

Distributed formation control of non-holonomic mobile robots

Implementation and testing in Robot Operating System

X.O.S. van der Geest

Master of Science Thesis



Distributed formation control of non-holonomic mobile robots

Implementation and testing in Robot Operating System

MASTER OF SCIENCE THESIS

For the degree of Master of Science in Systems and Control at Delft
University of Technology

X.O.S. van der Geest

September 30, 2020

Faculty of Mechanical, Maritime and Materials Engineering (3mE) · Delft University of
Technology

DELFT UNIVERSITY OF TECHNOLOGY
DEPARTMENT OF
DELFT CENTER FOR SYSTEMS AND CONTROL (DCSC)

The undersigned hereby certify that they have read and recommend to the Faculty of
Mechanical, Maritime and Materials Engineering (3mE) for acceptance a thesis
entitled

DISTRIBUTED FORMATION CONTROL OF NON-HOLONOMIC MOBILE ROBOTS

by

X.O.S. VAN DER GEEST

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE SYSTEMS AND CONTROL

Dated: September 30, 2020

Supervisor(s):

Assoc. prof. dr. ir. S. Grammatico

Reader(s):

Assoc. prof. dr. ir. J. Alonso-Mora

Ir. S. Krilašević

Abstract

The use of distributed multi-agent formation control for non-holonomic mobile robots shows great promise in practical environments. In this thesis, a Robot Operating System (ROS) package is developed in which various formation control algorithms are implemented for use on a group of skid-steer drive mobile robots. These formation control algorithms are tested in a simulated laboratory environment. For this, first seven distinct ways to adapt holonomic (omnidirectional) formation control algorithms to be used on skid-steer drive, differential drive or unicycle model mobile robots are presented. These seven adaptations can also be used to categorise existing non-holonomic formation control algorithms for unicycle model mobile robots. These adaptations are used to implement three formation control algorithms in a ROS package. Furthermore, obstacle avoidance and goal reaching is added to the package. Control inputs are generated from objects detected in the local coordinate frame of the robots, by means of the 2D LIght Detection And Ranging (LIDAR) and an open source obstacle detection package. The formation control package and the implemented formation control algorithms are tested on a group of skid-steer drive mobile robots, the Husarion ROSbot, in the Gazebo simulation environment. Simulations show the effectiveness of the developed formation control algorithms. Various conditions regarding control frequency and LIDAR noise levels are evaluated, and finally, a number of practical recommendations for the use of formation control algorithms on non-holonomic mobile robots are made.

Table of Contents

Preface	xiii
1 Introduction	1
2 Formation control algorithms	3
2-1 Classes of formation control	3
2-2 Distance based control algorithms	4
2-3 Adapting holonomic gradient-based control to teams of unicycle model robots . .	5
2-3-1 Non-holonomic gradient-based control	5
2-3-2 Simulation of the seven adaptations	9
2-3-3 Conclusions	11
3 Implemented algorithms	15
3-1 Formation keeping and obstacle avoidance	15
3-1-1 Virtual spring-damper system	16
3-1-2 Consensus algorithm	17
3-1-3 Exponential potential algorithm	18
3-2 Moving towards goal	19
4 Simulation environment	21
4-1 ROS structure	21
4-1-1 Robot identification	21
4-2 Identification of practical limitations	22
4-2-1 LIDAR	23
4-2-2 Control frequency	24
4-3 Simulation	24
4-3-1 Test scenario	24
4-3-2 Technical set-up	25

5	Results	27
5-1	Virtual spring-damper system	28
5-1-1	Ideal situation: using global distance measurements and running the skid-steer drive controller at 100 Hz	28
5-1-2	Using LIDAR for distance measurements and running the skid-steer drive controller at 100 Hz	29
5-1-3	Using global distance measurements and running the skid-steer drive controller at 15Hz	31
5-1-4	Using the LIDAR for distance measurements and running the skid-steer drive controller at 15 Hz	31
5-1-5	Finding acceptable LIDAR noise levels	34
5-2	Consensus Algorithm	35
5-2-1	Ideal situation: using global distance measurements and running the skid-steer drive controller at 100 Hz	35
5-2-2	Using LIDAR for distance measurements and running the skid-steer drive controller at 100 Hz	35
5-2-3	Using global distance measurements and running the skid-steer drive controller at 15 Hz	36
5-2-4	Using the LIDAR for distance measurements and running the skid-steer drive controller at 15 Hz	37
5-2-5	Finding acceptable LIDAR noise levels	40
5-3	Exponential potential function	42
5-3-1	Ideal situation: using global distance measurements and running the skid-steer drive controller at 100 Hz	42
5-3-2	Using LIDAR for distance measurements and running the skid-steer drive controller at 100 Hz	42
5-3-3	Using global distance measurements and running the skid-steer drive controller at 15 Hz	44
5-3-4	Using the LIDAR for distance measurements and running the skid-steer drive controller at 15 Hz	44
5-3-5	Finding acceptable LIDAR noise levels	47
6	Conclusions and Recommendations	49
6-1	Conclusions	49
6-1-1	Adapting holonomic control laws for use on non-holonomic robots	49
6-1-2	Effects of LIDAR distance measurements on formation control performance	50
6-1-3	Influence of control input frequency on object detection	51
6-1-4	Acceptable LIDAR noise levels	51
6-1-5	Main conclusion	52
6-2	Discussion	53
6-2-1	Integral Squared Error (ISE)	53
6-2-2	Obstacle detection	53
6-2-3	Tuning of the controllers	54
6-2-4	Identification of robots and obstacles	54
6-3	Recommendations	54
6-3-1	Adapting holonomic formation control algorithms	54

6-3-2	LIDAR noise	54
6-3-3	Control frequency	55
6-3-4	Choosing a formation control algorithm for the ROSbot	55
6-3-5	Improving performance	55
6-3-6	Future research	56
A	Simulation Environment	57
A-1	Robot Operating System (ROS)	57
A-2	Gazebo	58
A-3	Husarion ROSbot	58
A-3-1	ROSbot kinematics	59
A-4	LIDAR	60
A-5	Object Detector	61
A-6	Order of the system	61
	Bibliography	63
	Glossary	67
	List of Acronyms	67
	List of Symbols	67

List of Figures

2-1	Overview of the three different classes of formation control and the coordinate frames corresponding to each of these classes.	4
2-2	Determination of λ when the force drives the robot forward.	8
2-3	Determination of λ when the force drives the robot backwards, but is still taken positive.	8
2-4	Determination of λ when the force drives the robot backwards.	8
2-5	Sign of the rotational input when a potential "pulls" the robot towards the different quadrants.	10
2-6	Paths taken by the robot, reacting to the potential in Eq. (2-20), moving to the desired position where the distance to the other two static agents is 1 m.	12
2-6	Continued.	13
3-1	Mobile robots connected by virtual springs and dampers. Adapted from Wiech et al. [1]	17
3-2	Localisation of a weak source as proposed by Renzaglia and Briñón-Arranz [2]. Adapted from Renzaglia and Briñón-Arranz [2].	20
4-1	ROS nodes in each of the robots, and inter-robot communications.	22
4-2	Simulation scenario with LIDAR rays and sensed objects.	25
5-1	Paths of robots and distances between the robots, 100 Hz, global distance measurements	29
5-2	Distance errors and cumulative Integral Squared Error (ISE) for the group of robots using the spring-damper algorithms at 100 Hz, using global distance measurements.	29
5-3	Paths of robots and distances between the robots, 100 Hz, LIDAR distance measurements with 0.01 m noise standard deviation.	30
5-4	Distance errors and cumulative ISE for the group of robots using the spring-damper algorithms at 100 Hz, using LIDAR distance measurements with 0.01 m noise standard deviation.	30
5-5	Paths of robots and distances between the robots, running the skid-steer drive controller at 15 Hz, using global distance measurements.	31

5-6	Distance errors and cumulative ISE for the group of robots using the spring-damper algorithms at 15 Hz, using global distance measurements.	32
5-7	Paths of robots and distances between the robots, running the skid-steer drive controller at 15 Hz, using LIDAR measurements with 0.01 m noise standard deviation.	32
5-8	Distance errors and cumulative ISE for the group of robots using the spring-damper algorithms at 15 Hz, using LIDAR measurements with 0.01 m noise standard deviation.	33
5-9	Boxplots of ISE for 10 runs with the skid-steer drive controller running at 100 Hz, first with global distance measurements and then with LIDAR distance measurements with different standard deviations for LIDAR noise.	34
5-10	Boxplots of ISE for 10 runs with the skid-steer drive controller running at 15 Hz, first with global distance measurements and then with LIDAR distance measurements with different standard deviations for LIDAR noise.	34
5-11	Paths of robots and distances between the robots, using the consensus algorithm in the ideal situation of the skid-steer drive controller running at 100 Hz and using global distance measurements.	36
5-12	Distance errors and cumulative ISE for the group of robots using the consensus algorithms at 100Hz, using global positions for distance measurements.	36
5-13	Paths of robots and distances between the robots, using the consensus algorithm running the skid-steer drive controller at 15 Hz, and using LIDAR distance measurements with 0.01 m noise standard deviation.	37
5-14	Distance errors and cumulative ISE for the group of robots using the consensus algorithms at 100 Hz, using LIDAR distance measurements with noise standard deviation of 0.01 for distance measurements with 0.01 m noise standard deviation.	37
5-15	Paths of robots and distances between the robots, using the Consensus algorithm. 15Hz, global distance measurements.	38
5-16	Distance errors and cumulative ISE for the group of robots using the consensus algorithms at 15Hz, using global distance measurements.	38
5-17	Paths of robots and distances between the robots, using the consensus algorithm in the "real" scenario, with the skid-steer drive controller running at 15 Hz and using LIDAR distance measurements with 0.01 m noise standard deviation.	39
5-18	Distance errors and cumulative ISE for the group of robots using the consensus algorithms at 15 Hz, using LIDAR with noise standard deviation of 0.01 for distance measurements with 0.01 m noise standard deviation.	39
5-19	Boxplots of ISE for 10 runs with the skid-steer drive controller running at 100 Hz, first with global distance measurements and then with LIDAR distance measurements with different standard deviations for LIDAR noise.	40
5-20	Boxplots of ISE for 10 runs with the skid-steer drive controller running at 15 Hz, first with global distance measurements and then with LIDAR distance measurements with different standard deviations for LIDAR noise.	41
5-21	Paths of robots and distances between the robots, using the exponential algorithm. In this ideal scenario, the skid-steer drive controller is running at 100 Hz and global distance measurements are used.	42
5-22	Distance errors and cumulative ISE for the group of robots using the exponential algorithms at 100 Hz, using global distance measurements.	43
5-23	Paths of robots and distances between the robots, using the exponential algorithm. In this scenario the skid-steer drive controller is running at 100 Hz and LIDAR distance measurements with 0.01 m noise standard deviation are used.	43
5-24	Distance errors and cumulative ISE for the group of robots using the exponential algorithms at 100 Hz, using LIDAR with noise standard deviation of 0.01 m for distance measurements.	44

5-25	Paths of robots and distances between the robots, using the exponential algorithm. In this scenario the skid-steer drive controller is running at 15 Hz and global distance measurements are used.	45
5-26	Distance errors and cumulative ISE for the group of robots using the exponential algorithms at 15 Hz, using global distance measurements.	45
5-27	Paths of robots and distances between the robots, using the exponential algorithm. In this scenario the skid-steer drive controller is running at 15 Hz and LIDAR distance measurements with 0.01 m noise standard deviation are used.	46
5-28	Distance errors and cumulative ISE for the group of robots using the exponential algorithms at 15 Hz, using LIDAR distance measurements with noise standard deviation of 0.01 m for distance measurements.	46
5-29	Boxplots of ISE for 10 runs with the skid-steer drive controller running at 100 Hz, first with global distance measurements and then with LIDAR distance measurements with different standard deviations for LIDAR noise.	47
5-30	Boxplots of ISE for 10 runs with the skid-steer drive controller running at 15 Hz, first with global distance measurements and then with LIDAR distance measurements with different standard deviations for LIDAR noise.	47
A-1	Schematic of ROS communication. Adapted from Achmad et al. [3].	57
A-2	The Husarion ROSbot [4]	59
A-3	The four wheels of the ROSBot are simplified to two virtual wheels [5]	60
A-4	Fraction of measurements within $\pm 1\%$ at different distances for some standard deviation	61
A-5	Obstacle detection by the right robot.	62

List of Tables

4-1	Initial positions of robots and obstacles in the simulated scenario.	24
5-1	Control gains for the virtual spring-damper system.	28
5-2	Control gains for consensus system.	35
5-3	Control gains for exponential potential function.	44

Preface

This thesis is the final work to receive the Master of Science degree in Systems & Control at the Delft University of Technology.

Firstly, I would like to thank my supervisor, Associate professor dr. ir. Sergio Grammatico, for the opportunity to work in the DCSC lab with the group of ROSbots, and the feedback throughout the project. Unfortunately, because of the global pandemic, laboratory activities could not be continued. It would have been very cool to see the real ROSbots moving in formation while avoiding obstacles.

I would also like to thank my friends and family, for providing the necessary distractions, and Nelly and Eline specifically, for reading the thesis and providing feedback on the report.

Lastly, Marlein, thank you for supporting me throughout the process, for your very thorough remarks, and for bringing out the best in me. Without you, the final result would not have been the same.

X.O.S. van der Geest
Rotterdam, September 2020

Chapter 1

Introduction

In nature, large groups of animals often exhibit forms of collective behaviour. This allows for complex behaviour of the group, which can not be achieved by individual group members [6, 7]. Fish and birds, specifically, often move in some sort of formation. This behaviour serves as a way to evade predators, save energy, hunt for prey, or has other advantages for the group.

In the field of robotics, formation control may have similar advantages, and formations of mobile robots are being used to overcome various problems, such as surveillance, autonomous exploration, measurement of environmental research data [8], and search and localisation [2]. In the field of autonomous driving, cooperative control and formation control such as *Cooperative Adaptive Cruise Control* (CACC) and *Platooning* also hold promise for increasing highway capacity and flow, as well as safety [9].

A formation of mobile robots may be a more effective sensor than a single robot. e.g. if the position of an object has to be determined in three dimensions, at least four distance measurements of non-coplanar sensors with known position are needed. Moreover, multiple small sensors can be easier and cheaper to deploy than a single large sensor, and it can be quicker to cover an area with a group of sensors than with a single sensor [10].

One of the main challenges arising in the control of such multi-robot formations, is the architecture. One might think of a global controller which senses the position of all agents. This is possible for a small number of agents, but not scalable. To have scalable architecture, information and controllers should be decentralised. Therefore, *distributed formation control of multi-agent systems* is of interest specifically.

In recent years, most research to multi-agent formation control has been in the theoretical domain [11–15]. Here various formation control algorithms are explored, and conclusions can be drawn about properties such as stability, convergence, and about communication topologies. The formation of mobile robots is typically modelled by a single or double integrator system. The differential equations governing the system are simulated using MATLAB or similar software. Some research has also shown lab experiments on formations of mobile robots [16]. This M.Sc. Thesis aims to add practical knowledge on formation control for mobile

robots, by implementing various formation control algorithms on a team of non-holonomic skid-steer drive mobile robots, and testing these in various circumstances.

The objective of this M.Sc thesis therefore is to:

Identify practical limitations on formation control by implementing various formation control algorithms on a team of skid-steer drive mobile robots in a simulated laboratory environment.

To reach this objective, a main research question is formulated:

What are the practical limitations on formation control, and what effect do these limitations have on the functioning of different distance based control algorithms when applying them to a team of skid-steer drive mobile robots in a simulated laboratory environment?

- In what different ways can holonomic distance based formation control algorithms be implemented on a team of non-holonomic skid-steer drive mobile robots?
- Can object detection from LIght Detection And Ranging (LIDAR) data be used to measure distance between robots and obstacles, and how does this affect performance compared to using the global positions of the robots to measure distance?
- How does the control frequency of the low-level kinematic controller influence performance?
- What level of sensor noise is acceptable when using LIDAR distance measurements for formation control?

To answer the main research question and the subquestions, the first subquestion is answered by means of MATLAB simulations in Chapter 2. Then, different control laws are implemented in Robot Operating System (ROS) , which is thoroughly explained in Chapter 3. The ROS package is then tested in gazebo, which is explained in Chapter, 4. The results of the simulations are shown in Chapter 5. From these results, the other subquestions can be answered. Conclusions are drawn in Chapter 6, accompanied by limitations that were come across in this research, practical recommendations for implementing formation control, and suggestions for future research.

Formation control algorithms

In this chapter, different types of formation control algorithms are introduced and categorised. In literature, focus mostly lies with holonomic, or omnidirectional formation control algorithms [11, 14]. Various ways of adapting holonomic potential functions to work on teams of non-holonomic unicycle model mobile robots are devised, and MATLAB simulations are done to get a general understanding of the differences in behaviour of these adaptations. By the end of this chapter, the following research question will be answered:

In what different ways can holonomic distance based formation control algorithms be implemented on a team of non-holonomic skid-steer drive mobile robots?

First, the classes of formation control algorithms as defined by Oh et al. [12] are introduced, then one of these classes, distance based control, is further elaborated. Seven ways to adapt holonomic distance based control laws to work on teams of unicycle mobile robots are investigated and numerical simulations are done to draw conclusions about these adaptations, to answer the first research subquestion. The conclusions drawn from answering this subquestion are relevant for categorising non-holonomic formation control algorithms, and for designing non-holonomic formation control algorithms based on holonomic potential functions.

2-1 Classes of formation control

Oh et al. [12] categorises multi-robot formation control into three main classes, depending on sensing capability and interaction topology. These are *position based control*, *displacement based control* and *distance based control*. For the ROSbot, distance based control is the most applicable, as it requires the least sophisticated sensing capability. In distance based control, the agents sense the relative position of the other agents in their local coordinate frame. The control objective is to drive the distances to the desired distances, such that a desired formation shape is achieved. The formation is thus defined as a set of desired distances, or lengths of the graph edges. Besides the distinction between position-, displacement, and

distance based control, the degrees of freedom of the robots also play an important role. Omnidirectional robots, or holonomic robots, can move in any direction in the xy -plane, therefore, gradient based control algorithms are often implemented on this type of robot [11, 12]. Skid-steer drive robots, differential drive robots and unicycle model robots can rotate on the spot (around the z -axis) and move forward and backward (along the robot x -axis). This type of robot is represented by the unicycle model. In differential drive robots, not being able to move sideways is caused by the non-holonomic constraint that the wheels can roll, but can not slip. In this thesis, the term *non-holonomic mobile robot* is used for skid-steer drive and differential drive mobile robots which can be represented by the unicycle model, even though bicycle type mobile robots and car-like mobile robots are also types of non-holonomic mobile robots. In skid-steer drive robots, the wheels do slip (skid) when rotating, but no sideways movement is possible. The type of movement is the same as for differential drive robots, and thus, they are also represented by the unicycle model. In this thesis, *non-holonomic formation control algorithm* refers to a formation control algorithm for a group of *unicycle model mobile robots*. The ROSbot is a skid-steer drive mobile robot. However, to understand the control laws for this type of non-holonomic mobile robot, it is first necessary to look at the control laws for holonomic mobile robots.

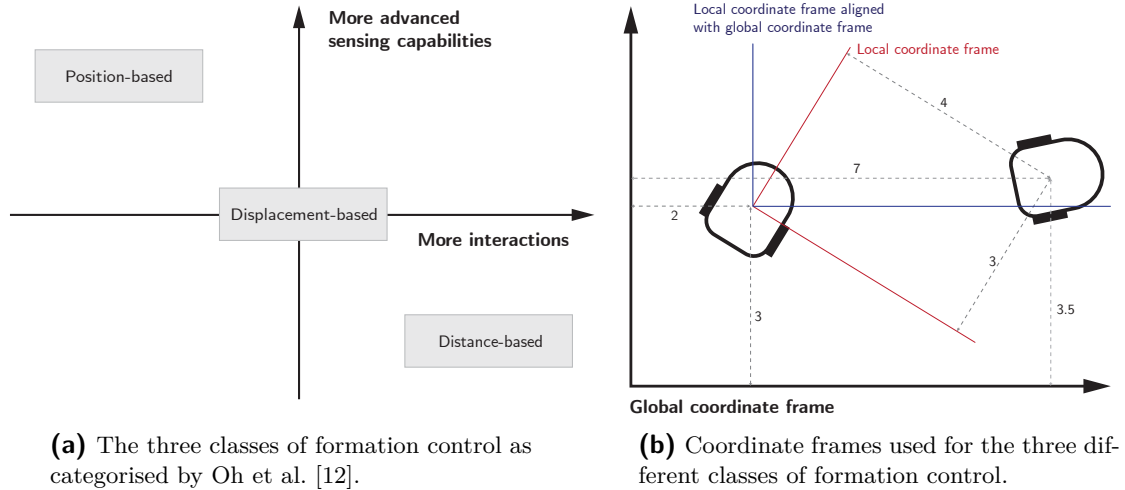


Figure 2-1: Overview of three different classes of formation control and the coordinate frames corresponding to each of these classes. A global coordinate frame is used for position based control, aligned local coordinate frames are used for displacement based control and local coordinate frames are used for distance based control.

2-2 Distance based control algorithms

In this section, the controls laws for holonomic robots are adapted to be suitable for non-holonomic unicycle model robots. The steps in this process are first illustrated based on an extensively studied potential function, found in literature [11, 17]: A consensus algorithm which drives each of the inter-agent distances to the desired distances. Later, in Chapter 3, this potential function will be varied to cover a wider range of control algorithms.

2-3 Adapting holonomic gradient-based control to teams of unicycle model robots

To reach the desired formation, gradient control laws are often used, where the robots move in the direction of the gradient of a potential function. The potential function is minimal when the agents are at the desired formation. Properties for the potential function to ensure exponential convergence are established in Sun [11]. An extensively studied [11, 17] formation potential function is given by:

$$V(p_1, \dots, p_N) = \frac{1}{4} \sum_{\{i,j\} \in \mathcal{E}} (\|p_i - p_j\|^2 - d_{ij}^{*2})^2 \quad (2-1)$$

Where $\mathcal{E}$ is the set of edges of graph $\mathcal{G}$, which is the graph representing the topology of the communication between the robots. The desired distance between agent i and agent j is denoted by d_{ij}^* . $p_i \in \mathbb{R}^2$ is the position $\begin{bmatrix} x_i & y_i \end{bmatrix}^\top$ of agent i in the global coordinate frame. If the agents move in the direction of the gradient of this potential, the control law is given by:

$$\dot{p}_i = u_i = -\nabla_{p_i} V(p) = \sum_{j \in \mathcal{N}_i} (p_j - p_i) (\|p_i - p_j\|^2 - d_{ij}^{*2}) \quad (2-2)$$

In Sun [11], exponential convergence for *rigid graphs* (see Anderson et al. [10]) to the desired formation is proven for class of gradient descent methods if the potential function has certain properties. Eq. (2-1) does meet these requirements, and thus, Eq. (2-2) will drive the robots to the desired formation exponentially fast. This control law can also be used in the local coordinate frames of the robot, since only the difference between the positions is taken. In this case, the relative position of robot j is measured in the coordinate frame of robot i : $p_j^i = \begin{bmatrix} x_j^i & y_j^i \end{bmatrix}^\top$ is the location of agent j in the coordinate frame of agent i . The final control input $\dot{p} = u_i$ is also given in the coordinate frame of robot i , as opposed to the global coordinate frame in Eq. (2-2).

$$\dot{p}_i^i = u_i = \sum_{j \in \mathcal{N}_i} p_j^i (\|p_j^i\|^2 - d_{ij}^{*2}) \quad (2-3)$$

Since the ROSbot cannot move sideways, it is not always possible to move in the direction of the gradient of the potential function, as in Eq. (2-2). Therefore, it is necessary to investigate ways to convert this desired movement in the direction of the gradient, $\dot{p}_i^i$ to an input of the form $\begin{bmatrix} v_x & \omega_z \end{bmatrix}^\top$.

2-3-1 Non-holonomic gradient-based control

Typically, formation control algorithms for non-holonomic unicycle model mobile robots are derived from potential functions for holonomic robots. The desired movement in the direction of the gradient of the potential function is given in the local coordinate frame of the robot by:

$$\dot{p}_i^i = \begin{bmatrix} F_{x_i} \\ F_{y_i} \end{bmatrix} = \begin{bmatrix} \sum_{j \in \mathcal{N}_i} \begin{pmatrix} F_{x_{ji}} \\ F_{y_{ji}} \end{pmatrix} \\ \sum_{j \in \mathcal{N}_i} \begin{pmatrix} F_{x_{ji}} \\ F_{y_{ji}} \end{pmatrix} \end{bmatrix} \quad (2-4)$$

Here, $\mathcal{N}_i$ is the set of neighbours of robot i . F_i is the potential, or virtual force, working on robot i , which is decomposed in the direction of the robot heading, F_{x_i} , and a component perpendicular to the robot heading, F_{y_i} . F_{x_i} is the sum of the individual components, $F_{x_{ji}}$, caused by the position of agent j in the local coordinate frame of agent i . Similarly, F_{y_i} is the sum of all $F_{y_{ji}}$. The velocity input for the holonomic mobile robot i would be

$$v_{x_i} = F_{x_i} \quad (2-5)$$

$$v_{y_i} = F_{y_i} \quad (2-6)$$

Where v_{x_i} is the velocity in the direction of the local x -axis, or the robot heading, and v_{y_i} is the velocity in the direction of the local y -axis, which is perpendicular to the robot heading. Now, for the unicycle model, the non-holonomic rolling-not-slipping constraint has the form:

$$v_{y_i} = 0 \quad (2-7)$$

Instead of the sideways movement, the unicycle robot can rotate around its z -axis. This rotational input is calculated as a function of the holonomic inputs, the forward velocity input remains the same, namely, the component of the holonomic input in the direction of the robot x -axis. The input for robot i is given by

$$v_{x_i} = F_{x_i} \quad (2-8)$$

$$\omega_{z_i} = f(F_{x_{ji}}, F_{y_{ji}}) \forall j \in \mathcal{N}_i \quad (2-9)$$

This function $f()$ thus generates a rotational velocity input ω_z from the given potentials. Seven variants are identified and simulated. Firstly, three where the sideways potential F_{y_i} is used directly, then four where the angle between the robot heading and the total potential is used to generate a rotational input.

Offsetting F_y to generate rotational velocity input The potential in the direction perpendicular to the robot heading doesn't move the robot, since that would violate the non-holonomic constraint on the unicycle model. If the potential is considered as a force, offsetting this sideways potential by a gain λ will cause a torque around the robot centre, thus creating a rotational movement of the robot. This is used to derive three ways to generate a rotational input for the robot. Firstly, the total sideways potential is translated along the vector F_x . i.e. λ is chosen to be in the same direction as F_x , as in Figure 2-2 and Figure 2-4, the robot will not favour moving forwards over moving backwards. The rotational input is given by:

$$\omega_{z_i} = \lambda \frac{\sum_{j \in \mathcal{N}_i} (F_{x_{ji}})}{\left| \sum_{j \in \mathcal{N}_i} (F_{x_{ji}}) \right|} \sum_{j \in \mathcal{N}_i} (F_{y_{ji}}) \quad (2-10)$$

Secondly, if it is chosen that λ is always positive, as in Figure 2-2 and Figure 2-3, moving backwards will become unstable, causing the robot to turn and move forward in the direction of the potential. This rotational input is given by Eq. (2-11):

$$\omega_{z_i} = \sum_{j \in \mathcal{N}_i} (\lambda F_{y_{ji}}) \quad (2-11)$$

Both scenarios have its uses. In the first scenario, the robot will take a shorter path to move in the direction of the potential, which is closer to the path the holonomic robot would take. The latter may be preferable if the robot has front-facing sensors, such as a camera. It is also possible to generate a rotational input from every separate potential $F_{y_{ij}}$, and summing these to get the final input. If λ is taken to be always positive, the result is Eq. (2-11). If λ is taken in the direction of F_x , the result is Eq. (2-12):

$$\omega_{z_i} = \sum_{j \in \mathcal{N}_i} \left(\lambda \frac{F_{x_{ji}}}{|F_{x_{ji}}|} F_{y_{ji}} \right) \quad (2-12)$$

This can lead to undesired equilibria where the rotational inputs cancel out. The behaviours are simulated, and the resulting paths are shown in Figure 2-6.

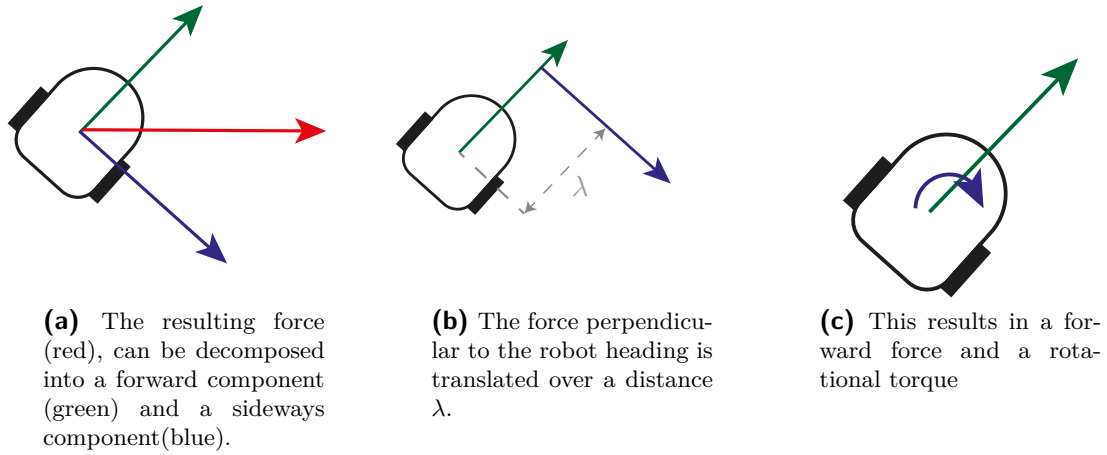


Figure 2-2: When the force drives the robot forward, λ is always taken positive.

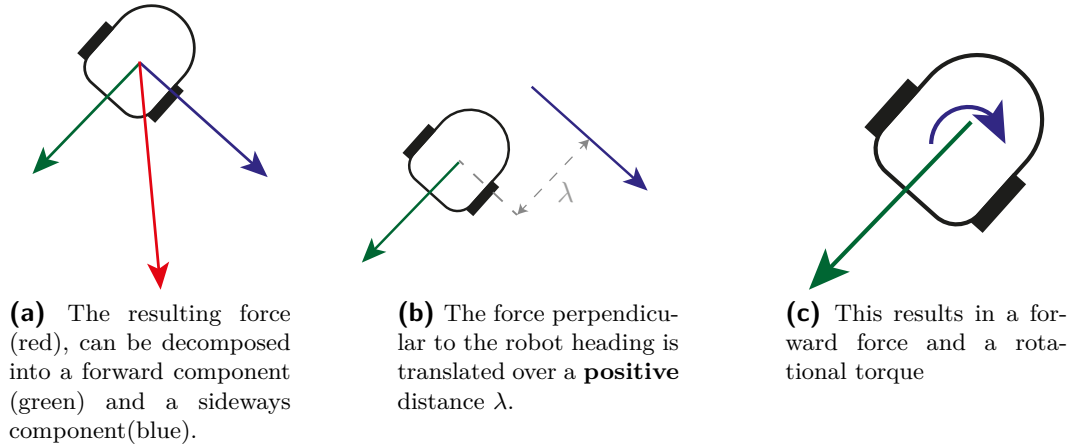


Figure 2-3: When λ is taken to always be positive, when the force drives the robot backward λ is still taken positive.

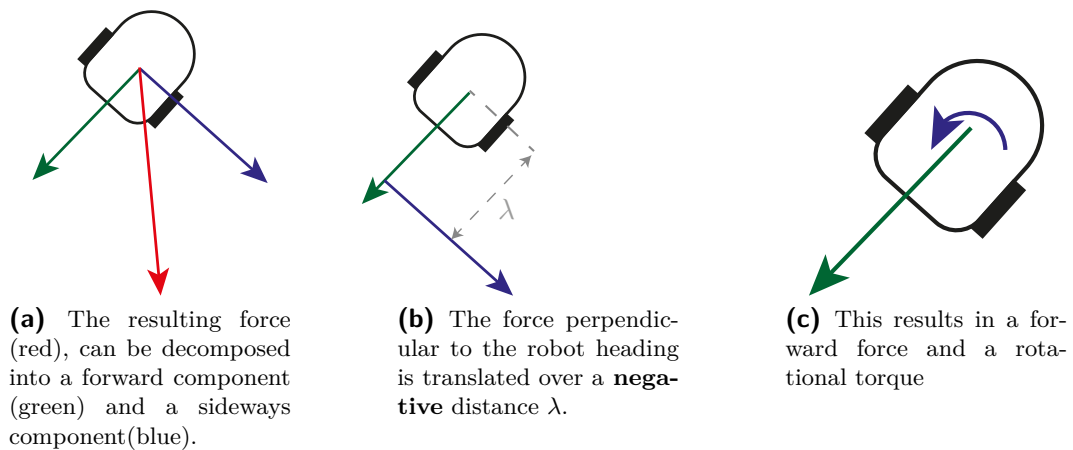


Figure 2-4: When λ is taken in the direction of F_x , when the force drives the robot backward, λ is taken negative.

Angle between heading and potential as input Besides the lateral potential F_y , the angle between the robot heading and the potential can also be used to calculate a rotational input. This angle, ϕ , can be defined as:

$$\phi = \arctan \frac{F_y}{F_x} \quad (2-13)$$

which will yield values between $-\frac{\pi}{2}$ and $\frac{\pi}{2}$, and there will be no difference between moving backwards or forwards. The angle ϕ can also be calculated as:

$$\phi = \text{atan2}(F_y, F_x) \quad (2-14)$$

which will yield values between $-\pi$ and π , and moving forward will be favoured. In this case, there is also a difference between calculating the rotational input from the total potential, or summing the rotational inputs calculated from separate potentials. This leads to four different ways to generate the rotational input. Firstly, taking the arctan of the separate potentials:

$$\omega_{z_i} = \sum_{j \in \mathcal{N}_i} \left(\arctan \frac{F_{y_{ji}}}{F_{x_{ji}}} \right) \quad (2-15)$$

Secondly, taking the arctan of the total potential:

$$\omega_{z_i} = \arctan \frac{\sum_{j \in \mathcal{N}_i} (F_{y_{ji}})}{\sum_{j \in \mathcal{N}_i} (F_{x_{ji}})} \quad (2-16)$$

Thirdly, taking the sum of the angles of the separate potentials as an input:

$$\omega_{z_i} = \sum_{j \in \mathcal{N}_i} \left(\text{atan2}(F_{y_{ji}}, F_{x_{ji}}) \right) \quad (2-17)$$

and finally using the angle between the heading and the total potential, calculated using Eq. (2-14):

$$\omega_{z_i} = \text{atan2} \left(\sum_{j \in \mathcal{N}_i} (F_{y_{ji}}), \sum_{j \in \mathcal{N}_i} (F_{x_{ji}}) \right) \quad (2-18)$$

2-3-2 Simulation of the seven adaptations

Now that seven different ways to calculate a non-holonomic input from the holonomic potential are devised, a MATLAB simulation is done to show the behaviours of the robot, moving backwards and forwards. Local unwanted equilibria are shown for the algorithms where the separate potentials first generate rotational inputs, which are then summed, i.e. Eq. (2-12), Eq. (2-15), and Eq. (2-17). In this simulation, one agent moves to a distance of 1m to both objects. The potential $\begin{bmatrix} F_{x_{ji}} & F_{y_{ji}} \end{bmatrix}^\top$ is the potential acting on this agent i in the direction of the local coordinate frame of agent i , as a result of the distance to object j .

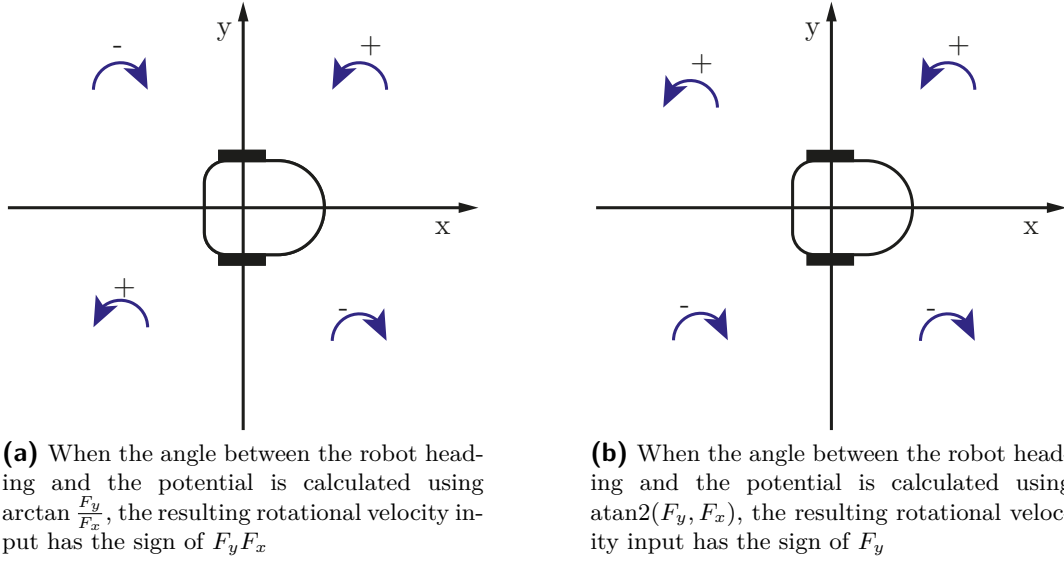


Figure 2-5: Sign of the rotational input when a potential "pulls" the robot towards the different quadrants.

The separate potentials are given by:

$$d_{ij} = \|p_j^i\| \quad (2-19)$$

$$\begin{bmatrix} F_{xji} \\ F_{yji} \end{bmatrix} = (d_{ij}^2 - d_{ij}^{*2}) p_j^i \quad (2-20)$$

Where $p_{ji} \in \mathbb{R}^2$ is the location of object j in the coordinate frame of agent i , d_{ij} is the euclidean distance between agents i and j , and d_{ij}^* the desired distance between agents i and j . A holonomic agent would move in the direction of the (total) potential. Which results in the system in Eq. (2-2). In this simulation, the path taken by a holonomic agent is compared to the paths taken by the agents utilising the seven functions that have been identified. These paths taken by the robots are shown in Figure 2-6.

In Figures 2-6a - 2-6c, the path can be seen if the agent does not have non-holonomic constraints and can move sideways to in the direction of the potential, for three starting locations. The desired final position, where the potential is 0, is shown with the red star. Figures 2-6d - 2-6f show the path taken if the rotational velocity input is generated by Eq. (2-11). Figures 2-6e and 2-6f show that when the potential moves the robot backwards, this movement is "unstable" and thus, the robot will rotate and move forward in the direction of the potential, i.e. moving forwards is favoured. By taking a larger value for λ , the curve radius will become smaller. Figures 2-6g - 2-6i show that using Eq. (2-12) to calculate the rotational velocity input will lead to undesirable equilibria. This is thus an unviable option. Figures 2-6j - 2-6l show the convergence to the desired equilibrium using Eq. (2-12). This will not favour moving forward over moving backward, thus, a shorter path is taken than when using Eq. (2-11), as can be seen in Figures 2-6h and 2-6i. In Figures 2-6m - 2-6o, results of Eq. (2-16) are shown, like with Eq. (2-10), moving backwards is stable. Figures 2-6p - 2-6r show the robot gets stuck in local equilibria when Eq. (2-15) is used. Using Eq. (2-18) and Eq. (2-17), moving

forward is favoured, and it is again seen that first, the total potential should be determined before taking the $\text{atan2}(y, x)$, as Eq. (2-17) gets stuck in a local equilibrium in Figure 2-6w.

2-3-3 Conclusions

With this classification of seven different ways to adapt holonomic distance based potential functions, the first research sub-question can be answered.

In what different ways can holonomic distance based formation control algorithms be implemented on a team of non-holonomic skid-steer drive mobile robots?

Seven different ways to implement holonomic formation control algorithms on unicycle model mobile robots are identified. Three of these show undesired behaviour where robots can get stuck in local equilibria. Two cause the robots to favour moving forwards, and two cause the robots to either move forward or backward, whichever is more in the direction of the gradient. Using this classification, the non-holonomic control laws discussed in literature can be analysed more easily, and non-holonomic control laws can be designed from existing holonomic control laws.

Some practical recommendations for the design of non-holonomic potential functions can be drawn from these results. Firstly, when adapting the holonomic potential to work on a non-holonomic robot, it is advised to first sum all the separate potentials, as otherwise the robot might get stuck in an undesired equilibrium. The systems in Eq. (2-10), Eq. (2-16), and Eq. (2-18) show this undesirable behaviour. Secondly, a choice can be made to favour moving forward over moving backwards. The systems given by Eq. (2-11) and Eq. (2-18) favour moving forward. The systems in Eq. (2-10) and Eq. (2-16) do not favour moving forward and the paths taken are more similar to the path taken by a holonomically moving agent. This poses a trade-off between shorter and more efficient paths and the ability to use front-facing sensors, and based on other requirements on the team of mobile robots and the task that has to be completed, a design choice can be made.

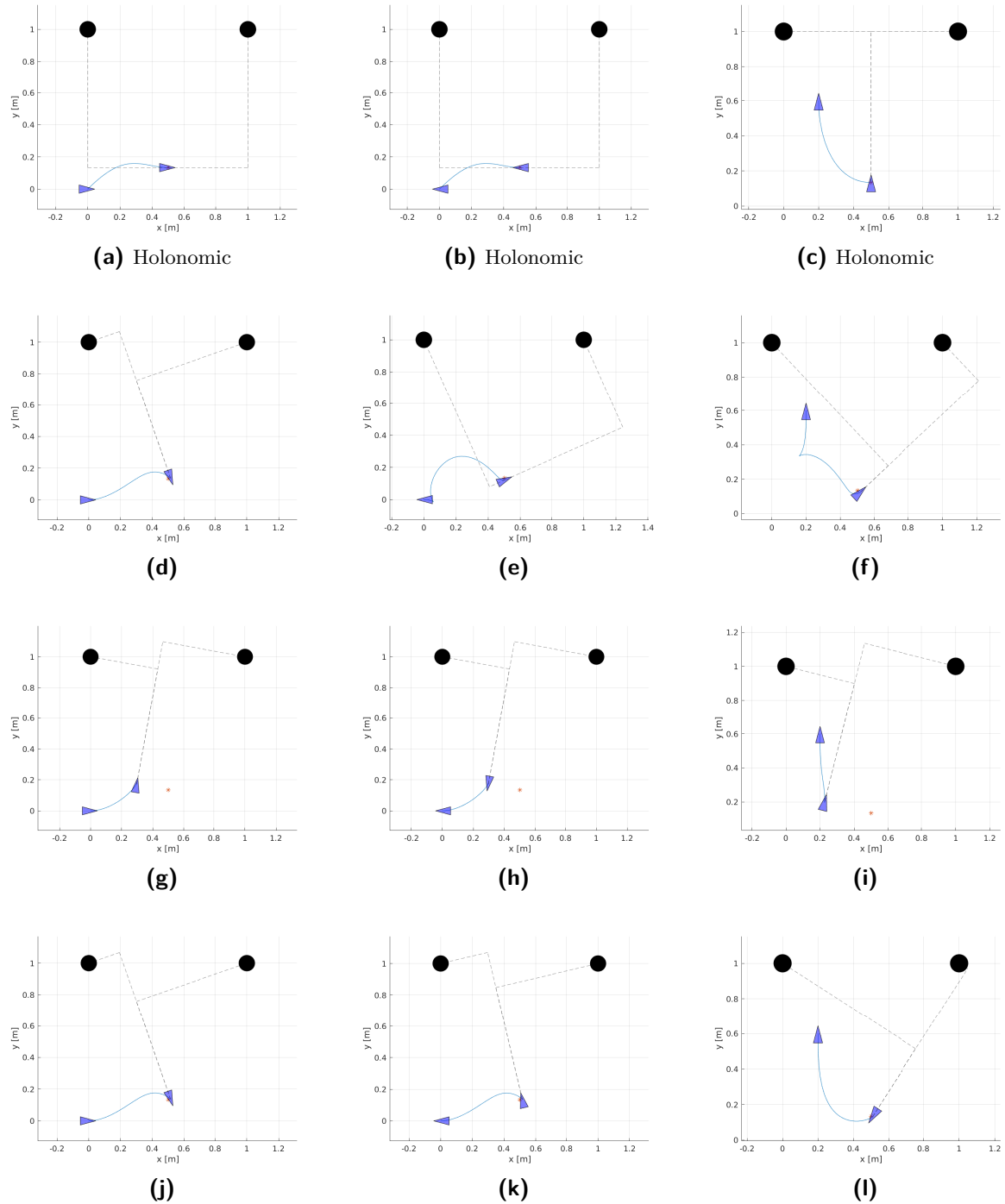


Figure 2-6: Paths taken by the robot reacting to the potential in Eq. (2-20), moving to the desired position (red star) where the distance to the other two static agents (black) is 1 m. from three different starting positions. The potential is converted to movement of the robot in 8 different ways. (a-c) Omnidirectional (holonomic) robot, using Eq. (2-6), and unicycle model (non-holonomic) robot, using: (d-f) Eq. (2-11), (g-i) Eq. (2-12), (j-l) Eq. (2-10), (m-o) Eq. (2-16), (p-r) Eq. (2-15), (s-u) Eq. (2-18), (v-x) Eq. (2-17).

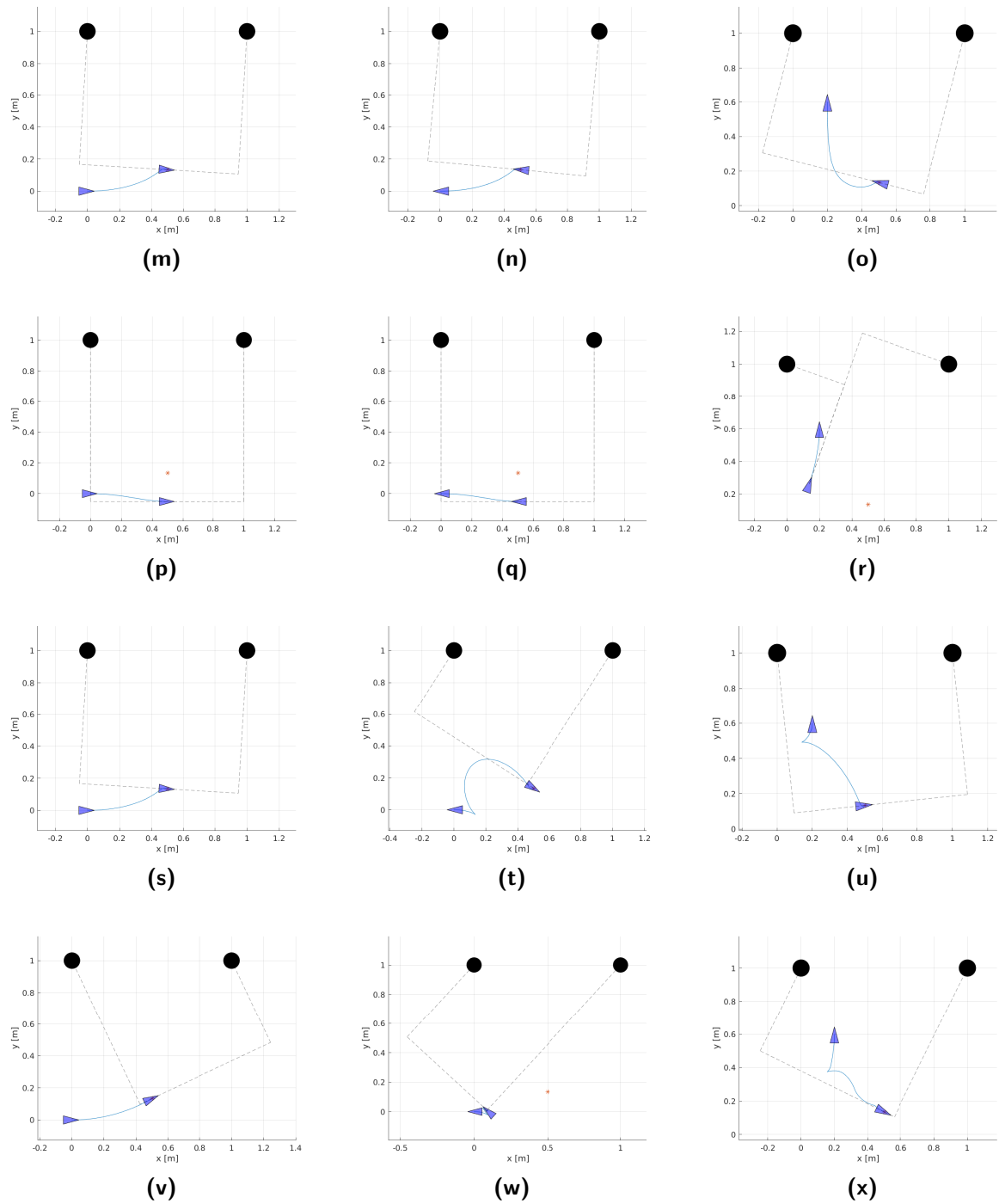


Figure 2-6: Continued.

Implemented algorithms

Now that the control laws have been categorised and some rules of thumb have been established, the next research subquestions to be answered are all related to the practical limitations on formation control, as well as the effects that these limitations have on the functioning of different distance based control algorithms when they are applied to a team of robots in a laboratory environment.

To answer these questions, different holonomic formation control algorithms are adapted to work on unicycle type mobile robots. A selection of different non-holonomic formation control algorithms is made and these algorithms are implemented into a Robot Operating System (ROS) package. The ROS package with the formation control algorithms are tested in the Gazebo simulation environment. In this chapter, the algorithms that are later implemented are introduced, based on a division between the specific goals for the team of robots. The overall goal for the group of mobile robots can be divided into three sub-goals:

- Move towards a goal location or in a desired direction;
- Converge to the desired formation and keep this formation;
- Avoid obstacles.

In this thesis, formation keeping and obstacle avoidance are taken as the focus. Therefore, three algorithms from literature are chosen and adapted to work on the skid-steer drive mobile robots. These algorithms are explained in this section—firstly covering the formation keeping and obstacle avoidance in Section 3-1, and then covering the goal of moving towards a specific location in Section 3-2.

3-1 Formation keeping and obstacle avoidance

For the combined goals of formation keeping and obstacle avoidance, three different algorithms are implemented, namely a virtual spring-damper system algorithm, a consensus algorithm, and an exponential potential algorithm. All three of these are elaborated below.

3-1-1 Virtual spring-damper system

One of the most intuitive ways to look at formation control for mobile robots, is to have a virtual spring-damper system between each of the neighbouring robots, where the desired distance between robots is given by the natural length of the virtual spring. This system is shown in Figure 3-1. The total force of all the springs and dampers can be decomposed to give a forward and a sideways force on the robot. This then can be translated into a forward velocity setpoint v_x and a rotational velocity setpoint ω_z .

The spring force on the robot i is given by:

$$F_{spring} = k_p \left(\|p_j^i\| - d_{ij}^* \right) \quad (3-1)$$

$$\begin{bmatrix} F_{spring_x} \\ F_{spring_y} \end{bmatrix} = k_p \left(\|p_j^i\| - d_{ij}^* \right) \frac{p_j^i}{\|p_j^i\|} \quad (3-2)$$

where k_p is the stiffness of the virtual springs between each of the neighbouring robots, and p_j^i is the position of robot j in the coordinate frame of robot i . The norm of this position is the distance between the robots. d_{ij}^* is the desired distance between robots i and j . The spring force is decomposed into two forces aligned with the coordinate frame of robot i . The damper force is given by:

$$F_{damper} = k_d \left(\dot{\|p_j^i\|} \right) \quad (3-3)$$

$$= k_d \left(\sqrt{\dot{x}_j^2 + \dot{y}_j^2} \right) \quad (3-4)$$

Where the k_d is the damping coefficient. By using the chain rule, the damper force can be expanded to:

$$F_{damper} = \frac{x_j^i \dot{x}_j^i + y_j^i \dot{y}_j^i}{\|p_j^i\|} \quad (3-5)$$

$$\begin{bmatrix} F_{damper_x} \\ F_{damper_y} \end{bmatrix} = \frac{x_j^i \dot{x}_j^i + y_j^i \dot{y}_j^i}{\|p_j^i\|} \frac{p_j^i}{\|p_j^i\|} \quad (3-6)$$

Each of the neighbours of i will exert a spring and damper force on robot i . When an obstacle is detected within a certain detection range, the obstacle is added to the set of neighbours of i , and will also exert a spring and damper force. The natural length of the spring is equal to the detection range, so the obstacles will only repel the robot. Summing these forces together, the total force acting on the robot is given by:

$$\begin{bmatrix} F_{x_i} \\ F_{y_i} \end{bmatrix} = \sum_{j \in \mathcal{N}_i} \left(\left(k_p \left(\|p_j^i\| - d_{ij}^* \right) + k_d \frac{x_j^i \dot{x}_j^i + y_j^i \dot{y}_j^i}{\|p_j^i\|} \right) \frac{p_j^i}{\|p_j^i\|} \right) \quad (3-7)$$

Two values for the spring stiffness and damping coefficient are taken, one for the neighbouring robots, and one for the neighbouring obstacles.

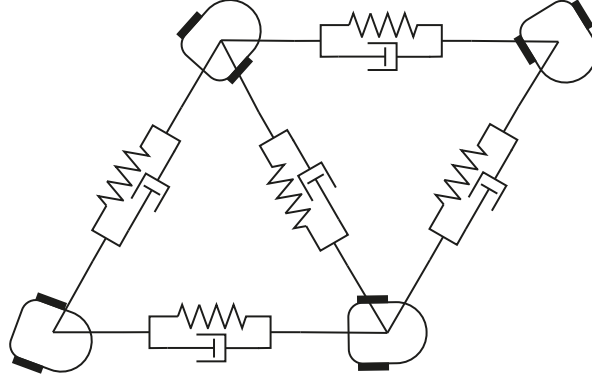


Figure 3-1: Mobile robots connected by virtual springs and dampers. Adapted from Wiech et al. [1]

In Wiech et al. [1], this system is modelled using

$$v_x = \int \frac{F_x}{m} dt \quad (3-8)$$

$$\omega_z = \lambda F_y \quad (3-9)$$

Thus, the rotational velocity is determined by Eq. (2-11). The forward force F_x causes an acceleration of the robot.

In this thesis, it is assumed that the kinematic controller is sufficient. Also, moving forward is preferred over moving backward. Therefore, like in Wiech et al. [1], ω_z is derived using Eq. (2-11). The velocity setpoint for each robot is given by:

$$v_i = F_{x_i} \quad (3-10)$$

$$\omega_i = \lambda F_{y_i} \quad (3-11)$$

3-1-2 Consensus algorithm

The consensus algorithm introduced in Section 2, in Eq. (2-3), is also implemented into the ROS package. As discussed in Section 2, one of the seven adaptations has to be chosen to get an input suitable for the ROSbot. Since the ROSbot has front-facing camera's that can be used while the robots move in formation, Eq. (2-18) is chosen to transform the holonomic potential into forward and rotational velocity inputs. This leads to the input given as:

$$v_i = k_{linear} \sum_{j \in \mathcal{N}_i} \left(x_j^i \left(\|p_j^i\|^2 - d_{ij}^{*2} \right) \right) \quad (3-12)$$

$$\omega_i = k_{rotational} \text{atan2} \left(\sum_{j \in \mathcal{N}_i} \left(y_j^i \left(\|p_j^i\|^2 - d_{ij}^{*2} \right) \right), \sum_{j \in \mathcal{N}_i} \left(x_j^i \left(\|p_j^i\|^2 - d_{ij}^{*2} \right) \right) \right) \quad (3-13)$$

With k_{linear} and $k_{rotational}$ being tunable control parameters. As with the spring-damper system, when an obstacle is within a certain detection range, this obstacle is added to the set of neighbours of i . The desired distance between agent i and the obstacle is equal to the detection range, thus repelling the robot away from the obstacle.

3-1-3 Exponential potential algorithm

In Listmann et al. [13], an exponential potential function is proposed. This system is given by:

$$k_{d_{ij}} = 1 - \exp\left(d_{ji}^{*2} - \|p_j - p_i\|^2\right) \quad (3-14)$$

$$v_i = - \sum_{j \in \mathcal{N}_i} \left([(x_i - x_j) \cos \theta_i + (y_i - y_j) \sin \theta_i] k_{d_{ij}} \right) + v_d \quad (3-15)$$

$$\omega_i = - \sum_{j \in \mathcal{N}_i} \left(\theta_i - k_{d_{ij}} \arctan \frac{y_j - y_i}{x_j - x_i} + (k_{d_{ij}} - 1) \theta_d \right) \quad (3-16)$$

Here, v_d is a desired velocity and θ_d is a desired angle in the global coordinate frame. However, this is a type of *displacement based control*, where the measurements should be aligned with a global axis system. The robot should be able to sense θ_i , for example with a compass. Besides driving the robots to the desired formation, the angles θ_i are driven to θ_d . Using only local measurements, the control law becomes:

$$k_{d_{ij}} = 1 - \exp\left(d_{ji}^{*2} - \|p_j^i\|^2\right) \quad (3-17)$$

$$v_i = \sum_{j \in \mathcal{N}_i} \left(k_{d_{ij}} \frac{p_j^i}{\|p_j^i\|} \right) \quad (3-18)$$

$$\omega_i = \sum_{j \in \mathcal{N}_i} \left(k_{d_{ij}} \text{atan2}(y_j^i, x_j^i) \right) \quad (3-19)$$

This, however, does not have the additional advantage that all the robots are aligned and the velocity offsets v_d would not be in the same direction. Therefore, the displacement based algorithm in Eqs. (3-14)-(3-16) is very sensitive to the desired distance, to overcome this, it is adapted to add more tuning parameters. The final implemented control law is given by:

$$k_{d_{ij}} = 1 - \exp\left(k_{formation} \left(d_{ji}^{*2} - \|p_j - p_i\|^2\right)\right) \quad (3-20)$$

$$v_i = - \sum_{j \in \mathcal{N}_i} \left([(x_i - x_j) \cos \theta_i + (y_i - y_j) \sin \theta_i] k_{d_{ij}} \right) + v_d \quad (3-21)$$

$$\omega_i = - \sum_{j \in \mathcal{N}_i} \left(\theta_i - k_{d_{ij}} \arctan \frac{y_j - y_i}{x_j - x_i} + (k_{d_{ij}} - 1) \theta_d \right) \quad (3-22)$$

When one of the robots gets too close to an obstacle, Listmann et al. [13] proposes a switching behaviour. The control law for any robot coming too close to an obstacle is given as:

$$\left. \begin{aligned} v_i &= k_u \sin(\alpha_o) \\ \omega_i &= -k_\omega \cos(\alpha_o) \end{aligned} \right\} \text{if } d_{obs} < d_{obs}^* \quad (3-23)$$

Where α_o is the angle between the robot heading and the obstacle, d_{obs} is the distance between the robot and the obstacle, and d_{obs}^* is the detection range at which the robot starts reacting to the obstacle. k_ω and k_u are control gains. This control law will make the robot circumvent the obstacle at a distance of d_{obs}^* with velocity k_u .

3-2 Moving towards goal

For the pack of robots to move towards a goal location, a direction of this goal, and a distance to this goal is necessary. For the formation control, it is assumed that all measurements of the robots where in the local coordinate frame of the robots. However, to move towards a global goal location, the robots should be able to sense the position of this goal relative to their location. In the simulation, the location of the goal relative to the robot is sensed by sensing the global location of the robot and of the goal location. In a real life scenario where no global coordinate frame is available, robots could plan a direction to move to based on sensed variables, as is also done in Renzaglia and Briñón-Arranz [2]. Here a signal strength is measured by each of the individual robots, and using their relative positions, a desired direction of movement is agreed on. In this thesis the goal finding is simplified to a global location, where the robots are pulled towards.

$$F_{goal} = k_{p_{goal}} d_{goal} \quad (3-24)$$

Here, d_{goal} is the distance to the goal location, and $k_{p_{goal}}$ a proportional gain. For the system in Section 3-1-3, no additional goal finding potential is added, because there is a desired angle between the global coordinate frame x -axis and the robots movement, and the pack of robots will move in this direction indefinitely.

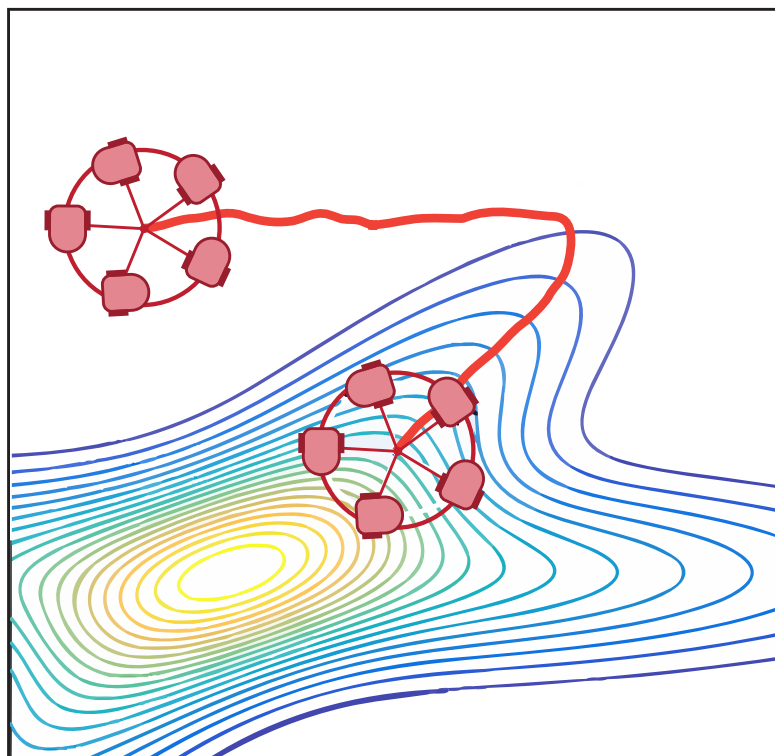


Figure 3-2: Localisation of a weak source as proposed by Renzaglia and Briñón-Arranz [2]. Adapted from Renzaglia and Briñón-Arranz [2].

Simulation environment

The formation control algorithms introduced in the previous chapter are implemented in a Robot Operating System (ROS) package, and tested in the Gazebo simulation environment. In Appendix A, the working principles of ROS and Gazebo are further explained. In this chapter, the specific communication structure of the different ROS nodes, as well as the testing methodology are discussed.

4-1 ROS structure

In this section, the ROS structure in which the formation control algorithms discussed in Section 3 are implemented is elaborated. These formation control algorithms are implemented into ROS nodes. The communication between the ROS nodes running on all the robots is schematically depicted in Figure 4-1.

The objects around the ROSbot are listed in the `/obstacles` topic. From this, actual obstacles and other robots are identified. Global locations of the robots are used for this. These global locations of the robots are taken from the `/tf` topic, which is published by Gazebo. The detections of robots and obstacles are fed into the control node, in which the formation control algorithm determines the velocity setpoints v_x and ω_z . These setpoints are sent over the `/cmd_vel` topic to the robot firmware, which then drives the motors to achieve the desired movement. The communication between robots happens by subscribing the control node to the detections of the other robots. Each robot runs the nodes and topics in its own *namespace*. e.g. the `laserscan` topic produced by the Light Detection And Ranging (LIDAR) of the first robot is published as `robot1/scan`, the velocity setpoints sent to the firmware of robot 3 are published as `robot3/cmd_vel`.

4-1-1 Robot identification

The LIDAR system combined with the obstacle detection ROS package allows for object detection. Object distances and directions are measured, and objects are tracked. However,

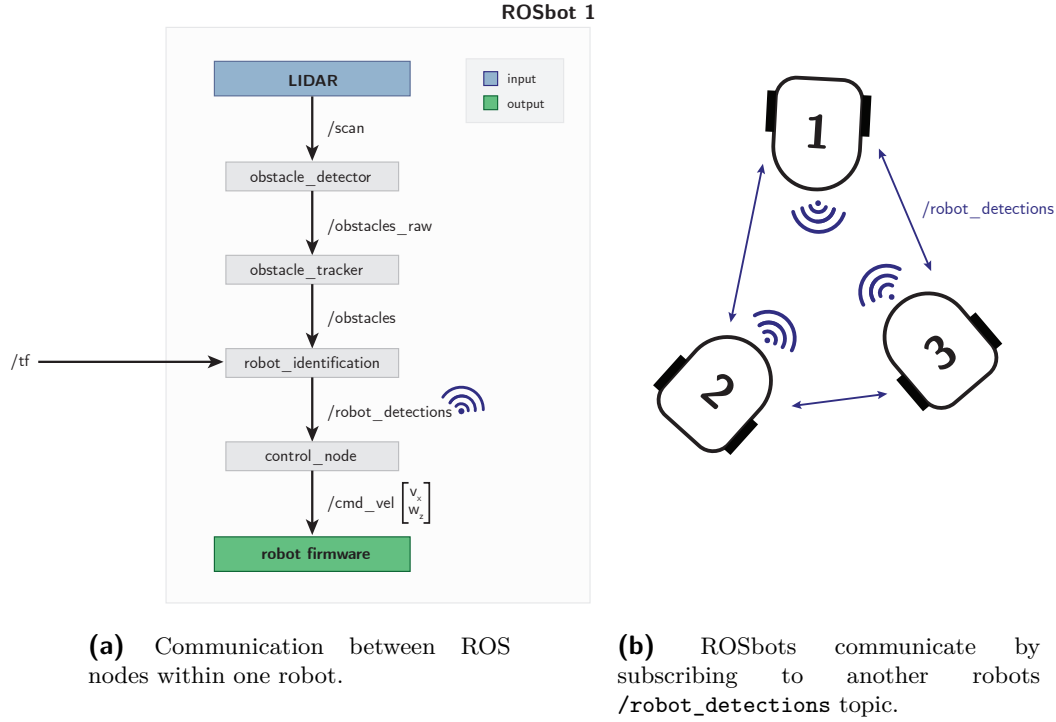


Figure 4-1: ROS nodes in each of the robots, and inter-robot communications.

with this setup, it is not possible to determine which of the sensed objects are obstacles, and which are the other robots. For this, a secondary system is used, which tracks all the robots in a global frame. A ROS package is made to cross-reference the locally detected obstacles with the global tracking system, to thus determine whether the object is another robot and which robot it is. Only robots are tracked, if no match is found, the object is treated as an obstacle. Outside of a laboratory scenario, more complex sensing capabilities would have to be implemented to overcome this identification problem. Possibilities include 3D LIDAR with shape identification, 360 deg camera with identification using Apriltag (See Wang and Olson [18]). Or, within the capabilities of the ROSbot, use the camera with Apriltags to identify the robots, and track them with the planar LIDAR when out of the camera angle.

The robot identification node logic is given by the pseudocode in Algorithm 1.

The detected objects have a position (X, Y) , in the local coordinate frame of the robot. The global positions of the robots have position (X, Y, θ) in the global coordinate frame, and a name. The list of detected robots is returned as R , and the list of detected obstacles is returned as O .

4-2 Identification of practical limitations

Compared to simulations in MATLAB or similar software packages, implementation in practice and simulations in Gazebo bring new limitations. First of all, using the LIDAR to detect obstacles brings noise on the distance measurement, and obstacles and robots may sometimes

Algorithm 1: Robot Identification

```

struct {
    string Name;
    float  $X, Y, \theta$ ;
} Detection;
IdentifyRobots ( $[Detection]D, [Detection]P$ )
    inputs : List of detected objects  $D$ , in the local coordinate frame of the robot; List
              of global positions of robots  $P$ 
    outputs: List of detected robots  $R$ ; List of detected obstacles  $O$ 
    Get global position of this robot  $p_{self}$  from  $P$ ;
    foreach Detected object  $obj_i \in D$  do
        Global x position of object  $p_{object}.X \leftarrow p_{self}.X + obj_i.X \cos p_{self}.\theta$ ;
        Global y position of object  $p_{object}.Y \leftarrow p_{self}.Y + obj_i.Y \sin p_{self}.\theta$ ;
        foreach Robot position  $r \in P$  do
            if  $|p_{object}.X - r.X| < 0.05$   $\&\&$ 
                 $|p_{object}.Y - r.Y| < 0.05$  then
                     $obj_i.Name \leftarrow r.Name$ ;
                    Add  $obj_i$  to  $R$ ;
                else
                    Add  $obj_i$  to  $O$ ;
                end
            end
        end
    end
    return  $O, R$ 

```

not be detected at all. Secondly, the frequency at which both the control algorithm (high level controller) and the robot firmware (low level controller) run may be of influence on the performance. Lastly, communication between robots may not always be perfect. The algorithms discussed in Section 3 are tested to see if the performance is adequate in these scenarios.

4-2-1 LIDAR

To test the performance of the formation control algorithms when using the LIDAR for detection of other robots and obstacles, each algorithm is tested in a number of scenarios. First, the global position of the robots is used to get the distance between the robots. In this case, the detection and distance measurement is very close to reality. In the second scenario, the LIDAR is used to detect the robots and obstacles. The LIDAR simulation in Gazebo contains zero-mean Gaussian noise. Multiple simulations are done over a range of noise levels, by changing the standard deviation of the gaussian noise. The distance measurements will be more noisy and robots and obstacles may not always be detected reliably. The performance for all the scenarios is evaluated by means of the Integral Squared Error (ISE), given by

Eq. (4-1):

$$ISE = \int_{t=0}^{t=t_{end}} \sum_{\mathcal{E}_{ij} \in \mathcal{E}} (d_{ij} - d^*)^2 dt \quad (4-1)$$

The squared error is strictly non-negative. The integral of this squared error gives a measurement of performance over a longer time. The value of the ISE is thus a measurement of formation-keeping performance over the total duration of the simulation, summed over all the agents. From the comparison on ISE, conclusions about acceptable levels of sensor noise in a real life scenario can be drawn.

4-2-2 Control frequency

The ROSbot firmware natively runs at 15 Hz. This means the linear and angular velocity inputs are processed once every 0.067 s. It is investigated if this is sufficient for the formation control algorithms implemented in this thesis, and if a higher control frequency will improve the results of the implemented algorithms. From this, practical recommendations are given regarding the hardware used in mobile robots designed with formation control in mind. A more ideal situation is simulated where this frequency is 100 Hz, to come closer to the continuous time models proposed in literature [11]. Very high control frequencies will make simulation slow, thus, based on this trade-off, 100 Hz is chosen in this thesis.

4-3 Simulation

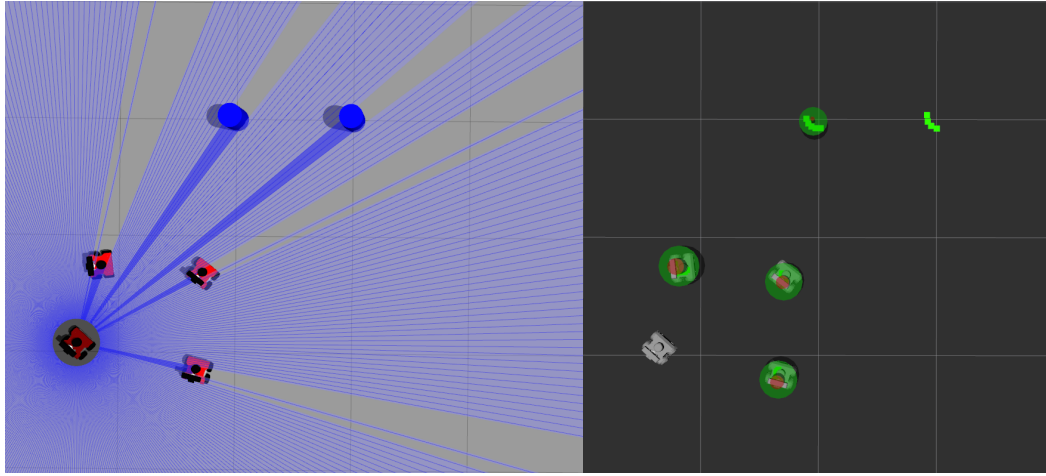
4-3-1 Test scenario

It is chosen to simulate a group of four robots in an environment with obstacles. The performance of the pack of robots is then evaluated based on collision avoidance, reaching the goal, and the integral squared formation keeping error ISE. The initial positions of the robots and the obstacles for the evaluated simulation are given in Table 4-1.

Table 4-1: Initial positions of robots and obstacles in the simulated scenario.

Initial positions	X	Y	θ
Robot 1	0	0	π
Robot 2	0	1.2	0
Robot 3	1.2	1.2	1.2
Robot 4	1.2	0	1.2
Obstacle 1	2	3	
Obstacle 2	3	3	

The desired formation shape is given by Eq. (5-1), which represents a rectangle with sides



(a) The simulation scenario in Gazebo, with LIDAR rays visible in blue. Since the Laser Range Finder (LRF) scans parallel to the ground, only objects at the height of the LRF are detected. This means the other robots bodies are not sensed, only the camera stand and the LRF

(b) The sensed LIDAR points and obstacles.

Figure 4-2: Simulation scenario with LIDAR rays and sensed objects.

0.75 m, and 1 m, and a diagonal of 1.25 m.

$$d^* = \begin{bmatrix} 0 & 0.75 & 1.25 & 1 \\ 0.75 & 0 & 1 & 1.25 \\ 1.25 & 1 & 0 & 0.75 \\ 1 & 1.25 & 0.75 & 0 \end{bmatrix} \quad (4-2)$$

4-3-2 Technical set-up

The system is tested with ROS Kinetic in Gazebo 9.13, using Open Dynamics Engine (OpenDE) to simulate the physics. The Gazebo simulation is run on a notebook computer with Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz processor and 16GiB RAM. For LIDAR simulation, GPU acceleration is used on NVidia Quadro T1000. The data from the simulation is captured using *rosvbag* [19], which is later analysed using MATLAB. A comparison is made between the use of the detections by the LIDAR measurements, and using the more precise global position published by Gazebo over the `/tf` topic. Also a comparison on the control frequency, or UpdateRate, of the skid-steer drive controller is made. For this, the ROSbot frequency of 15 Hz is compared to a faster controller with a frequency of 100 Hz. In Gazebo, the noise simulated on the individual LIDAR distance measurements (distance of individual points) is varied, with standard deviations ranging from $2.5e - 4$ m to $1e - 1$ m. 10 runs are done per algorithm, per noise level, per control frequency. From this data, box plots showing the ISE are used to draw conclusions on formation keeping performance.

Chapter 5

Results

The algorithms proposed in Chapter 3 are simulated in the Gazebo simulation environment (see Appendix A). Results of these simulations are presented in this chapter. These results are evaluated both qualitatively, e.g. by evaluating if collisions occurred and if the goal is reached, and quantitatively, by looking at the formation keeping errors (distance errors) between the desired inter-agent distance and the actual inter-agent distance. For this, the Integral Squared Error (ISE) is used as a measure of formation-keeping performance for the group of robots. The ISE is given by Eq. (4-1).

For each of the algorithms, the controller is tuned to work in an "ideal" situation with a high controller frequency of 100 Hz for the kinematic skid-steer drive controller in the ROSbot firmware, and using the global positions of the robot to derive the inter-agent distances, which are used as control inputs. First, a simulation run is done in this "ideal" situation. Second, to evaluate the sensitivity to the noise introduced when Light Detection And Ranging (LIDAR) is used for distance measurements, the distance used as an input for the controller is not generated from the global positions anymore, but from the LIDAR distance measurements and object detection. The noise on the LIDAR for these runs is set to have a standard deviation of 0.01 m. Third, to evaluate the sensitivity to a lower controller frequency, the formation is simulated with the skid-steer drive controller running at 15 Hz, while using the global position distances as the input for the controller. Lastly, the "real life" scenario is tested with the controller running at 15 Hz and using the LIDAR for the distance measurements.

Also, it is investigated how LIDAR noise levels affect the performance of the formation control algorithm. This is done by simulating the group of robots ten times per noise level, and evaluating the ISE. The LIDAR noise is taken to be Gaussian noise with zero mean. A larger standard deviation on this noise gives a noisier measurement. The noise for the RPLidar-A2M8 has a standard deviation between $2.5e - 5$ m and $1e - 2$ m. In Appendix A, the values for this standard deviation are further elaborated.

5-1 Virtual spring-damper system

In this section, the performance of the virtual spring-damper system is evaluated. In the first situation, the ROSbot firmware (skid-steer drive controller) runs at 100 Hz, and the global positions of the robots are used to derive distance measurements. This is referred to as an "ideal" situation. Then, a more realistic scenario where the LIDAR is used for the distance measurements is simulated. Additionally, a simulation is done with the ROSbot firmware running at 15 Hz using the global positions of the robot. Finally, a simulation is done using LIDAR for distance measurements, running at 15 Hz. These simulations are subsequently discussed in the following subsections.

5-1-1 Ideal situation: using global distance measurements and running the skid-steer drive controller at 100 Hz

In the ideal situation, the LIDAR is bypassed, and perfect distance measurements are taken from the global positions of the robots. The control node and the ROSbot firmware run at 100 Hz. The desired formation is a rectangle with sides of 0.75 m and 1 m. All the desired distances that define this rectangle are given by Eq. (5-1):

$$d^* = \begin{bmatrix} 0 & 0.75 & 1.25 & 1 \\ 0.75 & 0 & 1 & 1.25 \\ 1.25 & 1 & 0 & 0.75 \\ 1 & 1.25 & 0.75 & 0 \end{bmatrix} \quad (5-1)$$

The robots are initialised in a square formation with sides of 1.2 m. The interaction topology is fully connected, thus having a rigid, but not minimally rigid formation. The parameters are given in Table 5-1. The distance at which the robots start to react to the obstacle is $d_{obs}^* = 1$ m.

The results of this simulation are shown in Figure 5-1. It can be seen that, after the obstacles are avoided, the robots converge to the desired formation shape. All the distances are slightly smaller than the desired distances. This is due to the fact that the robots are grouped around the desired location, and are each slightly pulled toward this location, resulting in the final equilibrium position.

Table 5-1: Control gains for the virtual spring-damper system.

Parameter	Value
$k_{p_{robot}}$	1
$k_{d_{robot}}$	0.2
$k_{p_{obstacle}}$	2
$k_{d_{obstacle}}$	0.2
$k_{p_{goal}}$	0.1
$k_{d_{goal}}$	0

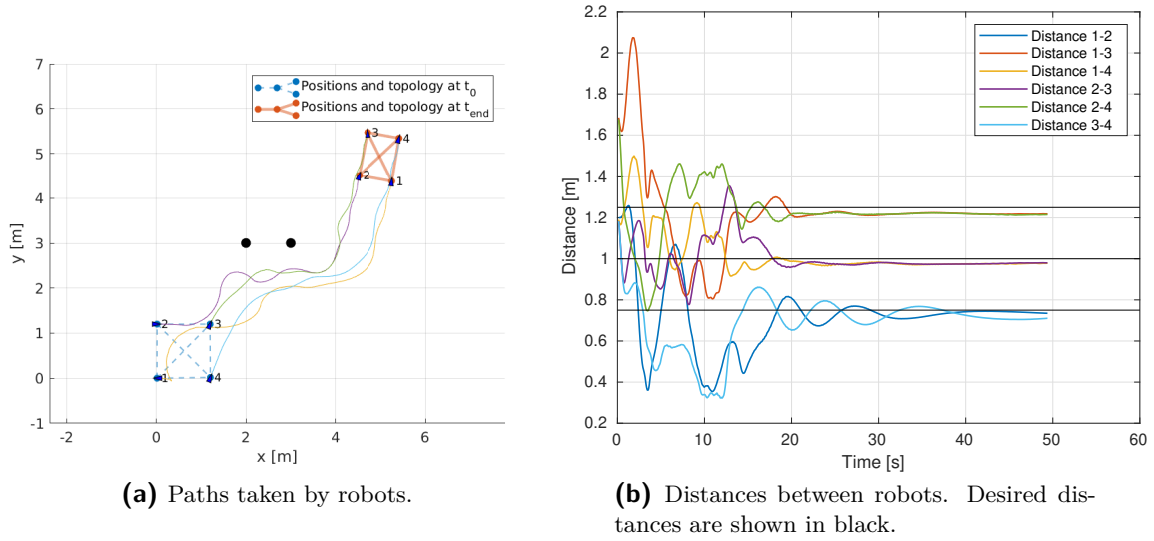


Figure 5-1: Paths of robots and distances between the robots, 100 Hz, global distance measurements

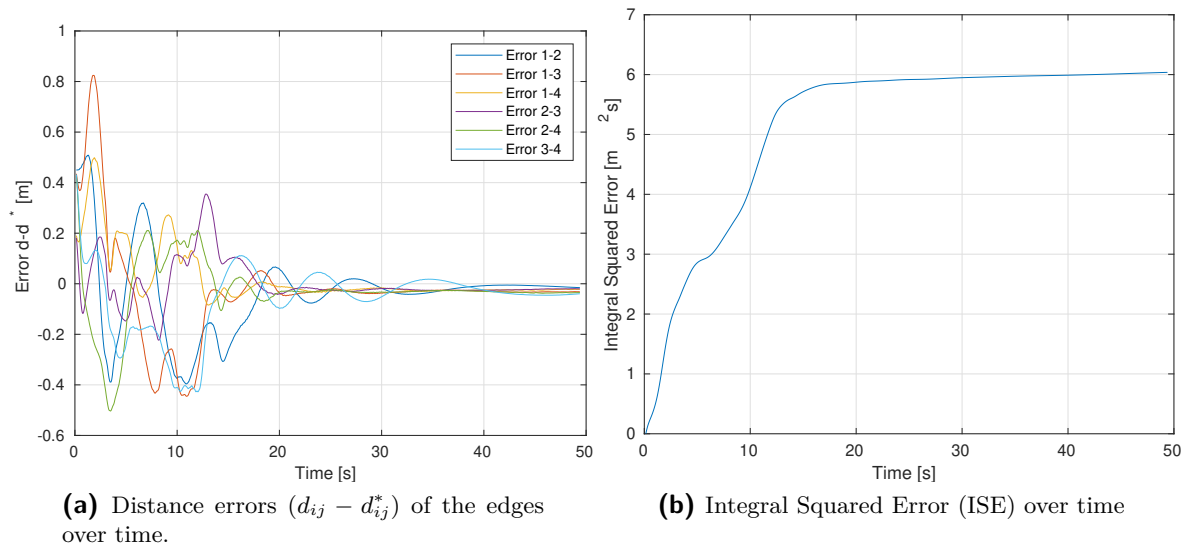


Figure 5-2: Distance errors and cumulative ISE for the group of robots using the spring-damper algorithms at 100 Hz, using global distance measurements.

5-1-2 Using LIDAR for distance measurements and running the skid-steer drive controller at 100 Hz

To get closer to the real scenario, the LIDAR is used by each robot to measure the distances of the other robots and obstacles locally. The desired formation is again given by Eq. (5-1), and control gains are given by Table 5-1. The results of this simulation using LIDAR are shown in Figure 5-3, which depicts the paths of the robots.

While moving around the two obstacles, the formation "flips over", resulting in a mirrored

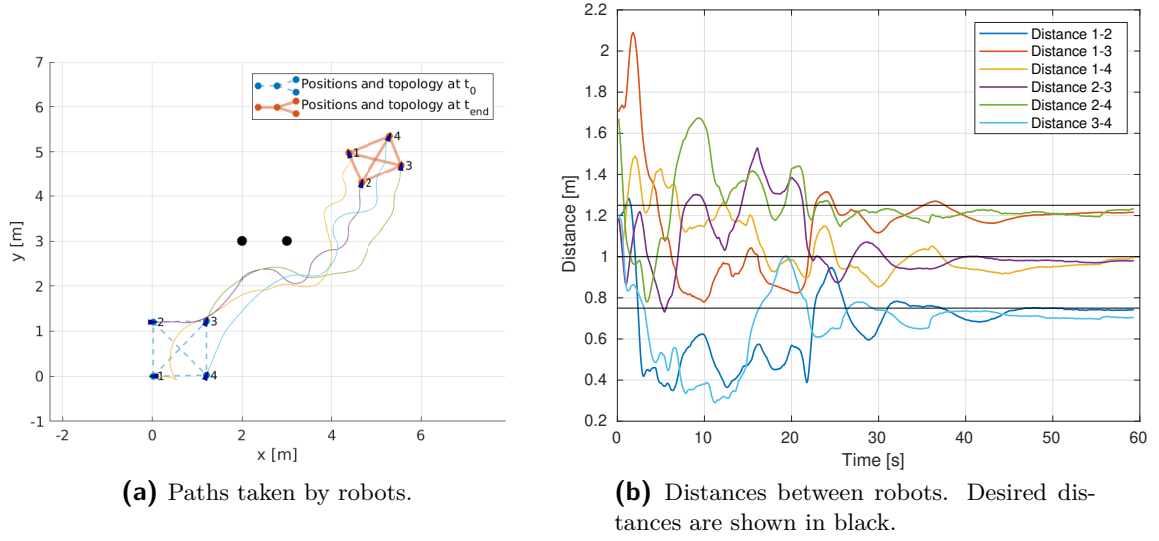


Figure 5-3: Paths of robots and distances between the robots, 100 Hz, LIDAR distance measurements with 0.01 m noise standard deviation.

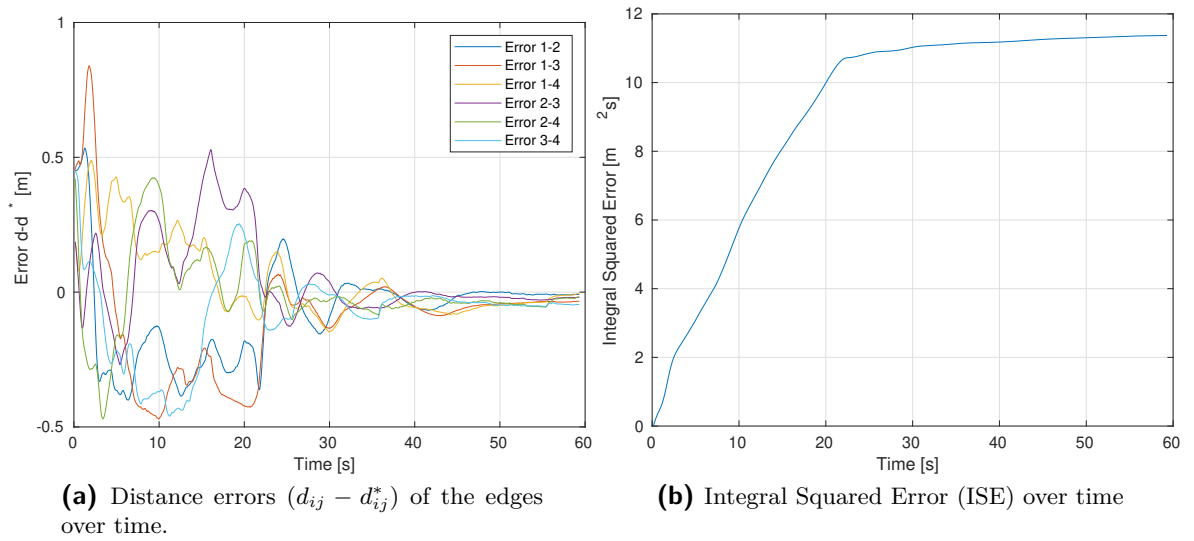


Figure 5-4: Distance errors and cumulative ISE for the group of robots using the spring-damper algorithms at 100 Hz, using LIDAR distance measurements with 0.01 m noise standard deviation.

formation at the end of the simulation compared to the starting orientation. Even though the final formation shows distance errors that are similar to the distance errors in the ideal situation, shown in Figure 5-1, this is not desired behaviour. This behaviour can be attributed to the line of sight between the robots being interrupted occasionally. The robot cannot react to the other robots which it has no line of sight to, since no distance measurement can be taken. From this, it is evident that it is necessary that the formation stays in shape such that line of sight is maintained.

5-1-3 Using global distance measurements and running the skid-steer drive controller at 15Hz

In this subsection, the 15 Hz controller used by the ROSbot is compared to the more ideal situation where the controller runs on 100 Hz. The distance measurements are calculated from the global positions of the robots. In Figure 5-5, behaviour similar to the behaviour seen in the previous section is observed, where the formation is mirrored. When robot 3 moves away from the obstacle, it passes by robot 4. When robot 2 moves away from the obstacle, it passes behind robot 1, and thus, the formation is flipped.

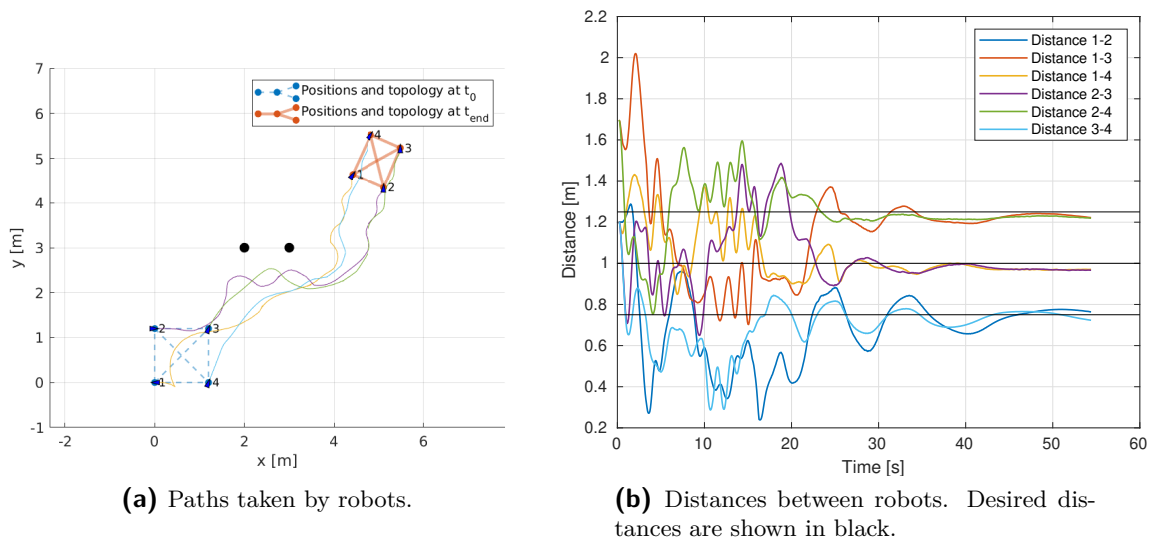


Figure 5-5: Paths of robots and distances between the robots, running the skid-steer drive controller at 15 Hz, using global distance measurements.

5-1-4 Using the LIDAR for distance measurements and running the skid-steer drive controller at 15 Hz

Lastly, the simulation of the controller and sensors as they would work in a real life scenario are simulated. The internal controller runs at 15 Hz, and the LIDAR is used for distance measurements between the robots. After robot 1, 3, and 4 have moved past the obstacle, robot 2 moves away from the obstacle and towards robot 1. This causes robot 1 to turn, and fall behind. Robot 3 tries to keep the distance to robot 1 and moves between robot 2 and robot 4. This causes the formation to get in the undesired shape with robot 3 and 4 switched locations.

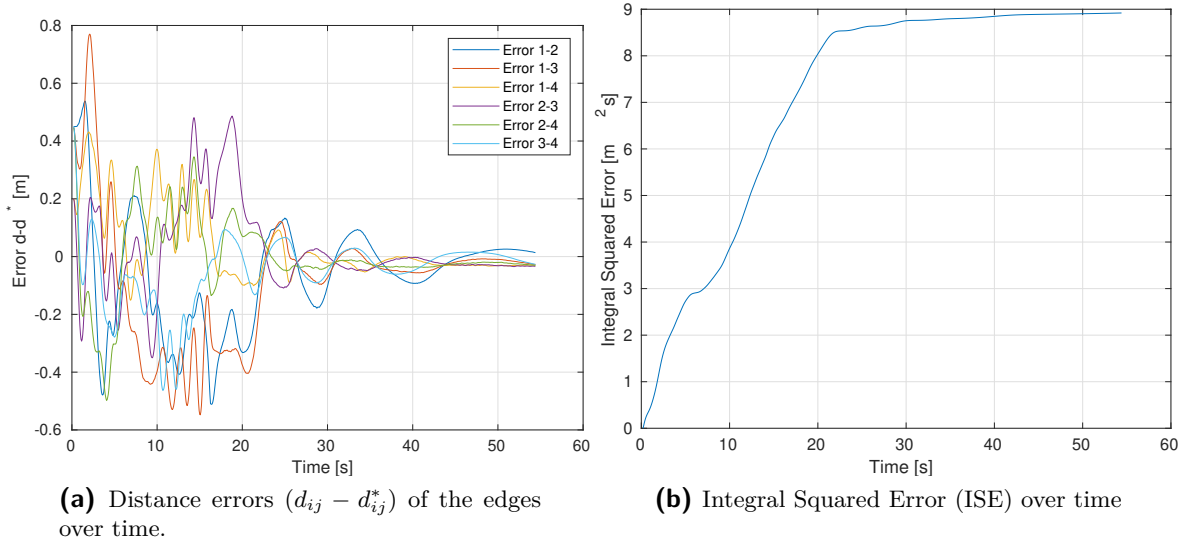


Figure 5-6: Distance errors and cumulative ISE for the group of robots using the spring-damper algorithms at 15 Hz, using global distance measurements.

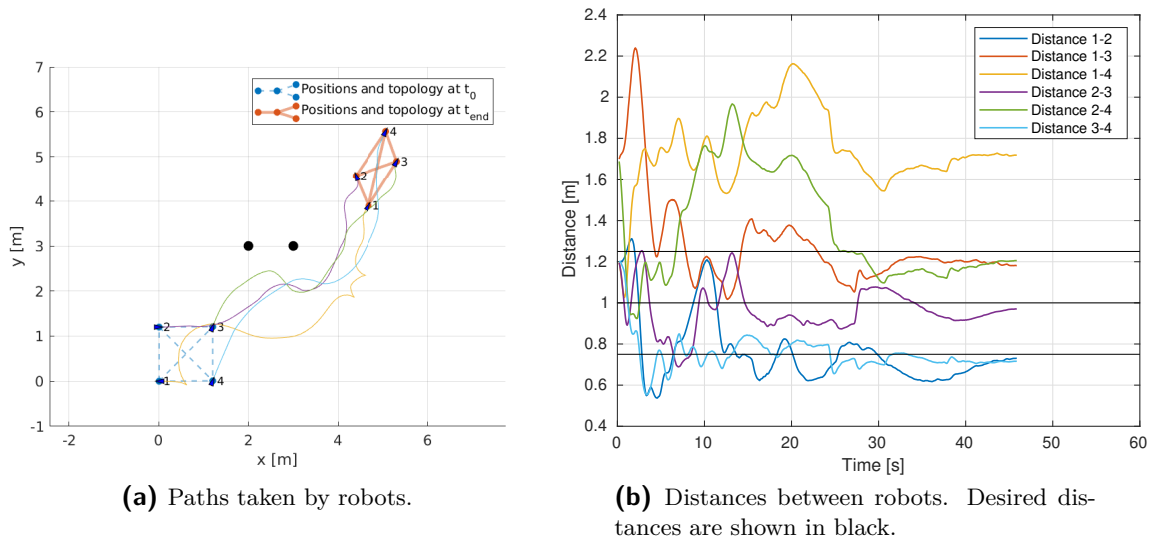
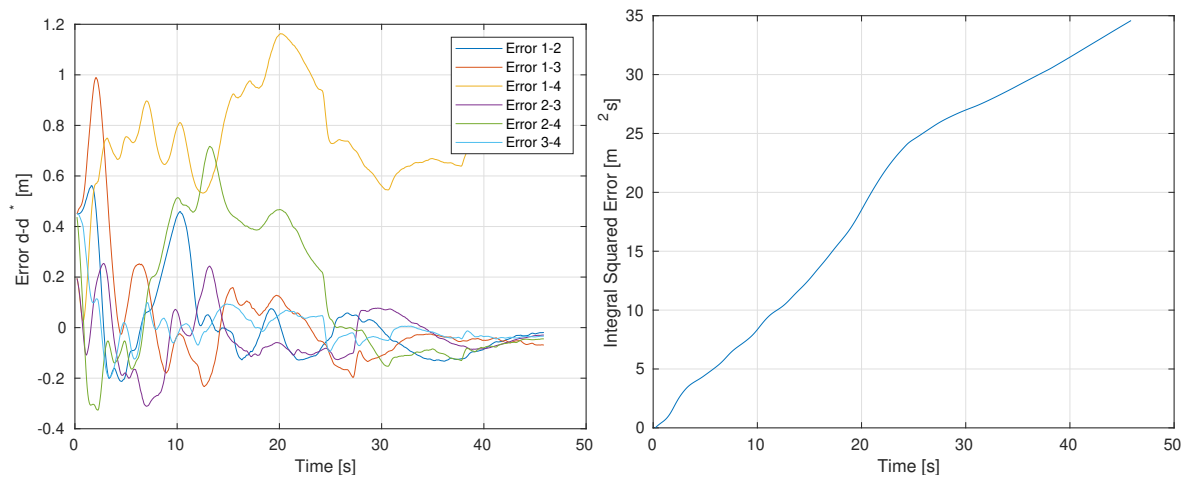


Figure 5-7: Paths of robots and distances between the robots, running the skid-steer drive controller at 15 Hz, using LIDAR measurements with 0.01 m noise standard deviation.



(a) Distance errors $(d_{ij} - d_{ij}^*)$ of the edges over time.

(b) Integral Squared Error (ISE) over time

Figure 5-8: Distance errors and cumulative ISE for the group of robots using the spring-damper algorithms at 15 Hz, using LIDAR measurements with 0.01 m noise standard deviation.

5-1-5 Finding acceptable LIDAR noise levels

Now that typical runs are shown for global distance measurements and LIDAR distance measurements with a noise standard deviation of 0.01, a multitude of simulation runs is done at various noise standard deviations, to investigate the sensitivity of the formation control performance to this LIDAR noise. It can be seen that the median ISE is similar for all of the simulations with noise standard deviation smaller than $3e-2$ m in both the 100 Hz scenario as the 15 Hz scenario. As the LIDAR noise gets larger, so does the ISE. This is in line with expectation.

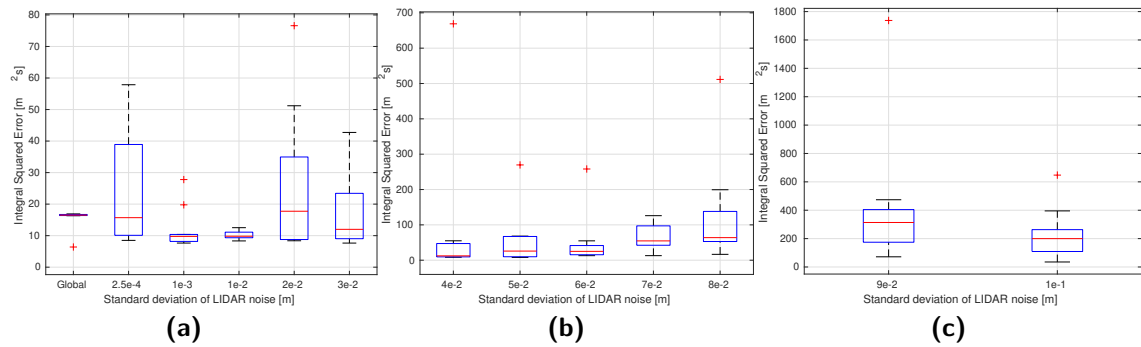


Figure 5-9: Boxplots of ISE for 10 runs with the skid-steer drive controller running at 100 Hz, first with global distance measurements and then with LIDAR distance measurements with different standard deviations for LIDAR noise: **(a)** global distance measurements, $2.5e-4$, $1e-3$, $1e-2$, $2e-2$, $3e-2$ and $4e-2$ noise, **(b)** $5e-2$, $6e-2$, $7e-2$, and $8e-2$ noise, and **(c)** $9e-2$ and $1e-1$ noise.

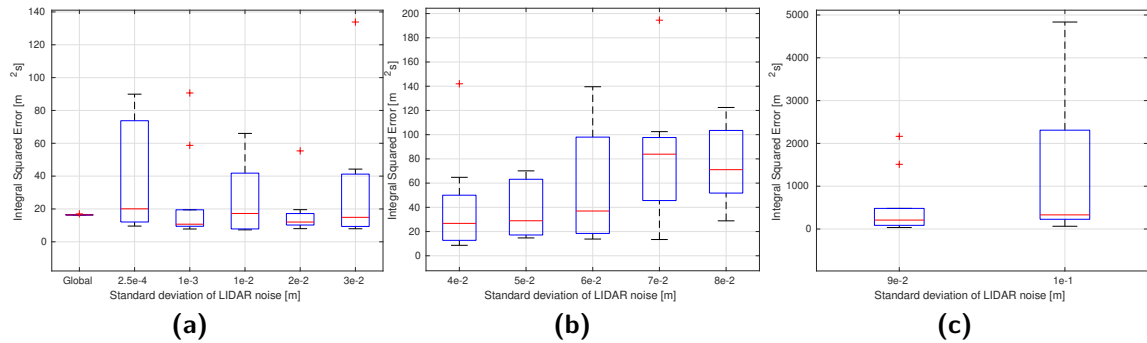


Figure 5-10: Boxplots of ISE for 10 runs with the skid-steer drive controller running at 100 Hz, first with global distance measurements and then with LIDAR distance measurements with different standard deviations for LIDAR noise: **(a)** global distance measurements, $2.5e-4$, $1e-3$, $1e-2$, $2e-2$, $3e-2$ and $4e-2$ noise, **(b)** $5e-2$, $6e-2$, $7e-2$, and $8e-2$ noise, and **(c)** $9e-2$ and $1e-1$ noise.

5-2 Consensus Algorithm

In this section, the consensus algorithm, given by Eqs. (3-12)-(3-13), is tested on the team of robots. As in the previous sections, an ideal situation is tested first. Then, the simulation is made more realistic first by using LIDAR for the distance measurements, then by setting the controller frequency to the value of 15 Hz while using perfect distance measurements, and finally by using LIDAR for distance measurements while also setting the controller frequency to the value of 15 Hz. The sensitivity to noise on the LIDAR is also evaluated by simulating the algorithm multiple times with different levels of noise. From these runs, looking at the ISE will give insight in what noise level is acceptable for this setup. The desired formation shape is given by Eq. (5-1).

5-2-1 Ideal situation: using global distance measurements and running the skid-steer drive controller at 100 Hz

First, the consensus algorithm is tuned for the situation where perfect distance measurements are used and the skid-steer drive controller runs at 100 Hz. The tuned parameters are given in Table 5-2. The paths and distances of the robots, as they converge towards the desired formation and move past the obstacles to the desired location, are shown in Figure 5-11. This figure shows excellent convergence to the desired formation and location. In Figure 5-12, the distance errors and cumulative ISE over time are shown. The ISE of this run is thus ± 2.75 , which is considered excellent performance.

5-2-2 Using LIDAR for distance measurements and running the skid-steer drive controller at 100 Hz

In this run, the LIDAR is introduced for distance measurements. The noise on the LIDAR measurements is zero-mean Gaussian noise with a standard deviation of 0.01. Still, the desired formation and location are reached, but the imperfect LIDAR distance measurements cause the robots to react, leading to the formation oscillating around the desired location. In Figure 5-13, this is shown. The individual errors and the cumulative ISE are shown in Figure 5-14.

Table 5-2: Control gains for consensus system.

Parameter	Value
$k_{rotational}$	2
k_{linear}	0.75
$k_{p_{goal}}$	0.1
$k_{d_{goal}}$	0

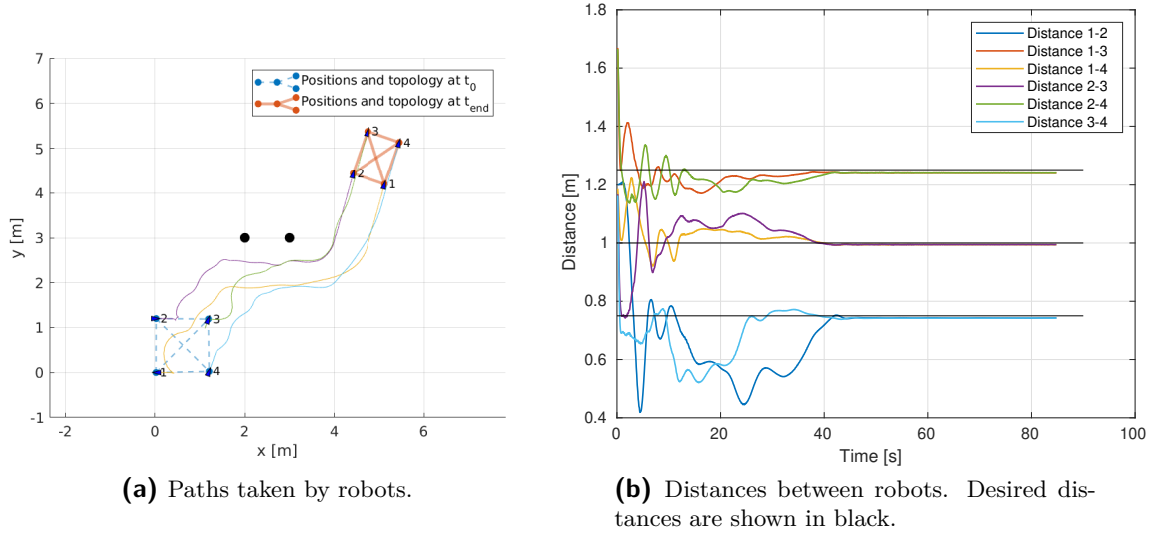


Figure 5-11: Paths of robots and distances between the robots, using the consensus algorithm in the ideal situation of the skid-steer drive controller running at 100 Hz and using global distance measurements.

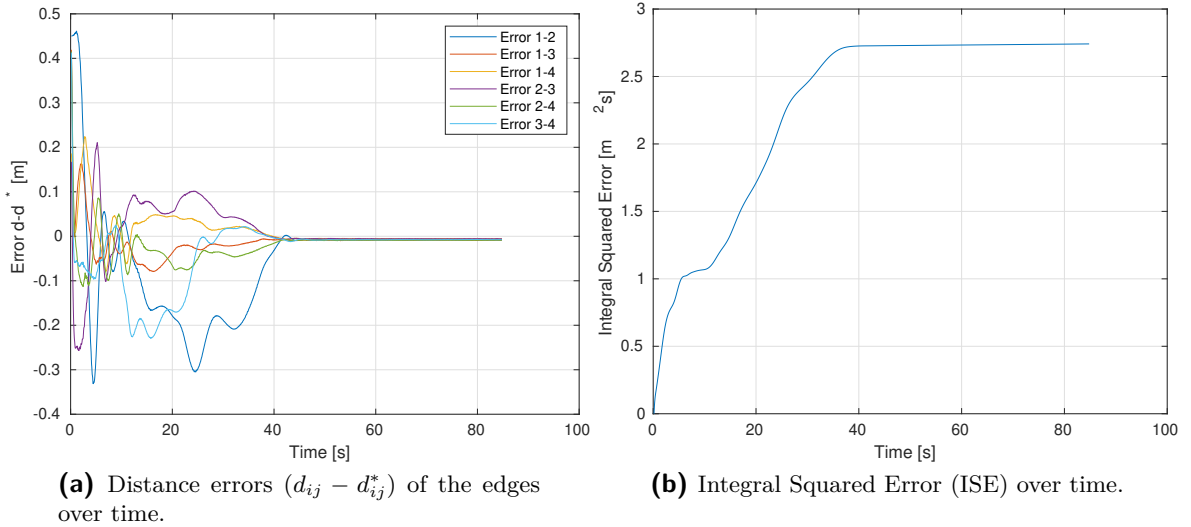


Figure 5-12: Distance errors and cumulative ISE for the group of robots using the consensus algorithms at 100Hz, using global positions for distance measurements.

5-2-3 Using global distance measurements and running the skid-steer drive controller at 15 Hz

In this subsection, the results of the consensus algorithm running at 15 Hz with global distance measurements are shown. It can be seen in Figure 5-15, that at 15 Hz the robots converge to the desired location and formation shape. After the obstacles are avoided, the distances quickly converge to the desired distances.

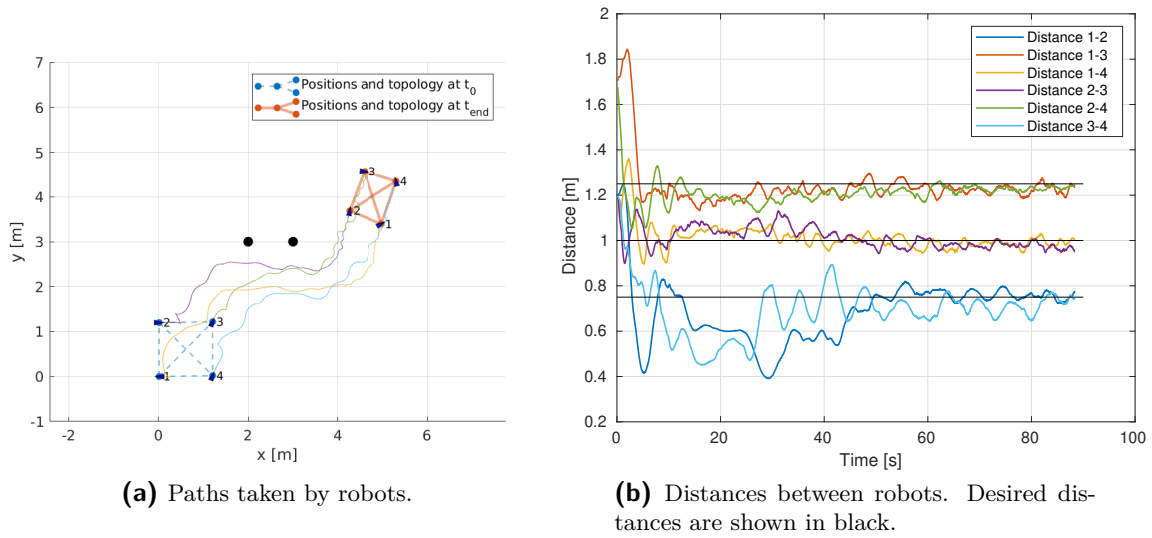


Figure 5-13: Paths of robots and distances between the robots, using the consensus algorithm running the skid-steer drive controller at 15 Hz, and using LIDAR distance measurements with 0.01 m noise standard deviation.

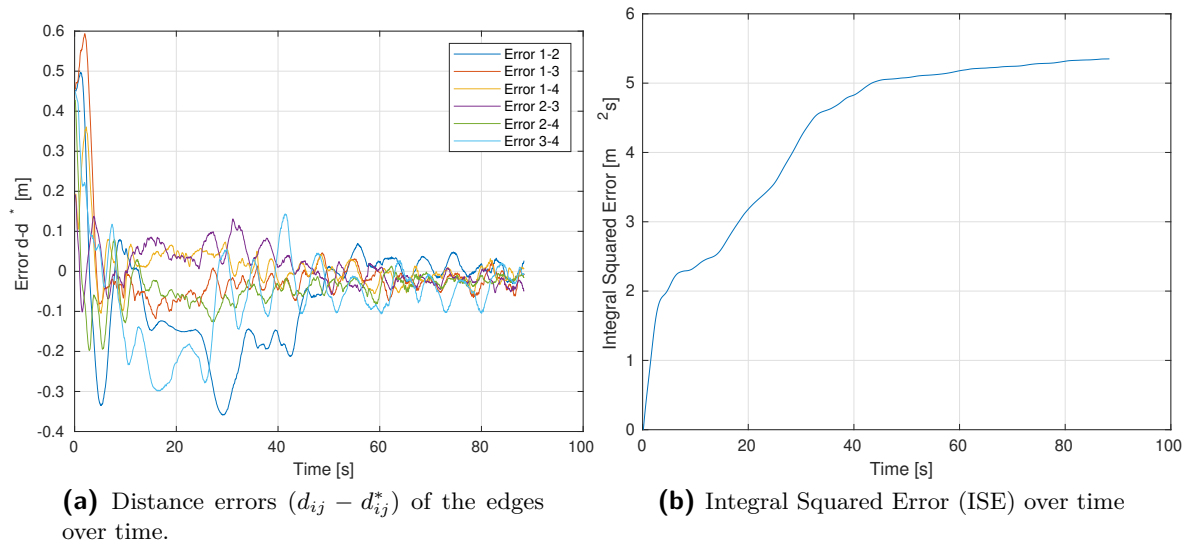


Figure 5-14: Distance errors and cumulative ISE for the group of robots using the consensus algorithms at 100 Hz, using LIDAR distance measurements with noise standard deviation of 0.01 for distance measurements with 0.01 m noise standard deviation.

5-2-4 Using the LIDAR for distance measurements and running the skid-steer drive controller at 15 Hz

Finally, the consensus algorithm is tested in the most accurate scenario, with the controller running at 15 Hz and the distance measurements coming from the LIDAR and object detector. Results are shown in Figure 5-17. Similarly to the results in the case of using the LIDAR for

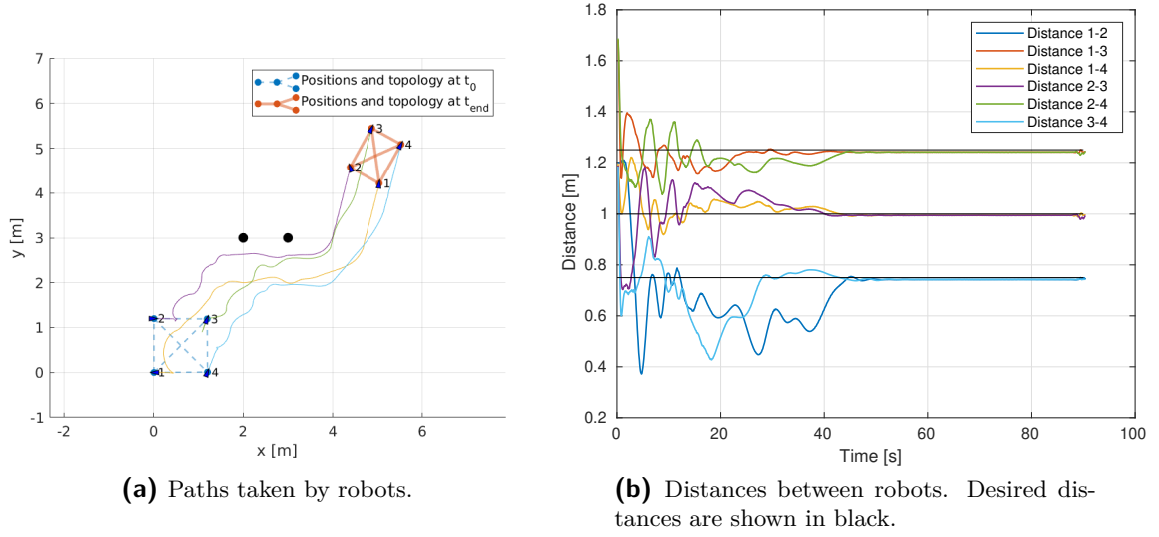


Figure 5-15: Paths of robots and distances between the robots, using the Consensus algorithm. 15Hz, global distance measurements.

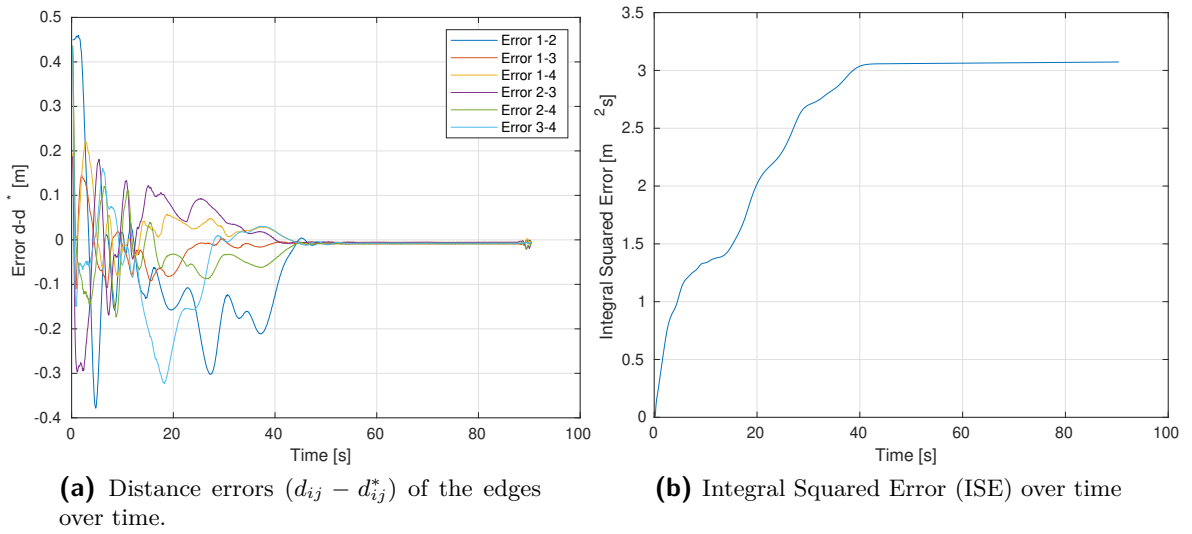


Figure 5-16: Distance errors and cumulative ISE for the group of robots using the consensus algorithms at 15Hz, using global distance measurements.

distance measurements and the controller at 100 Hz, the robots oscillate around the desired formation shape, but are still able to maintain the formation.

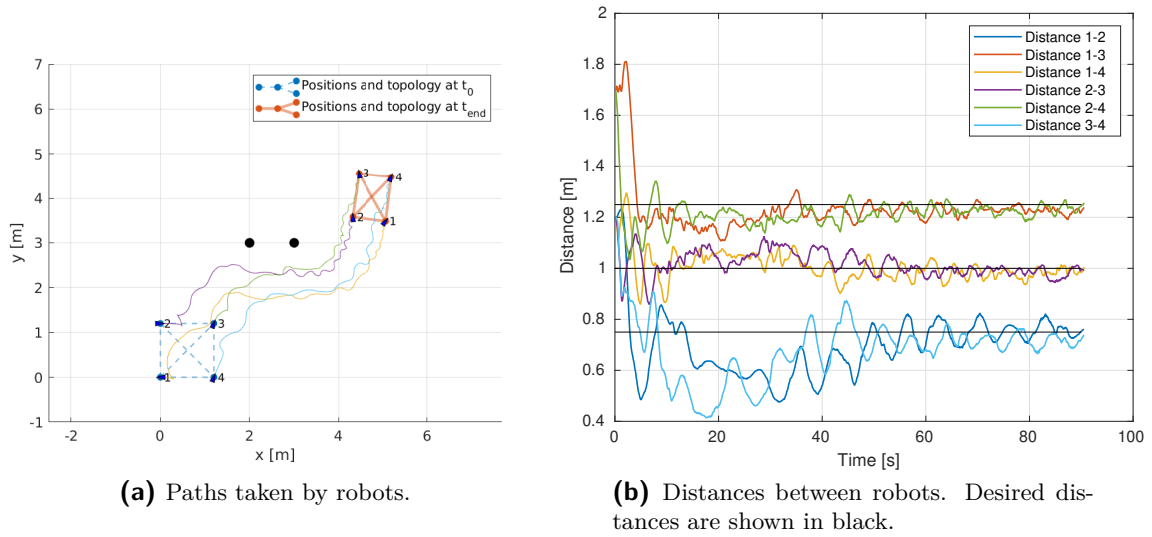


Figure 5-17: Paths of robots and distances between the robots, using the consensus algorithm in the "real" scenario, with the skid-steer drive controller running at 15 Hz and using LIDAR distance measurements with 0.01 m noise standard deviation.

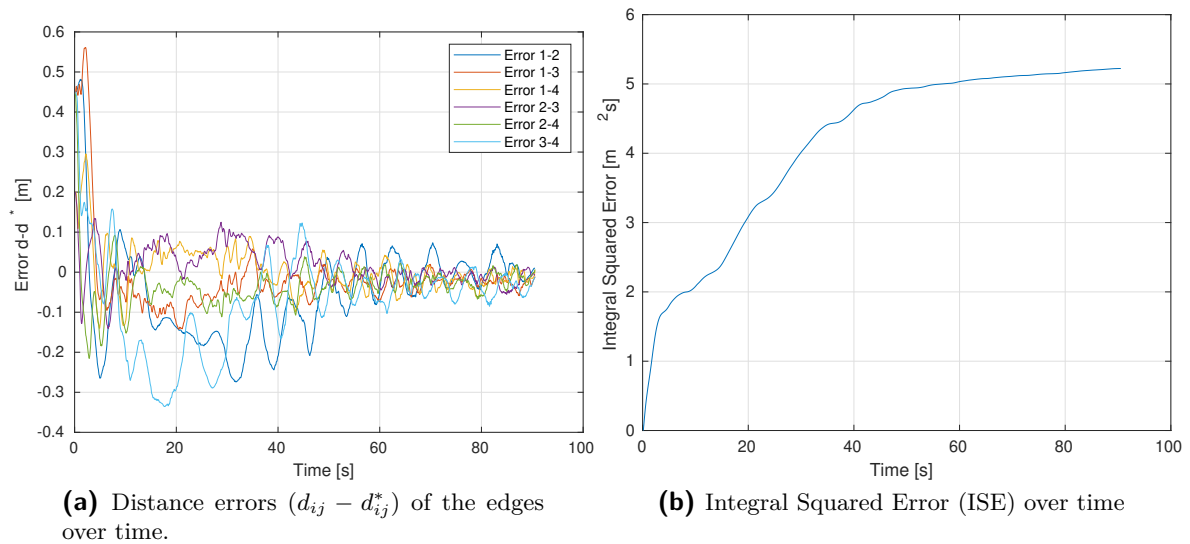


Figure 5-18: Distance errors and cumulative ISE for the group of robots using the consensus algorithms at 15 Hz, using LIDAR with noise standard deviation of 0.01 for distance measurements with 0.01 m noise standard deviation.

5-2-5 Finding acceptable LIDAR noise levels

In this subsection, a number of simulations are done with various different noise levels, to investigate the sensitivity of the consensus algorithm to the LIDAR noise. The scenario is simulated 260 times in total: 130 times at 100 Hz, and 130 times at 15 Hz, where for each of these frequencies 10 runs are done using global positions to calculate distances, and 10 runs are done for each of the 12 different standard deviations for LIDAR noise. The results in terms of ISE for the 100 Hz are shown in Figure 5-19, the results in terms of ISE for the 15 Hz are shown in Figure 5-20. These ISE values are shown as a box plot.

By looking at the movements and paths of the robot formation, it is found that for this scenario, typically, an $ISE < 10$ indicates that the robots were able to maintain formation without collisions. An ISE of 10 to 20 typically has minor collisions, where the formation is regained. A value of $ISE > 20$ means total failure, with robots tipping over and being unable to move.

From these box plots, it can be seen that the LIDAR distance measurements always perform worse than the global distance measurements, even at low noise levels. The performance of the group of robots, in terms of median ISE is not very sensitive to the noise standard deviation up to a standard deviation of ± 0.03 m. This is likely because the noise on the LIDAR is given as the noise on the individual points measured by the LIDAR. The distance measurement used by the developed formation control package comes from clustering several LIDAR measurements, and fitting a circle around this cluster. The distance of the centre of this circle is used as distance measurement by the formation control package. When the object is represented by seven LIDAR measurements in one time step, and by six in the next, the centre of the detected object moves. It is likely that these movements dominate over the noise on the individual LIDAR measurements, at low noise levels. At higher noise levels, the group of robots is no longer able to maintain the formation or to evade the obstacles, which is shown by the quickly increasing ISE.

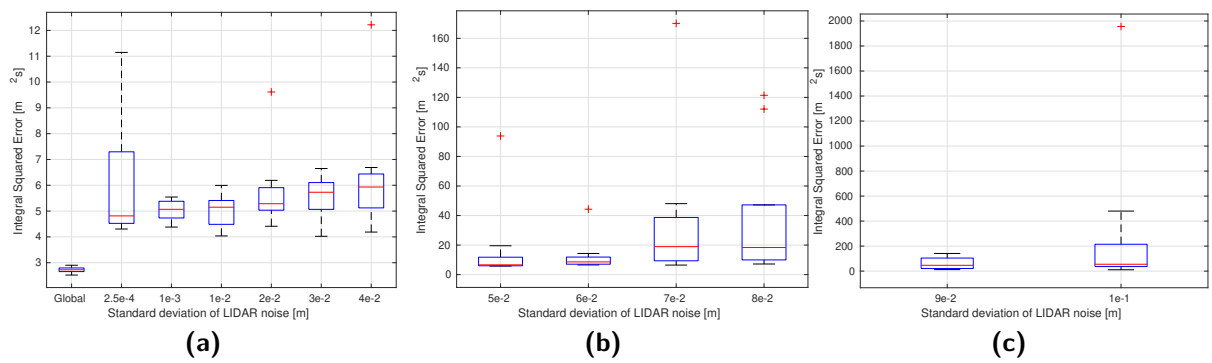


Figure 5-19: Boxplots of ISE for 10 runs with the skid-steer drive controller running at 100 Hz, first with global distance measurements and then with LIDAR distance measurements with different standard deviations for LIDAR noise: **(a)** global distance measurements, 2.5e-4, 1e-3, 1e-2, 2e-2, 3e-2 and 4e-2 noise, **(b)** 5e-2, 6e-2, 7e-2, and 8e-2 noise, and **(c)** 9e-2 and 1e-1 noise.

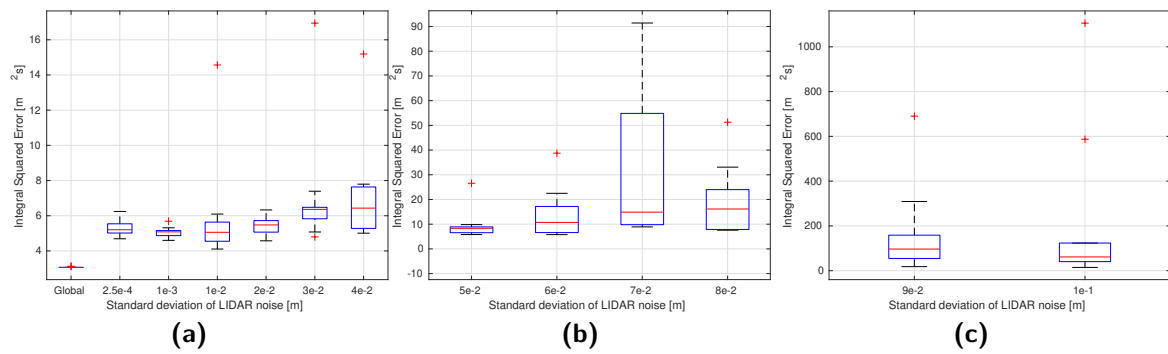


Figure 5-20: Boxplots of ISE for 10 runs with the skid-steer drive controller running at 15 Hz, first with global distance measurements and then with LIDAR distance measurements with different standard deviations for LIDAR noise: **(a)** global distance measurements, 2.5e-4, 1e-3, 1e-2, 2e-2, 3e-2 and 4e-2 noise, **(b)** 5e-2, 6e-2, 7e-2, and 8e-2 noise, and **(c)** 9e-2 and 1e-1 noise.

5-3 Exponential potential function

In this section, the exponential potential function given by Eqs. (3-14)-(3-16) is tested. As with the other algorithms, the controller is first tuned to work on the situation using perfect distance measurements, and then tested in the other scenarios to determine the sensitivity to distance measurement noise and a lower controller frequency. The tuned parameters are shown in Table 5-3.

5-3-1 Ideal situation: using global distance measurements and running the skid-steer drive controller at 100 Hz

First, the controller is tuned on the group of mobile robots, with the controllers running at 100 Hz and using the perfect distance measurements as input. This controller has a switching behaviour for the obstacle avoidance. In Figure 5-21, the purple line shows the path robot 2 takes, where the switching behaviour causes the robot to circle the obstacle. It can be seen that the robot switches to this behaviour when the first obstacle is within a range of 1 m, and takes a hard turn right to circle the obstacle. After the obstacles have been evaded, it keeps moving around the obstacle for too long, not switching back to the formation keeping and moving in the desired direction.

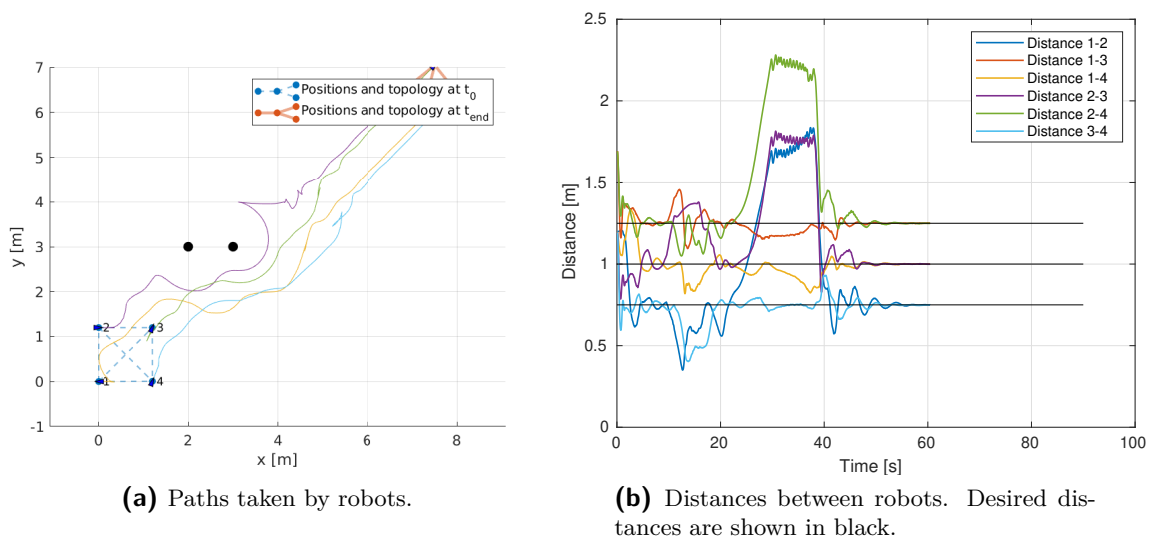


Figure 5-21: Paths of robots and distances between the robots, using the exponential algorithm. In this ideal scenario, the skid-steer drive controller is running at 100 Hz and global distance measurements are used.

5-3-2 Using LIDAR for distance measurements and running the skid-steer drive controller at 100 Hz

When using LIDAR combined with the obstacle detector for distance measurements, extra oscillations are introduced, but the desired formation shape is regained after the obstacles are

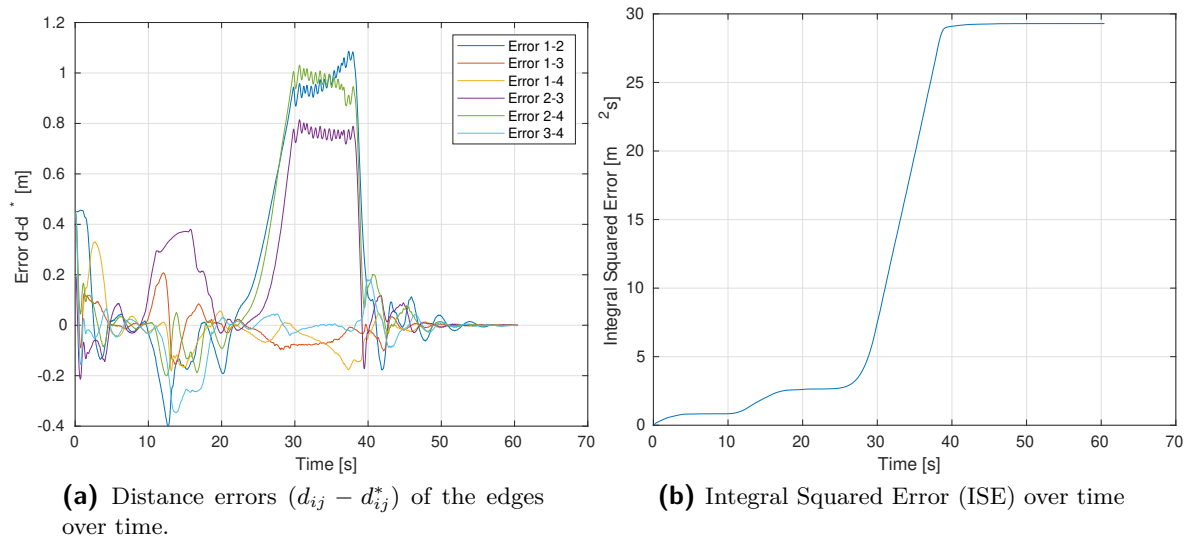


Figure 5-22: Distance errors and cumulative ISE for the group of robots using the exponential algorithms at 100 Hz, using global distance measurements.

avoided. These oscillations cause the robots to move forward slower. The time it takes for the robots to move forward and pass the obstacles, is much longer than when the global distance measurements are used. In Figure 5-23, the paths and inter-agent distances are shown for this scenario.

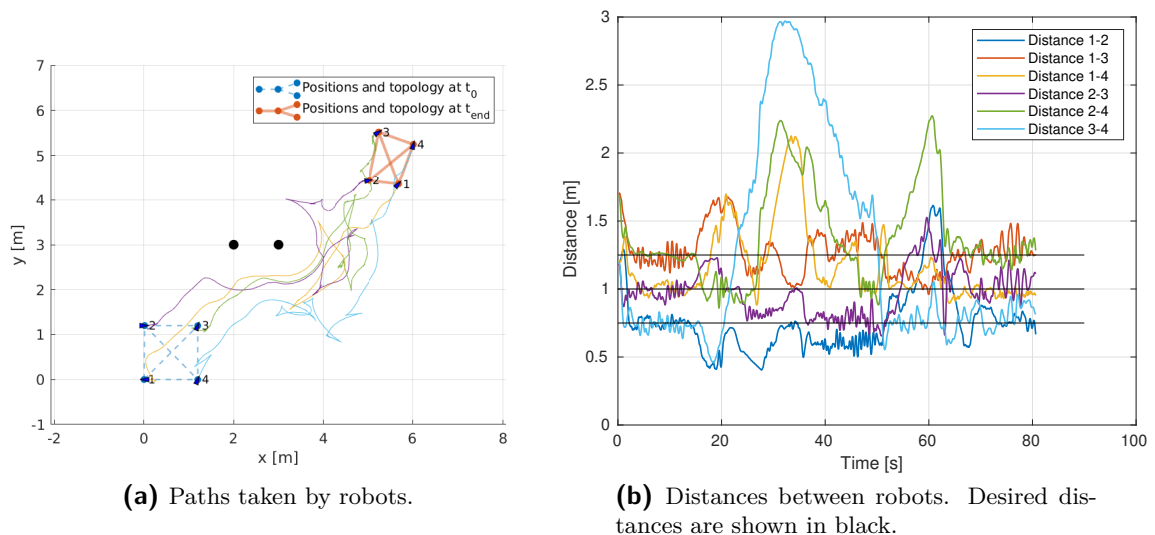
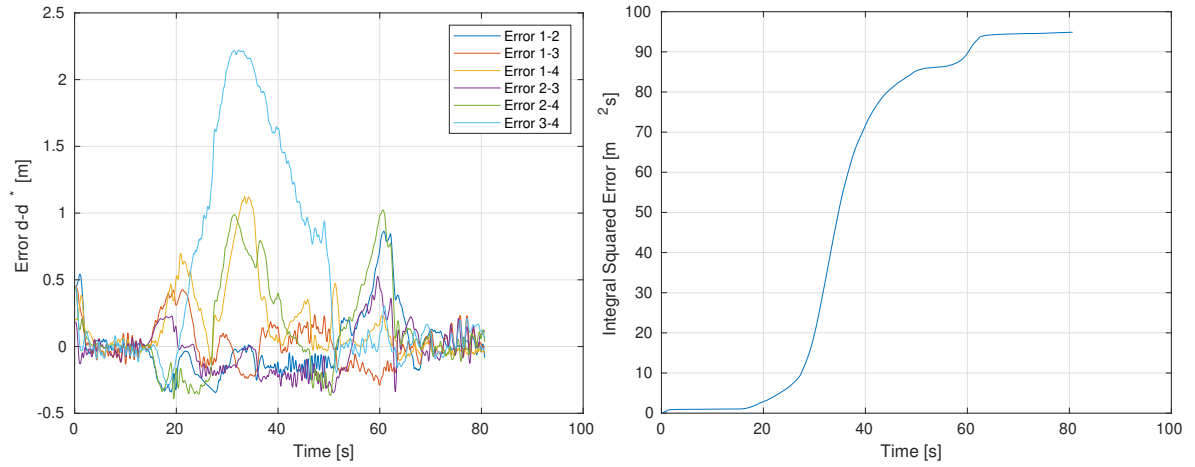


Figure 5-23: Paths of robots and distances between the robots, using the exponential algorithm. In this scenario the skid-steer drive controller is running at 100 Hz and LIDAR distance measurements with 0.01 m noise standard deviation are used.



(a) Distance errors ($d_{ij} - d_{ij}^*$) of the edges over time.

(b) Integral Squared Error (ISE) over time

Figure 5-24: Distance errors and cumulative ISE for the group of robots using the exponential algorithms at 100 Hz, using LIDAR with noise standard deviation of 0.01 m for distance measurements.

5-3-3 Using global distance measurements and running the skid-steer drive controller at 15 Hz

When the global distance is used combined with the controller frequency of 15 Hz, the high gain of the exponential controller causes overshoot on the desired distances. The robots are slow to react to one another, and will thus cause the formation to oscillate around the desired formation shape. In Figure 5-25b, these oscillations are shown. Even though there are many undesired movements, the robots stay close enough to the desired formation shape to not completely disrupt the formation.

5-3-4 Using the LIDAR for distance measurements and running the skid-steer drive controller at 15 Hz

Finally, the exponential consensus algorithm is tested in the real life scenario. In this scenario, the robots use LIDAR and obstacle detection to measure the distances, and the kinematic skid-steer drive controller runs at a frequency of 15 Hz. In Figure 5-27, the results of this

Table 5-3: Control gains for exponential potential function.

Parameter	Value
$\theta_{desired}$	$\frac{1}{4}\pi$
$k_{formation}$	2.8
v_d	0.2
k_u	0.2
k_ω	3

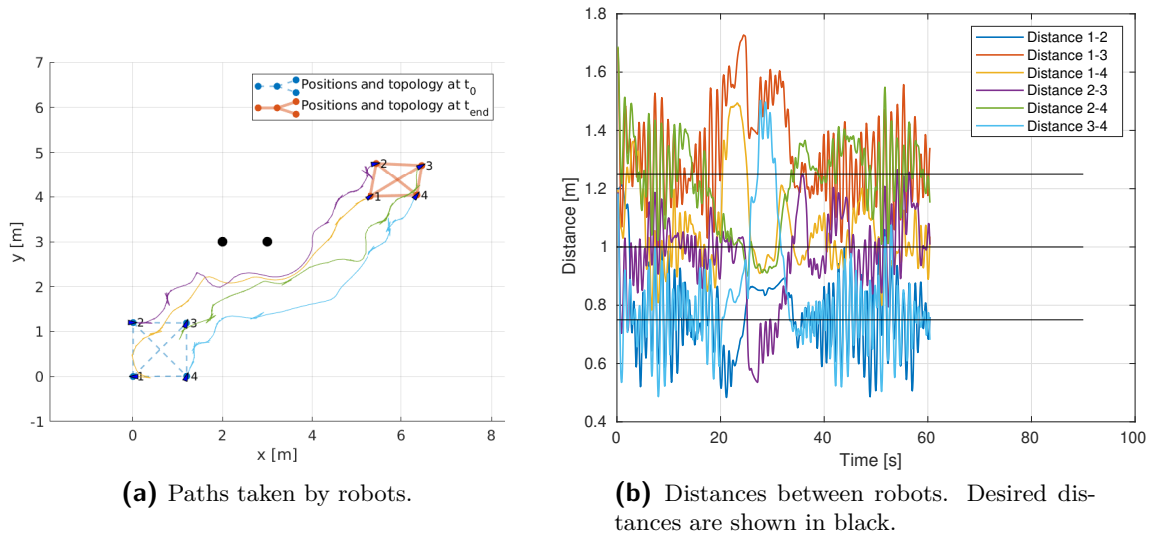


Figure 5-25: Paths of robots and distances between the robots, using the exponential algorithm. In this scenario the skid-steer drive controller is running at 15 Hz and global distance measurements are used.

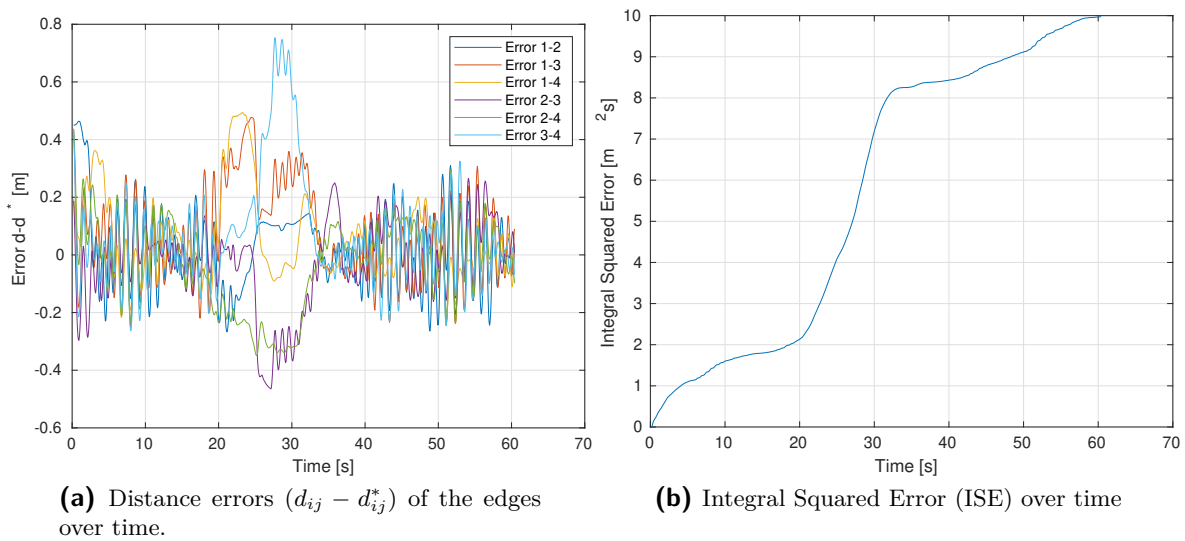


Figure 5-26: Distance errors and cumulative ISE for the group of robots using the exponential algorithms at 15 Hz, using global distance measurements.

simulation are shown. Due to large oscillations, robot 1 flips itself over without collision. Robot 2 and robot 3 circumvent the obstacles, and robot 4 tries to keep the formation with the now stationary robot 1 and the other two robots, which is not possible. The formation shape is broken and robot 1 is no longer functioning.

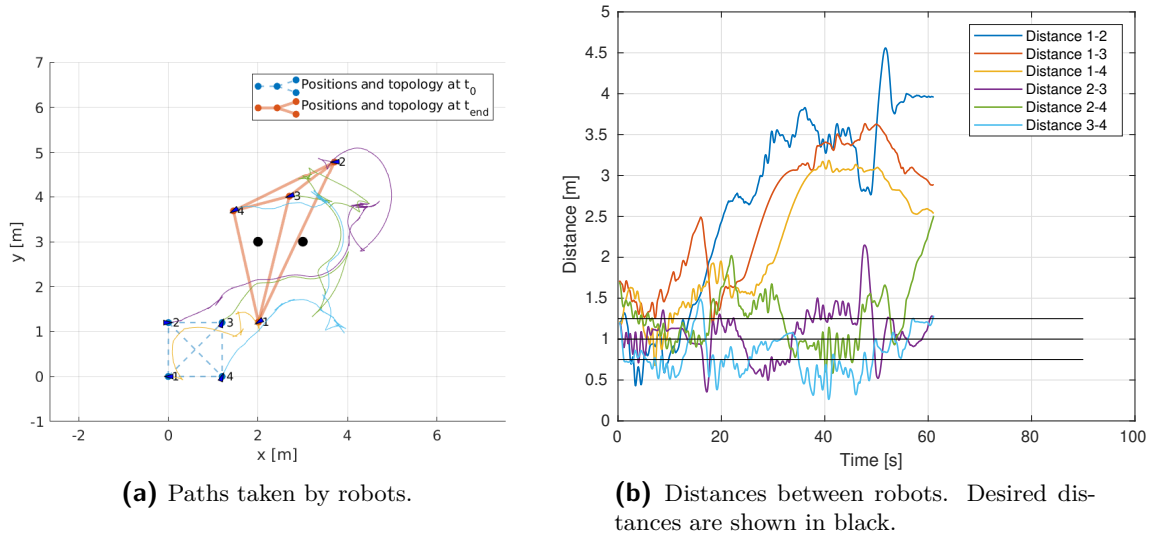


Figure 5-27: Paths of robots and distances between the robots, using the exponential algorithm. In this scenario the skid-steer drive controller is running at 15 Hz and LIDAR distance measurements with 0.01 m noise standard deviation are used.

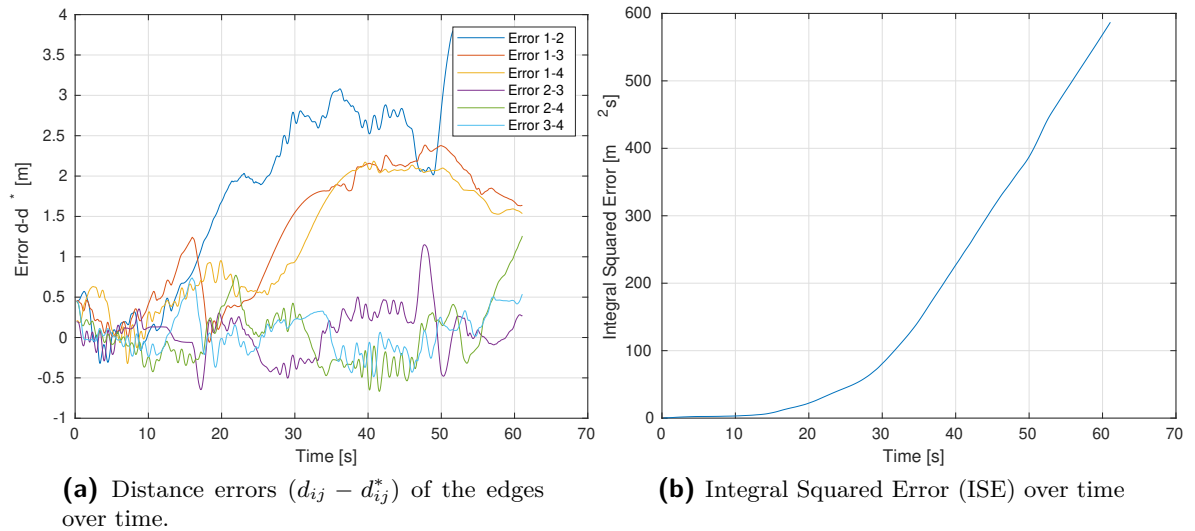


Figure 5-28: Distance errors and cumulative ISE for the group of robots using the exponential algorithms at 15 Hz, using LIDAR distance measurements with noise standard deviation of 0.01 m for distance measurements.

5-3-5 Finding acceptable LIDAR noise levels

Like with the previously discussed formation control algorithms, the scenario is simulated 10 times for global distance measurements and for each of the 12 LIDAR noise standard deviations, to find a critical noise level where the formation control is no longer possible. The results of these simulation are shown in Figure 5-29 for the controller running at 100 Hz, and in Figure 5-30 for the controller running at 15 Hz. For both the situations, it is very clear that the error is smallest when using global distance measurements. When running at 100 Hz, using LIDAR distance measurements, the ISE stays relatively constant as long as the noise standard deviation is lower than $6e-2$ m, with some high outliers even for the low noise levels. Interestingly, the 15 Hz scenario has similar performance at noise standard deviations lower than $5e-2$ m. Above this, the formation is not kept, which results in very high values for the ISE.

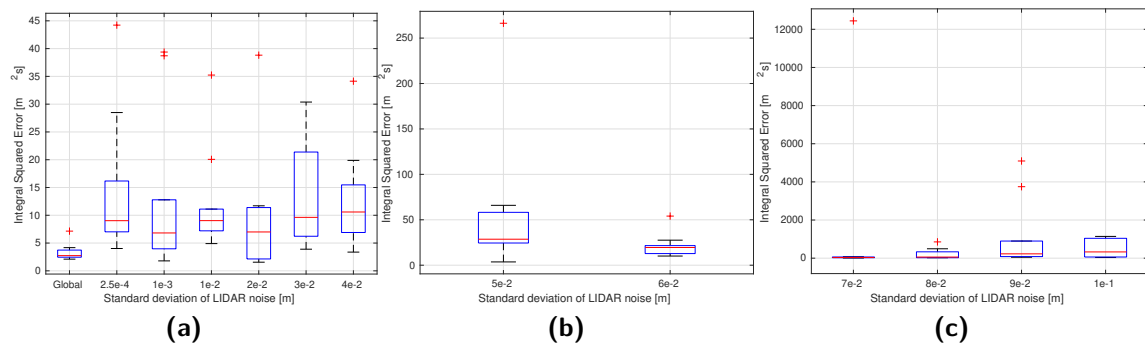


Figure 5-29: Boxplots of ISE for 10 runs with the skid-steer drive controller running at 100 Hz, first with global distance measurements and then with LIDAR distance measurements with different standard deviations for LIDAR noise: **(a)** global distance measurements, $2.5e-4$, $1e-3$, $1e-2$, $2e-2$, $3e-2$ and $4e-2$ m noise standard deviation, **(b)** $5e-2$, $6e-2$ m noise standard deviation, and **(c)** $7e-2$, $8e-2$, $9e-2$ and $1e-1$ m noise standard deviation.

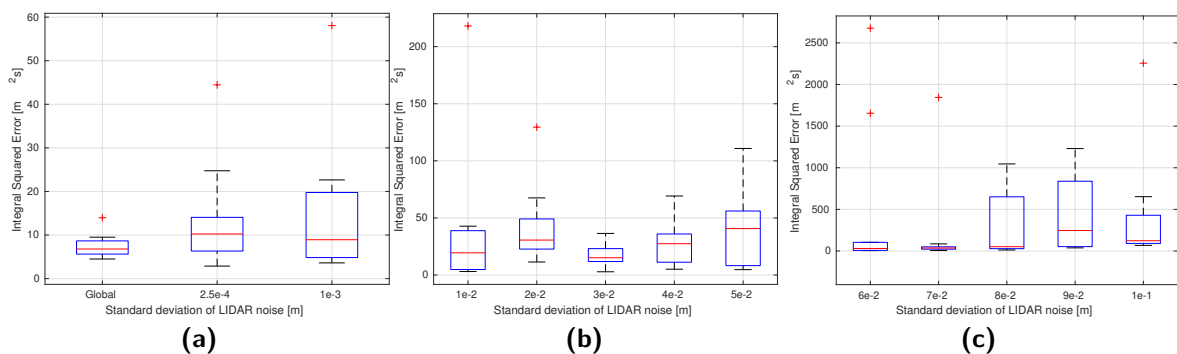


Figure 5-30: Boxplots of ISE for 10 runs with the skid-steer drive controller running at 15 Hz, first with global distance measurements and then with LIDAR distance measurements with different standard deviations for LIDAR noise: **(a)** global distance measurements, $2.5e-4$ and $1e-3$ noise, **(b)** $1e-2$, $2e-2$, $3e-2$, $4e-2$ and $5e-2$ noise, and **(c)** $6e-2$, $7e-2$, $8e-2$, $9e-2$ and $1e-1$ noise.

Conclusions and Recommendations

The use of distributed multi-agent formation control for non-holonomic mobile robots shows great promise in practical environments. In this thesis, it is shown that holonomic potential functions can be adapted to work on a team of non-holonomic skid-steer drive mobile robots, and a Robot Operating System (ROS) package is developed in which these potential functions are implemented. However, testing of this package shows that sensitivity to platform-specific parameters, such as controller frequency and the way distance measurements are sensed, or the number of robots in the formation, make it a difficult task to tune these controllers. In the following sections, both the conclusions and the corresponding recommendations are discussed. Section 6-1 covers the conclusions to the research questions introduced in Chapter 1, after which Section 6-3 covers the recommendations that follow from these conclusions.

6-1 Conclusions

This section elaborates on the overall conclusion to the main research question that was posed in Chapter 1. This conclusion is drawn on the basis of the answers to each of the four different research subquestions. Each of these research subquestions are answered in a separate section. Thereafter, it is assessed whether the research objective was reached.

6-1-1 Adapting holonomic control laws for use on non-holonomic robots

The first research subquestion that was composed to partly answer the main research question reads:

In what different ways can holonomic distance based formation control algorithms be implemented on a team of non-holonomic skid-steer drive mobile robots?

This question was answered in Chapter 2, and is briefly summarised here. In Chapter 2, it was shown that for a given holonomic distance-based potential function, seven distinct

adaptations can be used to implement these on non-holonomic skid-steer drive or unicycle model mobile robots. These seven adaptations are:

- Three different ways by offsetting the potential sideways to the robot direction of motion and using this as rotational input for the robot:
 - Always offsetting the sideways potential forward, causing the robots to favour moving forward;
 - Summing all the potential, then offsetting the sideways potential along the potential in the direction of the robot heading. This does not cause the robots to favour moving forward, and the paths taken are closer to the paths taken by holonomic mobile robots compared to the adaptations which favour moving forwards;
 - Offsetting each of the separate potentials along the potential in the direction of robot heading. This may cause unwanted local equilibria.
- Four different ways by taking the angle between the robot heading and the potential as rotational input for the robot.
 - Calculating the angle between the total potential and the robot heading using $\text{atan2}(F_y, F_x)$, which favours moving forward;
 - Calculating the angle between the total potential and the robot heading using $\text{atan}(F_y/F_x)$, which does not favour moving forwards over moving backwards. The paths taken by the robots will be closer to the paths taken by holonomic agents;
 - Calculating the angle between each of the separate potentials and the robot heading using $\text{atan2}(F_y, F_x)$, which can cause the robots to get stuck in local equilibria;
 - Calculating the angle between each of the separate potentials and the robot heading using $\text{atan}(F_y/F_x)$, which can cause the robots to get stuck in local equilibria.

From these seven adaptations, three were deemed unsuitable for use on a team of mobile robots, because there is the possibility to get stuck in local undesired equilibria. These adaptations used each of the separate potentials to derive rotational control inputs. The four remaining holonomic control law adaptations that were deemed suitable are the adaptations where the total potential is used. Two of the adaptations cause the robots to have a preferred direction of movement, i.e. the robots will move mostly forward. This is found to be advantageous if the robots have other front-facing sensors, such as a camera. The last two adaptations show no preferred direction of movement, i.e. the robot might move backwards towards its goal. This is found to be advantageous when no front-facing sensors are used, since the paths taken by the robot are generally shorter than the paths taken by adaptations which favour moving forward. These shorter paths are closer to the paths that would be taken by a holonomic robot.

6-1-2 Effects of LIDAR distance measurements on formation control performance

The second research subquestion can be answered based on the simulations that were shown and discussed in Chapter 5. This second subquestion, as posed in Chapter 1, reads:

Can object detection from Light Detection And Ranging (LIDAR) data be used to measure distance between robots and obstacles, and how does this affect performance compared to using the global positions of the robots to measure distance?

LIDAR distance measurements are shown to be useful for distance based formation control. The performance in terms of Integral Squared Error (ISE) is worse for the LIDAR distance measurements than for the global distance measurements, but acceptable results were seen using the consensus algorithm. Using the spring-damper algorithm, the formation "flipped" to a formation different than the initial positions of the robots, but where all the desired distances were met. Using the LIDAR distance measurements at 100 Hz in the exponential potential function, the noise causes much redundant movements, but the formation was regained after the obstacles were avoided.

6-1-3 Influence of control input frequency on object detection

The third research subquestion relates to the influence of the control input frequency on the performance of the team of robots in the formation task. It can be answered based on the presented results in Chapter 5. The corresponding research subquestion is:

How does the control frequency of the low-level kinematic controller influence performance?

Each of the algorithms was tuned to work at 100 Hz using the robots global positions for distance measurements. When running the skid-steer drive controller at 15 Hz, the performance is shown to be worse, with undesirable behaviour happening with the spring-damper algorithm, and even collisions and crashes for the exponential potential function. However, it was also shown that the consensus algorithm performance remains satisfactory, with no significant increase in ISE when the formation is simulated using a 15 Hz skid-steer drive controller. Using the spring-damper algorithm, the formation was not kept, but regained with robots having switched position, or the formation being "flipped over". When running the controller at 15 Hz, the exponential potential function showed large oscillations around the desired distance when using global distance measurements, and collisions when using LIDAR distance measurements. When running at 15 Hz, the consensus algorithm performed best.

6-1-4 Acceptable LIDAR noise levels

The fourth, and last, research subquestion that was posed in this thesis concerns acceptable LIDAR noise levels:

What level of sensor noise is acceptable when using LIDAR distance measurements for formation control?

By simulating the formation control scenario with different noise standard deviations, the effects of this noise on the performance is evaluated. This is done for the three implemented

formation control algorithms, with controller frequencies of 15 Hz and 100 Hz. In these scenarios, the performance of the global distance measurements is compared to LIDAR distance measurements with various standard deviations.

It is shown that in all cases, the global distance measurements give better median performance than the LIDAR distance measurements. With LIDAR distance measurements, the performance stays around the same level for low noise standard deviations, until a certain threshold is reached, after which the ISE gets larger with larger sensor noise.

For the spring-damper system, noise standard deviations up to 3×10^{-2} m produce satisfactory results. The consensus algorithm produces satisfactory results at noise standard deviations smaller than 4×10^{-2} m, in both the 100 Hz and 15 Hz scenario, with some outliers with higher ISE values below this threshold. Above this threshold, formation is not kept as well, resulting in higher ISE values. For the exponential potential function running at 100 Hz, the performance using LIDAR distance measurements stays roughly the same for standard deviations smaller than 4×10^{-2} m. For the 15 Hz case, standard deviations $< 1 \times 10^{-3}$ produce acceptable results. In each of the cases, except for the exponential potential function at 15 Hz, good results can be produced with the LIDAR distance measurements at noise standard deviations that the RPLidar-A2M8 produces.

6-1-5 Main conclusion

Now, on the basis of the answers to the four research subquestions discussed above, the main research question may be answered. The main research question of this thesis is:

What are the practical limitations on formation control, and what effect do these limitations have on the functioning of different distance based control algorithms when applying them to a team of skid-steer drive mobile robots in a simulated laboratory environment?

Based on the research presented here, the following limitations were identified and explored:

Firstly, most formation control algorithms presented in literature focus on omnidirectional, holonomic mobile robots. Seven adaptations modifying these holonomic control algorithms to work on non-holonomic mobile robots were presented. These can be used to expand the number of formation control algorithms available for non-holonomic mobile robots, as well as to categorise the existing non-holonomic formation control algorithms.

Secondly, one of the main practical limitations on formation control for non-holonomic mobile robots is the limited sensing capabilities of the mobile robots. From 2D LIDAR data, obstacles and robots can be detected, and their position can be sensed in the local coordinate frames of the robots. However, identifying the difference between robots and obstacles, and between the individual robots, could not be done using only 2D LIDAR data. If this limitation is overcome, LIDAR distance measurements combined with obstacle detection and tracking can be used to derive inter-agent distances and can be used for control. This scenario is explored in this thesis. The noise on the individual distance measurements of the RPLidar-A2M8 is—in most scenarios—small enough to produce acceptable results. Noise on the distances of the objects detected by clustering LIDAR measurements is relatively large, because the number of LIDAR measurements in the cluster varies each time step and the centre of the cluster

moves accordingly. This noise causes oscillations around the desired inter-agent distances. These oscillations were smallest when using the spring-damper system.

From analysing the performance of the formation control algorithms at different frequencies, it is shown that higher control frequencies generally perform better than low control frequencies. For the ROSbot, which has a 15 Hz kinematic controller and RPLidar-A2M8 Laser Range Finder (LRF), of the presented algorithms, the consensus algorithm presented in Eq. (3-12) and Eq. (3-13) is the most suitable, using the parameters in Table 5-2.

Lastly, it can be assessed whether the research objective is obtained. The research objective of this research, as drawn up at the start of this thesis, was to:

Identify practical limitations on formation control by implementing various formation control algorithms on a team of skid-steer drive mobile robots in a simulated laboratory environment.

The research objective is accomplished by implementing a ROS formation control package which can be used on a group of unicycle model mobile robots. By testing this formation control ROS package, research questions could be answered, and practical recommendations could be made. These recommendations will be elaborated on in Section 6-3.

6-2 Discussion

This section discusses and reflects on the work as a whole. Limitations that have arisen throughout the research are elaborated below.

6-2-1 Integral Squared Error (ISE)

The integral squared error is used as a measure of the formation keeping error over the total duration of the simulation. In Figure 5-26, quick oscillations are seen in the individual distance errors. Because these oscillations are at a high frequency, the areas under the curve remain small. This results in a relatively low value for the ISE even though this is not the desired behaviour for the group of mobile robots. Looking at the paths of the robots, the goal is reached and the formation is kept, thus, the low ISE did accurately represent this.

6-2-2 Obstacle detection

Since the number of points in a cluster is not constant for every measurement of the LIDAR, the centre of the cluster shows fluctuations, even for static objects. This causes the distance measurements to be a noisy signal. As the robots react to the distance measurements, oscillations form. It is shown that the consensus algorithm and the exponential potential function are most susceptible to these oscillations, as can be seen in Figure 5-13 and Figure 5-17.

6-2-3 Tuning of the controllers

Because each of the presented controllers have been tuned on this specific situation and the specific (simulated) hardware, the results cannot be generalised to any skid-steer drive or differential drive mobile robot. Also, the results are highly dependent on the tuning, and it cannot be proven to be an "ideal" tuning for the situation. In digital control, control gains are also dependent on control frequency. Therefore, re-tuning the 15 Hz scenario with new control gains will likely increase performance.

6-2-4 Identification of robots and obstacles

Besides the difficulties with tuning and the sensitivity to controller frequency and measurement noise, one assumption made in this thesis is that the sensed clusters of LIDAR data points can be identified as specific robots or as obstacles. The implementation in the ROS package in this thesis does this by cross-referencing the detected objects with the global locations of the other robots. This poses a practical limitation when using this system on a group of robots outside a laboratory environment. Increased sensing capability is necessary on the robots, to locally identify each other and differentiate between obstacles and robots.

6-3 Recommendations

In the previous sections, the four different research subquestions and subsequently the main research questions have been answered. From the conclusions drawn, some practical recommendations concerning designing a group of robots for formation control tasks, as well as some suggestions for future research, can be deduced. These are discussed below.

6-3-1 Adapting holonomic formation control algorithms

When adapting holonomic formation control algorithms to work on unicycle model mobile robots, seven adaptations can be made, of which three may get stuck in local equilibria, thus four viable adaptations remain. Two of these adaptations will favour moving forward, and is advisable for robots which have front-facing sensors, such as cameras. The other two might have a robot moving backwards towards its goal location, which is a shorter path than when the robots would rotate first. When no front-facing sensors are used on the robot, this is advisable, as the paths taken by the robots are shorter and closer to the paths taken by a omnidirectional (holonomic) mobile robot. Since these last four adaptations have different advantages based on the requirements on the team of robots, the choice between these is mostly a design choice. In future research, a trade-off can be made between moving forward or taking the shortest path.

6-3-2 LIDAR noise

It was shown that below a certain threshold, very little to no increase in performance is realised when LIDAR measurement noise is reduced. It is hypothesised that this is because

the number of LIDAR measurements clustered in a detected object is not constant, and therefore the cluster centre and its associated distance is changing each time step. To make this effect smaller, it would be advantageous to have as much LIDAR points in a cluster as possible. This can be achieved by changing the LRF to a new sensor with higher angular resolution. Another way of increasing the number of LIDAR points in the cluster would be to use a 3D LIDAR. Using 3D LIDAR allows the robot to sense the bodies of the other robots as well. This would allow for shape detection and more complex tracking. When the LRF is used in the current setup, but no global position measurements are available, another sensor should be used to identify the robots and obstacles. In case of the ROSbot, the camera could be used for this, perhaps utilising AprilTags [18, 20]. Tracking of the detected obstacles might have to be improved, to reliably track the objects when they are out of the cameras range of view.

6-3-3 Control frequency

It was shown that the formation control performance is influenced by control frequency. To get closer to the continuous time simulations done in previous research [1, 11, 13], it would be beneficial to use robot hardware with a high control frequency. The consensus algorithm has shown low sensitivity to the change in control frequency. This would be a great advantage when implementing the same algorithm on multiple robotic platforms, or when the pack of robots is non-homogeneous. The exponential potential function and spring-damper system both need to be tuned for the specific platform.

6-3-4 Choosing a formation control algorithm for the ROSbot

In simulations it was shown that, using the 15 Hz skid-steer drive controller and the LIDAR distance measurements, implementing the consensus algorithms presented in Eq. (3-12), Eq. (3-13), the formation reached its goal without collisions. However, oscillations around the desired distances were present. Alignment, as in the exponential potential function, or a damping term, as in the spring-damper algorithm could be advantageous to be implemented in the consensus algorithm, to reduce these oscillations and increase the performance of the formation.

6-3-5 Improving performance

The following improvements could be investigated as a way to improve formation control performance:

- Increase sensing capabilities
 - Increase 2D LIDAR angular resolution to improve object detection and tracking performance, reducing noise in the distance of the detected obstacles;
 - Cross-reference LIDAR with other sensor data to reliably identify robots and obstacles, without the need for global position data;

- Use another source of global position data, such as Global Positioning System (GPS) locations, for formation control without the need for external sensors;
 - Use 3D LIDAR with shape detection algorithms to identify robots and obstacles from the 3D point cloud.
- Improve control hardware
 - Increase control frequency of the robot skid-steer drive controller.

6-3-6 Future research

The goal of this thesis was to implement and test formation control algorithms in a laboratory environment. Due to the global pandemic, laboratory activities could not be continued, and the choice was made to do the experiments in the Gazebo simulation package. Therefore, verifying the conclusions of this thesis in the physical world, by running the developed ROS package on a team of mobile robots would add to the scientific value of this research, as well as to the practical knowledge on formation control. Doing the formation control using only locally sensed data, by cross-referencing the LIDAR data with other sensor data, such as the robot camera or GPS data, would allow the presented algorithms to work outside a laboratory environment as well. Applying formation control algorithms in real life situations shows great promise, and could be used to increase efficiency and reduce cost for search and rescue applications, environmental sensing, autonomous exploration and other applications.

Simulation Environment

A-1 Robot Operating System (ROS)

According to the ROS website [21], *"The Robot Operating System (ROS) is a flexible framework for writing robot software. It is a collection of tools, libraries, and conventions that aim to simplify the task of creating complex and robust robot behaviour across a wide variety of robotic platforms."*

The ROS framework consists of some building blocks like the *master*, *nodes*, *topics* and *services* which together make for easy development of robotics applications. These building blocks are briefly explained in this section. A schematic of the ROS communication is shown in Figure A-1.

ROS master The ROS master handles the registration of all nodes, topics and services. The master keeps track of subscribers and publishers to topics to enable the ROS nodes to find

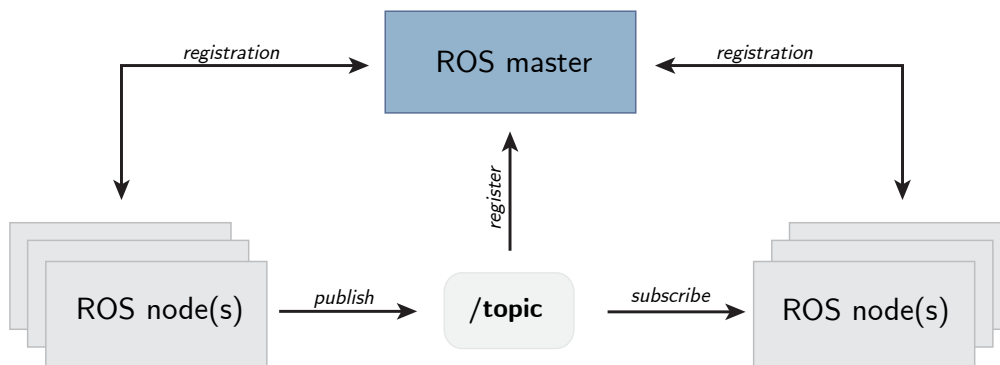


Figure A-1: Schematic of ROS communication. Adapted from Achmad et al. [3].

each other. The communication between the nodes is then carried out peer-to-peer i.e. the communication does not go through the master. In this sense, the ROS master is similar to a Domain Name Server. Another function of the ROS master is providing the parameter server, which is a dictionary where nodes can store and retrieve parameters at runtime. It is best suitable for static configuration parameters.

Nodes In ROS, tasks like handling devices or computations are carried out by nodes. Each node is a program which performs a separate task. Nodes communicate via topics and services. Multiple nodes can run on one system.

Topics A topic is a stream of data to which nodes can connect. A node might publish data, while other nodes can subscribe to this stream to obtain the data. The topic is strongly typed by the message type used by the publisher. The subscriber can only receive messages with matching type, and both parties should have definitions of the various message types.

Services Services are similar to the client-server model. One node (the server) will register the service. The ROS master keeps track of the registered services. Other nodes can send a request to this service, and get a response. In contrast to topics, two-way communication between the nodes is possible via services, since the request can contain data.

Example A simple robot with a sensor, control algorithm and actuator is used as an example: A ROS topic might be sensor data published by the node handling the sensor. A node running a control algorithm can subscribe to this topic and publish on a topic setting motor speed reference. A node handling the motor will then use this reference and drive the motor accordingly. Topics are registered by a unique name and have a defined message type. Multiple nodes can subscribe and publish to a topic, but one node can not subscribe and publish to the same topic.

A-2 Gazebo

Gazebo is an open source robotics simulator packaged with ROS . Gazebo simulates the robot dynamics using the physics engines Open Dynamics Engine (OpenDE), Bullet, Simbody and Dynamic Animation and Robotic Toolkit (DART). Sensors such as Light Detection And Ranging (LIDAR) , cameras and Inertial measurement unit (IMU) are simulated. Sensor data can be generated including noise to achieve a realistic simulation. The ROSbot gazebo model supplied by Husarion is made to work with OpenDE, so this is the physics engine used in this thesis.

A-3 Husarion ROSbot

The Husarion ROSbot is an autonomous, open source robot platform based on the Husarion CORE2-ROS controller [4, 22]. The goal is to aid rapid development of autonomous robots.

The robot integrates a four-wheel mobile platform with RGBD camera, inertial sensor, laser-scanner and distance sensors. The ROSbot has a CORE2-ROS controller with an Asus Tinker Board running Ubuntu 16.04 and ROS-Kinetic. To be more precise, the ROSbot contains:

- CORE2-ROS with Asus Tinkerboard
- RPLIDAR A2M8 360°Laser Range Finder (LRF)
- Orbbec Astra RGBD camera
- MPU9250 9 axis (Gyro, Accellero & Compass) IMU
- 4× VL53L0X Time-of-Flight (Laser) Distance Sensor

The robot is shown in Figure A-2.



Figure A-2: The Husarion ROSbot [4]

A-3-1 ROSbot kinematics

The ROSbot is a four-wheeled robot with separate drive for each wheel, also called a skid steer drive robot. This is simplified to a two-wheeled differential drive robot with two virtual wheels on the line through the robot geometric centre, this is shown in Figure A-3. This robot can be represented by a unicycle model, having three degrees of freedom, described in a global coordinate frame as:

$$\begin{bmatrix} x \\ y \\ \alpha \end{bmatrix} \quad (\text{A-1})$$

The forward velocity v_x , in direction of the local coordinate frames x-axis, x_r , and the rotational velocity ω_z around R_c are the two control inputs. In the global coordinate frame, this kinematic model is described by the following set of differential equations.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\alpha} \end{bmatrix} = \begin{bmatrix} v_x \cos \alpha \\ v_x \sin \alpha \\ \omega_z \end{bmatrix} \quad (\text{A-2})$$

In Husarion [5], a forward kinematic model is derived so encoder measurements can be used to derive position. Borst et al. [23] found this to be inaccurate and proposed an extended odometry incorporating IMU measurements.

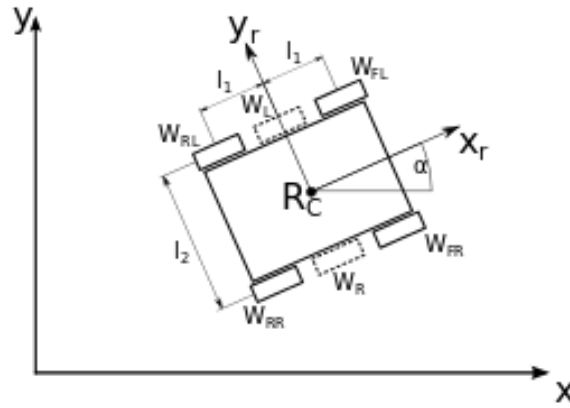


Figure A-3: The four wheels of the ROSBot are simplified to two virtual wheels [5]

Besides the actual robot, also a Gazebo model for the ROSbot is supplied by Husarion [24]. This model allows the ROSbot, its sensors and the designed ROS nodes to be simulated.

A-4 LIDAR

The ROSbot uses the RPLidar A2M8 LRF to scan its surroundings. This sensor consists of a Laser on a rotating platform, which senses distance 360° around the robot. The RPLidar A2M8 spins at 600rpm, or 10 rotations per second, while sensing distances at 4000Hz[25]. This means the final resolution of the LRF is 400 samples per rotation, or 0.9°. The 360°scan will be published with a frequency of 10Hz.

The distance resolution specified for the RPLidar A2M8 is given as $< 1\%$ of the distance, and $< 0.05\text{mm}$ if the distance is $< 1,5\text{m}$ [25]. In the scenarios proposed in this thesis, ranges are typically $< 3\text{m}$. In gazebo, Gaussian noise with zero mean is added to the distance of the individual LIDAR measurements. Zero-mean Gaussian noise with standard deviation of 0.01 will put 99.7% of the distances within $\pm 30\text{mm}$, or 1% of 3m. At smaller distances, this noise becomes more prevalent: for measurements at 0.5m, still 99.7% falls within $\pm 30\text{mm}$, but only 38.3% falls within $\pm 5\text{mm}$, or 1%. This is a shortcoming in gazebo, as the noise is simulated as zero-mean white noise with a fixed standard deviation, regardless of the size of the actual signal. If we assume most of the measurements to be $< 1.5\text{m}$, the specification

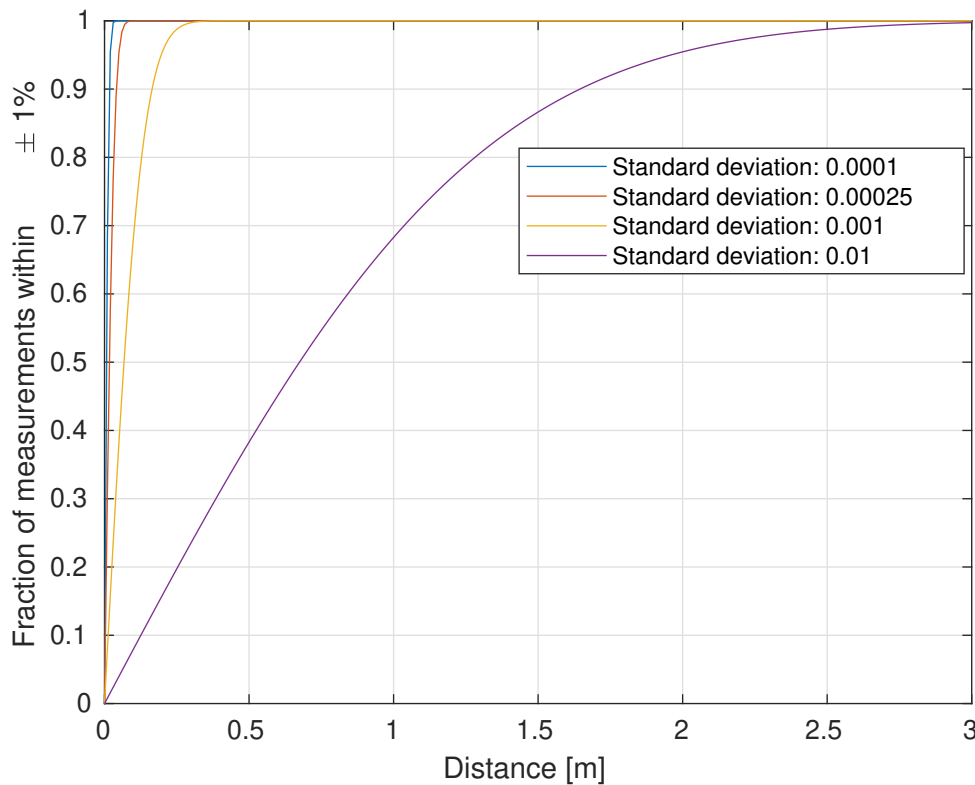


Figure A-4: Fraction of measurements within $\pm 1\%$ at different distances for some standard deviation

of $\pm 0.5\text{mm}$ can be used. If the noise is simulated with a standard deviation of 0.00025, this specification is met in 95.5% of the measurements. Thus, we can assume realistic values for the noise standard deviation as simulated by gazebo are in the interval $[0.0025, 0.01]$, for the scenarios proposed in this thesis.

A-5 Object Detector

For clustering of LIDAR measurements into circular objects, and for tracking these objects, an open source obstacle detector ROS package is used (see Przybyła [26]). In Figure A-5, the LIDAR measurements and resulting detected objects are shown.

A-6 Order of the system

The robot would best be approximated by a 2nd order system, however, since the robot has fast dynamics and takes a velocity setpoint as input, the assumption of a first order system is made.

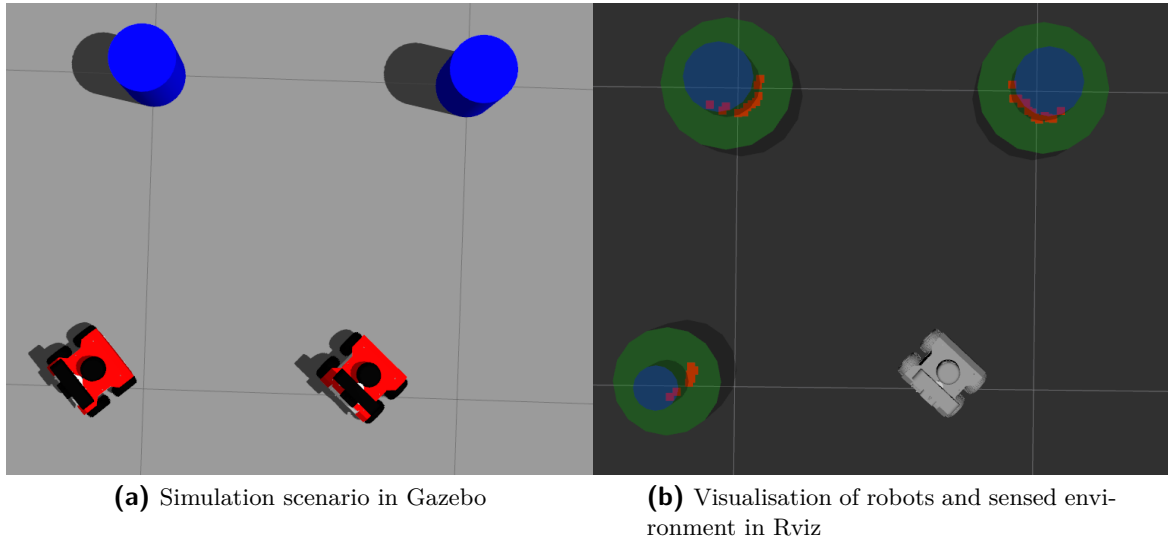


Figure A-5: Obstacle detection by the right robot. LIDAR scan measurements are shown as red points. From this points, a blue circular object is estimated. A safety margin is added to get the green circular object.

Integrating the input to give an input for acceleration slows the dynamics of the robot a lot, resulting in unwanted behaviour. Therefore, it is chosen to approximate the system by a first order system with a velocity input. In some algorithms, such as the spring-damper system, velocity measurements are used to calculate the desired input, which is a velocity setpoint. This results in a system shown in the following block diagram.

Bibliography

- [1] J. Wiech, V. A. Eremeyev, and I. Giorgio, “Virtual spring damper method for nonholonomic robotic swarm self-organization and leader following,” *Continuum Mechanics and Thermodynamics*, vol. 30, no. 5, pp. 1091–1102, 9 2018.
- [2] A. Renzaglia and L. Briñón-Arranz, “Search and Localization of a Weak Source with a Multi-robot Formation,” *Journal of Intelligent and Robotic Systems: Theory and Applications*, 2019.
- [3] M. Achmad, G. Priyandoko, R. Roali, and M. Daud, “Tele-Operated Mobile Robot for 3D Visual Inspection Utilizing Distributed Operating System Platform,” *International Journal of Vehicle Structures and Systems*, vol. 9, no. 3, 2017.
- [4] R. Jarema, L. Mitka, D. Nowak, S. Szantula, M. Zieliński, and H. Zwiercan, “Rosbot manual,” <https://husarion.com/manuals/rosbot-manual/>, 2017, accessed 2019-06-24.
- [5] Husarion, “Simple kinematics for mobile robot,” [Online; accessed 24-June-2019]. [Online]. Available: <https://husarion.com/tutorials/ros-tutorials/3-simple-kinematics-for-mobile-robot/>
- [6] S. Janson, M. Middendorf, and M. Beekman, “Honey bee swarms: How do scouts guide a swarm of uninformed bees?” *Animal Behaviour*, vol. 70, pp. 349—358, 2005.
- [7] S. Hubbard, P. Babak, S. T. Sigurdsson, and K. G. Magnússon, “A model of the formation of fish schools and migrations of fish,” *Ecological Modelling*, 2004.
- [8] W. Li, J. A. Farrell, S. Pang, and R. M. Arrieta, “Moth-inspired chemical plume tracing on an autonomous underwater vehicle,” *IEEE Transactions on Robotics*, vol. 22, no. 2, pp. 292–307, 2006.
- [9] S. Feng, Y. Zhang, S. E. Li, Z. Cao, H. X. Liu, and L. Li, “String stability for vehicular platoon control: Definitions and analysis methods,” *Annual Reviews in Control*, vol. 47, pp. 81–97, 2019.

- [10] B. D. Anderson, C. Yu, B. Fidan, and J. M. Hendrickx, “Rigid Graph Control Architectures for Autonomous Formations,” *IEEE Control Systems*, vol. 28, no. 6, pp. 48–63, 2008.
- [11] Z. Sun, “Cooperative Coordination and Formation Control for Multi-agent Systems,” The Australian National University, Canberra, Australia, Tech. Rep., 2018. [Online]. Available: <http://www.springer.com/series/8790>
- [12] K. K. Oh, M. C. Park, and H. S. Ahn, “A survey of multi-agent formation control,” *Automatica*, vol. 53, pp. 424–440, 3 2015.
- [13] K. D. Listmann, M. V. Masalawala, and J. Adamy, “Consensus for formation control of nonholonomic mobile robots,” *Proceedings - IEEE International Conference on Robotics and Automation*, no. May, pp. 3886–3891, 2009.
- [14] W. Ren and R. W. Beard, “Consensus seeking in multiagent systems under dynamically changing interaction topologies,” *IEEE Transactions on Automatic Control*, vol. 50, no. 5, pp. 655–661, 2005.
- [15] —, *Consensus algorithms for single-integrator dynamics*, 2008, no. 9781848000148.
- [16] J. Alonso-Mora, E. Montijano, T. Negeli, O. Hilliges, M. Schwager, and D. Rus, “Distributed multi-robot formation control in dynamic environments,” *Autonomous Robots*, 2018.
- [17] L. Krick, M. E. Broucke, and B. A. Francis, “Stabilization of infinitesimally rigid formations of multi-robot networks,” in *Proceedings of the IEEE Conference on Decision and Control*, 2008, pp. 477–482.
- [18] J. Wang and E. Olson, “AprilTag 2: Efficient and robust fiducial detection,” in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, oct 2016, pp. 4193–4198.
- [19] T. Field, J. Leibs, and J. Bowman, “Rosbag wiki,” <http://wiki.ros.org/rosvbag>, accessed 2020-06-24.
- [20] D. Malyuta, “Guidance, Navigation, Control and Mission Logic for Quadrotor Full-cycle Autonomy,” Master thesis, Jet Propulsion Laboratory, 4800 Oak Grove Drive, Pasadena, CA 91109, USA, Dec. 2017.
- [21] OSRF, “About ros.” [Online]. Available: <http://www.ros.org/about-ros/>
- [22] Husarion, “ROSbot quick start.” [Online]. Available: <https://husarion.com/tutorials/howtostart/rosbot---quick-start/>
- [23] T. Borst, D. D. Mol, and M. Schaap, “Software setup for the Husarion ROSbot,” pp. 1–7, 2018.
- [24] R. Jarema, L. Mitka, D. Nowak, S. Szantula, M. Zieliński, and H. Zwiercan, “Rosbot gazebo model,” https://github.com/husarion/rosbot_description, 2020.

- [25] Shanghai Slamtec, “RPLIDAR A2 datasheet,” 2017. [Online]. Available: https://www.robotshop.com/media/files/pdf2/ld208_slamtec_rplidar_datasheet_a2m8_v1.0_en.pdf
- [26] M. Przybyla, “Detection and tracking of 2D geometric obstacles from LRF data,” *11th International Workshop on Robot Motion and Control, RoMoCo 2017 - Workshop Proceedings*, pp. 135–141, 2017.

Glossary

List of Acronyms

DART	Dynamic Animation and Robotic Toolkit
GPS	Global Positioning System
IMU	Inertial measurement unit
ISE	Integral Squared Error
LIDAR	LIght Detection And Ranging
LRF	Laser Range Finder
OpenDE	Open Dynamics Engine
ROS	Robot Operating System

List of Symbols

ω_z	Rotational velocity setpoint
p_j^i	Position of agent j in the coordinate frame of agent i , given by the local coordinates $[x_j^i, y_j^i]$
$\mathcal{E}$	Set of edges of graph $\mathcal{G}$
$\mathcal{G}$	Graph representing the communication topology of the group of robots
$\mathcal{N}_i$	Set of neighbours of agent i
d_{ij}^*	Desired distance between agent i and agent j
F_i	Potential working on robot i
F_{x_i}	Potential acting on robot i in the direction of the x -axis of the local coordinate frame, or the heading of robot i
$F_{x_{ji}}$	Potential acting on robot i along the x -axis of the local coordinate frame of robot i , caused by the position of robot j
F_{y_i}	Potential acting on robot i , acting in the direction of the y -axis of the local coordinate frame, which is perpendicular to the robot heading

$F_{y_{ji}}$	Potential acting on robot i along the y -axis of the local coordinate frame of robot i , caused by the position of robot j
p_i	Position of agent i
v_x	Forward velocity setpoint