type is of the confirmable type will the client expect a response from the server, and will poll the message a set number of times is no response is given over a time interval [41, p. 21]. If no response is given will the client give up on sending the message. This type of response polling with confirmable messages is often called *reliable mode*. When it is sending non-confirmable messages, it is in *non-reliable mode*. Since CoAP is bound to UDP, which is an unreliable transport, it might result in messages arriving out of order and even not at all.

The MQ Telemetry Transport (MQTT) protocol was formerly developed by IBM and the current version was standardized by OASIS [43]. Instead of the request and response mechanism which CoAP operates on, this protocol is based on *publish and subscribe*. Clients communicate with each other through a central hub named a broker. In this mechanism, a subscriber subscribes to a certain *topic* on the broker. Publishers are then available to publish data on any topic, and the subscribers on said topic will receive the published payload. A client can both subscribe and publish to any number of topics. A topic name is a string, and its size can add a lot of weight to a constrained system. MQTT also supports the Internet of Things model, and works as an overlay for the TCP protocol, similar to CoAP being an overlay for UDP. TCP is more of an heavyweight compared to UDP; boasting a header size of 20 bytes. However, TCP is more reliable than UDP supporting the acknowledgement of sent messages and requiring a connection to be set up between nodes.

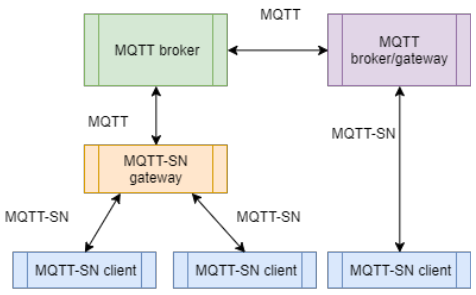


Figure 2.2: A typical network using both MQTT and MQTT-SN.

MQTT supports three different modes of operation which are called Quality of Service (QoS). QoS 0 mode sends a packet one time without requiring an acknowledgement. QoS 1 sends a packet several times while requiring an acknowledgement. The last mode, QoS 2, ensures that a packet is delivered exactly once. However, as stated previously, no matter the mode will TCP guarantee some sort of acknowledgements which could be sufficient. Acknowledgements require more traffic.

A more lightweight version of MQTT is the MQ Telemetry Transport for Sensor Nodes (MQTT-SN) [24]. MQTT-SN supports very much the same as MQTT, but also introduces some changes. A typical network utilizing both MQTT and MQTT-SN is shown in figure 2.2. Its a variant of the protocol which focuses on constrained devices similar to CoAP. This protocol is not required to adhere to the TCP/IP-model, and is normally implemented as an overlay for UDP [2], however it is not required. MQTT-SN supports topic ID instead of topic name, which is only 2-bytes. Upon registration the publisher requests a connection with both topic name and topic ID, and after registration messages are only published using the topic ID, which is less expensive. In order to ensure compatibility with MQTT will MQTT-SN introduce the notion of a gateway (GW). A gateway works as a translator and transforms MQTT-SN packets to standard MQTT packets. If the gateway is another device than the broker will the gateway connect to the broker through a TCP connection.

2.1.3.1 Further into the MQTT-SN protocol

A network using both MQTT and MQTT-SN will initially be established through a broker [24, p. 6]. Then, any device can either subscribe or publish to an arbitrary topic defined by the client. The broker keeps track on which devices are subscribed to what topic, and which topics are registered on the network. It is also the brokers responsibility to relay a published message to any subscribers to the topic in question. It should be noted that communication between a publisher and a subscriber is only done through the broker; because of this a simple peer-to-peer topology is not possible. The messages themselves is of course relayed within the mesh network to its appropriate end devices. It is possible for there to exist several gateways and brokers on the same network; mitigating the possibility of a single point of failure [44]. However, this will not be elaborated any further. The advantages of using MQTT-SN are that it minimises the number of messages being sent in an many-to-many mesh scenario through the broker, and it introduces little message overhead, seen in table 1.

Message Header (2 or 4 octets)	Message Variable Part (n octets)

Table 1: MQTT-SN general message overhead

The brokers communication is maintained over MQTT, which means that a MQTT-SN device must locate a MQTT/MQTT-SN gateway in order to communicate with the broker. The broker and the gateway can of course also be the one and the same device; but a MQTT-SN device must communicate trough the gateway regardless. A gateway could either be *transparent* or *aggregating*. A transparent gateway will setup and maintain a MQTT connection to the server for each and every MQTT-SN device, whilst a aggregating gateway maintains only one MQTT connection that is used for every MQTT-SN device. The transparent gateway works as a simple relay whilst an aggregating gateway is more complex and has to look at the actual data in order to understand what data should be sent. It is simply a matter of preference; a transparent gateway will require less computational power for the gateway unit, but will be more taxing for the server. When a gateway is connected to a server will it periodically advertise its presence through an ADVERTISE message, given by the time duration T_{ADV}. This will make it possible for clients to initiate a connection setup with the gateway. The ADVERTISE message is broadcasted to the entire network, and in order to avoid bandwidth congestion should the value of T_{ADV} at least be greater than 15 minutes [24, p. 20]. Instead of waiting for this long advertisement time is a client able to send a SEARCHGW message. The gateway will then wait for a set time $T_{SEARCHGW}$ and respond with a GWINFO message, containing the ID number of the gateway. The waiting time $T_{SEARCHGW}$ is employed in case that several clients is trying to search for a gateway at the same time, mitigating the responses needed from the gateway.

After the client receives a GWINFO message can the client setup a connection to the gateway. This must be done before normal communication and information exchange can be performed. The client will send information to the gateway and the gateway will then request the client's Will Topic and Will Message. This is information that may be sent to every other device in the network if the client fails, this information could be updated at any time by sending a TOPICUPD or a WILLMSGUPD message.

For a normal MQTT transmission from a client to the broker will a client first publish the topic name and then the payload. In MQTT-SN however is the topic name shortened into a topic ID to reduce the amount of bandwidth used. The client must therefore first register the topic name to the gateway, and the gateway will then return with a ID number for said topic name [24, p. 22]. The client can then start publishing data on the network using the topic ID continued with the payload. A similar procedure will be done for a subscription setup; the client will send a SUBSCRIBE message continued with the topic name to the gateway, which is then followed with a SUBACK and the topic ID from the gateway [24, p. 23]. Of course, some topic names are not legal and sometimes might the client not be available for communication at the moment since it is busy with another client. Several procedures are implemented to deal with these scenarios, and is described at great lengths in the MQTT-SN specification [24].

2.1.4 The ROMANO Protocol

As previously stated proposes Gosh *et al.* the Robotic Overlay coMmunicAtion prOtocol, ROMANO [2]. The ROMANO protocol is interesting since it works as an overlay on top of the already established MQTT-SN protocol. This means that the ROMANO protocol falls under the application layer together with MQTT-SN. The point of ROMANO is to work as a lightweight protocol for robotic applications with constrained computational power, introducing a message-format and -structure with small overhead.

Communication is done by establishing MQTT-SN topics. Any publisher of a topic is a transmitter node, while a subscriber to a topic is a receiving node. Because of the structure of MQTT-SN can any node be a transmitter, whilst a receiving node must first subscribe to a topic through the broker. This means that a ROMANO topology could either be one-to-one, one-to-many or many-to-many. ROMANO can also tell a receiver to subscribe to another topic if needed; and a receiver is able to publish certain kinds of data such as telemetry data whilst still maintaining the role as a receiver. One additional message which is possible through the ROMANO protocol is to publish a periodic "heartbeat" message, notifying the rest of the nodes of its presence. The ROMANO message formats are as follows [2, p. 3]:

ROMANO Message Format for Request Connected Nodes Info, Heartbeat Message, and ROMANO Connection Request

ROMANO Data Type (octet 0)	MSG Length (1)	ROMANO ID (2-9)

ROMANO Normal Data / Connected Nodes info message Format

ROMANO Data Type (octet 0)	MSG Length (1)	Data (2-k)

ROMANO MQTT SUB/UNSUB Control Message Format

ROMANO Data Type (octet 0)	MSG Length (1)	Topic to subscribe to or unsubscribe from (2-k)

ROMANO MQTT PUB Request Message Format

ROMANO Data Type (octet 0)	MSG Length (1)	MQTT Topic Length, (m) (2)	Topic ID (3-m)	Data (m+1-k)

ROMANO Movement Control Message Format

ROMANO Data Type (octet 0)	MSG Length (1)	Movement Control Type (2-3)	Movement Control Data (4-k)

ROMANO Sensor Data Message Format

ROMANO Data Type (octet 0)	MSG Length (1)	Sensor Type (2-3)	Sensor Data (4-k)

With the following movement commands:

Type	**Value**	**Movement Control Data**
Move Front	0x0000	Distance
Move Back	0x0001	Distance
Move Left	0x0002	Distance
Move Right	0x0003	Distance
Rotate Left	0x0004	Angle
Rotate Right	0x0005	Angle

The following datatypes are defined for ROMANO:

0x00: Normal data	0x06: Req. Con. Nodes Info
0x01: MQTT Subscribe	0x07: Con. Nodes Info
0x02: MQTT Unsubscribe	0x08: Heartbeat Message
0x03: MQTT Publish Req	0x09: ROMANO Con. Req.
0x04: Movement Control	0x0a: ROMANO Con. Ack
0x05: Sensor Data	0x0b-0xff: User Def. Message Types

ROMANO has a standard connection establishment/initialization phase which every device must follow in order to initiate the protocol:

1. Each end-device, such as a robot, must connect to a MQTT-SN broker/forwarder.

2. The 8 last characters of the device's IPv6 address is used as the device identifier. For example will a device with address ffff-ffff-1234-aaaa subscribe to the topic "1234aaaa". This address is published indefinitely by the end-device until a broker acknowledges the identifier, which can be used by the broker to communicate with the specific end-device.

3. The end-device then subscribes to the topic "common" which ROMANO uses for broadcasting data.

Based on this connection establishment/initialization phase is it clear that ROMANO has some requirements. Each end-device must run some sort of a MQTT-SN client, and the Network Layer must contain IPv6 or could be altered to another addressing-scheme protocol. ROMANO has shown to yield success with sending up to 200 messages a second, with no radio buffer failure [2, p. 5].

2.2 Control laws and strategies

In this section will four different control strategies be discussed that may be applicable to a robotic swarm. A control strategy deals with the control of a dynamic system in either continuous or discrete time. The goal of a control strategy is not only to achieve control, but doing so in a manner that ensures stability for a given plant. Control theory is vast and is discussed extensively [8, 9, 10, 11, 12]. What is of importance here is to discuss different control strategies on an implementation level, what principles govern each strategy in order to compute a control action, and how we ensure stability. First, in section 2.2.1 will two common linear controllers be discussed, the PID controller and LQR. Further will the notion of predictive control be introduced in section 2.2.2. Lastly will an interesting control strategy be introduced that uses the principle of artifical potential fields in section 2.2.3, the potential field controller. An abstract comparison between the control strategies on the user-level is given in figure 2.3.

2.2.1 Linear Controllers - PID and LQR

A common controller is the Proportional-Integral-Derivative (PID) controller [8]. The PID controller is an output feedback controller that computes an error value e as an difference between a desired setpoint and a measured output value. The control action can be expressed as:

$$u(k) = K_p e(k) + K_i \int_0^k e(k)dt + K_d \frac{de(k)}{dk}, \tag{3}$$

where K_p, K_i, K_d denote the coefficients for the proportional, integral and derivative terms respectively, and e(k) is the error term in discrete time between a measured and desired value, and u(k) is the control action. It is possible to remove one or several terms, which will in turn change the controllers dynamics.

The PID controller is fairly simple compared to other controllers, in that regard that one does not take the system dynamics *directly* into account. The controller is simply the summation of three unique terms that take an error term from a measured value into account, thus making it an output feedback controller. Given that the system is *stabilizable*, - that is if all unstable dynamics in the system is controllable -, then the controller itself is able to change the dynamics of the system. If one wants to tune the controller, one desires to change the system dynamics. One does this by altering the coefficients K_p, K_i, K_d for the controller, either directly on the control system or offline through computer simulations. The simulations will try to replicate the system dynamics and one can then use several techniques for PID-tuning, such as Ziegler-Nichols method [8, p. 44], step response or manual adjustments. One could also perform a mathematical analysis of the plant-controller system model and use tools such as frequency response [8, p. 137] and pole-placement and query with Routh-Hurwitz' method [8, p.126]. The PID controller has an easy implementation and thus sees use in industry.

The discrete time Linear Quadratic Regulator (LQR) minimizes a quadratic criteria given by [12]:

$$min_z f^{\infty}(z) = \sum_{k=0}^{\infty} \frac{1}{2} \Delta x_{k+1}^{\top} Q \Delta x_{k+1} + \frac{1}{2} \Delta u_k^{\top} R \Delta u_k \tag{4}$$

where

$$\Delta x_k = x_k^{mes} - x_k^{desired}$$

subject to equality constrains

$$x_{k+1} = \mathbf{A} x_k + \mathbf{B} u_k \tag{5a}$$

$$x_0 = given \tag{5b}$$

$$\mathbf{Q} \geq 0 \tag{5c}$$

$$\mathbf{R} > 0 \tag{5d}$$

The system model is given on a state space form in constraint equation 5a. Where **A** is the system matrix, **B** is the input matrix, whilst the matrices **Q** and **R** are weight matrices for the state vector x and input vector u respectively. The system is linear time invariant, meaning the linear system does not change its internal dynamics over time. The system dimensions are given by:

$$u_k \in \mathbb{R}^{n_u} \tag{6a}$$

$$x_k \in \mathbb{R}^{n_x} \tag{6b}$$

$$z^T = (u_0^T, \ldots, u_\infty^T, x_1^T, \ldots, x_\infty^T) \tag{6c}$$

Where n represents the complete time horizon in discrete-time. Minimizing this quadratic criteria yields a quadratic programming (QP) problem. Now, given that the system is controllable (and thus stabilizable), and the objective function in equation 4; one is able to give a solution to the system over an infinite time-horizon [12, p. 64, p. 65]:

$$u_k = -\mathbf{K} x_k \quad for \quad 0 \leq k \leq \infty \tag{7}$$

Where u_k is the control action, given by the feedback of the state vector x and **K** is the feedback gain matrix given by:

$$K = R^{-1} B^T P (I + B R^{-1} B^T P)^{-1} A \tag{8a}$$

$$P = Q + A^T P (I + B R^{-1} B^T P)^{-1} A \tag{8b}$$

$$P = P^T \geq 0 \tag{8c}$$

Where equation 8b is the algebraic riccati equation.

By inserting the solution of the objective function in equation 4 will one obtain the following result:

$$x_{t+1} = Ax_t + Bu_t = Ax_t - BKx_t = (A - BK)x_t \tag{9a}$$

Which shows that by employing state feedback into the system, and given that the system is controllable, one can in theory change the system poles. This will in turn change the dynamics of the system, meaning that an LQR could either be tuned with a pole placement technique, or by altering the weighing matrices Q and R, changing how much a change in either the states or an input has.

Another tool which can be done with a system model on state-space form is state estimation; the practice of estimating the state values through a model. Given an accurate model could a state estimator either work as a noise cancelling filter, or one could get a measure of previously non-detectable states. For this to be possible must the system be observable, if not it is not possible to detect every state of the system. A state estimator could be open loop and rely only on the model itself, or closed loop with an input from some sensory units. Employing a closed loop estimator will turn the system into an output-feedback controller similar to the PID. A common method is to combine an LQR with a Kalman Filter, yielding a Linear Quadratic Gaussian (LQG) controller [12, p. 68]. State estimation and LQG will not be further discussed here.

2.2.2 Model Predictive Control

An interesting approach in control theory is Model Predictive Control (MPC). A possible solution to the control problem is to solve an *open loop optimization problem* from a time horizon $t = 0$ to $t = N$, and then employing the precomputed control action to the actual plant. However, judging that a system might be time invariant, the model contains inaccuracies or the system itself might be perturbed in any way, this might be non-practical. Enter the MPC principle, which rather solves a *closed loop optimization problem*, as stated by Mayne et al. [12, p. 44]:

> *Model predictive control is a form of control in which the current control action is obtained by solving, at each sampling instant, a finite horizon open loop optimal control problem, using the current state of the plant as the initial state; the optimization yields an optimal control sequence and the first control in this sequence is applied to the plant.*

A Linear MPC applies the MPC concept to an objective function of a quadratic nature with linear constraints. as shown here:

$$min_z f(z) = \sum_{t=0}^{N-1} \frac{1}{2} x_{t+1}^{\top} Q_{t+1} x_{t+1} + d_{xt+1} x_{t+1} + \frac{1}{2} u_t^{\top} R_t u_t + d_{ut} u_t + \frac{1}{2} \Delta u_t^{\top} R_{\Delta t} u_t \tag{10}$$

subject to

$$x_{t+1} = \mathbf{A_t} x_t + \mathbf{B_t} u_t \tag{11a}$$
$$x_0, u_{-1} = given \tag{11b}$$
$$x^{low} \leq x_t \leq x^{high} \tag{11c}$$
$$u^{low} \leq u_t \leq u^{high} \tag{11d}$$
$$-\Delta u^{low} \leq \Delta u_t \leq \Delta u^{high} \tag{11e}$$

where

$$\mathbf{Q_t} \geq 0 \tag{11f}$$
$$\mathbf{R_t} \geq 0 \tag{11g}$$
$$\mathbf{R_{\delta t}} \geq 0 \tag{11h}$$
$$\Delta u_t = u_t - u_{t-1} \tag{11i}$$
$$z^T = (x_1^T, ..., x_N^T, u_0^T, ..., u_{N-1}^T) \tag{11j}$$
$$n = N \cdot (n_x + n_u) \tag{11k}$$

It should be noted that the MPC principle will also work on the non-linear case, however the quadratic problem is a good approximation of real-world problems such as economics [12, p. 21]. The MPC principle, as previously stated, revolves around applying some solver to the optimization problem presented in equations 10 and 11. The problem will be solved for each time iteration over a finite horizon, and the computed control action for that given iteration will then be applied.

The benefits of performing the MPC principle as a control action is mainly the fact that one will compute a (believed) optimal control action for each iteration, whilst taking the system dynamics into account. Surely will the accuracy of the system model be crucial for each computation. The MPC is able to work for both linear and non-linear systems, and time-invariant and time-variant systems alike. However, the longer the time-horizon, the more states must be computed and thus leading to a high demand of computational power. The states, as shown in equation 11k, will actually increase exponentially with the time-horizon, requiring a large amount of memory in short succession. One could reduce the time-horizon, but this could lead to a loss of accuracy of the MPC. However, the MPC leads to very intuitive tuning, as one can alter the weights for states and inputs in the objective functions and the constraints themselves for said states and inputs.

2.2.3 Potential Field Controller

Another approach to control is the *artificial field method* [15, p. 14]. The method involves projecting one or several artificial potential fields over the robots workspace. The potential field will either add or subtract to a value that the robot want to minimize. The potential fields will then dictate what control action the robot should perform in order to reach a local or global minimum. Often the artificial fields are scalar in the domain $\mathbb{R}^2$ or $\mathbb{R}^3$.

For a robotic swarm will the potential fields be used to move within a space, from an initial position to a goal position. Let x and y be coordinates describing the robots position in $\mathbb{R}^2$, and the potential field given by:

$$\phi(x, y) \tag{12}$$

The negative gradients is defined as:

$$\dot{x} = -\frac{\delta\phi}{\delta x} \tag{13a}$$

$$\dot{y} = -\frac{\delta\phi}{\delta y} \tag{13b}$$

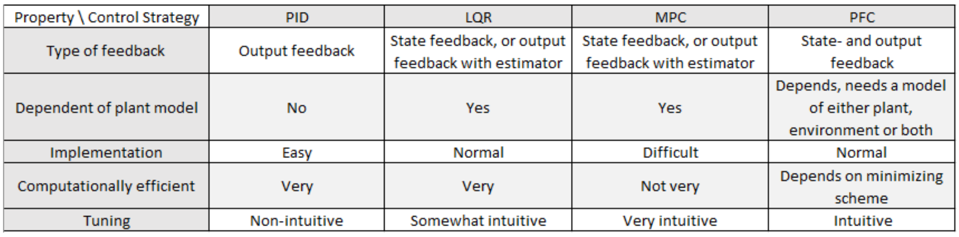
Which could then determine the body velocity. A potential field function could be the sum of several fields, such as the distance from a goal location, or the vicinity of an obstacle. A potential field function can be complex and the gradient might not be trivial to derive symbolically. One could either approximate the negative gradient as:

$$\frac{\delta\phi}{\delta x} = \frac{(\phi(x+\Delta) - \phi(x-\Delta))}{2\Delta} \tag{14}$$

or one could find a local/global minimum through linear, quadratic or non-linear programming [12], or one could solve the problem as a search problem [21, ch. 3].

A search problem needs to define an initial state, a description of possible actions and a goal state [21, p. 67]. For the potential field problem will the initial state be the robots initial position, and the goal state would be the robots goal position. The possible actions are the control actions that actuate the motors, and the potential fields themselves would be the *costs* by moving within the search-space. Several algorithms could be employed to search within this space, both uninformed and informed. An uninformed search algorithm have only access to the problem definition, whilst an informed search algorithm has access to an heuristic function indicating the distance from the goal state. Common uninformed algorithms are bread-first-search, depth-first-search and Dijkstra's algorithm [21, p. 85]. Common informed search algorithms are greedy best-first search and the A* search algorithm. These algorithms will not be further discussed here however.

The PFC's performance is greatly based on how the potential function is minimized. It is however very possible to make the PFC require small amounts of memory if it is needed. The potential field controller is somewhat intuitive also, and is indirectly tuned by how one defined the potential function. The PFC is also diverse; since the potential function is time-variant and changes based on sensory input. However, the system requires some degree of complexity in order to arrive at a goal state. If the potential function is non-convex it can contain several local minima which may or may not hinder the robot in arriving at a true goal state.



Property \ Control Strategy	PID	LQR	MPC	PFC
Type of feedback	Output feedback	State feedback, or output feedback with estimator	State feedback, or output feedback with estimator	State- and output feedback
Dependent of plant model	No	Yes	Yes	Depends, needs a model of either plant, environment or both
Implementation	Easy	Normal	Difficult	Normal
Computationally efficient	Very	Very	Not very	Depends on minimizing scheme
Tuning	Non-intuitive	Somewhat intuitive	Very intuitive	Intuitive

Figure 2.3: Comparison of the different control strategies.

2.3 Sensory systems

This section covers a couple of different sensory units that may be applicable to a swarm application. Sensory systems are discussed at great lengths in any respectable cybernetics program [14, 11, 45, 46, 47], and are by extension second nature to any engineer. However, it is of interest to discuss the advantages and disadvantages of some sensors that could be used as a feedback to the control part of the algorithm. First, in section 2.3.1 will three different type of proximity sensors be discussed that could be used for obstacle detection. These types of sensors are infrared sensors, ultrasonic sensors and laser sensors. Secondly, section 2.3.2 consists of a brief introduction to magnetometers and how they could be calibrated to give a better estimate of a units heading.

2.3.1 Obstacle detection

Obstacle detection is the method of using sensory systems to give some sort of input to a system indicating an obstacle in the vicinity. One of the more sophisticated ways of performing obstacle detection is through a vision system, i.e. a camera or similar; such as the Pixy CMUCam5 or the Arducam for the micro controller case [48, 49]. A system like this would make it possible to distinguish between different objects and store different data like size, distance and velocity to a robots memory and perform actions based on this data. However, as of working on this project are there no visual based system implementations for the nRF52-series. Creating such a system would not fit the time-scope of this project. An easier implementation would be to use proximity sensors for simple obstacle detection.

The three types of proximity sensors that will be discussed are infrared sensors (IR), ultrasonic sensors (US) and laser sensors (LS). Their basic principle is the same, a signal is emitted onto an surface and then bounced back to a receiver from said surface. The distance measurement is done based on the bounced back signal. For an IR sensor is this signal an infrared light. The distance measurement is done based on the reflected light intensity. As the receiver is irradiated with an infrared light source will it begin to carry a current [50, p. 2], which can then be measured via a micro controller or similar. As infrared sensors use light for measurements will they naturally inherit a low response time. Generally will a IR-receiver have a wide angular sensitivity cone, and is therefore widely used in remote controls. This will of course be determined by the receiver's lens. However, their non-linear behaviour and high dependence of the surface's reflectance results in a relatively low resolution for distance measurements [51, p. 1]. This can be mitigated by IR-sensors that base their measurements on the bounced back signal's phase shift, but these in general has a much higher cost [51, p. 1]. Additionally are there several sources of infrared light that could exist in an unknown environment; such as the sun or a fire source, making an IR receiver subject to noise. This could of course be dealt with through a specialised filter, without further elaboration in this report.

US sensors uses high-frequency ultrasound in order to measure distance. The measurements are most often based on the time of flight of the sound wave between the emission and the subsequent arrival of said sound wave, illustrated in equation 15. US sensors are often used in echo locating systems.

$$d = \frac{v_{sound} \cdot (t_{received} - t_{emitted})}{2} \tag{15}$$

Where d is the distance from a surface, v_{sound} is the speed of sound, and $t_{received}$ and $t_{emitted}$ are the time between receiving a sound signal and emitting one respectively. US sensors are widely used by the hobbyist today, as they are low cost and offer a fair precision. It is also shown to have an accuracy up to 3mm [52, p. 1]. They offer an advantage over IR sensors since they are not dependent of a surface's light reflective properties; meaning that dark or grey objects are not an issue. However, a soft surface such as cloth or carpeting could absorb the sound wave all together; or the sound wave might be deflected because of the surface's orientation, i.e. a corner or similar. The accuracy of a US sensor is highly dependent of the speed of sound of the traversed medium; which might change based on the temperature and humidity of the air it is being used in.

Lastly, LS sensors works on a similar principle as an IR sensor. The emitted signal is a laser, a focused amplified light source. The methods for distance measurement could either be based on the reflected light intensity [53], or by time of flight as for US sensors [54]. The main difference between traditional IR sensors and LS sensors are that for LS sensors is the signal more *focused*. This in a practical sense means that the signals amplitude range is relatively higher, but the sensitivity cone is scarcer. The higher amplitude range results in a much greater precision than for regular IR, opening up for imaging technologies such as LIDAR [55].

2.3.2 Magnetometer calibration

Magnetometers are sensors often used for navigation systems. Through this sensor is it possible to compute a units heading based on the earth's magnetic field. Today is the implementation of a magnetometer straight-forward, as there are a multiple of integrated circuits that houses a magnetometer solution. It should be noted that a magnetometer is prone to noise through magnetic interference from the sensors own integrated circuitry, or through nearby magnetic fields such as motors, inductors and nearby circuitry. These errors resulting from magnetic fields are often classified into two groups. *hard iron distortion* and *soft iron distortion* [56, p. 1]. Hard iron biases are the result of constant or slowly time-varying fields generated by ferromagnetic sturctural magnetic materials near the magnetometer. Soft iron biases are the result of magnetic fields generated by nearby circuitry as a response to an externally applied field. The magnetometer output for the two-dimensional case is given by Hermanth *et. al.* [56]:

$$h^2 = (\frac{h_x - B_x}{\gamma_x} - \frac{h_y - B_y}{\gamma_y}) \tag{16}$$

where
B_x and B_y are the hard iron bias parameters
γ_x and γ_y are the scale factor and soft iron bias parameters
h_x and h_y are the actual magnetic field strength relative to the earth.

A normal non-distorted output in conjunction with a distorted output is shown in figure 2.4. Here it can be seen that the magnetometer output has a circular nature. For this case is the device heading with respect to magnetic North given by trigonometry:

$$\theta = -\arctan\left(\frac{h_y}{h_x}\right) \tag{17}$$

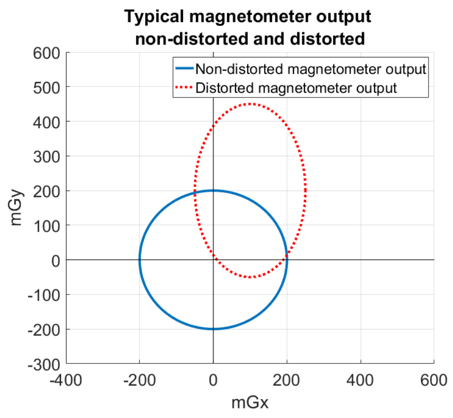


Figure 2.4: Typical distorted magnetometer output.

If hard iron biases are present will the centre of the magnetometer output be offset along the x and y axis. Furthermore will the presence of soft iron biases change the nature of the magnetometer's output from a circular to an elliptical one, as illustrated in 2.4. It is rare that these biases are absent in any electrical system. It is apparent that some sort of calibration is necessary for a biased output. Magnetometer calibration is a complex and broad field [57, 56], and performing a proper calibration could be an assignment in itself. A possible starting point is to use a simple min/max method based on non-calibrated output data from the magnetometer [58]. The min/max calibration method is based on computing an estimate for the hard- and soft iron biases. This estimation is based on a data series of the magnetometer output after performing figure-eight patterns. The estimates are then according to Winer given by [58]:

$$B_x = \frac{max(h_x) + min(h_x)}{2} \tag{18a}$$

$$B_y = \frac{max(h_x) + min(h_x)}{2} \tag{18b}$$

$$\gamma_x = \frac{max(h_y) - min(h_y)}{2} \tag{18c}$$

$$\gamma_y = \frac{max(h_x) - min(h_x)}{2} \tag{18d}$$

The estimates are very simple, but are shown to be suited for a sensor-fusion case with an accelerometer and gyrometer in order to get orientation data [59, 69]. Although speculative is it seen as a good start in order to get sensible heading data.

3 Control- and communication algorithm

In this chapter will the choices for design of the swarm algorithm be discussed. The architecture and technologies used for both the communication and control parts are covered, as they make up the complete algorithm. It is believed that both communication and control play an important part into ensuring a robust system for robotic swarms. First, in section 3.1 will the communication part of the algorithm be covered. The section then proceeds to discuss the proposed control strategy in section 3.2. Lastly will the joint communication and control architecture for robotic swarms be presented in section 3.3.

3.1 Choices for the communication protocol

As discussed in section 2.1 are there several aspects that define digital communication. Everything from a networks topology, to its layered structure, package and header size and the overarching network topology are all aspects that define digital communication. Since the discussion in this report focuses on robustness are there several principles that should hold:

- **Topology**. The network topology should contribute to the certainty that a message arrives at its destination in a dynamic, time-varying network.
- **Message size.** The messages should have word-length that is as small as possible, making the messages lightweight and giving the data smaller odds to be corrupted.
- **Layered structure.** The layered structure of the communication protocol should add little data overhead.
- **Communication technology.** The underlying protocols and layers used for the communication protocol should be readily available (supported by already-established hardware solutions with a software infrastructure), low-power and low-complex for devices with constrained computational power and low energy reservoirs.

On the matter of topologies are there several to choose from in this project, namely point-to-point, star, broadcast and mesh. The swarm application is typically many-to-many, meaning the topology must support two-way communication from every device to every device. By this definition will a point-to-point topology be counter-intuitive, meaning the network will support one-to-one communication. Now, a broadcast topology would be doable; a central hub could broadcast messages and the devices within the swarm could process this message and act upon it. This would result in a one-to-many communication structure; the devices would have to change from a listening device to a broadcasting device in order to relay its messages, which is cumbersome. The star topology is a good candidate, and is a feasible solution. A central hub could relay data in-between all nodes, establishing communication between all devices. However, in the case that the central node fails and can no longer fulfill its role; the network itself would terminate. For the sake of robustness would the mesh network be superior, adding much-needed redundancy to the network and no single-point of failure. The cost would be the complexity of network. Robustness is the most important aspect of the algorithm, and it is therefore chosen to employ a mesh topology.

As previously stated in section 2.1.2 are there three established communication technologies to choose from; Zigbee, Thread and Bluetooth. These are all competing Internet-of-Things protocols. As previously discussed is the mesh configuration considered for all these technologies, as the desired communication topology would be a many-to-many network. All protocols operates in the same frequency bands, and they all are constructed with low power in mind and sharing the same range. Zigbee is the oldest of the protocols, and has an established infrastructure and sees use in a lot of devices. Bluetooth Mesh inherits the vast infrastructure from BLE, although with added complexity. Thread however is a fairly new protocol and does not see as much use as of yet, but the Thread Group is supported by large power houses such as Google and Samsung. Since Thread uses the same physical layer as Zigbee can it more often than not be used on the same devices. The nRF52-series supports all protocols, and currently has software development kits ready for both Thread and Bluetooth Mesh. Because of this would it be too time-consuming to write a working implementation of Zigbee for the nRF52-series, and Zigbee is therefore no longer considered. Bluetooth Mesh is an interesting new technology that will most likely see use in the future, but its complexity is jarring for a swarm solution, as it is stacking layer upon layer on BLE. Also, Bluetooth Smart's flooding technique is promising for the sake of redundancy, but could pose difficulties for a time-sensitive application such as a robotic swarm. As the swarm increases in size will possibly several devices flood the network, leading to a noisy and saturated frequency band.

Zigbee is shown to be robust and is already used in several robotic swarms [15, 17, 6, 2]. As Thread is a minimalist protocol by using UDP and 6LoWPAN, - adding as little header data as possible -, and it uses the same physical layer as Zigbee, one could expect the same robustness if not greater. Thread will be used further in this project, over Zigbee and Bluetooth Mesh. Thread's network layer contains IPv6, and it also does not define an application layer, making it perfectly suited for ROMANO over MQTT-SN.

ROMANO, discussed in section 2.1.4, has shown promise in the case of robustness. As it builds upon MQTT-SN, will MQTT-SN be chosen over CoAP. ROMANO promises that up to 200 messages a second will be successfully received within a network with a rate of 95%, see section 2.1.4. This is promising, but the ROMANO Movement Control Message Format is lacking. This control scheme is created with a central hub in mind that controls one robot at a time with an arbitrary movement, which needs to be continuously updated for complex movement. Surely, it would be more efficient to rather send positional data and incorporate a controller to handle the path planning. Instead of arbitrary movement commands such as *Move Front* and *Rotate Left*, is it desired to send data such as *Heading* and *Speed*, which are reference values given to the robots controller.

A possible solution would be to introduce an altered version of the ROMANO protocol with the swarm application in mind, called ROMANOs. The main ideology of ROMANOs is to compress the amount of data overhead even further, removing unnecessary data and non-essential functionalities. ROMANOs inherits all the functionalities from ROMANO, but the following changes apply:

ROMANOs Normal Data Message Format

ROMANOs Data Type (octet 0)	MSG Length (1)	Message Data (2-k)

ROMANOs Movement Control Message Format

ROMANOs Data Type (octet 0)	Movement Control Type (2-4)	Movement Control Data (4-k)

ROMANOs Request Sensor Data

ROMANOs Data Type (octet 0)	Sensor Type (2)	Device MAC Address (6)

ROMANOs Sensor Data

ROMANOs Data Type (octet 0)	Sensor Type (1)	Device MAC Address (6)	Sensor Data (1-k)

ROMANOs Connected Nodes Info Format (Mutual for request)

ROMANOs Data Type (octet 0)	Device MAC Address (6)

With the following movement commands:

Type	Value	Movement Control Data
Heading	0x0000	Heading control reference
Speed	0x0001	Speed control reference
Pre Programmed Motions	0x0002 - 0xFFFF	User defined data

and the following sensor types:

Type	Value	Length (octet)	Sensor Data
RSSI	0x0000	(1)	Perceived signal strength of the device
Obstacle sensor	0x0001	(8)	Distance from obstacle NESW (2 byte per sensor)
Heading	0x0002	(2)	Heading attitude in the NED frame
Voltage	0x0003	(2)	Battery voltage of the device, given in floating number

The following datatypes are defined for ROMANOs:

0x00: Normal data	0x04: Req. Con. Nodes Info
0x01: Movement Control e	0x05: Con. Nodes Info
0x02: Request Sensor Data	0x06: Heartbeat Message
0x03: Sensor Data	0x07-0xff: User Def. Message

The Heartbeat message is of especial importance, allowing devices to both transmit and receive signal strength data that can be used for control, discussed in section 3.2. It should be noted that for ROMANOs is it not required for most message formats to inform the device about the message length. This is because the behaviour for each message format should be pre-programmed for each device in order to reduce message overhead. The following behaviours for each message type is expected by the receiving device:

- **Normal data.** The receiving device should write the data over RTT or UART in order for a user to interpret the data.
- **Movement control.** The receiving device should update its movement control references.
- **Request Sensor Data.** The receiving device should send out data of the given type to a topic name of the given MAC address.
- **Sensor Data.** The receiving device should store the given data and write it over RTT or UART for user interpretation.
- **Reg. Con. Nodes Info.** The receiving device should broadcast its own MAC address.
- **Con. Nodes Info.** The receiving device should store the given data and write it over RTT or UART for user interpretation.
- **Heartbeat Message.** A heartbeat message to be broadcast to the network by any device over a given time period, in order to get an estimate of any device's signal strength within the network.

To summarize will the proposed communication network be a Thread network with a mesh topology, using ROMANOs and MQTT-SN at the application layer. The communication architecture for the algorithm will give the protocol stack shown in figure 3.1.

There is also a necessity to employ a method of understanding a single device's signal strength within the mesh network. The joint control and communication algorithm does not contain any methods for localization, which makes it difficult to receive a measure of distance between network members. As previously discussed in section 2.1.1 are RSSI values an indication for signal strength. In the case that a device has several RSSI values to choose from, - i.e. the device has connections to several other devices -, then some decision-making should be present. The decisions are non-trivial. One cannot simply take the seemingly *best* value as it can't be stressed enough that RSSI values are by nature an estimation. The node with the largest signal strength might send low-quality data which can be discovered by the CRC algorithm. Additionally could a device have shut down entirely without the receiving device knowing. A device should check the quality of the RSSI indication and how long it has been since said signal was received.

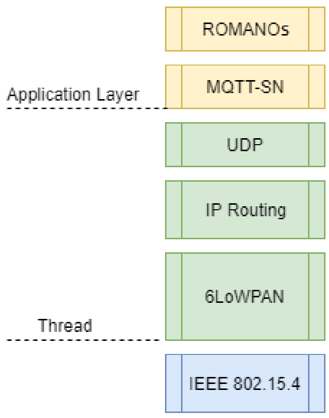


Figure 3.1: Protocol stack for the proposed communication architecture.

The following method is proposed:

- When a data package arrives, check the sender ID, CRC- and RSSI value.
- If the CRC value is approved, update the RSSI value for the sender ID and update its *time to live* (TTL) value with 100. The TTL value indicates how long it has been since the RSSI value for said sender ID has been updated. If the TTL value becomes zero should the recipient disregard the RSSI value and not use it further for any computations. The RSSI value should be updated in a simple moving average computation with length 5 to reduce noise.
- Decrement TTL values for all sending devices. A failed CRC value should decrement the TTL value tenfold. If a TTL value goes to zero, remove the RSSI value from memory.

This kind of *moving average CRC-filtration* should both remove noise from the RSSI values, check for bad data quality and detect if a sending device is no longer participating in communication. Figure 3.2 illustrates normal operations of this filter. For the first scenario will device A get a data package from both device B and C. The package from device C has a CRC value of 0 and will therefore not store the connection links RSSI estimate. In the second scenario receives device A a package only from device C, this time will this signal strength estimate be stored as the CRC value is 1. The TTL value of the previous data package from device B is decremented. Note that the system might use the signal strength estimate from device B as long as the TTL value is non-zero, and its estimate is greater than that of device C.

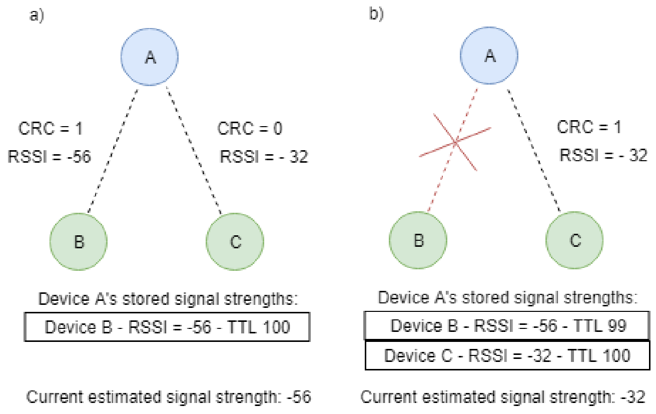

a)
A
CRC = 1
RSSI = -56
CRC = 0
RSSI = - 32
B
C
Device A's stored signal strengths:
Device B - RSSI = -56 - TTL 100
Current estimated signal strength: -56
b)
A
CRC = 1
RSSI = - 32
B
C
Device A's stored signal strengths:
Device B - RSSI = -56 - TTL 99
Device C - RSSI = -32 - TTL 100
Current estimated signal strength: -32

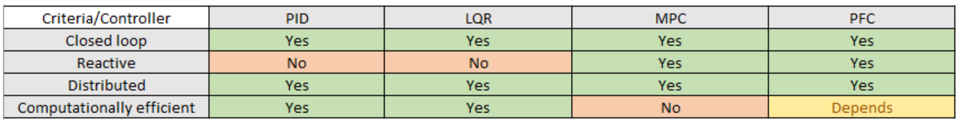
Figure 3.2: Normal operations of the moving average CRC filter.

3.2 Choices for the control strategy

There are a myriad of control strategies to choose from, as shown in section 2.2. Those that are presented in section 2.2 are only a small subset of the available options. The strategies that were presented, PID, LQR, MPC and PFC, are all general control strategies that are applicable to a number of applications; such as the swarm robot application. For the sake of robustness is it practical to choose a control strategy that satisfy certain criteria. As stated by Dunbar and Esposito should a control strategy be [15]:

- ***Closed loop***: *It must continually update its motion plan based on its current position, allowing it to correct for localization errors or noise that may be introduced in the course of the experiment.*
- ***Reactive***: *It must be able to adapt to a changing environment. Each member of the swarm functions as a moving obstacle, leading to a constantly changing environment.*
- ***Distributed***: *Each member of the swarm should be able to synthesize its own motion plan using only locally available information, without relying on a "master" robot.*
- ***Computationally efficient***: *It must be able to recompute its motion plan in realtime, as new information becomes available.*

The comparison of the four control strategies against the aforementioned criteria yields the comparison schema shown in figure 3.3. All control strategies are both closed loop and distributed, but since both PID and LQR are linear time-invariant are they not able to change to a dynamic environment. For a PID controller in a dynamic environment would one have to retune the controller for every change in the environment, using for example tools mentioned in section 2.2. This is cumbersome and non-feasible. For an LQR would one have to change the costs in equation 4 and then recompute the Riccati Equation again (equation 8b), leading the LQR to be computationally non-efficient. Now, the MPC on the other hand, is able to accept time-varying and even non-linear models in its computations. However, the MPC has to solve at best a quadratic problem, and at worst a non-linear problem for each iteration over a given time-horizon. This makes it heavyweight and therefore not suited for the swarm application.



Criteria/Controller	PID	LQR	MPC	PFC
Closed loop	Yes	Yes	Yes	Yes
Reactive	No	No	Yes	Yes
Distributed	Yes	Yes	Yes	Yes
Computationally efficient	Yes	Yes	No	Depends

Figure 3.3: How the different control strategies fare againts different criteria.

The PFC looks promising based on figure 2.3 and 3.3. It is not difficult to implement, is intuitive and can be computationally efficient. The PFC's efficiency depends on the potential function. The potential function should guide a robot from an initial state to a goal state without being exposed to hazards such as collisions or being led on redundant paths. This poses a problem; creating a sure model that will guarantee an optimal or sup-optimal path to the goal state requires some degree of complexity, as the world itself is complex. This complexity however can lead to a bigger chance of modelling errors, a more difficult implementation or a more computationally inefficient control algorithm. As stated previously is the PFC flexible, as there are many methods that can be used for minimizing the potential functions value. One way of alleviating the complexity of the potential function is to employ a method that will not only minimize the PFC's value, but also adding a secondary level of control. This method could also smooth out the control action that may or may not be subject to modelling errors or noise. Rasekhipour *et. al* has done exactly this, they add the principles of the PFC into the MPC [26]. Rasekhipour *et. al* proposes a potential-field based model predictive controller. It works by creating potential fields and using the potential function as the objective function within the MPC. The control action will then be applied after the MPC has minimized the potential fields. The real strength of this algorithm is that it ensures optimality; as the MPC is an optimality based control algorithm. The algorithm itself is shown to have promising results for the sake of robustness [26, p. 1265], as was mentioned in section 1.1 .

Adding a coupled PFC and MPC is not feasible for the swarm application as the micro controller would struggle with efficiency. A lightweight alternative is proposed; a potential-field based PID controller. The idea is that a PFC assigns different potential fields and computes their values based on input data. The PID controller will then compute a control action which tries to minimize the value of all fields.

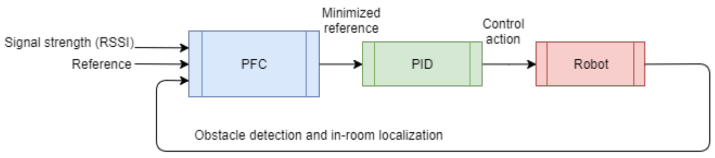


Figure 3.4: Illustration of the proposed control strategy with PFC and PID in a cascade solution.

Before proceeding any further should it be noted that the initial idea was that the PFC was given a reference path. This reference path was altered by the PFC based on the potential fields and a local path-finding search, and then fed into the PID controller. For this case was a potential field based on the euclidean distance from the present device position to a goal position proposed. However, in order for this to be realizable would the system need some sort of localization method. This is not a part of the proposed algorithm, but could potentially be explored further in future research. What is out of interest is to draw potential fields primarily on signal strength, but also for the device's heading and its distance from nearby obstacles. Both the signal strength heading field and the obstacle avoidance field are believed as essential for a robust system. The heading field could be used to address a movement direction for the swarm.
The potential field functions will be as follows:

$$U_{PFC} = U_{Obstacle} + U_{RSSI} + U_{Heading} \tag{19}$$

where $U_{Obstacle}$ is the obstacle avoidance potential field, U_{RSSI} is the potential function given by the signal strength within the swarm, and $U_{Heading}$ is the potential function for the distance between the devices heading to the goal heading in the world frame.

The obstacle avoidance potential field $U_{Obstacle}$ is the distance from the robot and the nearest objective. It is defined by Latombe as [15, p. 16]:

$$U_{Obstacle} = \begin{cases} K_{Obstacle}(\frac{1}{Obstacle(x,y)} - \frac{1}{D_0})^2 & d(x,y) \leq D_0, \\ 0 & d(x,y) > D_0 \end{cases} \tag{20}$$

Where $K_{Obstacle}$ is a positive scaling factor, $Obstacle(x,y)$ is the minimal distance to the nearest obstacle, and D_O is a cutoff distance where the potential field will not be influenced by an obstacle, removing noise. This potential field will be repulsive, as it increases as the distance $d(x,y)$ decreases. It should be noted that the field $U_{Obstacle}$ is the sum of each and every detected obstacle and/or obstacle sensor. This means that unique computations of $U_{Obstacle}$ should be performed for every sensor. A similar potential field for the signal strength potential field is defined, U_{RSSI}:

$$U_{RSSI} = \begin{cases} K1_{RSSI}(\frac{1}{100+R_{Value}} - \frac{1}{100+R_{Minimum}})^2 & R_{Value} \leq R_{Minimum}, \\ 0 & R_{Minimum} < R_{Value} < R_{Maximum} \\ K2_{RSSI}(\frac{1}{R_{Value}} - \frac{1}{R_{Maximum}})^2 & R_{Value} \geq R_{Maximum} \end{cases} \tag{21}$$

Where $K1_{RSSI}$ and $K2_{RSSI}$ are positive scaling factors. R_{Value} is the current RSSI value, while $R_{Minimum}$ and $R_{Maximum}$ are threshold values. The RSSI-value will increase when a robot increases its signal strength to the rest of the swarm. In order to understand the values of the different fields is it reminded that the RSSI values are strictly negative within the area of -20 and -90. Since it is desired that the RSSI values are within a minimal value, will it be defined an attractive potential field towards higher RSSI-values. To avoid clustering will the potential field stop attracting over a certain threshold value, and if the RSSI-value becomes too high will the potential field become repulsive. This will keep each node within a certain area of each other, increasing robustness. The potential function $U_{Heading}$ is simply the arc length from the current heading to a goal heading:

$$U_{Heading} = \begin{cases} K_{Heading}(\theta_{mes} - \theta_{ref})r & \theta_{mes} - \theta_{ref} \leq \pi, \\ -K_{Heading}((\theta_{mes} - \pi) - \theta_{ref})r & \theta_{mes} - \theta_{ref} > \pi \end{cases}$$

Where $K_{Heading}$ is a positive scaling factor, θ_{mes} and $\theta_r ef$ is the current heading and reference heading given in radians respectively, and r is the device radius. Note that the field will reverse its direction if the difference between θ_{mes} and θ_{ref} is greater than π. The potential function will attract the robot towards a goal heading by rotation; rotating the device towards the shortest path. The expected push and pull behaviour of the potential fields during a movement is illustrated in figure 3.5.

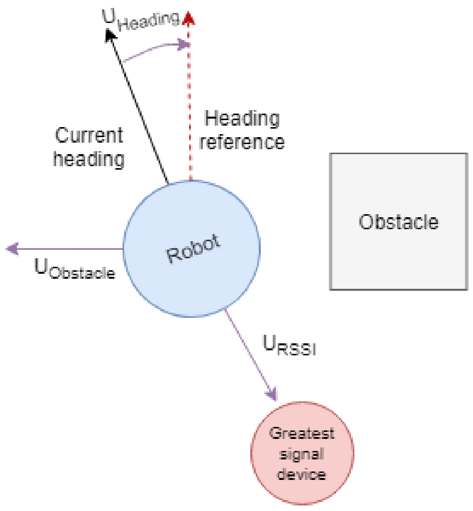


Figure 3.5: The expected behaviour of the potential fields during movement.

Both potential fields $U_{Obstacle}$ and U_{RSSI} are added with robustness in mind. The obstacle avoidance potential field will enable each robot to have a distributed collision avoidance. The signal strength potential field is important since it deals with communication strength and ensures that a robot does not make an illicit movement that makes it lose network coverage. The importance of these potential fields will be further discussed in section 3.3. The PFC and PID controller work in cascade, where the former is the higher-level controller and the latter is the lower-level controller. The potential field function will be controlled through a local search algorithm that feeds into a PID controller, as shown in figure 3.4. An important comment is that the sum of the potential fields shown in equation 19 is a generalization. It is highly dependent of the number of obstacle detection sensors. Additionally will the robots physical configuration determine in what *direction* the potential functions will either push or pull the device. The implementation is discussed in section 4.4.

3.3 Proposed algorithm

In the two former sections are the choices for both communication and control discussed. These aspects will now be combined in one package, the *joint control and communication algorithm for robotic swarms*. The idea is that both communication and control must be designed in tandem in order to achieve a greater order of robustness. The control strategy should maintain a stable communication, and the communication protocol should be both lightweight and responsive in order for the signal strength indication to be directly implemented into the control strategy. The following is a proposal for a joint control and communication algorithm that draws artificial potential fields based on signal strength, this project's swarm algorithm.

Algorithm 1 Control and communication algorithm for robotic swarms

```
while no gateway found do
    search for gateway
end while
Perform ROMANOs Initalization phase.
while no new data from broker do
    MAIN LOOP:
    Poll a heartbeat message and receive a signal strength indication from the network
    Use sensory system for heading data and obstacle detection
    Update artificial potential fields
    Compute control action and actuate motors
end while
ROMANOs Data Type = New data from broker
if ROMANOs Data Type == Normal Data OR ROMANOs Data Type == Sensor Data
OR ROMANOs Data Type == Con. Nodes Info. then
    Print data over RTT or UART.
else if ROMANOs Data Type == Movement Control then
    Update movement control references.
else if ROMANOs Data Type == Request Sensor Data then
    Send out sensor data of requested type.
else if ROMANOs Data Type == Reg. Con. Nodes Info then
    Broadcast device MAC address.
else if ROMANOs Data Type == Heartbeat Message then
    Check CRC register.
    if CRC register approved then
        Update RSSI potential field with the same PAN ID of messenger device.
    else if CRC register denied then
        Review if the RSSI potential field of the messenger device should be removed.
    end if
end if
return to main loop.
```

As mentioned in section 1.1 was it believed that the combination of PFC and RSSI values was an original idea. This was especially believed because until only a couple of years ago would there not exist a microchip solution with enough processing power to single handedly establish a communication network, and then apply a control strategy that took this network into account. However, this idea was already proposed as early as 2011 by Penders *et al.* [5]. But, as also mentioned in section 1.1 is their report not that descriptive on the implementation on both a theoretical and practical level. What can be taken from their study is that this idea shows promise in a dynamic environment for the sake of robustness [5, p. 112].

The control and communication algorithm lays a framework that swarm applications can be built upon. It establishes a mesh network that uses Thread as the networking technology, and ROMANOs for definition of the data payload. Thread is of interest because of its self-healing properties and 6LoWPAN. This results in an adaptive network with support for internet connectivity with little overhead. ROMANOs is especially constructed for the deliverance of relevant signal strength data packets, movement control and local communication in-between members of the network with as little overhead as possible. The mesh topology will make it possible for every member of the robotic swarm to communicate with all members of the swarm, guaranteeing that the best signal strength value will be taken into account. The RSSI value will then be used directly by each robots distributed control strategy, affecting the robots artificial potential field. The robot also has potential fields drawing it towards a goal heading, and a potential field for obstacle avoidance. The controller will then compute potential fields, and their values are then sent to a PID controller in order to compute a minimizing control action. This essentially bridges communication and control together in one package. A flow chart of the algorithm can be found in figure 3.6. It should be noted the implementation of CRC, discussed in section 2.1.1. Essentially will the algorithm check the *validity* of drawing a potential field based on the current RSSI data for the specific device being communicated with. If the CRC register shows that the data is corrupted is it not out of interest to update this potential field, and if the CRC register has shown a corruption multiple times in succession should the potential field be deleted entirely. Signal strength data from devices who has been unresponsive for a while should also be disregarded.

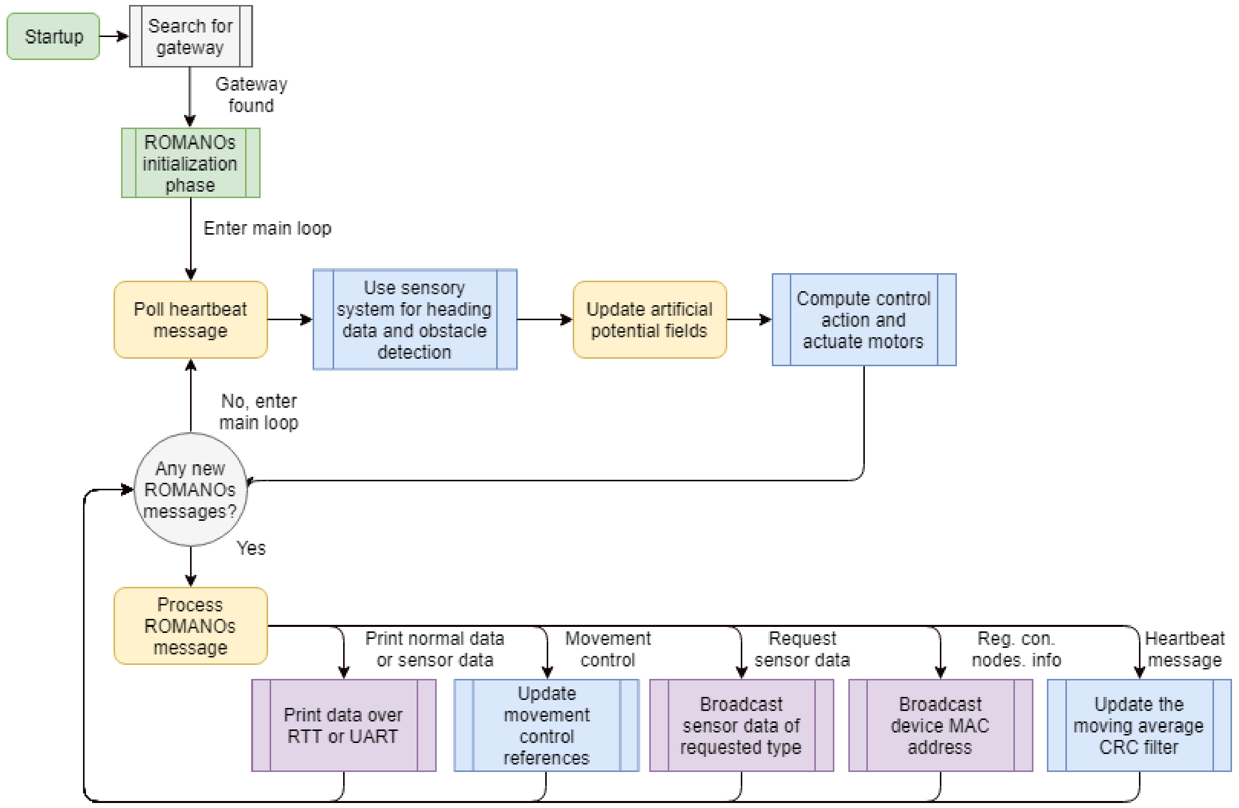

Startup
Search for gateway
Gateway found
ROMANOs initialization phase
Enter main loop
Poll heartbeat message
Use sensory system for heading data and obstacle detection
Update artificial potential fields
Compute control action and actuate motors
No, enter main loop
Any new ROMANOs messages?
Yes
Process ROMANOs message
Print normal data or sensor data
Movement control
Request sensor data
Reg. con. nodes. info
Heartbeat message
Print data over RTT or UART
Update movement control references
Broadcast sensor data of requested type
Broadcast device MAC address
Update the moving average CRC filter

Figure 3.6: Flowchart of the joint control and communication algorithm.

4 System architecture

A point of interest is to create a physical platform, a robot, that could run the algorithm presented in section 3.3. The robot will be tailored for the algorithm at hand, with enough flexibility to run alterations of the algorithmic design if needed. The main idea of the robot is to be able to test whether or not the control and communication algorithm contributes to robustness. First, in section 4.1 will the the backbone of the hardware platform be introduced, consisting two printed circuit boards (PCB). Further will the chassis design be shown in section 4.2, which houses all the components for the robot. In section 4.3 will the firmware be discussed, which is the manifestation of the algorithm presented in section 3.3, and the necessary code around it in order make it run. Lastly will the complete robot architecture and swarm hierarchy be presented in section 4.4.

4.1 Design of hardware platform

For this project are there created two PCB's that together create the entire hardware platform for the robotic devices in the swarm. One main central board houses all components and functionalities, except from obstacle detection which is covered by the secondary board. The main board is the most important component of the robot, as it contains the nRF52840 System-on-Chip as the processing unit [36], which is a requirement for this project. The main design philosophy for both boards is that they house all necessary electronics in order to run the control- and communication algorithm discussed in section 3.3. A full list of their respective specifications are given in section 4.1.2 and 4.1.3. Key decisions and design choices are presented in section 4.1.4.

4.1.1 The nRF52840 System on Chip

The nRF52-series by Nordic Semiconductor is a family of system micro controllers that focuses on ultra low power wireless communication. It contains all necessary components for a lightweight computing device in one single chip, making it a complete *System-on-a-Chip* (SoC). The integrated circuitry is created around either a 32-bit ARM Cortex M4 or a ARM Cortex M4F micro processor, dependent of the model in the series. The nRF52840 will be used in this porject, as it is the more *beefy* member of the series, boasting both higher memory and a higher number of peripherals.

Its specifications that are relevant for this project are the following:

Processor	32-bit ARM Cortex M4F Processor
Voltage level	1.7v to 5.5v operation
Memory	1MB flash + 256kB RAM
Radio	Bluetooth 5 and 802.15.4 radio support
GPIO	48
SPI	4 x Master/Slave SPI
I^2C	2 x Two-wire interface
PWM	4 x PWM modules
ADC	One 12-bit ADC
NFC	On-chip NFC present
PPI	External PPI-module present

It should be noted that as of writing this report is the nRF52840 under development. This means that some anomalies might occur. The version used in this project is the nRF52840 IC. Rev engineering B, and the following anomalies are of interest:

- The RSSI values accuracy is dependent on the temperature of the SoC, and might give a false indication with ±3 of the actual RSSI level.
- Higher current consumption than desired during normal operations.

4.1.2 Specifications for the robot PCB

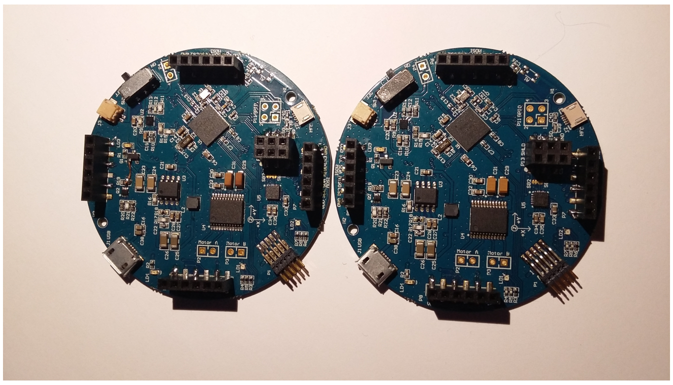


Figure 4.1: Two fabricated and soldered main boards.

All specifications for the main board comes naturally from the design of the swarm algorithm presented in section 3.3, along with the requirement to use the nRF52 System-on-chip (SoC). As the algorithm is still in development is a main design philosophy of the main board to give it as much lee-way as possible without going overboard. This is why it is decided to use the most powerful SoC in the 52-series, the nRF52840, but also why the platform is given some additional functionalities, mostly based on intuition.

The main functions of the main board are as follows:

- Contains an nRF52840 System-on-Chip.
- Circular form factor.
- Interface to connect expansion cards.
- Interface to connect sensory units.
- Power system that regulates an input voltage from 4.5-14.8v to 3.3v.
- Battery plug.
- Motion sensor with gyroscope, accelerometer and magnetometer.
- Programming- and debug interface.
- Antenna for wireless communication. (BLE and IEEE 802.15.4 or proprietary)
- Motor control system for two DC motors that can deliver currents up to 1A.
- Protective circuitry for reversed voltages.

The main board has the following extra functions:

- Integrated circuitry for battery charging and voltage measurement.
- Micro-USB plug.
- Contact for near-field communication, NFC.
- Two programmable RGB LED.
- Green indication LED on active voltage.
- Transistor circuitry with red LED indication if PCB is applied reverse voltage.

4.1.3 Specifications for the obstacle detection PCB

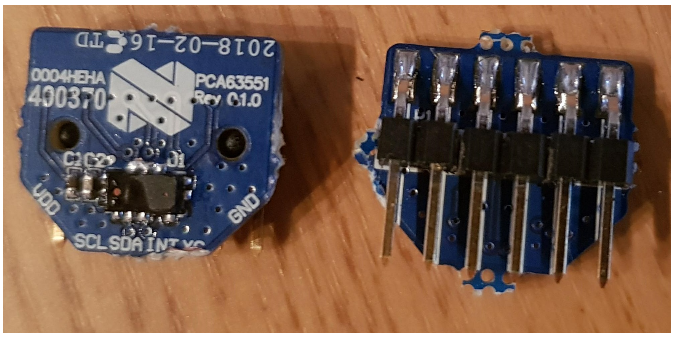


Figure 4.2: Two fabricated and soldered secondary boards. Front side shown to the left, back side shown to the right.

The secondary board's only purpose is to employ some sort of obstacle detection method, and the methods to choose from are presented in section 2.3.1. As the test platform is used to check for robustness should not the obstacle detection method itself contribute to less robustness. What type of method used for obstacle detection is not defined by the algorithm, only the fact that some sort of obstacle detection should be present. The strongest arguments for an obstacle detection method is that it should both be *scalable* and *precise*. Scalable in the sense that it should be able to work well in a noisy medium introduced by multiple robots with multiple sensors, and precise in the sense of giving range measurements with a high resolution as possible. Arguably would LS sensors work best for this case. US sensors are precise but has a large sensitivity cone, and several US sensors might interfere with each other. IR sensors on the other hand struggles with precision from their non-linear behaviour from different surfaces. The amplified LS sensors mitigates the precision problems of IR, and as it has a smaller sensitivity cone makes it more suited than US sensors for a noisy medium. This method of obstacle detection could be seen as a method for in-room localization.

The specifications for the secondary board are the following:

- Integrated circuitry for laser sensing.
- Interface to connect with the main board.

4.1.4 Layered structure and key decisions

The main and secondary board are developed in Altium Designer 17.0, a tool for creating embedded systems [27]. Altium Designer works in two parts, first you create the schematic drawings, and then you draw the circuit board itself. The secondary board is a common two-layered PCB where the top layer is used for signal routing and the bottom layer is used for grounding. The main board is more complex and consists of four layers. This reduces surface area and is needed to utilize all general purpose input and outputs (GPIO) from the nRF52840 SoC. The schematics of the main board is shown in *Appendix 1 - Main board schematics*. The schematics of the secondary board does not require any illustrations as it is simply an break out board for the laser sensor. The four layers for the main board are as follows:

Top Layer Signal layer.

Mid Layer 1 Ground layer.

Mid Layer 2 Power layer.

Bottom Layer Signal layer.

For the main board was the top layer and the bottom layer grounded to mid layer 1. For both boards were all components installed on the top layer, and board routing was done in-between each and every layer were it was needed. There were several key practices that were used during board routing. First of all should the length between routing points be as small as possible in order to reduce impedance and possible noise. Further should the number of vias be reduced in-between the layers, as it would be lead to inefficient and sloppy routing. Lastly was it decided to keep signal routing on the top- and bottom layers as much as possible, and power routing on mid layer 2. This is to ensure a systematic routing scheme. Board placement on the main board was highly influenced by the placement of the nRF52840 SoC. The nRF52840 was placed on the outskirts of the board whilst still making headroom for the interface for the sensory units. It was then rotated 45 degrees in order to make the path to the antenna as small as possible, see figure 4.3.

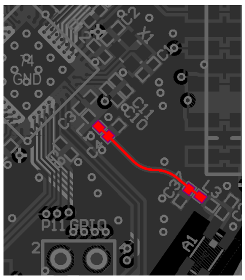


Figure 4.3: Illustration of the radio line on the PCB, marked in red.

Figure 4.4 shows an abstraction of the placement of the different integrated circuits on the main board. Note that the power system was put on the "north" side of the system chip. This was because the nRF52840 has the fewest GPIO's on this side, and the power system requires almost none. Additionally, circuitry that require some GPIO's has been placed on the "south" side of the system chip; whilst the motor driver that requires both power and GPIO's is placed in-between. Both boards contain an interface in order for the boards to connect to each other. This interface is designed to be highly flexible and could conform to both TWI, SPI or other communication protocols and also supply the secondary board with power.

Generally is the interface designed as follows:

VDD	SCL	SDA/MOSI	MISO	CS	GND

However, all signal lines are essentially general purpose and can be tailored to better suit the application. For the obstacle detection board was the user interface used as follows:

VDD	SCL	SDA	INT	XS	GND

Two important integrated circuits that the nRF52840 SoC communicates with are the MPU-9250 and the VL53L0X. They are both small sensory units that are communicated with over the TWI bus. The MPU-9250 is a nine-axis gyro, accelerometer and compass, making it a MEMS motion tracking device. The MPU-9250 is a combination of two other devices; the MPU-6500 gyro and accelerometer and the AK8963 compass. It is low power, and the compass' full scale measurement range of $\pm 4800 \mu T$ shows promise for the sake of robustness. The obstacle detection board is equipped with a VL53L0X ToF sensor [54]. This is a small integrated circuit that contains a LS ranging sensor with a precision up to $1mm$ over $2m$. The laser is working in the $940nm$ range, making it a laser in the IR band. However, the IC itself is equipped with internal physical infrared filters [54, p. 1] giving it a higher resistance to ambient light and better robustness against glass surfaces. It is also shown to work against dark surfaces [54, p. 26], making it well suited for a changing environment.

The printed circuit boards themselves are fabricated at an external company, but is soldered by hand using a stencil, a microscope and an infrared heating chamber.

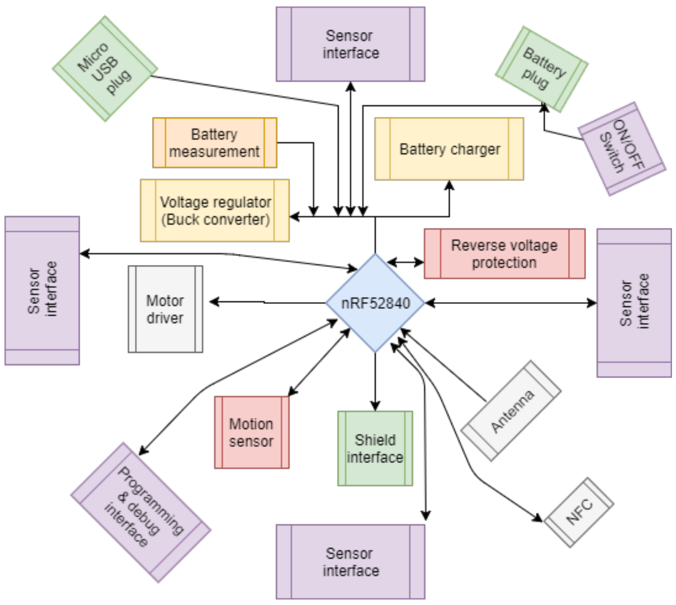


Figure 4.4: Abstraction of the PCB, showing the placement of important parts.

4.2 Robot chassis and 3D-model

In order to complete the physical platform was it necessary to create a chassis that could hold all components in a practical manner. This chassis should be able to mount both PCB's and two DC-motors. All interfaces on the PCB's should be accessible, and the motors should be oriented in such a way that it would be possible for the robot to rotate in all directions. With the tools available was it natural to make a model of the chassis and later print the model using a 3D-printer. The model itself was designed in Onshape, and then printed with a Mojo 3D-printer [29, 60]. The model was tried and tested multiple times, and the fourth and final iteration will be shown here. The STL file for the 3D model will be added in *Appendix 3 - Robot chassis 3D model*. Illustrations of the chassis model is shown in figure 4.5 and 4.6 with explanations of its design functionalities.

The model has a height of $3.314cm$ and a diameter of $8.0cm$. As can be seen in figure 4.5 is the main board mounted on top of the chassis, with spacings for the different connectors and interfaces on the board. One thing that should be noted from figure 4.6 is the alignment of the motors; they are both facing inwards. As the center of force comes close to the center of mass will the robot be able to turn quickly and effortlessly. Additionally will this reduce the chassis width. The chassis itself is made to give a $0.5mm$ clearing between the chassis itself and the flooring for the wheels. This will make the chassis skid along the floor. The motors are fitted through a friction fit and does not require any tools to mount. The fitting is created with micro metal motors in mind.

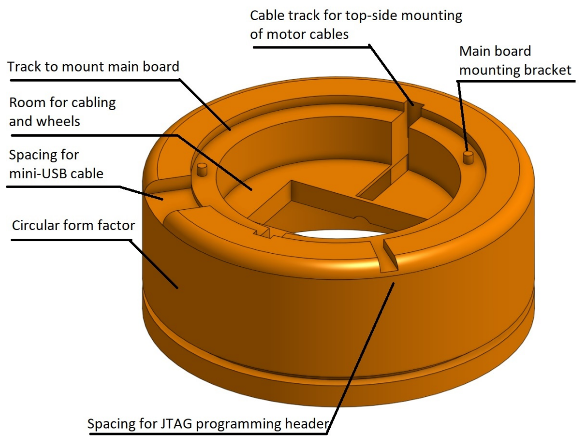

Cable track for top-side mounting of motor cables
Main board mounting bracket
Track to mount main board
Room for cabling and wheels
Spacing for mini-USB cable
Circular form factor
Spacing for JTAG programming header

Figure 4.5: Illustration of the chassis model seen from the top.

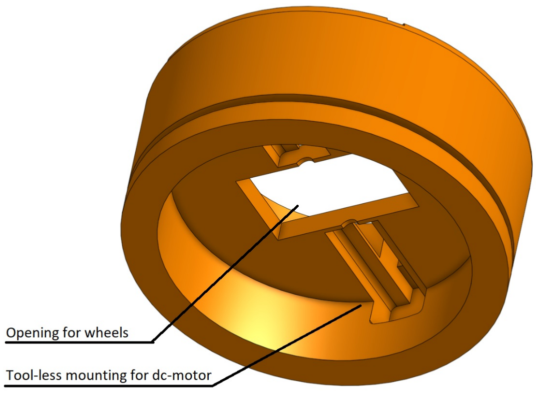


Figure 4.6: Illustration of the chassis model seen from the bottom.

4.3 Firmware

The firmware is is the programming that runs on the nRF52840, and realizes the algorithm discussed in 3.3. It is essentially the bridge between the hardware platform and the swarm algorithm. Its purpose is to *initiate* the algorithm and *execute* its functionalities. On a technical level does this mean that the nRF52840 must establish control with all necessary peripheral units, initiate digital communication, perform necessary computations and perform outputs based on input data. All of these actions should conform to the specifications of the control and communication algorithm, and also be applicable to the physical platform.

Because of the time scope of this project was it decided to use the Software Development Kit (SDK) developed by Nordic Semiconductor, more specifically version 11 of the Thread SDK [32]. The SDK is a library that could be used as an abstraction layer for embedded programming. Instead of writing to registers in memory directly, one could instead call upon functions and edit structures in order to realize different functionalities. The drawbacks of using the SDK is that one has less control over what the programming does at the lower levels, which results in suboptimal memory usage. The SDK should not be a problem for the sake of robustness, as it has been subject to rigorous testing and development. The programming works as a layer on top of the SDK, and as its size is naturally very large is it difficult and counter-intuitive to present it as a whole. Therefore will it be introduced different ideas and design philosophies used to develop the firmware, and most importantly lay an overview, to better suit the scope for this report.

The firmware can be divided into two main phases; the *initialization* phase and the *main cycle* phase. During the initialization phase will the nRF52840 start up all protocols and peripherals which is needed by the system, and a complete breakdown is given in figure 4.7. The main cycle phase is essentially the implementation of the joint control- and communication algorithm discussed in section 3.3, and the firmware is able to enter a *interrupt* sub phase which deals with interrupts from data sent by the rest of the swarm. The interrupt phase deals with scenarios as depicted in ROMANOs, presented in section 3.2. An illustration of the main phase is shown in figure 4.8.

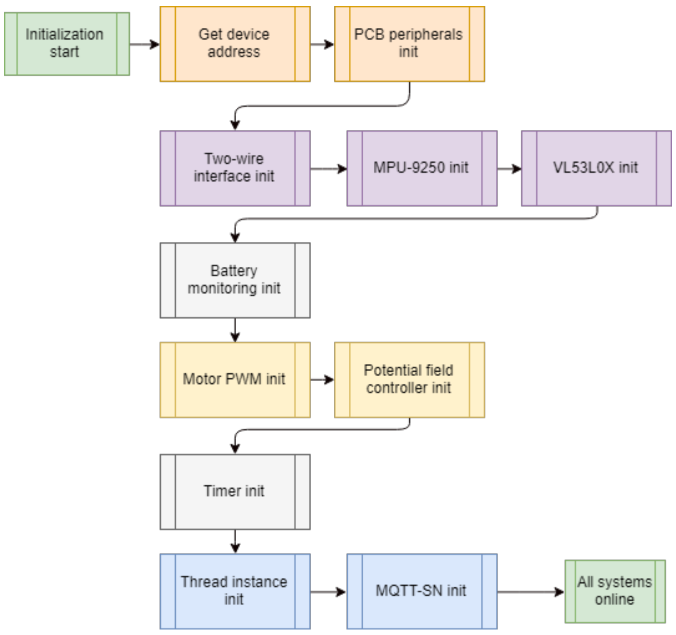


Figure 4.7: Flow chart for the initialization phase of the firmware.

A short description of the main point and usage of each part of the firmware will be presented here. As the firmware is complex is it not suitable to discuss every detail in this report. It is heavily encouraged to check out the complete firmware in *Appendix 3 - Firmware code* to get the full picture. The initialization phase, as depicted in figure 4.7, performs the following tasks:

- **Get device address**. Initially on start up will the device get its unique MAC-address. This will be used when connecting to a MQTT-SN gateway along with distinguishing sensor data used for analysis. This step is crucial as it removes the need to hard code an ID for each device, which would be cumbersome.
- **PCB peripheral init**. The nRF52840 then allocates all necessary pins as either inputs or outputs used for communication with external devices. It will then wait for 100 ms in order to ensure that all external IC's has started up.
- **Two-wire interface init.** The two wire interface (TWI) is used to perform communication with the secondary board's VL53L0X ToF sensor and with the on-board MPU9250 MEMS device. The hardware layer of the TWI is initialized, which is an external part of the nRF52840 CPU. This means that the CPU will write data and read data to a buffer when desired, the communication itself will not require any CPU resources.
- **MPU-9250 init**. Communication with the MPU-9250 is established. The nRF52840 will try to read the *Who am I?* register of the MPU-9250, as the unit might not be ready for communication yet. If no answer is given will the nRF52840 continuously poll the *Who am I?* register up to 100 times, and if no response is ever present will the initialization process be cancelled. The initialization process itself is interested in configuring the magnetometer only and getting it ready for communication. The MPU-9250's magnetometer is an external device existing within the same package; the AK8963. The MPU-9250 must therefore first be configured into *bypass mode*, a mode where all TWI-communication is relayed through the MPU-9250 to the AK8963. The magnetometer is then configured with data such as data resolution and update frequency. This configuration is based on sparkfun's API for the MPU-9250.
- **VL53L0X init.** Every VL53L0X will on start up default to a hard coded TWI address. As all laser sensors on the board share a TWI bus is it required to change the TWI address of three sensors before it is possible to communicate with each and every one. This is done by putting all *Xshut* lines low, which will turn off the sensor. One by one will a sensor then be turned on by raising the level of *Xshut*, and then writing a new, temporary, TWI address to this device. Each sensor is then in turn configured according to the API by STMicroelectronics, which consists of status checks and calibration methods. The sensors are set in single ranging mode, which means that the sensors will send a discrete laser signal rather than a continuous one for ranging. This mitigates noise.

- **Battery monitoring init.** Battery monitoring is done external from the CPU. This is done by configuring a Programmable Peripheral Interconnect (PPI) module, a module that performs tasks on the basis of a trigger. This trigger event is performed by a timer; over a set time will the timer which is linked to the PPI module send an interrupt signal to the module, which then reads from the ADC module and stores it in memory. The nRF52840 sets up the PPI module, the ADC module and the timer, and also allocates the memory which will store the ADC data. If the voltage level is discovered to be too low will the unit shut off in order to protect the longevity of the battery.

- **Motor PWM init.** The motors will be controlled with a PWM signal, and the module is set up by the nRF52840 with its appropriate pins and switching frequency. The PWM module is configured in *individual mode*, which means that the PWM signal for both motors are independent of each other. The PWM switching itself is done by a peripheral unit in the nRF52840, the CPU will only need to update the duty cycle and/or the switching frequency.

- **Potential field controller init.** The potential field controller is an abstraction layer used by the nRF52840 for computing the control signal to be sent to each motor. During initiation will the controllers be stored in memory with its appropriate controller values and set points.

- **Timer init.** The main timer is initiated. This timer will perform an interrupt over a set time interval, setting a flag to true that the main algorithm should be performed. This makes the main algorithm run at a set frequency rather when the CPU is available, resulting in a more fluid control procedure rather than a sporadic one.

- **Thread instance init.** The Thread module will start, which contains all software functions for Thread communication and interfacing with the antenna. This, along with the code for MQTT-SN, is directly taken from the SDK and is not created by me. However, alterations have been done such that the CRC register will be checked for each data package, along with an interrupt informing the unit of the current data packages RSSI value.

- **MQTT-SN init.** The application layer MQTT-SN is put on top of the Thread instance. This part of the software is taken from the SDK. This concludes the initialization phase, and all necessary systems needed for the algorithm to run are online.

When the initialization phase is performed will the device be ready to react to the world around it. During initialization will the device inform a user of the procedure in real-time over RTT if this is so desired. The device will immediately start to search for a nearby gateway and continue this search indefinitely. If a gateway is found will the ROMANOs initialization phase begin. The main algorithm will trigger by each interrupt by the main timer. The main algorithm phase is as follows:

- **Obtain magnetometer data.** The nRF52840 uses TWI to communicate with the AK8963 through the MPU-9250 in bypass mode. It will then use the magnetometer data to compute a device heading in the NED-frame.
- **Obtain range data.** The nRF52840 uses TWI to communicate with each and every VL53L0X TOF sensor to get ranging data. All computations and filtration is done by the ranging sensors themselves, so the nRF52840 will only obtain the data and parse it to the potential field controller.
- **Check real time clock.** The nRF52840 checks a timer running at 64 MHz for how long time has passed since the last iteration of the algorithm. This timing is done by PPI and will therefore not tax the CPU in idle performance. Although the main algorithm is initiated at a set frequency, - thanks to the main timer -, is it no guarantee that each iteration of the algorithm has the same exact time difference. Interrupts from data communication and processing might occur resulting in slower execution times. It is therefore crucial to check with a timer exactly how long time has passed (in microseconds) to get a more accurate control action.
- **Update PFC controller**. The PFC controller will be fed with the input data of current heading, obstacle ranging and signal strength. The signal strength data is acquired during the interrupt phase. It will then perform potential field computations which is then applied to the PID controllers which tries to minimize its values. The control action from each PID controller is summed and formatted to suit the motor domain by interpreting the data as a duty cycle and rotation direction.

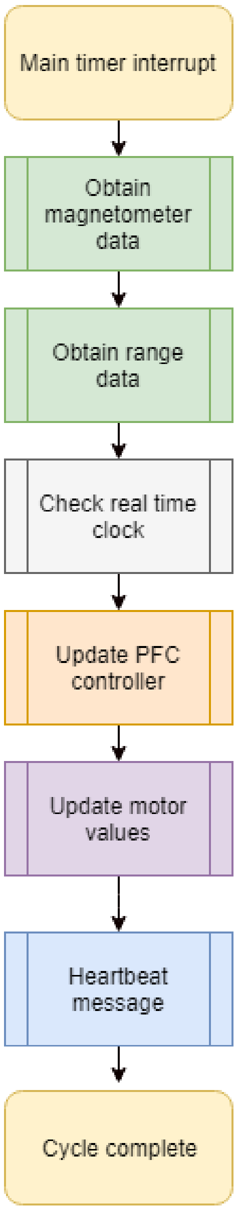


Figure 4.8: Flow chart for the main phase of the firmware.

- **Update motor values**. The computed motor values from the controller layer is then updated to the motor driver. This driver will update the duty cycle of the PWM signal and raise/lower directional indication pins.
- **Heartbeat message**. The device will send out a heartbeat message to the rest of the network that can be used by the other devices to arrive at the interrupt phase.

After the main algorithm phase has the algorithm performed one iteration, and will wait for the next cycle indicated by the main timer to perform another iteration. The Thread communication is maintained outside of the main algorithm phase. The interrupt phase is very important since the nRF52840 can only arrive at this state during Thread communication, but is essential for the control architecture as it updates the estimates for signal strength.

The interrupt phase performs the following actions:

- **CRC filter and RSSI update.** When a data package is arrived will the interrupt phase begin. The CRC value of said package will first be checked; if it is 0 will the package be disregarded, if the value is 1 will the contents of the package be examined. The nRF52840 will store the current RSSI value of each and every unique Personal Area Network (PAN) ID, and use the best unique RSSI value for control computations. If a PAN device has not sent a data package after a certain number of data transmission (i.e. 100), will said device be removed from the equation and not be used for control computations. This penalty will also account for messages with a bad CRC value, meaning it will be seen as no package has been sent at all. This is important as one cannot, in any circumstance, use corrupted data for control.
- **ROMANOs data received procedure.** The data will be processed in the ROMANO abstraction layer after an approved CRC check. Every action to be done based on the payload is described in section 3.1.

As the RSSI value is important for control will part of the interrupt phase have priority over the main algorithm phase. This means that the RSSI value can even be updated during the main algorithm phase; but ROMANOs procedures will not have priority. It is especially important to have a new signal strength value for control, but computing the control action itself is seen as more important than communicating with the network. ROMANOs procedures will take place outside of the main algorithm phase.

The firmware itself has been programmed to be as modular as possible. This means that every part in each phase can be removed without crashing the system. Each part of the programming is entirely independent of each other and could simply be changed or removed as the control- and communication algorithm develops. Every sensory information however is stored into one single, main structure. Data such as the device MAC address, voltage level and control inputs are stored here. This data can later be sent over the network for analysis or be printed over RTT.

During development and analysis was real-time indications a useful tool for error detection and performance testing. This was done by defining a LED indication scheme:

LED Color	**Indication**
White	Firmware booting
Green	Firmware ready
Flashing green	Searching for gateway
Flashing blue	Gateway found
Blue	Connected to swarm network

4.4 Robot architecture and swarm hierarchy

By combining the swarm algorithm presented in section 3.3, the hardware platform in section 4.1, the chassis in section 4.2 and the firmware discussed in section 4.3, one arrives at the swarm architecture. This architecture defines the design, construction and behaviour of one stand-alone member of the swarm. It also characterizes the behaviour of each swarm member and how they should interact with each other. The complete physical package is shown in figure 4.9.

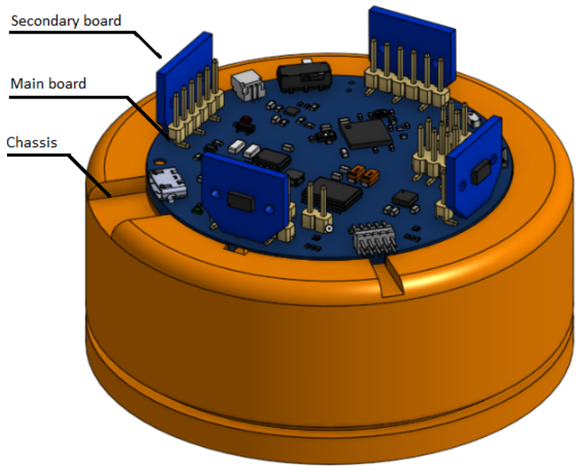


Figure 4.9: Complete 3D model of the robotic device.

It was mentioned in section 3.2 that the direction of the potential fields push- and pull-direction was highly dependent of the *configuration* of the robot. There is no localization method present in our architecture. This means that not only should the scalar value of a sensory input be taken into account, but the orientation of the sensor itself, too. A sensors orientation would make it possible to derive a vector out of each potential field. Figure 4.10 illustrates this case, where the obstacle detection fields at the front and back either pulls the robotic device forwards, or pushes it backwards respectively. Similarly, the obstacle detection fields at the sides will actuate a *rotation* either towards or against the clock. This rotational field will be similar for the magnetometer. Lastly, since the position of a sending device is not known to the recipient will it not make any real sense to make a rotation based on signal strength alone. However, the signal strength potential field could push or pull the recipient device forwards and backwards based on the RSSI value, as shown in equation 21. It is therefore always assumed that the current signalling device is behind the recipient device on the same axis at all times. The motor actuation then becomes the following:

$$Motor_L = Obst_{back} + Obst_{left} - Obst_{right} - Obst_{front} + Signal_{strength} \tag{22a}$$

$$Motor_R = Obst_{back} - Obst_{left} + Obst_{right} - Obst_{front} + Signal_{strength} \tag{22b}$$

Where $Motor_L$ and $Motor_R$ arre the motor actuation for the left and right motor respectively. $Obst_{back}$, $Obst_{left}$, $Obst_{right}$ and $Obst_{front}$ are the control actions for its respective obstacle avoidance field. Similarly is $Signal_strength$ the control action from the signal strength field. Figure 4.10 illustrates the different vectors pushing and pulling a device in different directions.

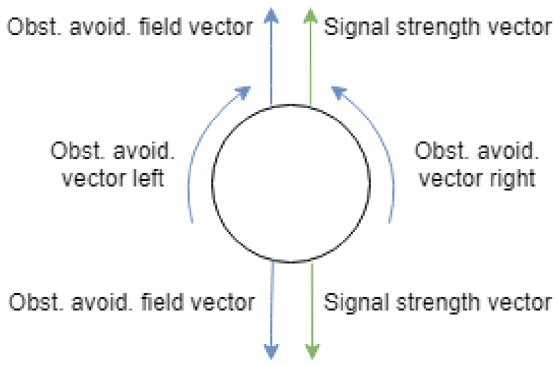

Obst. avoid. field vector
Signal strength vector
Obst. avoid. vector left
Obst. avoid. vector right
Obst. avoid. field vector
Signal strength vector

Figure 4.10: The different vectors pushing and pulling a device in different directions, introduced by the potential fields.

The swarm will naturally be established by connecting a multiple of these devices together. From the users perspective is the initialization of the swarm seamless, as the device itself will search for a network and establish connection upon detection. As the choice for the communication protocol discussed in section 3.2 is to use MQTT-SN, it is needed to introduce a broker to the network. A broker requires more processing power to be maintained than is available in the nRF52-series or similar microchips. Usually is a broker maintained over *at least* a single threaded operating system [44, 61]. An external broker of some sorts would therefore be needed to be introduced to the network. Creating a broker from scratch is not feasible within the time-scope of the project. A broker/gateway solution is sophisticated and could be seen as a project in and of itself. The simplest solution would be to connect the swarm to an external broker over the internet. Nordic Semiconductor already provides the necessary software to set up an MQTT/MQTT-SN gateway

[62], which makes the swarm compatible with all traditional brokers. The Nordic Semiconductor Thread Border Router works as a border router for the mesh network and as a gateway to interface with a broker. It uses a nRF52840 development kit that communicates with a Raspberry Pi Model 3B over UART. The nRF52840 development kit is only used for its 802.15.4 antenna to interface with the thread network. The Raspberry Pi, on the other hand, performs all gateway tasks discussed in section 2.1.3.1. The broker used in this project is the open source eclipse broker [63]. The swarm hierarchy will no longer be flat by opting for an external broker, as shown in figure 4.11. Yet, one could introduce a robot architecture that used a stronger processing unit than of the nRF52-series to run the joint control- and communication algorithm, and work as a MQTT/MQTT-SN broker/gateway simultaneously. The drawbacks of this architecture is that it would be less cost-efficient and have a larger current draw. The added complexity could also prove difficult to design, construct and program within this projects time-scope. For this swarm application will an external broker be sufficient for initial testing, and no further elaboration will be done on an alternative architecture.

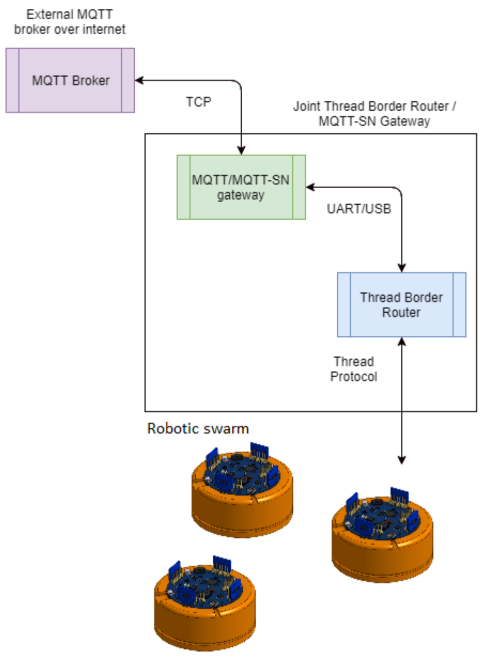


Figure 4.11: Complete swarm hierarchy with the different communication protocols.

Figure 4.12 shows the real physical construction of six swarm robots. The physical Thread Border Router / MQTT-SN gateway solution is shown in figure 4.13, consisting of a nRF52840 development kit and a Raspberry Pi Model 3.

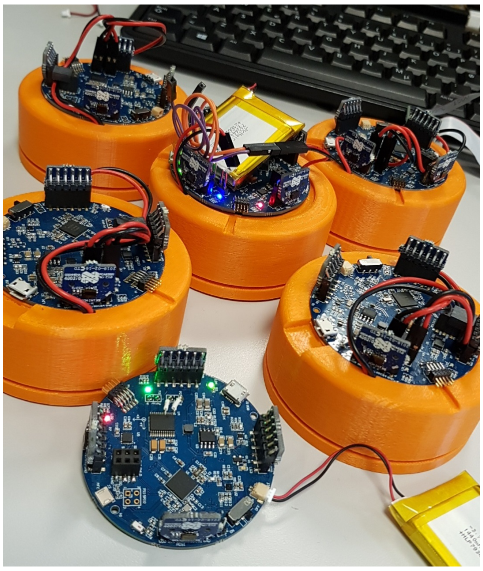


Figure 4.12: Six real-world swarm robot devices.

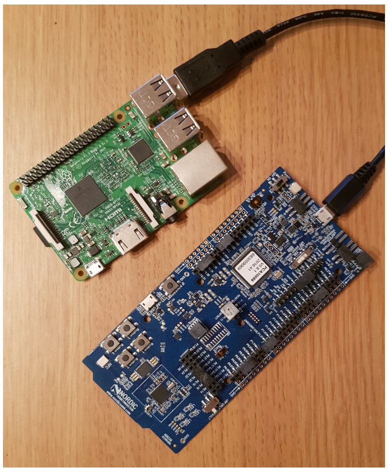


Figure 4.13: The Thread Border Router / MQTT-SN gateway solution.

5 Test platform

In this section will the methods for acquiring analytical data from the algorithm and system architecture be discussed. The swarm algorithm and the swarm architecture are discussed in section 3 and 4 respectively. The main philosophy is to establish a platform for testing the system that produces results for individual part of the swarm system. These results are vital to to get a measure of the systems robustness for further analysis. First, in section 5.1 will the methods for acquiring analytical data be presented. Secondly will there be a discussion of what kinds of tests are of importance in this project, how they will be performed, and what kind of data is to expected.

5.1 Data acquisition

Useful data acquisition in this project would be the method of acquiring sensory inputs, computed outputs and stored data from each and every member of the swarm. This would paint an image of how the swarm acts on a group level and on an individual level. The most important factor when acquiring analytical data was the fact that the acquisition itself would not tamper with the systems performance in any way. A natural decision was therefore to rely on existing infrastructure used by the swarm system, without changing any major parts of the swarm.

The robotic swarm itself is connected to the internet through the Nordic Thread Border Router. This makes it possible for a listening device, - like a computer -, to connect to the same broker as the Border Router, and subscribe to an MQTT topic used by the swarm. Every member of the swarm will then publish relevant data to said topic which is then picked up by the listening device. The listening device will then store all data it picks up to a database for further analysis. This is all possible trough the use of Node.js [33]. Node.js is a JavaScript runtime that through its MQTT library makes it possible to connect to any broker and any topic within said broker. The Node.js module is then programmed to save any data it has acquired to an SQL database [34]. This data could later be analysed in MATLAB as it supports SQL database connections. The communication topology with the listening device present in the network is illustrated in figure 5.1.

All members of the swarm are programmed to broadcast all data of interest by every other heartbeat message. This is to ensure that at least half of the heartbeat messages have as little payload as possible, resulting in the data broadcast to have a minimal interference with the control strategy. For ease of use from the analytical side will all data messages be sent in the JavaScript Object Notation (JSON) format [64]. By adhering to this messaging format will it be easier to parse all data for further analysis as it is naturally compatible with Node.js.

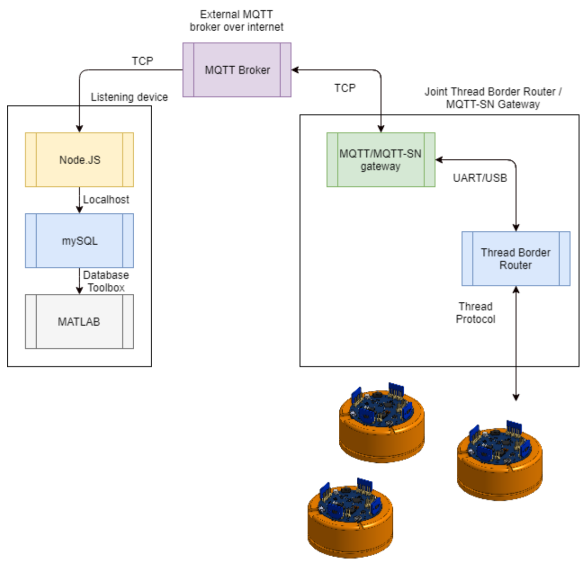


Figure 5.1: Complete swarm hierarchy with the different communication protocols and the listening device present.

The data message will contain the following data:

Stored data from device:

Device MAC Address (octet 6)	CRC (1)

Sensory input data from device:

Signal strength (RSSI) (octet 1)	Obstacle dist. (LS 1-4) (8)	Mag Out.(Axis XYZ) (8)

Computed outputs:

Heading (octet 2)	PID Out. RSSI (2)	PID Out Obst. Avoid. (1-4) (8)	PID Out Heading (2)

All of this data is sent in one single data package. The SQL data table will adhere to this message format and define columns for all of the different data types. Additionally will the SQL table consist of a column for the test ID and a timestamp for easier processing. This infrastructure makes it possible for further analysis in MATLAB, which is used in section 6.

5.2 Scenarios and expected behaviour

The complete swarm system presented in section 3 and 4 contains several layers, from both hardware, firmware, and the algorithm structure. The layers introduces techniques and methods that are believed to have not been combined before. This results in an interesting case where it is desired to create a mapping of the systems performance, from each individual layer to the complete system. It may be possible that some parts of the system are performing better than others, and some layers could either be revised or excluded entirely. The most desirable trait for the system is robustness. This means that each sub part of the system should contribute to the whole system, making it **scalable**, **efficient**, **adept** and **redundant**. These are all broad terms, and their meaning would differ from each part of the system. For the sake of *scalability* would it mean that the size of the swarm would not infer on performance. *Efficiency* would mean that the system has an optimal performance, with low-cost, fast execution tasks. An *adept* system would perform as well no matter the environment, and a *redundant* system would be able to still work although some failure has occurred.

The most natural way to get an understanding of the swarm system would be to perform several different tests, and run an analysis based on the results. Different tests and scenarios that are out of interest are as follows:

- **Test A - Calibration.**
  - Calibrate the magnetometer for further control strategy testing.
  - Define maximum and minimum values for RSSI for further control strategy testing.
  - Tune all controllers for further control strategy testing.
- **Test B - System performance.**
  - Check memory usage of the firmware. Use a logic analyser to monitor the CPU usage of the nRF52840.
  - Illustrate the antenna gain topology, search for dead spots.
  - Check the quality of all sensory inputs from the magnetometer, antenna and obstacle ranging.
- **Test C - Communication protocol.**
  - Network topology illustration.
    - Map the network topology with only a few devices connected close together.
    - Map the network topology with all devices connected close together.
    - Map the network topology with all devices connected far from each other.
  - Check how many messages per second the system is able to successfully deliver.

  - Check how the CRC values change depending on noise in the 2.4GHz frequency band.

- **Test D - Control strategy.**
  - Run a performance test of the signal strength control strategy.
  - Run a performance test of the obstacle avoidance control strategy.
  - Run a performance test of the compass heading control strategy.
  - Swarm behaviour:
    - Map how the system searches for a local minima.
    - Introduce badly tuned parameters and check how the system would now search for a local minima.
    - Introduce noise to all sensory inputs and check how the system behaves.
    - Remove the cyclic redundancy check of the network and check if the system performs as well.

The initial calibration tests are necessary in order to get the system up and running. This will make it possible to arrive at experimental values that could be used for further testing. It is desirable to perform simulations and use different techniques for tuning the controllers themselves, as discussed in section 2.2.1. However, as this project is time constricted and there exists a physical system wil it be more efficient to perform experimental tuning and calibration. It is then possible to move further and run performance tests for how the algorithm runs in general. The joint control- and communication algorithm covers several taxing parts for a micro controller, and it is not a given that it is able to run on a single chip solution. However, it is believed that the algorithm is able to run on an nRF52840 because of its modern floating point processor running at 64MHz, but it is not certain on what frequency. By using a logic analyser is it possible to see how often the processor is in sleep mode versus when it is active, which can be used as a measure to see how much leeway the processor has. The logic analyser in question is the Saleae Logic 16 Pro [65]. It is also important to map out the gain topology for the antenna, as it will illustrate possible areas where the antenna has no coverage, so called dead spots, if any. This would inform us if some orientations for the device are less ideal in order to obtain greater signal strengths. The antenna gain topology could be measured through an antenna measurer, in particular the Emscan RFX2 [28]. It is also out of interest to perform diagnostics of all sensory inputs and get an understanding of whether or not they are troubled with noise.

Test C and test D are directed towards the communication and control parts of the system respectively. All data from these tests are gotten from the data acquisition scheme presented in section 5.1. The first part of the communication tests are to illustrate the network topology, and see how data is transmitted in-between nodes in a different configuration. The second part of the communication tests are directed towards the performance of the network. It is desirable to have a high messaging frequency within the network, and to find the networks upper limits will a broadcasting device transmit $10k$ data packages with different messaging frequencies. The receiving device will transmit how many of these packages arrived. It would also be beneficial if the CRC values would indicate that a device is subject to a noisy environment for communication. All members of the swarm will receive $10k$ data packages and one swarm member will be set next to an active Wi-Fi router. Since the Wi-Fi router transmits on the same frequency band as Thread is it expected that this device will get more non-approved CRC values than its fellow swarm members. The control strategy defines all movements for the robot, and its behaviour is set by the three potential fields. All potential fields will be tried out on their own, and they will subsequently work together in order to give a behavioural pattern for all members of the swarm. It is believed that all devices will search for a local minima for all potential fields. Lastly will some parts of the system be removed or altered, as they might be redundant and unnecessary.

It is believed that when performing all of these examinations and scenarios one would arrive at an understanding of the systems performance. These tests would try to find the strong points and the shortcomings of the system design, and could result in possible revisions of the system architecture or the joint control- and communication algorithm.

A very important point to make is that it has its difficulties to describe the performance and behaviour of the complete swarm system on paper. It has therefore been created a supplementary demonstration video added in *Appendix 4 - Demonstration video* that displays the entire swarm architecture in action. It is highly recommended to watch this video before proceeding.

6 Results and further discussion

In this chapter will the test results for the complete swarm application be discussed. The swarm application consists of the algorithm presented in section 3 and the system architecture introduced in section 4. The swarm application consists of several parts and the tests are specialized to focus on one part at the time. In section 6.1 will the test results be presented. Secondly, in section 6.2 will the complete swarm application be reflected upon with the test results as a basis. Further will shortcomings of this project be discussed as a whole, and how it might have affected the design choices for the swarm application and its results. Lastly will the way forwards be considered, as in how the system can be improved and what remains to be perfected.

6.1 Test results

Section 5.2 describes all tests that are out of interest, and all tests from test A to test D will be covered in section 6.1.1 to section 6.1.4 respectively.

6.1.1 Initial calibration

The goal of the magnetometer test is to arrive at an estimate of its hard- and soft iron biases. This is necessary to be able to use the magnetometer for heading control. Sensory data from the magnetometer is acquired by performing movements in a figure-eight pattern of the complete robot device. It is important to test the entire configuration and not only the PCB that houses the magnetometer, as all electrical components present could infer a magnetic disturbance upon the magnetometer. The data plot is shown in figure 6.1.

Two different data sets are shown here, one where there is no motor actuation, and one where the motors are actuated at 1.5v. For the case of no motor actuation is the output nature of the magnetometer as expected from section 2.3.2. It shows a slight elliptical form with the major axis along the x-axis, and an offset onto the third quadrant. The distribution of the outputs are not symmetrical, and several test shows the same trend towards the area around the point (-400,-50). It is speculated that this trend is due to the presence of the battery. However, by actuating the motors at any voltage will the magnetometer output become greatly altered, as shown in figure 6.1. The output will be moved to the first quadrant and will converge towards a point rather than the expected circular nature. This will make it difficult to employ a calibration method as the hard- and soft iron biases are highly time-varying. A possible solution would be to design a configuration where the magnetometer is further away from the motors and the battery, as their influence on the magnetometer would be reduced. Because of the time-scope of this project will this problem not be further researched, and it is chosen to disregard the systems heading control for the continuing tests.

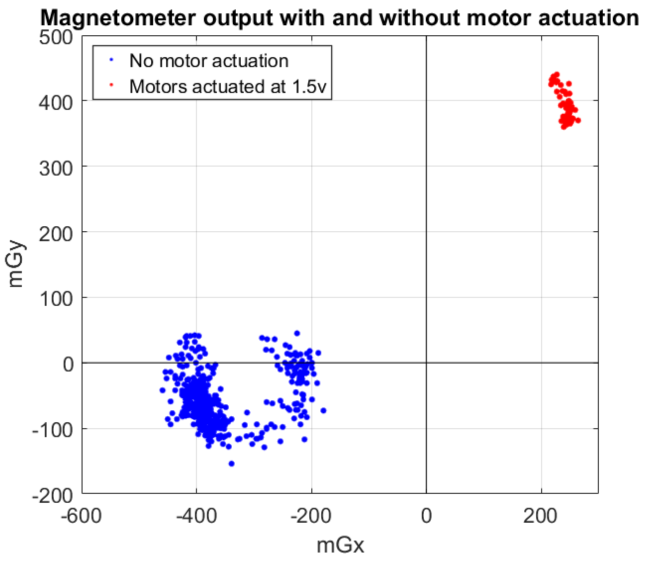


Figure 6.1: Data plot of the magnetometer output after performing movements in a figure-eight pattern.

A simpler, but although necessary test was to define the desirable area of signal strength values for a device. The control system wants to move the device towards a boundary where the signal strength value is neither too large or too small. As discussed in section 2.1.2 is the estimated maximum range of the Thread protocol from 10 to 100 meters. The test area is not large enough to arrive at critically low signal strength levels. The test was performed by placing five devices stationary on arbitrary positions in the test area, and then moving one device in said area over a duration of five minutes. By computing the average RSSI value over this duration was the desirable signal strength area defined as:

Average RSSI Value	Lower bound RSSI	Upper bound
-53	-55	-50

When the RSSI values are within the lower and upper bounds should the signal strength potential field have a zero value, as defined in equation 21. The lower and upper bounds are defined due to the presence of noise. Small noisy alterations in the RSSI value should not push the signal strength potential field from a zero value to a non-zero value, as this would be a false indication. Similarly was the critical distance from any obstacle defined as 500mm. If the distance is any larger than this will the control strategy ignore any obstacle, but for any distance under 500mm will the control strategy try to avoid said obstacle.

The final calibration necessary for the system is to arrive at experimental values for all controllers. As previously discussed was it out of interest use simulations in MATLAB to arrive at some reference values, that could be later fine tuned. However, as the project is time constricted was it decided to rather use experimental methods as they could be faster to employ when the physical system is present. This is feasible for the swarm application, as all devices are small with little mass, thus resulting in relatively little inertia and fast time constants. All devices in the swarm application are equal, and arriving at control values for one device would be sufficient for all devices. The difficulty of experimentation arrives at the architecture of the control strategy, discussed in section 3.2. The signal strength potential field is entirely discontinuous, as the potential field is zero in-between an upper and a lower value, and non-zero for all values outside of this boundary. All ranging potential fields used for obstacle detection on the other hand are entirely non-negative by design. The PID controller in this case is not used in the traditional sense, but rather used as a minimizing scheme for an arbitrary scalar value introduced by the potential field. The controllers are not directly controlling a reference value by applying a positive or negative control action, but indirectly by minimizing the potential fields.

It would be difficult to use well-known experimental tuning methods, such as Ziegler Nichols' method [8, p.44], as the system would have discontinuous oscillations at critical controller parameters, and the period of the oscillations themselves would be obscured. The method of applying a step response to the system would estimate the system as a first order process [8, p.50], which is a too broad of an estimation for a discontinuous system. Another well known method is to stress the system with an *infinitely* large P-value, and then measure the resulting output oscillations and their period [8, p.47], a form of simple auto tuning. However, as some of the potential fields are entirely non-negative would none of these oscillations be present. It was therefore decided to tune the system without any reference values, and entirely based on the following criteria:

- **The obstacle avoidance controller on the front and back** sensors should guarantee no collisions. This calls for a fast response and would result in a PD-controller.
- **The obstacle avoidance controller on the front and back** should have a fast response without being under damped, resulting in too large rotational movements. A PD-controller will have both dampen a system and have a naturally faster response.
- **The signal strength controller** should be somewhat slower than the obstacle avoidance controller as it has less priority. However, it is important to stay within an area with a desirable signal strength value, and the controller should therefore perform an increasingly stronger control action when outside of this area. This will result in a PI-controller. It is not recommended to apply a derivative control part here as the RSSI values are, in general, subject to noise. See figure 6.6.

The potential field scalar values defined in equation 20 and 21 were all set to 1000 due to the fact that the scalar value 1000 in the firmware is equal to a control action at 100% . By trial-and-error with the previously stated criteria are the final controller values as follows:

Controller	**Kp**	**Ki**	**Kd**
Obstacle avoidance front/back	135	0	27
Obstacle avoidance left/right	40	0	10
Signal strength controller	120	15	0

The performance of the controllers are shown in figure 6.7 and 6.6.

6.1.2 System performance

Before testing the joint control and communication algorithm itself is it important to see how the entire system is able to run. This would make it possible to discover possible bottlenecks that could hold back the system's performance. It was noticed during compiling that the firmware itself uses 32% flash memory, and 27% RAM. This would mean that there should be more room for further programming. Yet, this would also boil down to how much computing the programming requires. The nRF52840 was programmed to set an arbitrary output pin as high whenever the chip was in a sleeping state, i.e. it was not performing any tasks. The change in the state of the output pin could then be measured through logic analysis [65].

Figure 6.2 shows the sleep behaviour of the nRF52840 when the system is not connected to the network. It has been found that the algorithm is too taxing to run simultaneously as the chip is looking for a network, and when establishing a connection to the network. The control strategy is therefore halted whilst the device is looking after a suitable network. Figure 6.2 shows that the system is only sleeping for a little bit under 60% of the time when searching for a network. This shows the repercussions of using the software development kit. As the networking code was not written specifically for this project, is it naturally not optimized and uses more resources than it possibly should. For this project is it only required for the connection establishment to periodically advertise a *Search Gateway* message, which intuitively would not use over 40% of the system chip's resources. The sleep behaviour of the nRF52840 after connection when the complete algorithm is running at $25Hz$ is shown in figure 6.3. It shows that the system is sleeping for approximately 14% of the time, meaning that the algorithm takes it toll on the system. It shows that the joint control- and communication algorithm is indeed able to run on a single chip solution with the nRF52840. The absolute threshold for the chip is when running the algorithm at $50Hz$, as seen in figure 6.4. The system is sleeping at 0%, but performing as expected. When turning the frequency up will the system not be able to maintain communication with the network resulting in troublesome disconnects. If the algorithm were to be run at even higher frequencies should either the software development kit be disregarded, or one could employ a dual chip solution with the communication part and the control part of the algorithm separated in its own chip.

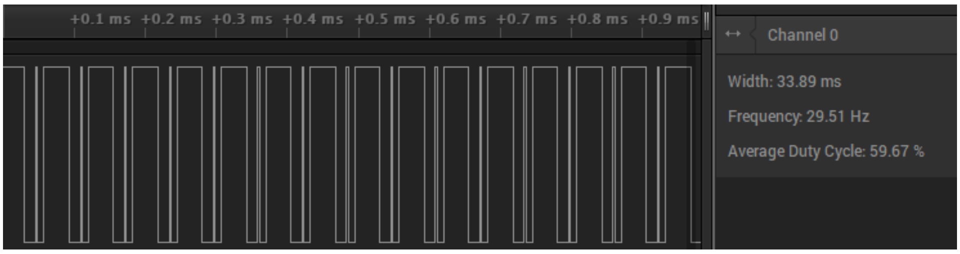


Figure 6.2: Logic analysis of the sleep behaviour of the nRF52840 when searching for a network.

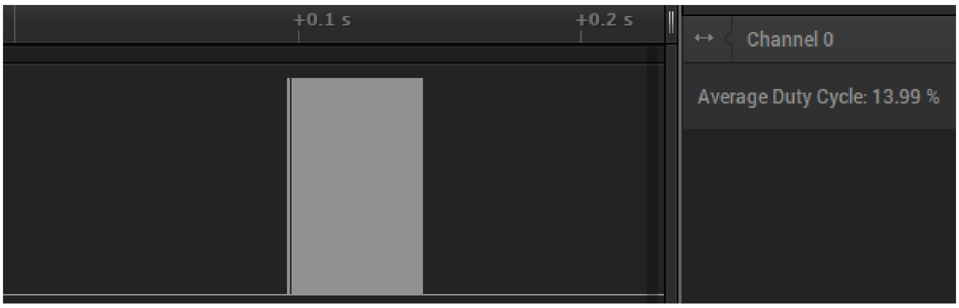


Figure 6.3: Logic analysis of the sleep behaviour of the nRF52840 when running the algorithm at $25Hz$.

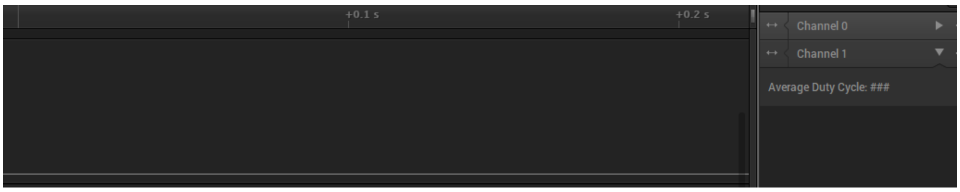
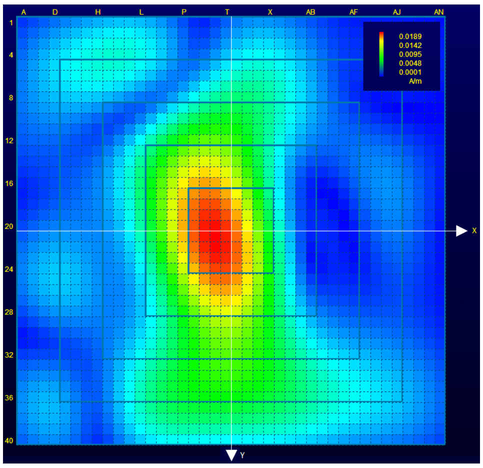
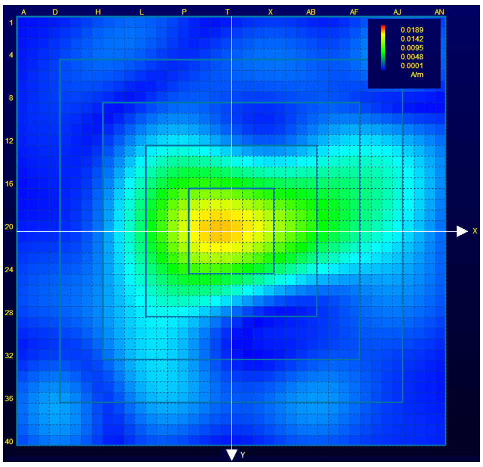


Figure 6.4: Logic analysis of the sleep behaviour of the nRF52840 when running the algorithm at $50Hz$.

Further was the antenna gain topology of the chip antenna present in the main board measured. The test was performed by placing the main board on an antenna gain measurer, specifically the RFX2 [28]. The main board was programmed to continuously publish arbitrary messages in the 2.4GHz frequency band. The antenna gain measurer would then use its own antenna to measure and receive messages on the same frequency band and see how the signal strength amplitude of said packages will be altered to create a mapping of the test board's topology. The first test was performed by sending measurement packages along the x-axis, and then along the y-axis. This is due to the chip antennas physical form, as it is not symmetrical will one axis naturally have a stronger resonance [66].



(a) Measurements done along the x-axis.



(b) Measurements done along the y-axis.

Figure 6.5: Antenna gain topology of the main board, seen from a top-down perspective.

A two-dimensional illustration of the antenna gain topology when measuring along the x-axis and the y-axis respectively is shown in figure 6.5. Similarly is the three-dimensional illustration of the antenna gain topology for the x-axis case shown in figure 6.5b. The tests show that the system has no dead spots, which means that any orientation of the robotic device should be able to receive a message from the network. It is also noted that the topology of the antenna is not spherical, but rather elliptical with the largest gain along the z-axis. It can also be seen that the gain is stronger when receiving a package along the x-axis rather than the y-axis. Both of these results show that the signal strength, and consequently the RSSI-values, are highly dependent of the orientation of the system will have a somewhat non-linear nature. This means that a rotation will yield different RSSI-values, although the robotic device has not changed its position. Although undesirable shouldn't this result halt the system, but it will take longer for a device to find a local minima within the signal strength potential field. Additionally, as this project focuses on the two-dimensional case is it not needed for the antenna to be the strongest along the z-axis. None of these aspects are critical for further testing, but they could be improved upon by performing antenna tuning [67].

Lastly were the sensory inputs of the signal strength measurements and the ranging sensors looked into. It is necessary to see if the sensory inputs are troubled with noise or if any filtering methods should be applied. A noisy input would intuitively result in a noisy output and should be avoided. The change in RSSI values was realized by moving the receiving device around the network. For the case of the LS sensors was simply a hand gestured in front of the sensor a couple of times to see how the measurements changed. It should be noted that henceforth will all ranging measurements above 1000mm be defined as no obstacle is present.

Figure 6.6 and 6.7 illustrates the signal strength measurement and the ranging measurements respectively. Figure 6.6 illustrates that it is indeed necessary to employ the moving average filter discussed in section 3.1 in order to remove most of the noise. After using the moving average filter is the signal strength values seen as satisfactory. Figure 6.7 shows that the ranging measurements are adequate.

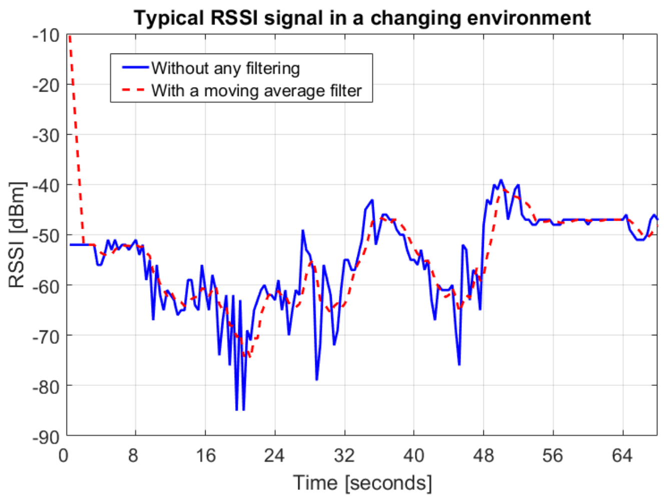


Figure 6.6: RSSI signal values with and without a moving average filter.

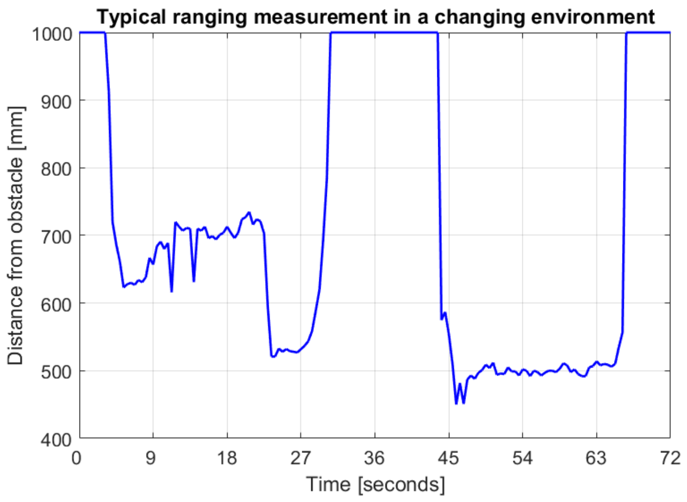


Figure 6.7: Normal behaviour of the range measurements from the laser ranging sensor.

6.1.3 Communication protocol tests

The communication tests consists of mapping out the performance of the communication protocol and its upper limits. The communication protocol consists of ROMANOs and MQTT-SN running upon Thread, as illustrated in figure 3.1. This combination is new as ROMANOs is first introduced in this project. It is expected to have similar performance as traditional ROMANO however, if not even greater as ROMANOs has the same message overhead but less message payload. The initial test was to get a mapping of the networks topology and how it changed depending on the distance between the number of nodes in the network, and the distance between them. It is possible to get an illustration of a thread networks topology in real-time by using the nRF Thread Topology Monitor (nRF TTM) [68] and any development kit equipped with a 802.15.4 antenna. The nRF TTM is an application that scans any Thread network by using the aforementioned antenna. As mentioned in section 5.2 is it desirable to illustrate the case where a small number of nodes are connected to each other, and then see how the network grows when additional devices are added.

Figure 6.8 shows the monitoring result for three different cases from A to C. Case A illustrates the network topology created due to the presence of the three swarm robots, one Thread Border Router and the scanner itself. The network is typically many-to-many and showcases that the network does indeed behave as expected. The network is expanded to five swarm robots in case B, and it shows that the number of connections increases exponentially. For the sake of robustness is this redundancy desirable. This redundancy should not affect the performance of individual devices either, except from that every link is stored in each routers memory. Case C illustrates the case where a device is far from the rest of the network, but still has connection to one routing device. If the routing device suddenly loses connection to the network would this be a single point of failure for the end device.

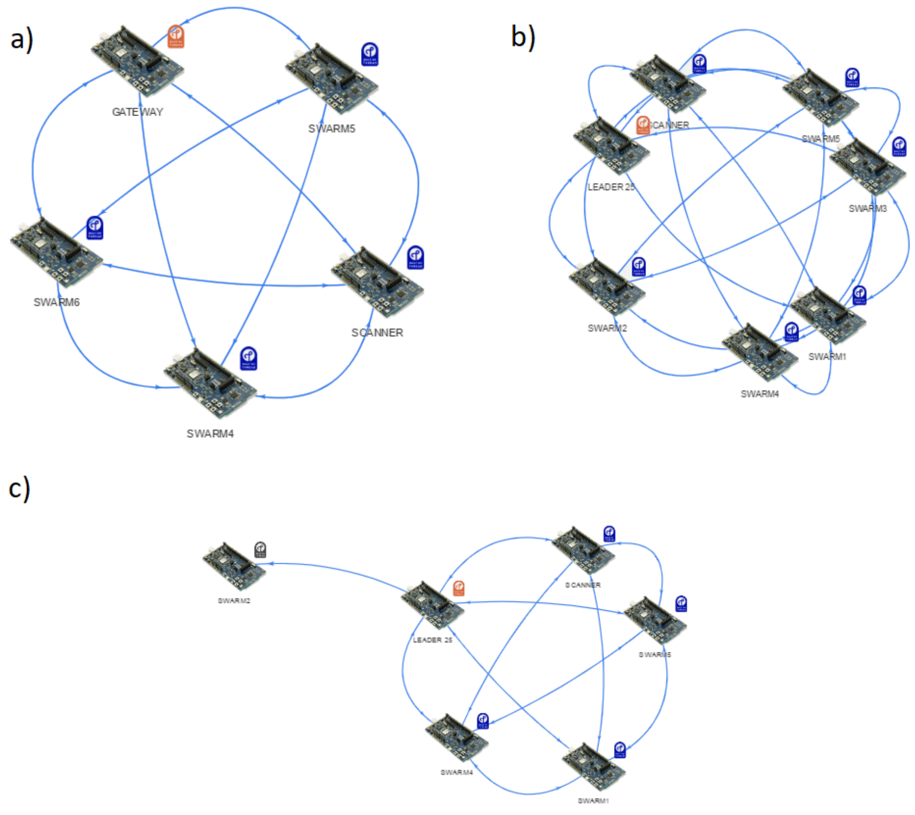


Figure 6.8: Topology monitoring of the swarm network for three different cases. The orange tag denotes the leader of the network, whilst a blue tag denotes a routing device. Similarly is a black tag an indication of a end device.

It was also necessary to define the upper limits of the messaging frequency of the network. One swarm device was therefore configured to broadcast 10 000 heartbeat messages at different messaging frequencies. This would result in the network to have six receiving devices, the gateway and five other swarm devices. The average number of data messages that were received at differing messaging frequencies are as follows:

Messaging frequency	Average number of messages received
5 Hz	100%
10 Hz	95%
50 Hz	73%
100 Hz	46%
500 Hz	1%

This result shows that the system is not able to transmit messages above 5Hz, as some messages would not be successfully transmitted through the network. This result is surprising, as traditional ROMANO has been reported to have a stable messaging frequency as high as 200Hz [2, p. 5]. For the sake of robustness is this a huge weakness, as the system would not be scalable. Nevertheless, as ROMANOs is especially created to require a small messaging frequency should a frequency of 5 Hz be sufficient for a small swarm system. It is believed that the gateway/broker solution used in this project is the bottleneck for the swarm system. It is recommended for further development to rather create a custom gateway and a local broker instead of using ready-made solutions used for general projects.

It was also out of interest to check if there would be an apparent change in the CRC values if a swarm device was set in a noisy environment for data communication. This would mean that the swarm device is able to detect the quality of its current position in the data network and possible recovery actions could be applied. The swarm device is first set in a normal environment with little noise for reference measurements. Afterwards will the same swarm device be set near an active Wi-Fi router broadcasting on the 2.4GHz frequency band. One gets the following result when transmitting 10 000 packages for both cases:

Environment	**Avg. number of not OK CRC values per 10k msg.**
Normal	10
Noisy	530

This is indeed an interesting result as it shows that any swarm device is able to detect that it is currently in a noisy environment. It is recommended that this is researched further and there should possibly be designed recovery methods to get out of this environment. It is an important factor for robustness to deal with a noisy environment, which the system is currently not. It would be interesting to perform swarm behavioural tests by removing the cyclic redundancy checks, however, no environment that were noisy enough to have a continuous effect on the CRC values during movement were available.

6.1.4 Control strategy and system behaviour

The control strategy of the system is the most defining property of the swarm application, as it defines the movement of every device. For the current swarm system will the control strategy be the only decision maker, and those decisions will be done based on RSSI values and ranging data. A desirable outcome of this decision making is that each device is able to arrive at a minima of its own potential fields, a local minima. Before running the full control strategy will each sub part be tried out on their own. Both the signal strength control strategy and the obstacle avoidance strategy has to arrive at a minimal potential field value for their own respective fields, in order for the system to arrive at a perceived *true* local minima within the swarm.

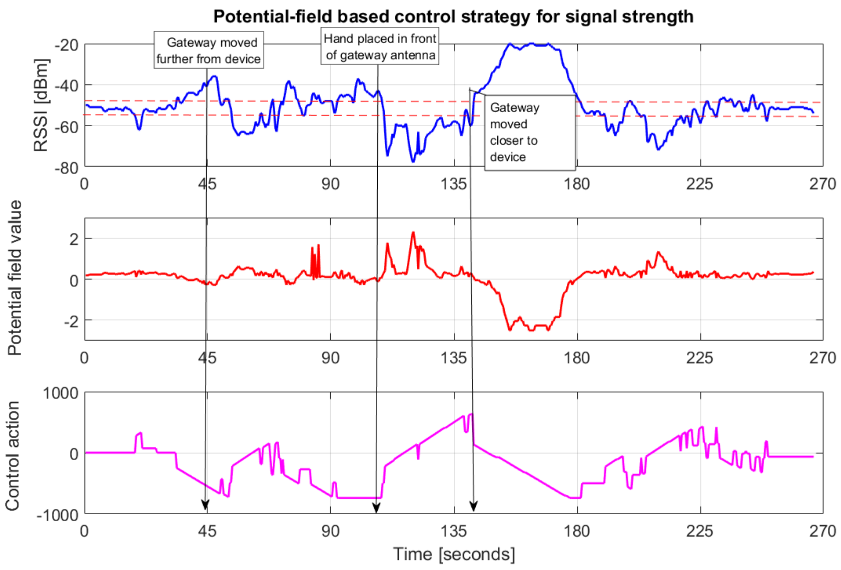


Figure 6.9: Control strategy with only signal strength control enabled.

Figure 6.9 shows the change in the RSSI values, the signal strength potential field value and the following control action when only running the signal strength control strategy. The test was performed by introducing a simple point-to-point network with one swarm robot and the Thread Border Router/Gateway for easier control of the swarm robot's RSSI value. The gateway was moved during testing in order to change the RSSI-value in real time. For reference is a value of 1000 a motor actuation of 100%, a control action of around 100 is needed to make the device moving. One can see that initially is the RSSI value within the defined desirable area, put the input is rather noisy and leads to some small hiccups. The first disturbance applied to the system is at t = 45, when the gateway is moved away from the robot device and the RSSI value decreases. One can see that the potential field value then grows slightly, and the necessary control action to increase the RSSI value is applied. One can see that the system is seemingly underdamped. It should be noted that the RSSI values are in general quite noisy, and slight rotations will change the signal strength, as discussed in section . Nevertheless can one see that after the first perturbation is the signal strength value slightly above the desirable area. The second disturbance is applied t = 105, where a hand is covered over the gateways antenna. It is apparent that if something physically comes in-between a connection will this change the signal strength, and also for this case will the control strategy act accordingly and apply a control action. However, one can see that the signal strength converges towards a value slightly below the desirable area. The last disturbance applied to the system is to move the gateway very close to the robot device at t = 140. The potential field becomes repulsive and the potential fields scalar value turns negative rather than positive. The swarm device is successful in computing the right control action sequence and when the system achieves a steady-state within the desirable signal strength area.

It is clear that the signal strength control strategy does indeed work, and the control action has a somewhat smooth nature. It is noted that the system is rather slow, as for the worst case it used almost 100 seconds to hit a steady state. Additionally did the system only hit a steady state in the reference area in some of the cases, even though the controller has an integral term. It is very apparent that additional controller tuning is indeed very much needed, but the current system works as a proof-of-concept.

A similar situation for the obstacle avoidance control is shown in figure 6.10. As mentioned in section 6.1.1 will the obstacle avoidance control strategy try to avoid any obstacle closer than 500mm. Any distance above that is disregarded. It can be seen in figure6.10 that the control strategy is largely successful, and as expected will the potential field values be entirely non-negative. It should be noted however that the controller has been tuned with a fast response in mind, as it was seen as crucial to avoid any obstacle. This is a matter of preference. It can be seen that the derivative part of the control strategy results in large spikes in the control action, which results in a fast avoidance but a greater toll on the motors. It could be beneficial to tune the PD-controllers to have a smoother nature in the future.

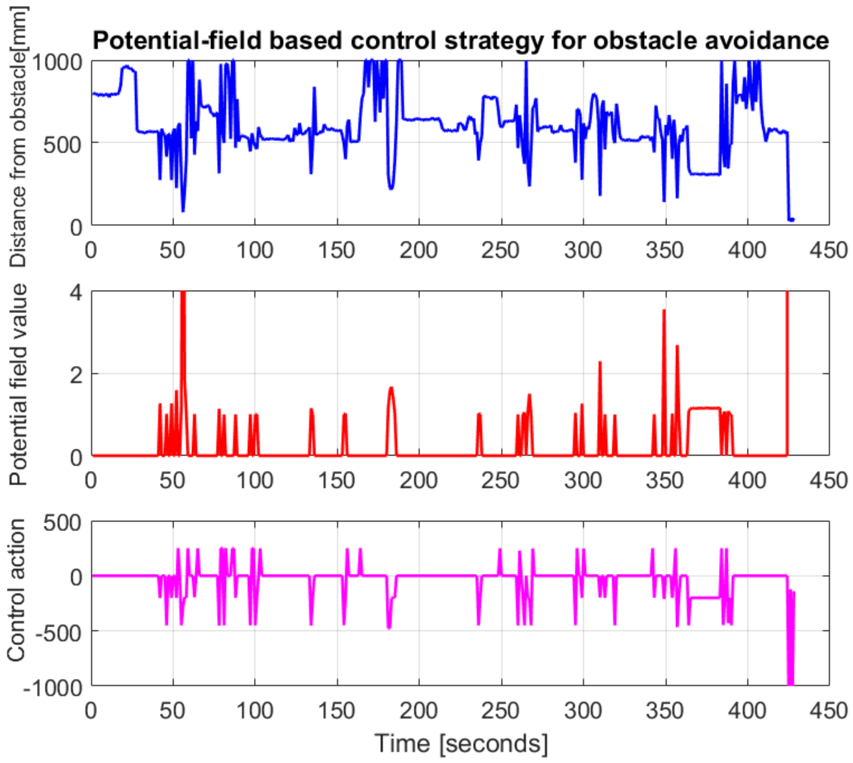


Figure 6.10: Control strategy with only obstacle avoidance enabled.

Further was the complete control strategy tested, as can be seen in figure 6.11. This figure holds a lot of information, but it is necessary to show the entire picture with all sensory inputs for this case. This figure represents the complete joint control- and communication algorithm running in a swarm apllication. Each column shows different sensory data, - from signal strength to ranging sensors -, their respective potential field, and their respective control action. The test was performed by initiating the complete swarm system and then monitoring a device that tries to find a local minima within the swarm. The test is divided into three sections, section S1, S2 and S3. In section S1 is the device searching continually after a local minima for all potential fields. It will not stop moving before every potential fields have a zero value. In section S3 has some change appeared in the network and the actual robot device ends up with a lower signal strength than desired. This results in the device searching for a new local minima. Section S1 and S2 shows that the system is actually capable of finding a local minima and remains stationary in that point. This is the expected swarm behaviour, and this is a huge result for the entire swarm application. The results shows that it is possible to design a complete swarm application that finds local minima with a potential-field based controller. Similarly, as shown in section S3, will the device continue to search for a new local minima whenever there is a change in the network, resulting in a flexible and responsive swarm system.

It was also interesting to test how the system would perform with badly tuned controller values. This would indicate if the system even needs a well tuned minimizing strategy and if the system is sensitive to different tuning values. The same test as in 6.11 was performed by first setting all PID controllers as pure P-controllers with a value of 1, and then by testing the system with 10 times higher of the normal controller values. The first case with only P-control is depicted in figure 6.12, and the case with 10 times the normal values is shown in figure 6.13. For the case of simplicity will only one ranging field be shown. Both cases act as expected. For the case with only P-control is it apparent that the values of the potential field themselves are not high enough to achieve a sensible motor actuation. Too high values on the other hand would result in an unstable system with very high oscillations. A potential field system requires a relatively well tuned minimizing strategy in order to find a local minima.

Lastly was the case of sensory noise tried out. It was interesting to see how well the control strategy would handle noise, as it would give an indication of its robustness. The averaging filter on the RSSI values was therefore removed, and a random noise on every ranging signal were added by adding 1% of the left adjacent ranging sensor and removing 1% of the right adjacent ranging sensor. The result is shown in figure 6.14. It is apparent that the system is not able to handle noise that well, as a noisy obstacle avoidance system would result in unwanted rotations. These rotations would in turn further alter the RSSI values that are already riddled in noisy, resulting in even worse control and even stronger oscillations. This would make it difficult and highly unlikely to find a local minima. It is indeed very important to employ methods to mitigate noise over the entire system.

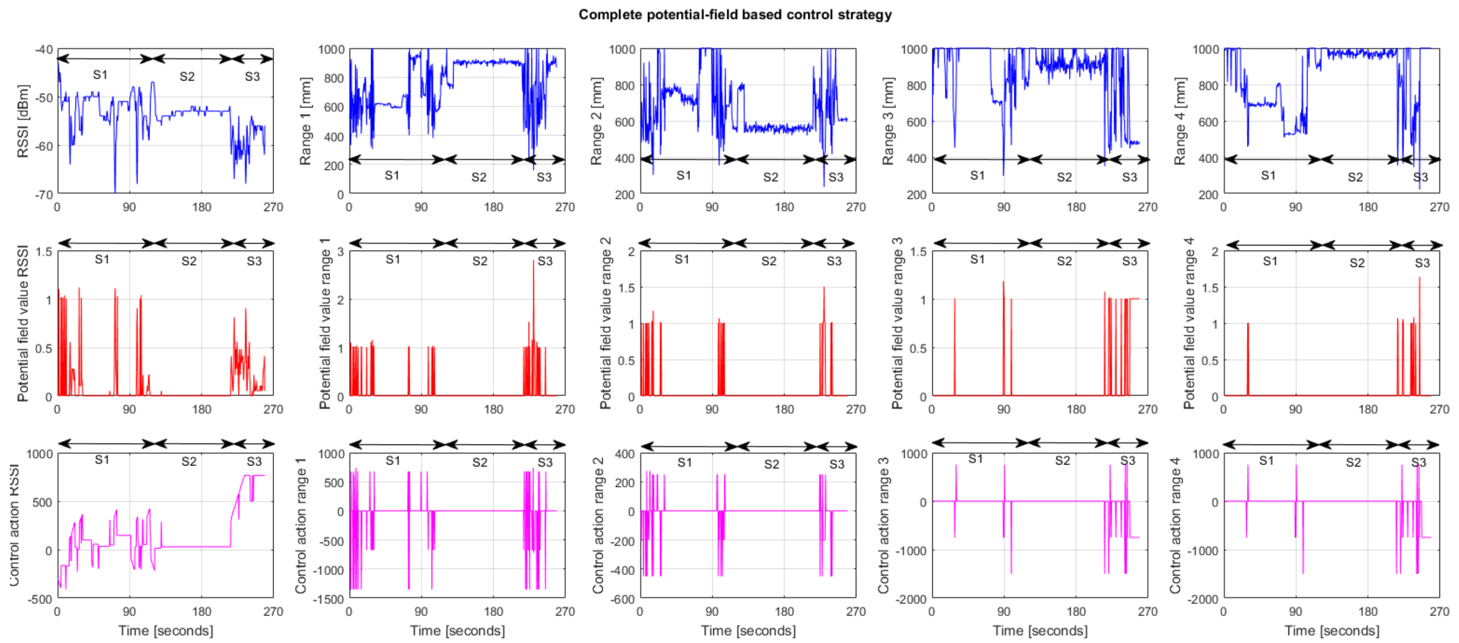


Figure 6.11: Complete control strategy with both obstacle avoidance and signal strength control.

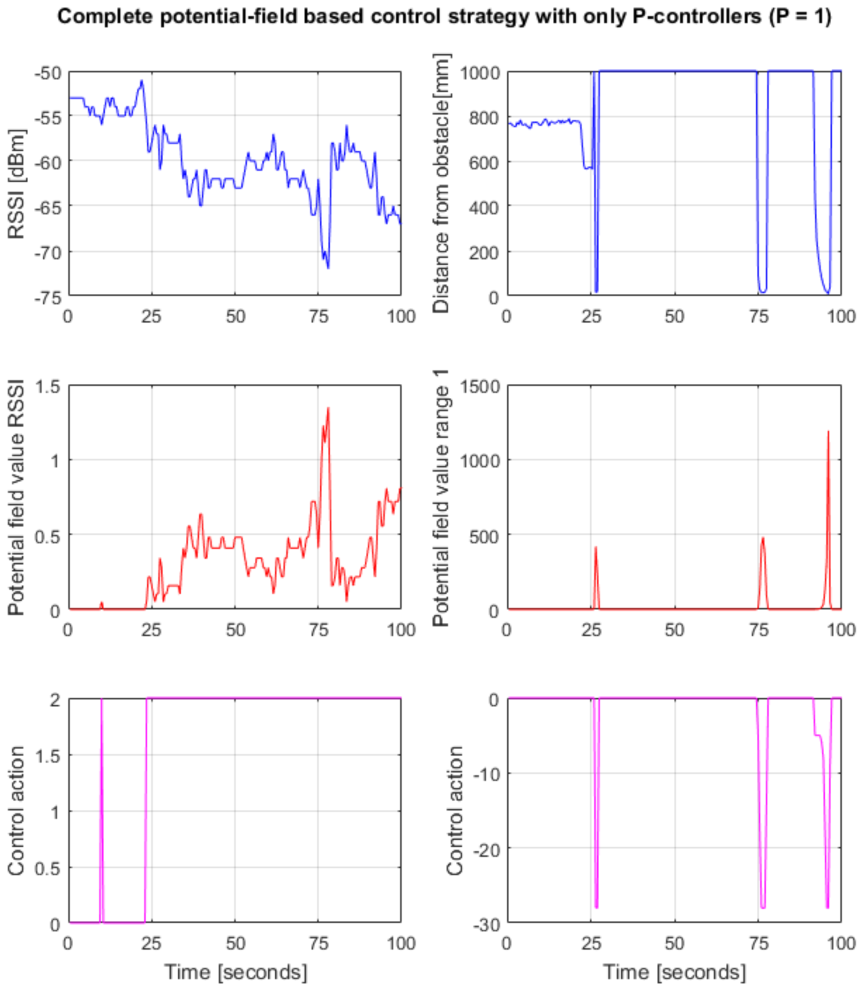


Figure 6.12: Complete control strategy with P-controllers, where P = 1.

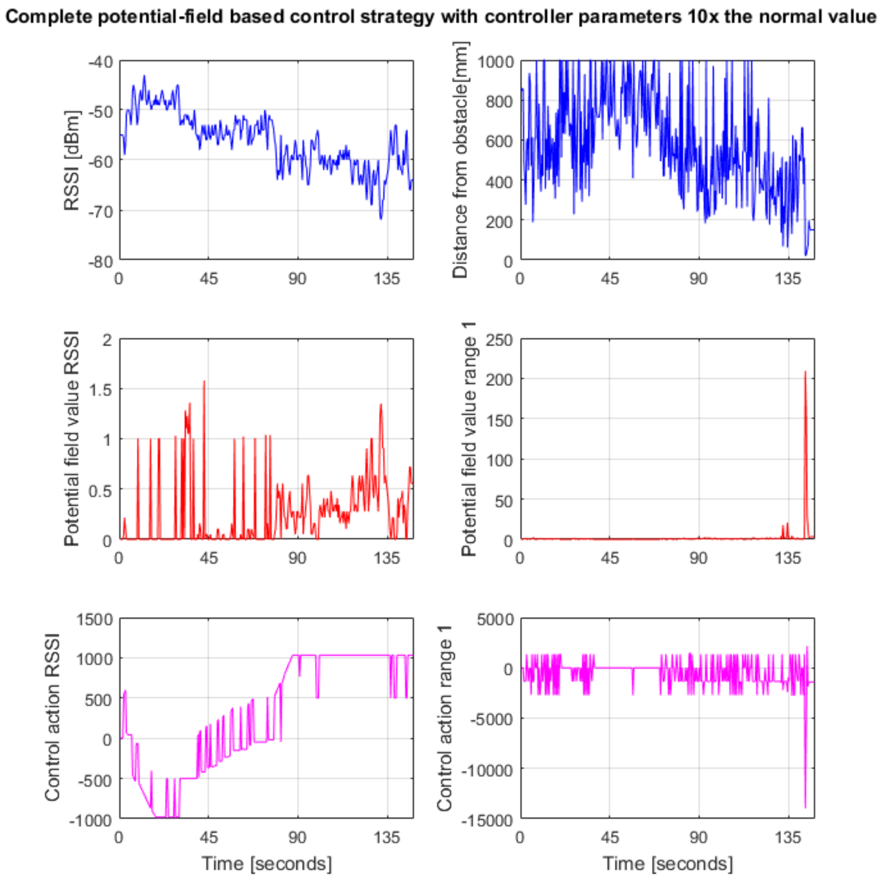


Figure 6.13: Complete control strategy with 10x the normal control values.

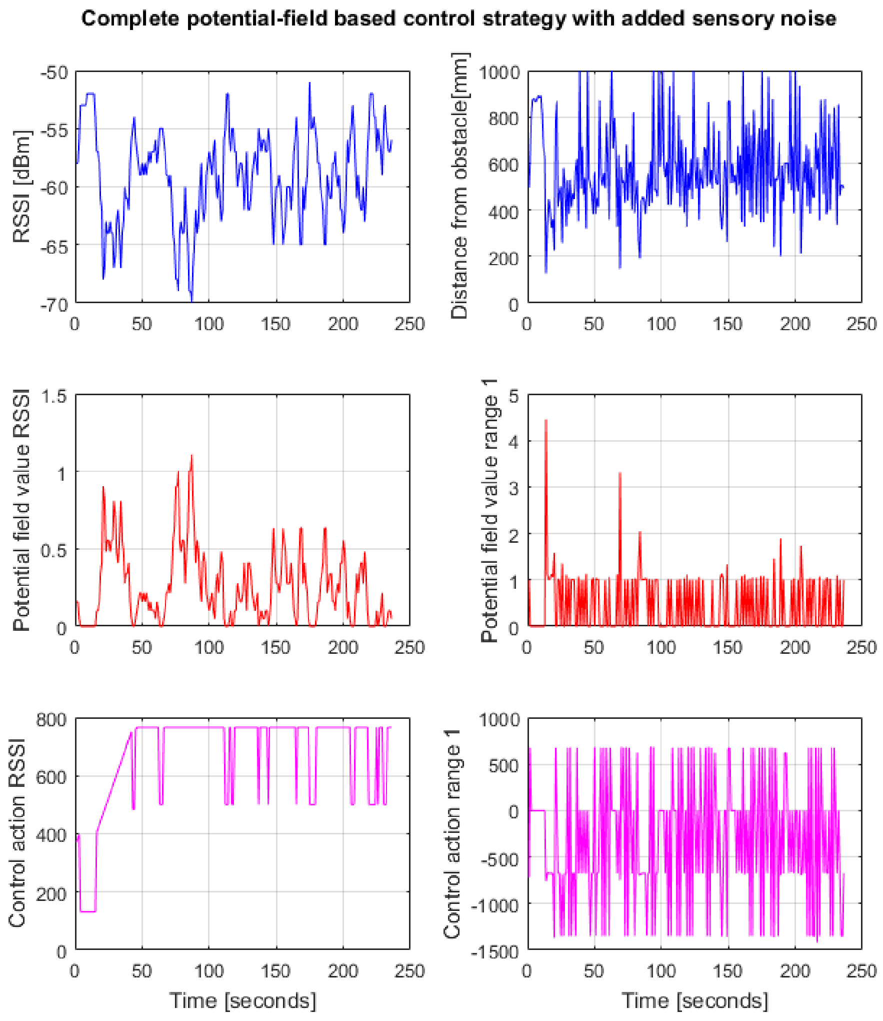


Figure 6.14: Complete control strategy with added sensory noise.

6.2 Discussion of proposed algorithm and hardware platform

The tests performed in section 6.1 yields a mixed result. The swarm application is a combination of a multiple of different parts, and it is not surprising that some parts performs better than others. All of the tests described in section 5.2 are designed to create a mapping of the systems performance, and give a measure of its robustness. It is non-trivial to decide what tests should be performed and exactly what information they should search for. As previously discussed is the swarm application complex, and one could perform a great magnitude of tests to map out every single detail of the system. However, it is inefficient to discuss every single little detail, and not suited for the scope of this project. It is believed that the most important aspects of this application is covered and the tests are sufficient to come to any conclusion.

The most interesting result, by far, is illustrated in figure 6.11. It illustrates that when given enough time will any member of the swarm find a local minima within the network. It is an interesting result because it illustrates that the entire system exhibits some sort of behaviour that could be perceived as intelligent, although no intelligence is actually present in the system. The decision making is simply due to the fact that the communication network is directly linked to the control strategy. One could almost call this behaviourism as some sort of natural *instinct* inherited from the joint control and communication algorithm. All decision making is based solely on heartbeat messages and ranging data. The members of the swarm are not performing any direct communication and collective decision making, similar to normal honeybee behaviour [69]. From a practical standpoint would this indicate that a multiple of parts in the system is working as expected, from algorithmic design to hardware design, and anything in-between. However, the discovery of a local minima alone is not a strong enough indication of a robust swarm system. As discussed in section 5.2 is it desirable for a true robust system to be scalable, efficient, adept and redundant.

The presented swarm system contains proposed solutions for hardware design, in-room localization, communication and control. The system performance tests showed that both the hardware design and the in-room localization scheme was favorable for the most part. The hardware profile consists of two PCB's mounted on a printed cylindrical chassis. One could argue that the hardware exhibits some sort of redundancy, since there are a multiple of devices present. The firmware itself don't even utilize half of the memory available on the nRF52840, indicating the adaptability of the firmware. Yet, the logic analysis indicates that the nRF52840 is pushed to its limits by running the system at any higher frequencies than 50 Hz. It is also a clear disadvantage that the system is not able to connect to the network while the algorithm is running. A possible solution would be to either refine the firmware, or to introduce a dual chip solution to reduce workload for each chip. The current system proved to be sufficient and could fortunately be used for further testing, but it indicates that the firmware is neither efficient nor scalable. Measures should indeed be taken to achieve these criteria.

The in-room localization method was decided to not rely on any external infrastructure, as the swarm should be able to adhere to any environment. It is therefore a part of each swarm device to ensure adaptability. The localization method is distributed, and each device is able to synthesize its own understanding of its localization on locally available information. This would result in no intuitive method for realizing a global coordinate system for the entire swarm, which could be an interesting field of study in the future. Localization is instead based on signal strength data and ranging data, and both figure 6.6 and 6.7 indicates that the input data is sufficient for further use, although the signal strength values needs some sort of filtration. A global coordinate system would be beneficial in order to establish some sort of formation control. Currently will each member of the swarm see each other as an obstacle for the most part. Studies performed by Rubenstein et al. indicates that the current laser sensing method could possibly be used for some sort of direct communication in-between the swarm [1, p. 797]. Further, as illustrated in figure 6.5 is the antenna topology not symmetrical, resulting in a somewhat non-linear nature for the RSSI-values. It could be benficial to perform antenna tuning in the future. As the ranging method uses laser sensors with a relatively small sensitivity cone, and the signal strengths measurement would only improve with a growing network, one could indeed say that the localization method is scalable. There are a multiple of ranging sensors on each device, and the antenna will gather signal strength data from all members of the network, introducing a much needed redundancy. Additionally is it only required for the nRF52840 to poll input data from either the antenna module or the ranging sensors, as all data is efficiently computed on the external modules. It was initially desired to use a magnetometer for heading control, although this was never critical for the rest of the system to work. It is an interesting idea since it could be used to realize for example formation control or a swarm based guidance system. As figure 6.1 indicates is this not possible with the current hardware configuration, as the magnetometer only exhibits the expected nature during no motor actuation.

The communication protocol is based on Thread, MQTT-SN and ROMANOs. The Thread protocol establishes a mesh network that consists of all swarm devices in a many-to-many configuration with a joint Thread Border Router/MQTT-SN Gateway and an external broker. Figure 6.8 illustrates the Thread network and how the number of communication lines grows exponentially by each networking device, resulting in a much needed redundancy. It should be noted that on some rare occurrences would a device establish a single connection to the network, introducing a single point of failure. The idea however is that this would be a rare occurrence due to the signal strength control strategy. The signal strength control strategy would most likely attract a device to greater RSSI values. The moving average CRC-filtration method stores RSSI-values for all nodes in the network and removes static values based on *polling* rather than *timing*, see section 3.1. It means that the signal strength control scheme will always be active, regardless if a end-device has lost its connection to its router or not. One could argue that the storage of RSSI-values introduces some sort of adaptability. This could make it possible for the end-device to converge towards the network upon a loss of connection to its router, but it is not guaranteed as the RSSI-values would not be updated when no connection is present.

Perhaps the most surprising part of the communication was its poor performance on messaging frequency. It is sufficient for a small swarm network, but a frequency of 5 Hz with six receiving devices is absolutely not scalable for a larger system. This would result in increasingly worse control and perhaps the non-discoveries of local minima, resulting in inefficiencies. However, it is expected that the current architecture could sustain higher frequencies [2, p. 5], and a probable cause is the use of a general Thread Border Router/MQTT-SN Gateway solution. The swarm system introduces a hierarchy consisting of the robotic devices, the gateway and the broker. The introduction is beneficial since it enables the swarm to communicate over great distances, but it also introduces a single point of failure. It is possible to introduce a multiple of brokers working together [44], or one could create a completely flat hierarchy with larger robotic devices that are able to both run the swarm algorithm and simultaneously work as a MQTT-SN gateway and a local MQTT broker. Such a task would require a much more powerful computing device able to run a single threaded operative system, and would result in a much more cost inefficient system, but could be needed to ensure true robustness.

The control strategy was proposed to be a potential-field based PID controller. This is a trade-off from the potential-field based MPC controller in order to ensure efficiency. The potential field philosophy is an interesting one, as it is rather intuitive in its design and adaptable to many environments. The computations are rather efficient as it revolves around fast multiplications. The control strategy is shown to be largely successful on signal strength control and on obstacle avoidance, as illustrated in figure 6.9 and 6.10 respectively. The number of potential fields are of course scalable and can work together. As it has been discussed previously will the combination of the signal strength field and the obstacle avoidance fields result in a system behaviour that searches for a local minima within any given environment. There were performed no simulations to better fine tune the system. It is highly recommended to pursue further with establishing a suitable swarm model in order to arrive at some reference values. Even though the system is successful shows the obstacle avoidance behaviour that the control action is rapidly changing, resulting in a shorter longevity of the motors. Additionally, as seen in figure 6.11 does it take around 100 seconds to find a local minima. It is probable that different controller values would reduce this time. Most likely would some sort of intelligence or additional decision-making be employed to further reduce this time, if desired. The importance of well tuned controller values was further shown in figure 6.12 and 6.13, illustrating that it is not possible to locate a local minima with a badly tuned controller strategy. Is it also noticed in figure 6.14 that the system is especially sensitive to noise. This sensitivity could result in unwanted motions, and it could be important to employ some sort of filtration or an estimator to smooth the control action in the future.

It is not wise to define the current proposed swarm application as robust. Yet, it can't be stressed enough that this proposal shows great promise. All parts of the system introduces some sort of scalability, efficiency, adaptability and redundancy to a different degree. It is indeed fully possible to refine the less favorable parts of this application, possibly leading to a true robust system. The algorithm and swarm architecture proposed in this report are a stepping stone in the right direction, but the execution itself needs refinement.

6.3 Shortcomings

Based on the discussion in section 6.2 is it apparent that the limitation of time affected the results. There have been proposed solutions for both hardware design, in-room localization, control protocol and control strategy for robotic swarms, but the execution is lacking on different parts. The complete system yields favorable results for the most part, but for the sake of robustness is there much to be done.

The largest drawback of the system is the communication protocol's low messaging frequency. It takes a large amount of time to program a full implementation of the Thread protocol with MQTT-SN, the Thread Border Router solution and the MQTT-SN gateway. It was therefore chosen early to not use that much time to develop these aspects, but rather adapt ready-made solutions provided by Nordic Semiconductor. It is clear that these parts should be made from the ground up, especially for the swarm application case, to increase the messaging frequency of the system. Similarly is it led to believe that the usage of the Thread 11 Software Development Kit halts the efficiency of the firmware, as the system only sleeps at around 60% when the firmware is in an idle state and searching for the network. See figure 6.2. It would be possible to work without this SDK, but it would not fit the time-scope of this project.

Time is an apparent limitation for most of the project. The system shows great promise and could possibly lead to a true robust system, but especially the executions of each parts had a little room for development. It resulted in the fact that the system had to be used *as is*, and tested at its current state with little to no revisions. This is indeed apparent with the magnetometer, as it shows an expected behaviour at no motor actuation. A revision of the magnetometers position could lead to an usable heading control system. In general would it be interesting to both revise the swarm system and introduce new parts to tests its improvements, such as state estimation and swarm intelligence.

As discussed previously is the field of swarm robotics vast and ever changing. The field is experienced as new, and the contributions to the field grows exponentially in many different directions. There are a myriad of solutions and different implementations to research, whilst still performing project planning, algorithmic design, construction, testing and so forth. It is not feasible to believe that the literature study covered all previous work, and further research should be done to get the entire picture of swarm robotics. The proposed swarm algorithm itself is backed up by theory and looks promising based on previous research. However, as is stated by Gosh et al. [2] is it currently believed that ROMANO is the only option as a lightweight application overlay protocol for robotic swarm applications. This would make the proposed swarm algorithm in this project the second option as of writing. As this is uncharted territory is there a very real possibility that several aspects of the algorithm must be revised during the continuation of the project.

6.4 Suggestions of future research

This project introduces a promising platform for the continuation of further work within the field of swarm robotics. The framework introduced by this project needs both revisions and possible additions to achieve a true robust result. There are several paths to take for the continuation of this project, the following bullet-point list is only a recommendation for future work.

Suggestion of possible areas that could be researched for this swarm application are as follows:

- Write the complete firmware from ground up in order to create a more efficient system.
- Refine the configuration of the robotic device in order to get sensible data out of the magnetometer during motor actuation.
- Create a specialized Thread Border Router / MQTT-SN gateway unit from the ground up for the swarm application with high messaging frequencies in mind.
- Create a specialized local MQTT broker unit from the ground up for the swarm application. Make the network able to house multiple brokers and switch seamlessly between them upon a single broker failure.
- Introduce a flat swarm hierarchy with only one type of device, removing all dependencies of non-robotic swarm devices.
- Perform simulations and further experimental tuning of the controllers to get smoother control and possible faster movements. Tune the controllers to better suit the system with robustness in mind. Possibly introduce new control strategies or additional potential fields.
- Introduce intelligence to the system for stronger decision-making. Make the system take recovery action when a robot is lost to the swarm, and if a robot is asked to make an illegal move, take action to make this move legal (if possible). Implement methods to reduce the discovery time of a local minima.
- Perform additional testing to further establish an understanding of the systems degree of robustness. Use this as a basis for further revisions.
- Use intelligence to make the swarm be able to perform complex tasks, working together in a multi-agent system.
- Possibly use intelligence to learn the environment and introduce a Simultaneous Localization and Mapping (SLAM) system. This should be possible trough the usage of the current laser sensors.
- Since the system uses IPv6, make several swarms communicate with each other over the internet. Possibly performing larger tasks in a large building, or several.

- Perform revisions of the PCB for better protection against reversed currents and strong magnetic fields.
- Perform antenna tuning to further enhance its range, and to create a more spherical gain topology for a more linear change in the RSSI values.
- Further develop the hardware platform to make it able to run possible new versions of the joint control- and communication algorithm.
- Introduce a dual chip solution (or more) to further enhance the performance and frequency of the swarm algorithm. A suggestion would be to run communication on one chip, and the control scheme on another.

Clearly, some points should be prioritized over others, but it is difficult to understand the complexity, time-requirement and importance of each task. It is a task in itself to make a mapping and find out what tasks should be prioritized and in what order. It is recommended that this will be done first before further development is done with this system.

7 Conclusion

This study has attempted to find out a possible solution to the following question: *Which technologies and algorithmic design are suitable for both the control strategy and the communication protocol to ensure robustness within a robotic swarm*

To answer this question was a research performed on the field of swarm robotics, and a proposal to a possible robust platform for swarm applications was made. The main philosophy was to get an understanding of which technologies would result in a true robust system, and test the combination of said technologies on real-world scenarios. This final chapter will sum up the results of the aforementioned test platform, before it presents some concluding remarks.

7.1 Evaluation of the proposed platform

The presented swarm system contains proposed solutions for hardware design, in-room localization, communication and control. The innovative part of this swarm system is the idea of joining communication and control in a cooperative manner. The signal strength associated with the communication network would directly influence the potential field based controller, and the controller should initiate control actions to maintain said network. The decisions for the hardware design and the in-room localization method were based on giving a physical realization of the joint control- and communication algorithm. The findings for each part are as follows:

1. The hardware platform, consisting of two PCB's connected with each other, is successful with running the complete swarm algorithm and peripheral programming on a single chip solution. The nRF52840 proves favorable when establishing a self healing Thread mesh network. It is noticed that running the programming is taxing for the processor, and the upper limit for the algorithm's frequency would be set at 50 Hz. This is satisfiable for the swarm system created for this project, but it is not scalable for a larger swarm system in a more demanding environment. It is believed that the system is halted by the Software Development Kit, as the library works as an abstraction layer, resulting in added complexity for the firmware. Measures could be taken to remove this layer in order to increase the run frequency of the algorithm, or one could introduce one or several additional processing units to create a more discrete and specialized system.

2. The in-room localization method is based on signal strength data and ranging data, and their measurements are shown to be suitable for control, albeit the signal strength values needs filtration. Another discovery was the fact that the antenna itself had a non-linear topology, this could be resolved by tuning the antenna further. The system was designed to use a magnetometer for heading control, but the current configuration resulted in too much noise to get any sensible data during motor actuation. The current localization method will perceive all neighbouring devices as an

obstacle due to the ranging method, a future implementation however could introduce a more sophisticated ranging method using direct communication in-between members of the swarm.

3. The communication protocol was designed to establish a mesh topology using the Thread protocol, MQTT-SN and the newly introduced ROMANOs. This protocol stack was designed to add little data overhead, with much needed redundancy and self-haling capabilities. The underlying protocols runs on the 802.15.4 physical layer, meaning that the algorithm could be run on a established and popular infrastructure with a large device support. The most important design philosophy of the protocol was thati t had to guarantee that each and every member of the swarm would get a steady supply of signal strength estimates, as they were further used in the control strategy. This is shown to be largely successful. A surprising result was the fact that the protocol would not support large messaging frequencies, as it would only support a frequency at 5Hz. This is most likely due to the current Thread Border Router / MQTT-SN Gateway solution.

4. The control strategy introduced a potential-field based PID controller. It was designed with the philosophy to be reactive, distributed and computationally efficient while being intuitive for the swarm application. The potential fields were based on signal strength data and ranging data, with the possibility to add heading data in the future. The PID controller would then compute a minimizing control action based on the values of the potential fields. This control strategy was shown to be successful for both signal strength control and obstacle avoidance, and by combining these together would any swarm device, if given enough time, find a local minima within its environment. This uninformed swarm behaviour is a really promising result, as it shows that the complete system would introduce some sort of *instinct* for the swarm system. It is also noted the importance of well tuned controller values, and steps should be taken in the future to further smooth out the control action and reduce the search time of a local minima. The control strategy itself is also very sensitive to noise, and possible filtrations or state estimators should be employed to increase its tolerance.

The current proposal for hardware design, in-room localization, communication and control results in a swarm system that shows great promise, but there is still much to be done from a robustness point-of-view. It is not wise to define the current system as robust. Several of the executions should be more refined and fine-tuned, and the limitation of time is apparent on some accords. However, the system itself is working well in a smaller scale. The execution of the system in a real-world scenario should be more efficient to increase its performance in order to be both scalable, and work for larger scale systems. It is believed that the current proposals for both control strategy, communication protocol, general algorithmic design and system architecture are suitable to ensure robustness within a robotic swarm in the future.

7.2 Concluding remarks

On a personal note has it been an absolute pleasure to take part in this field of research. It combines several aspects from control theory to computer science, resulting in a broad sub-field in the study of robotics. Swarm technology is without a doubt a very interesting field of study.

This study sought to find the answer of a true robust robotic swarm. Swarm technology is complex, vast, and most importantly a unique and a relatively new research area. It is non-trivial to decide which technologies should be used and what design choices to make, as there are a myriad to chose from. There are many different philosophies in how one designs and implements a robotic swarm. It is believed that this project introduces a new and interesting approach, but as is natural with a new approach does it have its difficulties, since it is an entirely new path. Surely, for the swarm robotics field are there still a lot to be done.

A personal wish is that this research will be continued in one way or the other. There have been several suggestions for future work in this project that could be researched for this swarm application, but they are exactly that; suggestions. No matter what direction one decides to take is it guaranteed to be of interest. This study has been shown to yield mixed results, but also shows promise on several occasions. It is important to understand that the work done in this project could be used as a framework for greater swarm applications, given that the necessary refinements are present.

References

[1] M. Rubenstein, A. Cornejo, and R. Nagpal, “Programmable self-assembly in a thousand-robot swarm,” *Science*, vol. 345, no. 6198, pp. 795–799, 2014. [Online]. Available: http://science.sciencemag.org/content/345/6198/795

[2] P. Ghosh, J. A. Tran, D. Dsouza, N. Ayanian, and B. Krishnamachari, “ROMANO: A novel overlay lightweight communication protocol for unified control and sensing of a network of robots,” *CoRR*, vol. abs/1709.07555, no. 1, pp. 1–6, 2017. [Online]. Available: http://arxiv.org/abs/1709.07555

[3] F. Arvin, J. Murray, C. Zhang, and S. Yue, “Colias: An autonomous micro robot for swarm robotic applications,” *International Journal of Advanced Robotic Systems*, vol. 11, no. 7, p. 113, 2014. [Online]. Available: https://doi.org/10.5772/58730

[4] M. Li, J. Harris, M. Chen, S. Mao, Y. Xiao, W. Read, and B. Prabhakaran, “Architecture and protocol design for a pervasive robot swarm communication networks,” *Wireless Communications and Mobile Computing*, vol. 11, no. 8, pp. 1092–1106, 2011. [Online]. Available: http://dx.doi.org/10.1002/wcm.856

[5] J. Penders, L. Alboul, U. Witkowski, A. Naghsh, J. Saez-Pons, S. Herbrechtsmeier, and M. El-Habbal, “A robot swarm assisting a human fire-fighter.” *Advanced Robotics*, vol. 25, no. 1/2, pp. 93 – 117, 2011. [Online]. Available: http://search.ebscohost.com/login.aspx?direct=true&db=a9h&AN=55754400&site=ehost-live

[6] K. Hattori, N. Tatebe, T. Kagawa, Y. Owada, L. Shan, K. Temma, K. Hamaguchi, and K. Takadama, “Deployment of wireless mesh network using rssi-based swarm robots,” *Artificial Life and Robotics*, vol. 21, no. 4, pp. 434–442, Dec 2016. [Online]. Available: https://doi.org/10.1007/s10015-016-0300-y

[7] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, third edition ed., M. I. of Technology, Ed. The MIT Press, 2009.

[8] K. Bjørvik and P. Hveem, *Reguleringsteknikk*, K. Forlag, Ed. Kybernetenes Forlag, Aug. 2014.

[9] C. Chen, *Linear System Theory and Design*, ser. The Oxford Series in Electrical and Computer Engineering. Oxford University Press, Incorporated, 2013. [Online]. Available: https://books.google.no/books?id=XyPAoAEACAAJ

[10] H. Khalil, *Nonlinear Systems*, third edition ed., Pearson, Ed. Ashford Colour Press, 2015.

[11] M. W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot Modeling and Control*, C. F. Shultz, Ed. John Wiley & Sons, Inc., 2006.

[12] B. Foss and T. A. N. Heirung, “Merging optimization and control,” Norwegian University of Science and Technology. Faculty of Inromation Technology, Mathematics, and Electrical Engineering. Department of Engineering Cybernetics, Tech. Rep., 2016.

[13] O. Egeland and J. T. Gravdahl, *Modeling and Simulation for Automatic Control*, corrected second printing ed., T. Trykkeri, Ed. Marine Cybernetics AS, 2003.

[14] J. Catsoulis, *Designing Embedded Hardware - Create New Computers and Devices*, 2nd ed., A. Oram, Ed. O'REILLY, 2005.

[15] T. W. Dunbar and J. M. Esposito, "Artificial potential field controllers for robust communications in a network of swarm robots," in *Proceedings of the Thirty-Seventh Southeastern Symposium on System Theory, 2005. SSST '05.*, March 2005, pp. 401–405.

[16] A. O. de Sá, N. Nedjah, and L. de Macedo Mourelle, "Distributed efficient localization in swarm robotics using min–max and particle swarm optimization," *Expert Systems with Applications*, vol. 50, no. Supplement C, pp. 55 – 65, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S095741741500809X

[17] J. Timmis, A. Ismail, J. Bjerknes, and A. Winfield, "An immune-inspired swarm aggregation algorithm for self-healing swarm robotic systems," *Biosystems*, vol. 146, no. Supplement C, pp. 60 – 76, 2016, information Processing in Cells and Tissues. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S030326471630034X

[18] Zigbee Alliance, *ZigBee Specification*, rev. 20 ed., ZigBee Alliance, Sep. 2012. [Online]. Available: http://www.zigbee.org/wp-content/uploads/2014/11/docs-05-3474-20-0csg-zigbee-specification.pdf

[19] Thread Group, *Thread Technical White Paper*, Thread Group, Jul. 2015, rev. 2.0. [Online]. Available: https://portal.threadgroup.org/DesktopModules/Inventures_Document/FileDownload.aspx?ContentID=633

[20] Bluetooth SIG, *Bluetooth Core Specification v 5.0*, Bluetooth Special Interest Group, Dec. 2016, rev. 5.0. [Online]. Available: https://www.bluetooth.org/DocMan/handlers/DownloadDoc.ashx?doc_id=421043

[21] S. Russell and P. Norvig, *Artificial Intelligence - A Modern Approach*, M. (CTP-VVP), Ed. Pearson Education Limited, 2016.

[22] H. M. Halvorsen, "Robust control and communication algorithm for robotic swarms using the nrf52 system on chip," *Specialization Project*, vol. 1, no. 1, pp. 1–35, Dec. 2017.

[23] (2017) Robot operating system. [Information accessed 9. November 2017]. [Online]. Available: http://www.ros.org/

[24] A. Stanford-Clark and H. L. Truong, *MQTT For Sensor Networks (MQTT-SN)*, version 1.2 ed., International Business Machines Corporation, Nov. 2013. [Online]. Available: http://mqtt.org/new/wp-content/uploads/2009/06/MQTT-SN_spec_v1.2.pdf

[25] M. H. Amaran, N. A. M. Noh, M. S. Rohmad, and H. Hashim, "A comparison of lightweight communication protocols in robotic applications," *Procedia Computer Science*, vol. 76, no. Supplement C, pp. 400 – 405, 2015, 2015 IEEE International Symposium on Robotics and Intelligent Sensors (IEEE IRIS2015). [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877050915038193

[26] Y. Rasekhipour, A. Khajepour, S. K. Chen, and B. Litkouhi, "A potential field-based model predictive path-planning controller for autonomous road vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 5, pp. 1255–1267, May 2017.

[27] (2018) Altium designer 18. Online. [Information accessed 20. May 2018]. [Online]. Available: https://www.altium.com/

[28] (2018) Rfx2 product description - bench-top antenna measurement equipment. Online. [Information accessed 10. May 2018]. [Online]. Available: https://www.emscan.com/products/antenna-testing/

[29] (2018) Onshape - modern cad. Online. [Information accessed 20. May 2018]. [Online]. Available: https://www.onshape.com/

[30] (2018) Atom - a hackable text editor for the 21st century. Online. [Information accessed 20. May 2018]. [Online]. Available: https://atom.io/

[31] (2018) Segger embedded studio | the cross platform ide by segger. Online. [Information accessed 20. May 2018]. [Online]. Available: https://www.segger.com/products/development-tools/embedded-studio/

[32] (2018) Nordic semiconductor infocenter - nrf5 sdk for tread v0.11.0. Online. [Information accessed 20. May 2018]. [Online]. Available: http://infocenter.nordicsemi.com/index.jsp?topic=%2Fcom.nordic.infocenter.threadsdk.v0.11.0%2Findex.html

[33] (2018) Node.js. Online. [Information accessed 6. May 2018]. [Online]. Available: https://nodejs.org/en/

[34] (2018) Get started with sql server. Online. [Information accessed 6. May 2018]. [Online]. Available: https://www.microsoft.com/en-us/sql-server/developer-get-started/

[35] M. Sauter, *From GSM to LTE - An introduction to mobile networks and mobile broadband*. John John Wiley & Sons, Ltd., 2011. [Online]. Available: https://books.google.no/books?id=uso-6LN2YjsC&printsec=frontcover&hl=no#v=onepage&q&f=false

[36] Nordic Semiconductor, *nRF52840 Objective Product Specification v0.5*, version 0.5 ed., May 2016. [Online]. Available: http://infocenter.nordicsemi.com/pdf/nRF52840_PS_v1.0.pdf

[37] (2016) Understanding and implementing crc (cyclic redundancy check). Online. [Information accessed 28. March 2018]. [Online]. Available: http://www.sunshine2k.de/articles/coding/crc/understanding_crc.html

[38] Bluetooth SIG, *Bluetooth Mesh Specification*, Bluetooth Special Interest Group, Jul. 2017, rev. 1.0. [Online]. Available: https://www.bluetooth.org/docman/handlers/downloaddoc.ashx?doc_id=429633

[39] Thread Group Inc., *Thread Specification*, 1st ed., Feb. 2017.

[40] Silicon Labs, *UG103.11: Thread Fundamentals*, rev. 0.7 ed., Silicon Labs, 2015. [Online]. Available: https://www.silabs.com/documents/public/user-guides/ug103-11-appdevfundamentals-thread.pdf

[41] Z. Shelby, Arm, K. Hartke, and C. Bormann, *RFC 7252 - The Constrained Application Protocol (CoAP)*, Internet Engineering Task Force (IETF), Jun. 2014. [Online]. Available: https://tools.ietf.org/html/rfc7252

[42] (1999, Jun.) Hypertext transfer protocol – http/1.1. Online. [Information accessed 20. May 2018]. [Online]. Available: https://tools.ietf.org/html/rfc2616

[43] OASIS Open, *MQTT Version 3.1.1*, Oct. 2014, version 3.1.1. [Online]. Available: http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.pdf

[44] (2018) Hivemq - enterprise mqtt broker. Online. [Information accessed 31. March 2018]. [Online]. Available: https://www.hivemq.com/company/

[45] A. S. Sedra and K. C. Smith, *Microelectronic Circuits*, 5th ed. Oxford University Press, 2004.

[46] R. L. Boylestad, *Introductory Circuit Analysis*, 12th ed. Upper Saddle River, NJ, USA: Prentice Hall Press, 2010.

[47] N. Mohan, T. M. Undeland, and W. P. Robbins, *Power Electronics. Converters, Applications and Design*, 3rd ed. John Wiley and Sons, Inc, 2003.

[48] (2018) Arducam usb3.0 camera shield. Online. [Information accessed 20. May 2018]. [Online]. Available: http://www.arducam.com/

[49] (2018) Pixy (cmucam5). Online. [Information accessed 20. May 2018]. [Online]. Available: http://charmedlabs.com/default/pixy-cmucam5/

[50] VISHAY, *IR Receiver Modules for Remote Control Systems*, rev. 1.3 ed., Jan. 2009. [Online]. Available: https://www.sparkfun.com/datasheets/Sensors/Infrared/tsop382.pdf

[51] G. Benet, F. Blanes, J. Simó, and P. Pérez, "Using infrared sensors for distance measurement in mobile robots," *Robotics and Autonomous Systems*, vol. 40, no. 4, pp. 255 – 266, 2002. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0921889002002713

[52] Elec Freaks, *Ultrasonic Ranging Module HC - SR04 - Datasheet*, version 1.0 ed., Elec Freaks, 2015, information accessed 2018-08-04. [Online]. Available: https://www.electroschematics.com/wp-content/uploads/2013/07/HCSR04-datasheet-version-1.pdf

[53] (2018) Laser sensors for displacement, distance and position. Online. [Information accessed 8. April 2018]. [Online]. Available: https://www.micro-epsilon.com/displacement-position-sensors/laser-sensor/

[54] (2018) World smallest time-of-flight ranging and gesture detection sensor - vl53l0x datasheet. Online. [Information accessed 8. April 2018]. [Online]. Available: http://www.st.com/resource/en/datasheet/vl53l0x.pdf

[55] (2018) Lidar—light detection and ranging—is a remote sensing method used to examine the surface of the earth. Online. [Information accessed 20. May 2018]. [Online]. Available: https://www.webcitation.org/6H82i1Gfx?url=http://oceanservice.noaa.gov/facts/lidar.html

[56] K. Hemanth, V. Talasila, and S. Rao, "Calibration of 3-axis magnetometers," *IFAC Proceedings Volumes*, vol. 45, no. 1, pp. 175 – 178, 2012, 1st IFAC Workshop on Embedded Guidance, Navigation and Control in Aerospace. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1474667015350400

[57] D. Gebre-egziabher, G. H. Elkaim, J. D. Powell, and B. W. Parkinson, "Calibration of strapdown magnetometers in magnetic field domain," *J. Aerosp. Eng*, vol. 1, no. 1, pp. 87–102, 2010.

[58] K. Winer. (2018) Simple and effective magnetometer calibration. Online. [Information accessed 20. May 2018]. [Online]. Available: https://github.com/kriswiner/MPU6050/wiki/Simple-and-Effective-Magnetometer-Calibration

[59] P. He, P. Cardou, A. Desbiens, and E. Gagnon, "Estimating the orientation of a rigid body moving in space using inertial sensors," *Multibody System Dynamics*, vol. 35, no. 1, pp. 63–89, Sep 2015. [Online]. Available: https://doi.org/10.1007/s11044-014-9425-8

[60] (2018) Mojo desktop printer - mojo magic at your desk. Online. [Information accessed 20. May 2018]. [Online]. Available: http://www.stratasys.com/3d-printers/mojo

[61] (2018) Eclipse mosquitto - an open source mqtt broker. Online. [Information accessed 31. May 2018]. [Online]. Available: http://mosquitto.org/

[62] (2018) Thread border router - nordic semiconductor infocenter. Online. [Information accessed gotten 6. May 2018]. [Online]. Available: https://infocenter.nordicsemi.com/index.jsp?topic=%2Fcom.nordic.infocenter.threadsdk.v0.9.0%2Fthread_border_router.html

[63] (2016) Iot eclipse - getting started. Online. [Information accessed 3. June 2018]. [Online]. Available: https://iot.eclipse.org/getting-started

[64] Ecma International, *The JSON Data Interchange Syntax*, 2nd ed., Ecma International, Dec. 2017. [Online]. Available: http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf

[65] (2018) Saleae logic analysers. Online. [Information accessed 10. May 2018]. [Online]. Available: https://www.saleae.com/

[66] Johanson Technology, *Johanson 2.4GHz SMD Antenna, Edge Mount Design*, ver 3.0 ed., 2015. [Online]. Available: https://www.johansontechnology.com/datasheets/antennas/2450AT18D0100.pdf

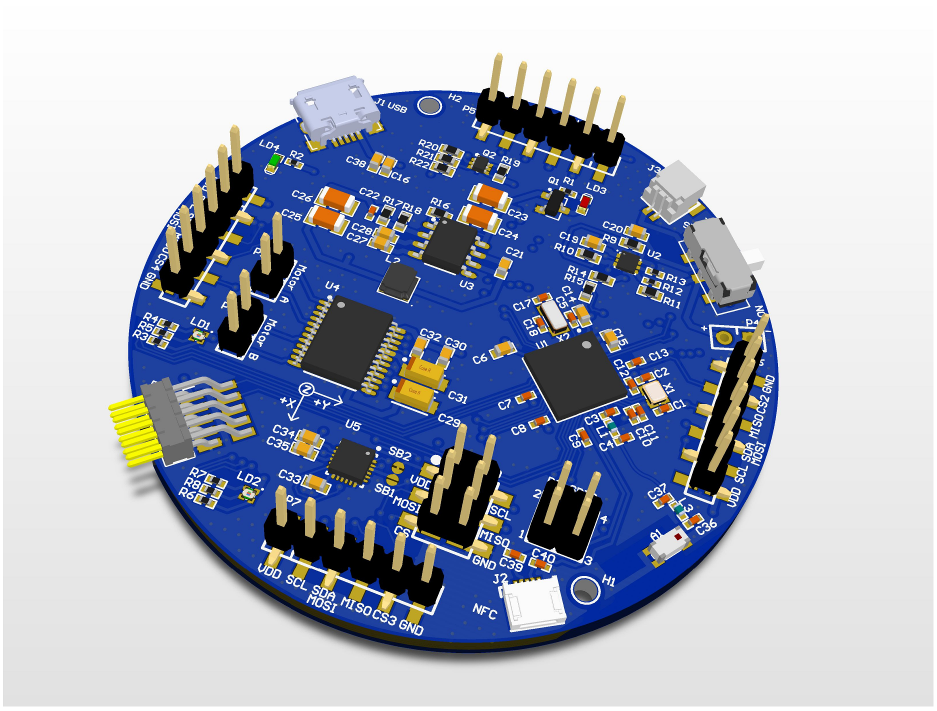
[67] Nordic Semiconductor, "Antenna tuning - nwp-017 white paper," *Infocenter*, vol. 1.0, no. 1, pp. 3–38, 2012. [Online]. Available: http://infocenter.nordicsemi.com/pdf/nwp_017.pdf

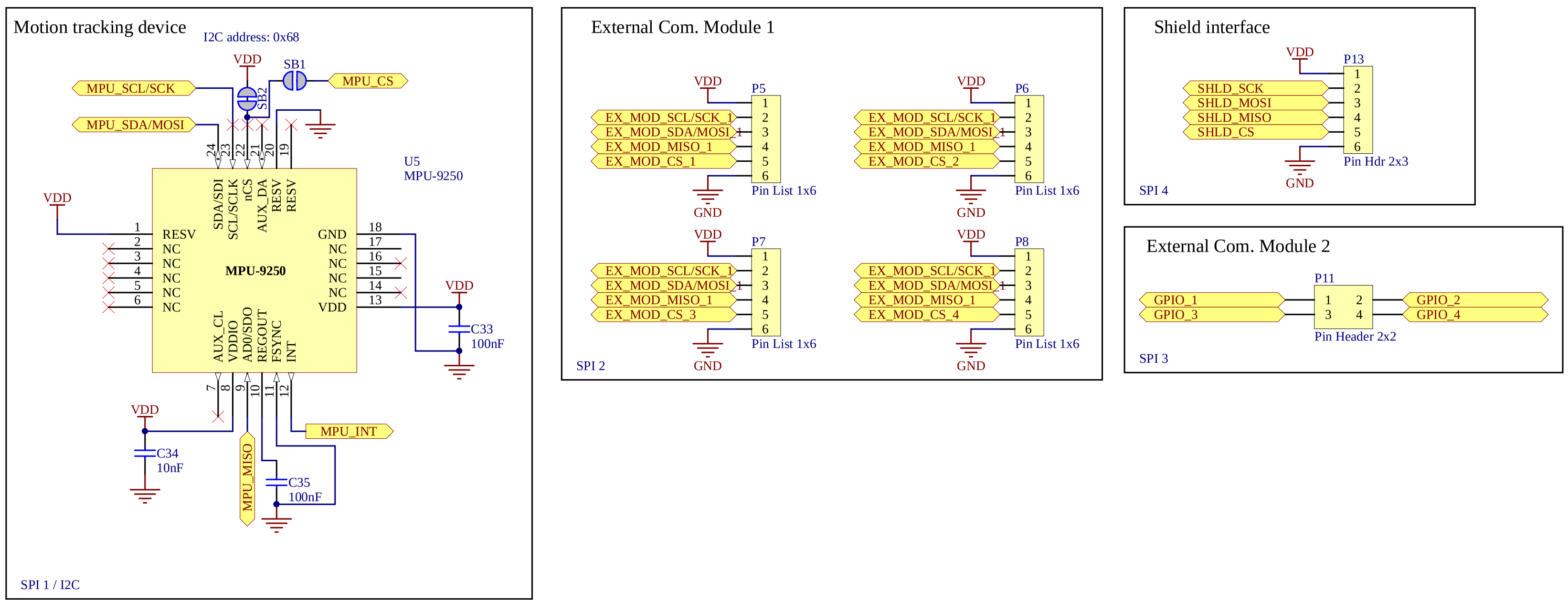
[68] (2018) Thread network topology monitor - nordic semiconductor infocenter. Online. [Information accessed 16. May 2018]. [Online]. Available: http://infocenter.nordicsemi.com/index.jsp?topic=%2Fcom.nordic.infocenter.threadsdk.v0.11.0%2Fthread_topology_monitor.html&cp=4_2_1_1_10

[69] M. Bodi, R. Thenius, M. Szopek, T. Schmickl, and K. Crailsheim, "Interaction of robot swarms using the honeybee-inspired control algorithm beeclust," *Mathematical and Computer Modelling of Dynamical Systems*, vol. 18, no. 1, pp. 87–100, 2012. [Online]. Available: https://doi.org/10.1080/13873954.2011.601420

Appendices

Appendix 1 - Main board schematics

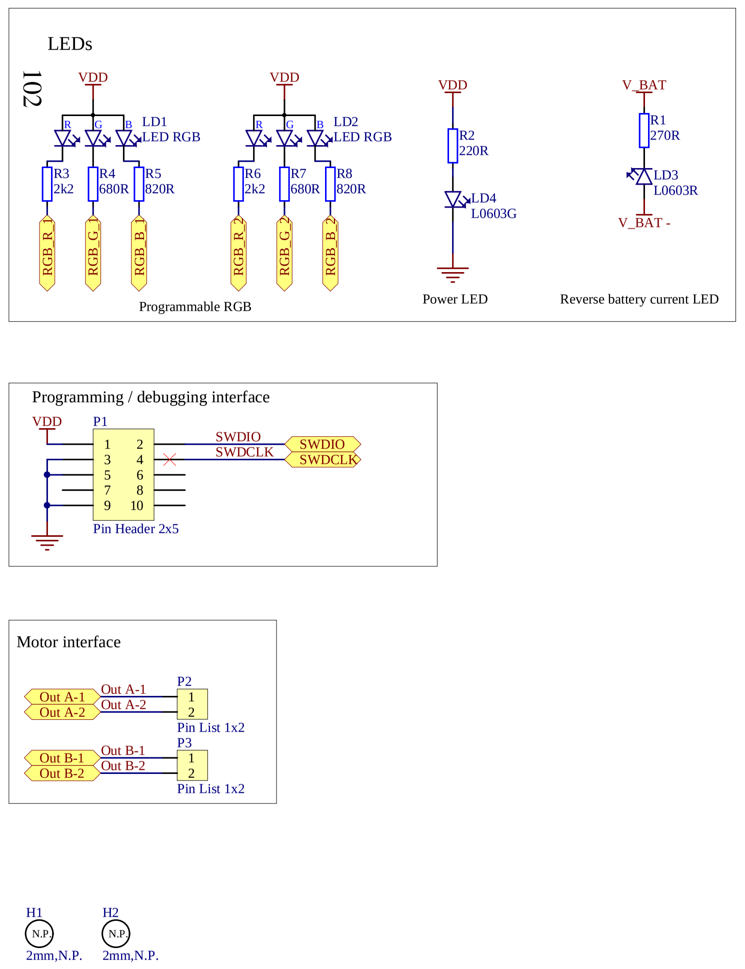




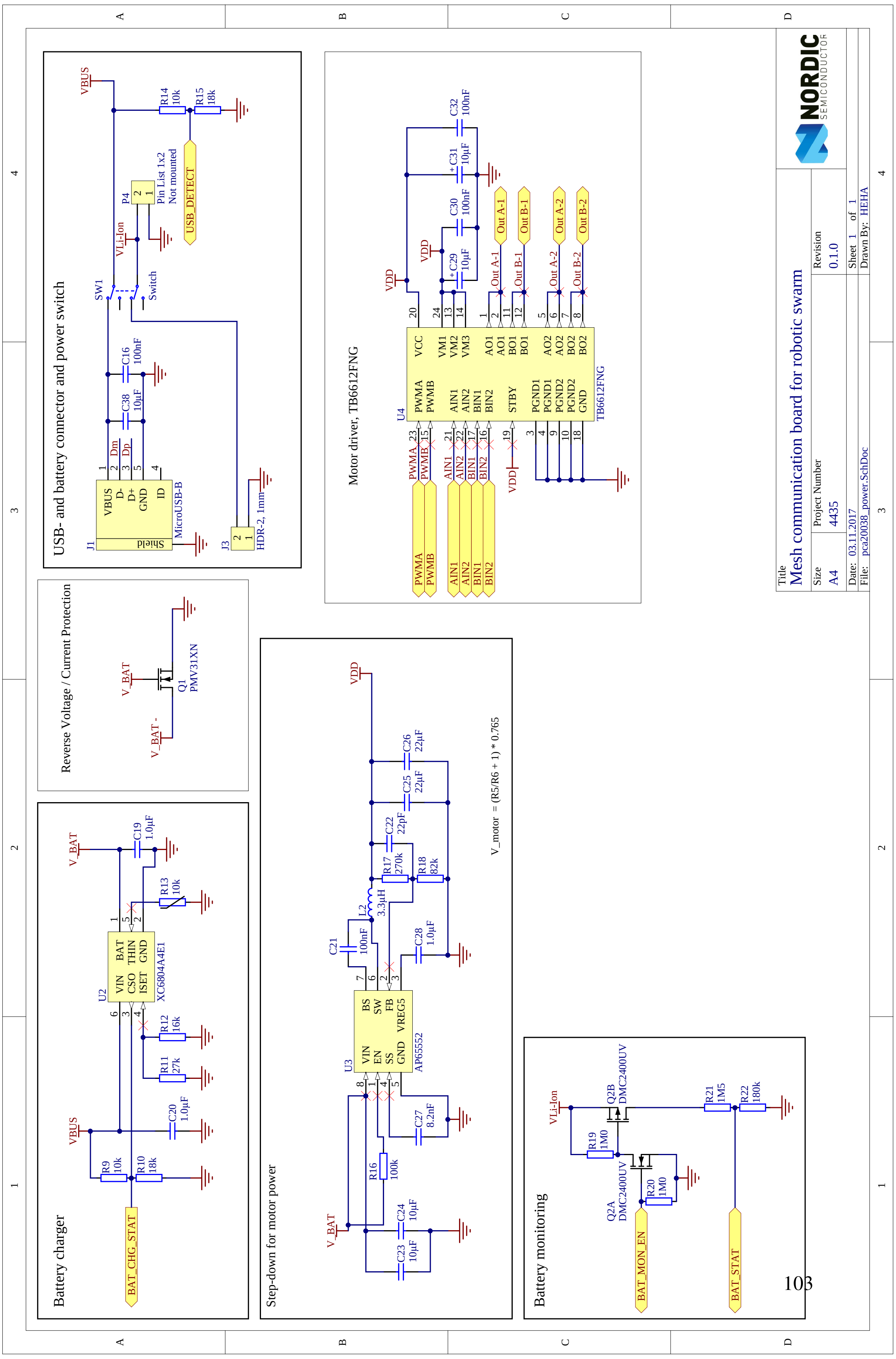
Title		
Mesh communication board for robotic swarm		
Size	Project Number	Revision
A4	4435	0.1.0
Date: 03.11.2017		Sheet 1 of 1
File: pca20038_sensory.SchDoc		Drawn By: HEHA

NORDIC SEMICONDUCTOR

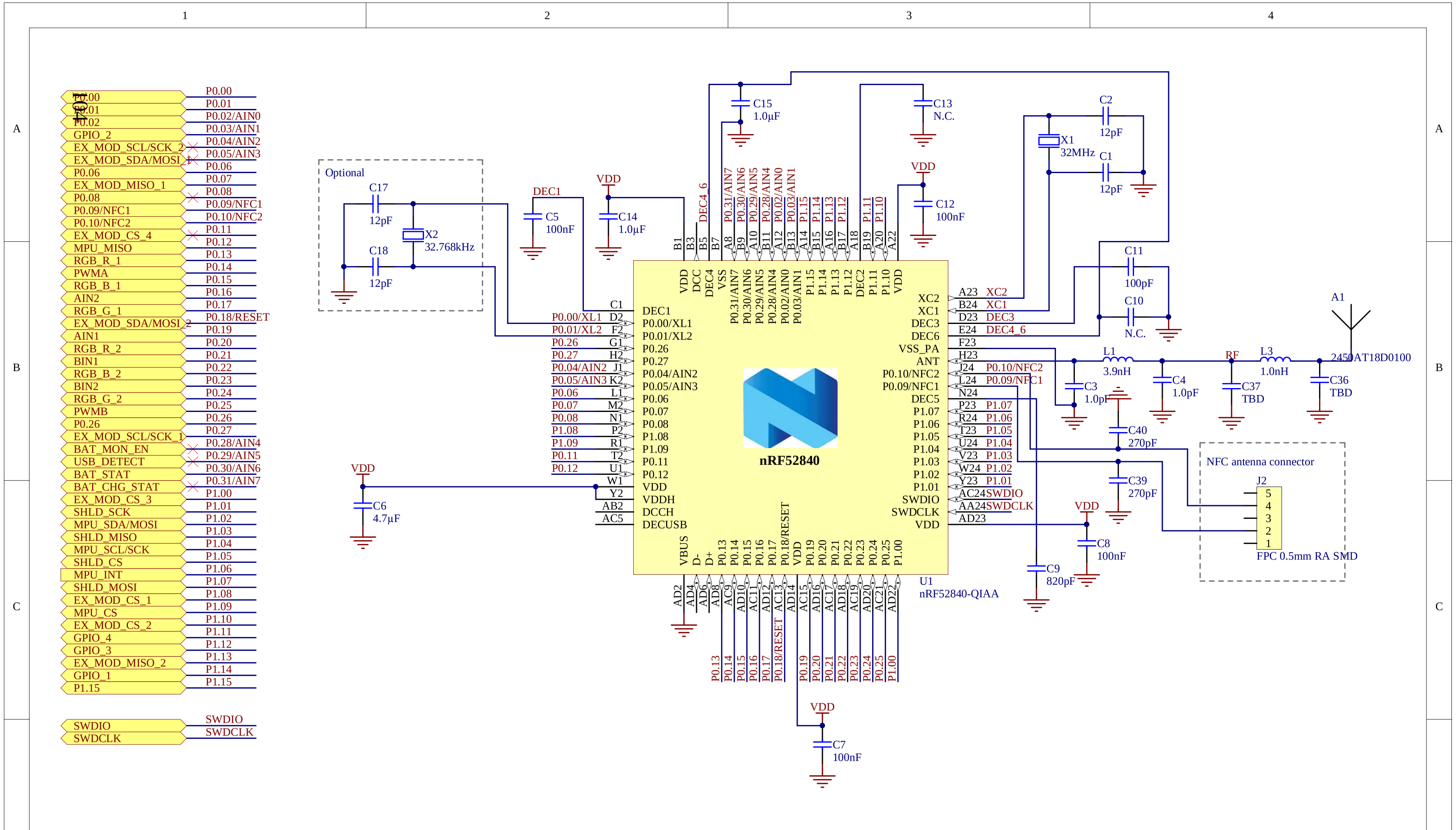
102



Title		
Mesh communication board for robotic swarm		
Size: A4	Project Number: 4435	Revision: 0.1.0
Date: 03.11.2017		Sheet 1 of 1
File: pca20038_misc.SchDoc		Drawn By: HEHA



USB- and battery connector and power switch
Battery charger
Reverse Voltage / Current Protection
Step-down for motor power
Motor driver, TB6612FNG
Battery monitoring
V_motor = (R5/R6 + 1) * 0.765
J1 MicroUSB-B
SW1 Switch
P4 Pin List 1x2 Not mounted
J3 HDR-2, 1mm
U2 XC6804A4E1
Q1 PMV31XN
U3 AP65552
U4 TB6612FNG
Q2A DMC2400UV
Q2B DMC2400UV
VBUS
V_BAT
V_BAT -
VDD
VLi-Ion
USB_DETECT
BAT_CHG_STAT
BAT_MON_EN
BAT_STAT
PWMA
PWMB
AIN1
AIN2
BIN1
BIN2
Out A-1
Out B-1
Out A-2
Out B-2
Title
Mesh communication board for robotic swarm
Size A4
Project Number 4435
Revision 0.1.0
Date: 03.11.2017
Sheet 1 of 1
File: pca20038_power.SchDoc
Drawn By: HEHA



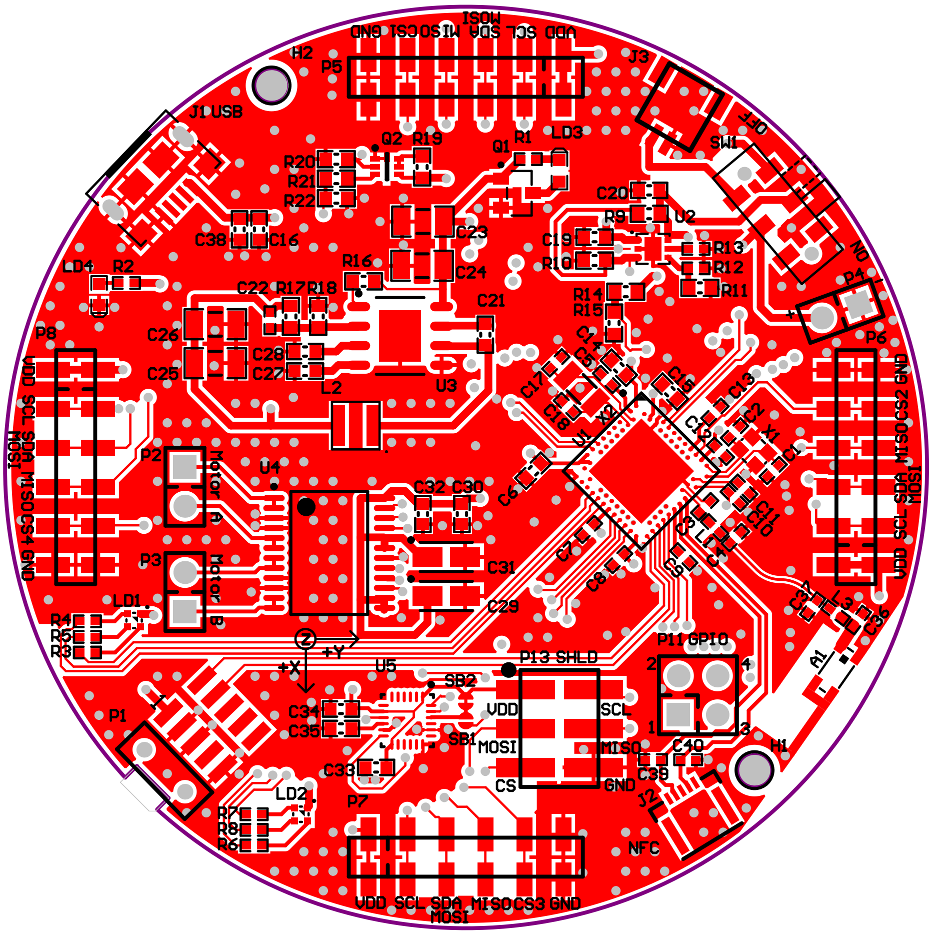
104

Title
Mesh communication board for robotic swarm

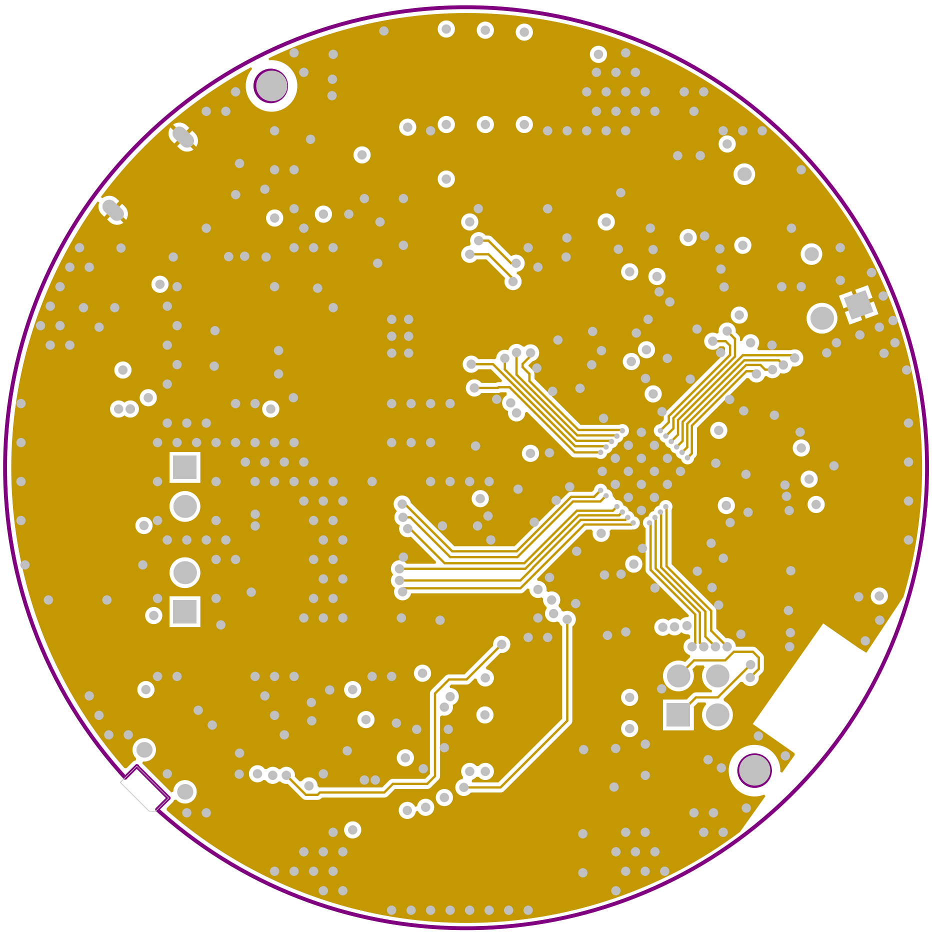
Size	Project Number	Revision
A4	4435	0.1.0

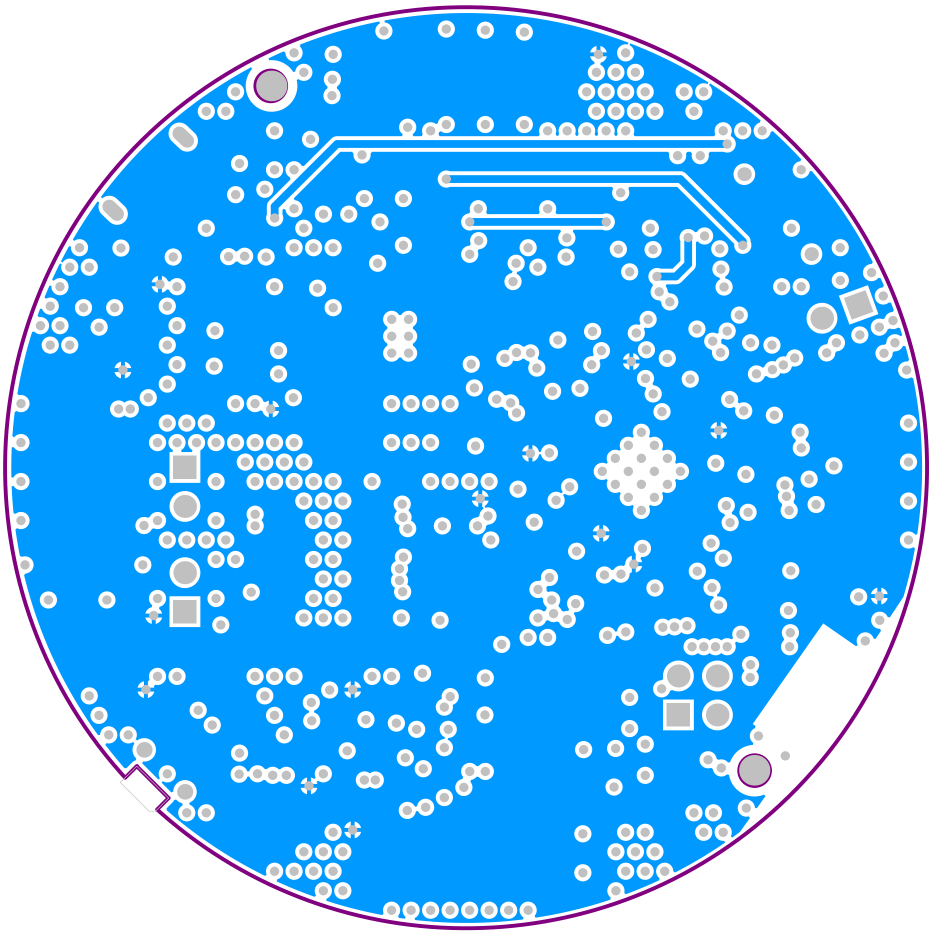
Date: 03.11.2017 Sheet 1 of 1
File: pca20038_mcu.SchDoc Drawn By: HEHA

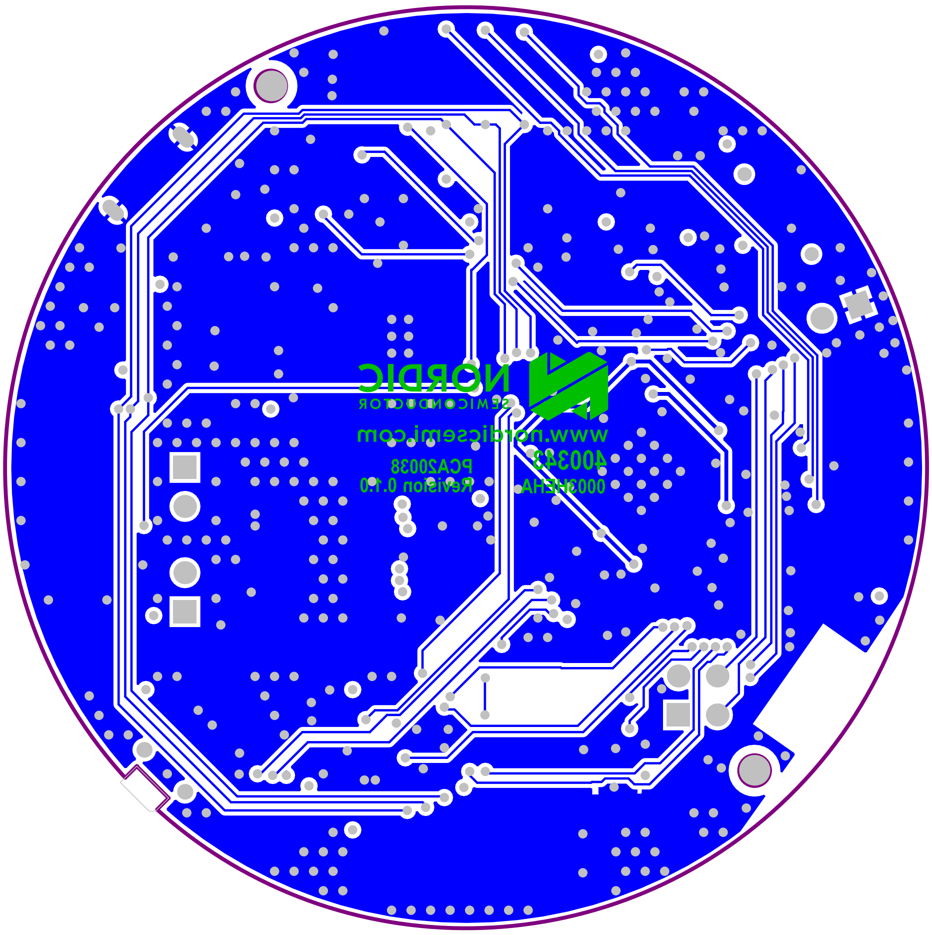
NORDIC SEMICONDUCTOR



GND CS1 MISO MOSI SDA SCL VDD
H2
P5
J3
J1 USB
OFF
Q2 R19
Q1 R1 LD3
SW1
R20
R21
R22
C20
R9 U2
C38 C16
C23
C19
R10
R13
R12
ON
LD4 R2
R16
C24
R11
P4
C22 R17 R18
R14
C21
R15
+
P8
C26
C14
P6
VDD SCL SDA MISO CS4 GND
MOSI
C28
C25
C27
L2
U3
C17
C5
C15
C13
C18
X2
C12
C2
X1
C1
U1
GND CS2 MISO SDA SCL VDD
MOSI
P2
Motor A
U4
C6
C32 C30
C11
C10
C3
C7
L1
C4
C9
C8
C31
P3
Motor B
C29
LD1
C37
L3
C36
R4
R5
R3
Z
+Y
+X
U5
P13 SHLD
P11 GPIO
2
4
A1
SB2
VDD
SCL
1
3
C34
C35
SB1
MOSI
MISO
C40
H1
P1
C33
C39
CS
GND
J2
LD2
P7
R7
R8
R6
NFC
VDD SCL SDA MISO CS3 GND
MOSI






NORDIC
SEMICONDUCTOR
www.nordicsemi.com
400343
PCA20038
Revision 0.1.0

The following appendices are submitted digitally:

Appendix 2 - Robot chassis 3D model file

Appendix 3 - Firmware code

Appendix 4 - Demonstration video