rijksuniversiteit groningen

Displacement- and Distance-based Formation Control in the port-Hamiltonian Framework

Final Report for Bachelor Integration Project

24-06-2022

Student:
J.M. Oeben, s4101839
j.m.oeben@student.rug.nl

1st Supervisors:
Msc. N. Li & prof. dr. ir. J.M.A Scherpen
ningbo.li@rug.nl & *j.m.a.scherpen@rug.nl*

2nd Supervisor:
Dr. ir. M. Taheri
m.taheri@rug.nl

Abstract

This research explores two different kinds of formation control in a port-Hamiltonian framework. Moreover, the agents in this research are modelled by double-integrator dynamics. In section 1, an introduction to formation control, and the port-Hamiltonian framework will be given based on previous research. Second, some preliminary knowledge on graph theory, port-Hamiltonian systems and formation control will be presented in section 2, after which the research problem is formulated in section 3. In section 4, displacement-based formation control is researched, proven to be stable and simulated in Matlab, after which distance-based formation control is researched in a similar fashion in section 5. Naturally, the conclusion and future research suggestions follow in section 6.

Contents

1 Introduction

Animals have been expressing their behaviour in forms of patterns for as long as mankind exists. Take as an example bird flying formations, where the birds fly with a certain geometric shape (formation) to optimize their use of energy as a collective, or a herd of animals flocking together to minimize the risk of being attacked by predators [1]. In the past few decades, this behaviour has been studied in the form of formation control. Formation control in this research is described as group behaviour being exhibited to achieve a desired geometric shape [2].

1.1 Applications

Formation control is a concept widely used in both civil and military applications [3]. Often, the purpose of formation control is either surveillance or exploration of a given region. For example, much research has been and is being conducted in the field of formation flying, such as unmanned airborne vehicles (UAV's). Some research in this field was done by [1], [3] and [4], which demonstrate the various opportunities for practical applications and further research on this subject.

Another purpose of formation control is inspection and maintenance. For example, in the ROSE project [2], robots and sensors were used in a specific formation to inspect the quality of dykes. By using robots in a triangular formation that travel along the length of a dyke (one robot on top of the dyke, and one on either side of the dyke), an image could be made of the internal dyke structure to spot failures, and repair it accordingly.

In paper [5], some other applicability's of formation control are mentioned, such as environmental disaster monitoring, rescue cases (exploration), environmental monitoring, chemical spill searching, etc. In conclusion, it is safe to say that this field has a wide practical applicability.

1.2 Current Research

In the past few decades, a lot of research has been conducted in the field of formation control. A book [6] published in the end of 2019 contains a detailed overview of the current methods used for formation control, and serves as a basis for discovering relevant formation control methods. One of the methods that can be used is using gradient control laws [7][8]. A paper published in 2020 still used this method to control the formation of mobile robots [9].

Another approach of formation control is aligning the orientation of agents in a system [6], which is used in papers such as [10] and [11]. The last method widely used is when bearing measurements are used for formation control. This is a combination of the aforementioned two methods combined with graph theory [6] An example of formation control using this approach is [12] which is a theoretical research on formation control.

1.3 Port-Hamiltonian Systems

For this research, the input-state-output port-Hamiltonian form will be used to model formation control. This is part of a mathematical framework that describes a system based on the energy it contains, and offers a great starting point for control design for the above mentioned formation examples [2]. Additionally, by using port-Hamiltonian framework, multi-domain interconnections of a physical system are revealed which allow for better understanding of the system. In addition, port-Hamiltonian systems provide insights into system properties such as stability, passivity, and scalability [13] which are very useful for designing a controller for formation control. A more in-depth explanation will be given in the preliminaries.

In order to check the stability of the designed controllers using the input-state-output port-Hamiltonian form, the Lyapunov's stability theorem and the LaSalle's invariance principle are u. sed in many researches on the topic of formation control. A few examples are given in [1][2][6][14]. Subsequently, they will also be used for this research.

1.4 Contribution

Despite such favourable characteristics for modelling control, the port-Hamiltonian framework has only been used a handful amount of times in the field of formation control. Examples of researches using this framework include [8], [14], [15], [16], [17], [18] and [19].

However, the topic of this research is not covered by these papers. For example, in [19], a gradient control law is used, and in [16], the passivity assumption of port-Hamiltonian systems is relaxed and the underlying graph topology of the system described in this research are considered to be acyclic. In [17], only the Hamiltonian equation is used instead of the energy preserving port-Hamiltonian system. Last, in [18], Linear Matrix Inequalities had to be solved, which is a very complicated process.

This research aims to use the port-Hamiltonian framework for describing physical systems to design a controller for formation control in a way that has not been done before. Passivity is assumed in this research, and the agents will be modelled by second-order dynamics. Additionally, the stability of distance-based formation control on cyclic graphs is explored, which has not been done so far.

2 Preliminaries

2.1 Graph Theory

Graph theory is essential when it comes to formation control. Hence, a brief introduction of this field will be given in this section.

A graph is composed of a set of vertices $\mathcal{V}$ and edges $\mathcal{E}$, and is a mathematical structure defined as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ to model pairwise interactions between objects [2]. In this research, the objects will be agents in a system defined in the node set $\mathcal{V}$, and an edge set $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ that corresponds with the way two agents interact with each other.

Node set $\mathcal{V} = \{n_1, n_2, \dots, n_N\}$ has $N = |\mathcal{V}|$ elements, which in this research, will only be $N = 2$ or $N = 3$, and the edge set $\mathcal{E} = \{e_1, e_2, \dots, e_E\}$ has $E = |\mathcal{E}|$ elements.

The graphs that will be used in this research can be seen below in figure 2.1. It shows a system where $N = 2$ and $E = 1$, and where the agents denoted by $i \in \{1,2\}$, and the edge by $j \in \{1\}$. On the right, $N = 3$, $E = 3$ and $i, j \in \{1,2,3\}$

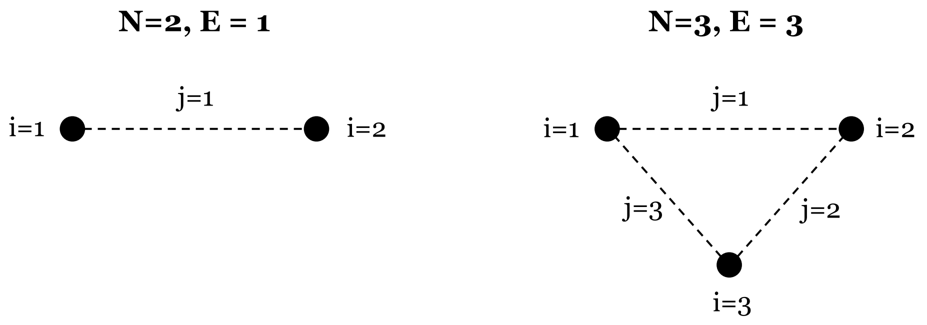

Figure 2.1: Graphs of two- and three agent systems considered in this research

Graphs in this research are considered to be undirected, which means that an edge of a pair of nodes is unordered ($(n_1, n_2) \in \mathcal{E}$ if and only if $(n_2, n_1) \in \mathcal{E}$). For these types of graphs, an arbitrary orientation can be assigned to each edge, denoted by the positive (head) sign and negative (tail) sign in figure 2.2.

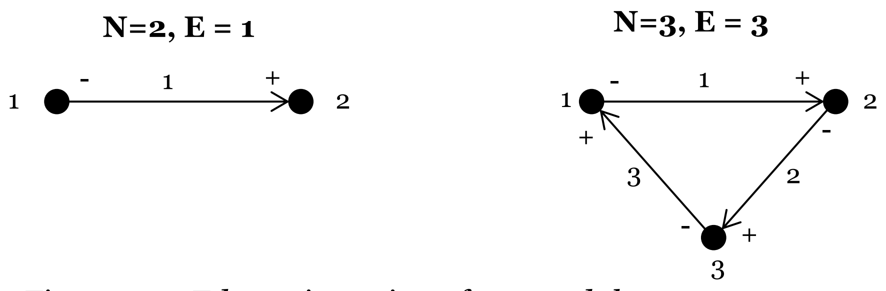

Figure 2.2: Edge orientation of two and three agent systems

Graphs can simply be described using various matrices, namely a degree matrix $\Delta \in \mathbb{R}^{N\times N}$, an adjacency matrix $\mathrm{A} \in \mathbb{R}^{N\times N}$, and incidence matrix $\mathrm{B} \in \mathbb{R}^{N\times E}$. The degree matrix is defined as $\Delta = diag\big(d(n_1), \ldots, d(n_N)\big)$, where $d(n_i)$ is the degree of node n_i, or the number nodes adjacent to it in $\mathcal{G}$. Adjacency matrix A represents adjacency relationships within $\mathcal{G}$, and is defined by

$$a_{ij} = \begin{cases} 1 \text{ if } (n_1, n_2) \in \mathcal{E} \\ 0 \text{ if } (n_1, n_2) \notin \mathcal{E}. \end{cases} \tag{2.1}$$

The most important matrix for this research is incidence matrix B, which is defined as

$$b_{ij} = \begin{cases} +1 & \text{if node } n_i \text{is the head of edge } e_j, \\ -1 & \text{if node } n_i \text{is the tail of edge } e_j, \\ 0 & \text{otherwise.} \end{cases} \tag{2.2}$$

Further, some minor background in graph rigidity is needed. For this, the reader is referred to [6], as Sun, Z. has explained these concepts thoroughly. With respect to graph rigidity, it must be noted that the graph for distance-based control is minimally and infinitesimally rigid. This means that the graphs used in this research contain the minimal amount of edges that it requires in order to be considered rigid.

2.2 Formation Control

In general, formation control can be described by a system of agents, of which the desired (controlled) output behaviour is a spatial configuration (the formation) [6]. This could mean both that the agents must form, and maintain the formation [20].

A visual representation of a simple case of formation control can be seen in figure 2.3. Here, a global coordinate system (large coordinate system in figure 2.3) can be seen containing three agents, where each agent has its own coordinate system, called a local coordinate system (smaller coordinate systems in figure 2.3). The position $q_i \in \mathbb{R}^{2\times 1}$ of each agent i (i = 1,2,3) is represented using a solid black circle, and its desired position ${q_i}^*$ is denoted by a dotted circle. The goal of formation control is to design a controller such that the current position q_i of each agent i converges to the desired position q^* (figure 2.3). In mathematical terms, the goal is then $q_i \rightarrow {q_i}^*$ for $i = 1,2,..,n$ number of agents.

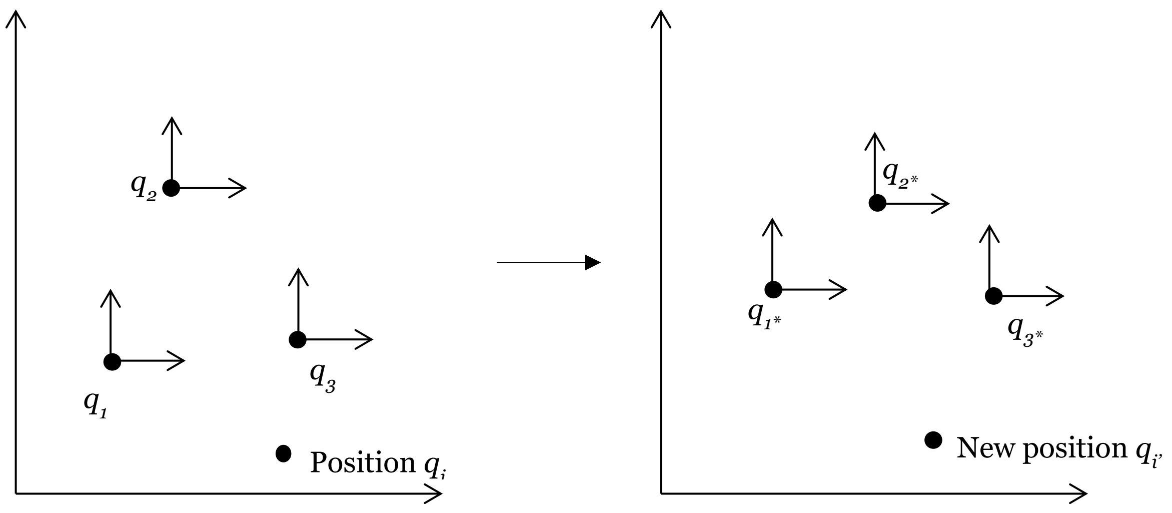

Figure 2.3: Formation with (right) and without (left) control

According to papers from various researchers such as [21] and [22], at least three approaches of formation control can be distinguished in the existing literature. Two of those are relevant for this research, and are explained in more detail below.

2.2.1 Displacement-based Formation Control

Displacement-based formation control is widely researched and is based on only a global coordinate system, where the agents are able to sense the position of their neighbouring agents with respect to their own local coordinate system[21]. Based on this information, the agents can control their own displacement. Additionally, with this form of formation control, the local coordinate systems of all agents are aligned with respect to a global coordinate system, which can be seen in figure 2.4.

Displacement-based formation control is based on displacement information of the edges $\mathcal{E}$ between agents in the system, defined as $z_j = q_1 - q_2 \in \mathbb{R}^{2\times 1}$ [16][21], where j denotes the edge between two agents in a system. Therefore, the formal control objective for displacement-based formation control is stated as follows:

$$z_j \rightarrow {z_j}^*. \tag{2.3}$$

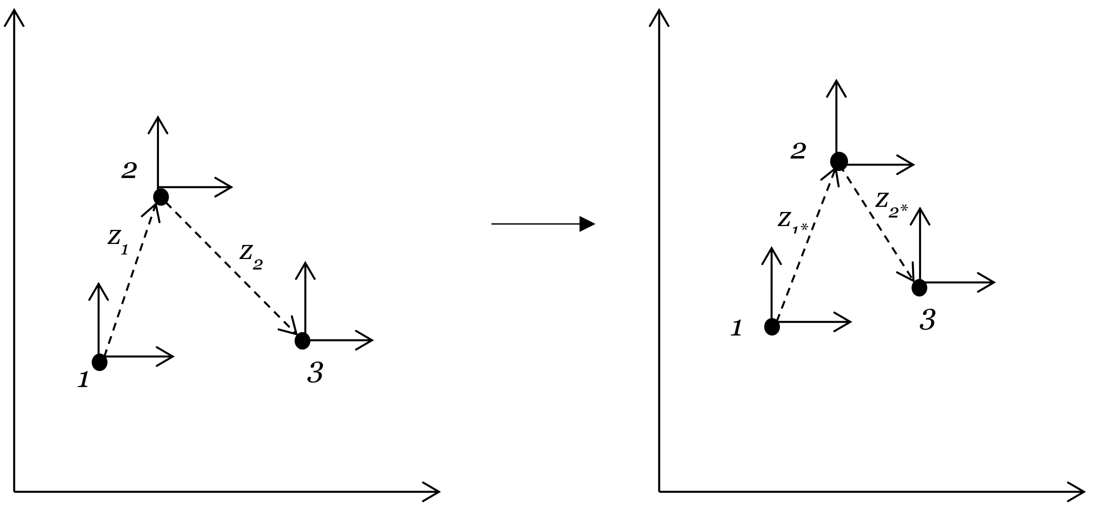

Figure 2.4: Displacement-based formation control

For this research, the displacement-based formation control is based on an acyclic graph with two edges. This can be attributed to the fact that controlling two edges is sufficient to achieve the desired displacement formation.

2.2.2 *Distance-based Formation Control*

Distance-based formation control is the main topic of this research. Here, the agents in a system only have access to the inter-agent distances between themselves and other agents, based on their own local coordinate system (see figure 2.5) [21]. In contrast to displacement-based formation control, these local coordinate systems are not aligned. Logically, this form of formation control is useful in cases where there is no information on global coordinates of agents [22], for example space and planet exploration missions.

For determining the distance $d_j \in \mathbb{R}^{1\times1}$ between a pair of agents in a system, the modulus of the difference between their relative positions in a two-dimensional space is used. Alternatively, the distance d_j between two agents is given by the modulus of edge z_j such that $d_j = \|z_j\|$ (see displacement-based formation control). The formal formation control problem is then given by

$$d_j \rightarrow {d_j}^*. \tag{2.4}$$

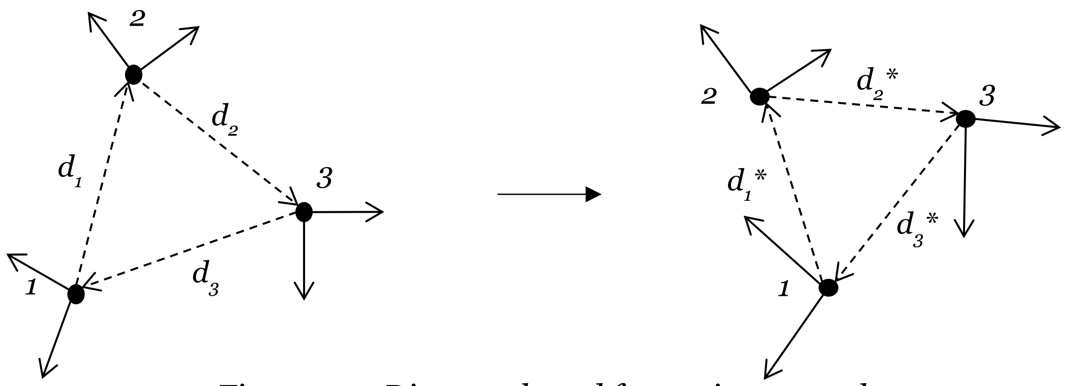

Figure 2.5: Distance-based formation control

It can be noted that in contrast to displacement-based formation control, distance-based formation control is nonlinear due to taking the norm of edge z_j . Additionally, the graphs used for distance-based formation control are cyclic, meaning all agents are connected through edges as shown in figure 2.5. This can be attributed to the fact that using three edges is the only way to achieve the desired formation when using distance-based control formation.

Furthermore, the ultimate interest of this research is distance-based formation control. However, to get familiar with the topic of formation control altogether, it has been advised to first start with displacement-based formation control and work onwards from there. Displacement based formation control is simpler due to its linearity, and more widely researched than distance-based, which makes it easier to get a grasp of the theory and concepts.

Table 2.1: Summary of crucial information on different kinds of formation control [21]

	Displacement-based	Distance-based
Sensed variables	Relative positions of neighbours	Relative position of neighbours
Controlled variables	Relative positions of neighbours	Inter-agent distances
Coordinate system	Orientation aligned local coordinate system	Local coordinate system

2.3 Input-State-Output port-Hamiltonian Systems

The port-Hamiltonian theory consists of two important parts, namely the Hamiltonian equation and power-port theory. First, the Hamiltonian will be discussed.

2.3.1 *The Hamiltonian*

The Hamiltonian equation describes the total energy stored in a system [2], which is the sum of all kinetic and potential energy in a system:

$$H(x) = E_{kinetic} + E_{potential}. \tag{2.5}$$

In this research, the systems that will be analyzed are two-dimensional systems. That means that the potential energy $E_{potential}$ is zero, which simplifies the system. Then, the Hamiltonian is given by

$$H(x) = E_{kinetic}, \tag{2.6}$$

where

$$E_{kinetic} = \frac{1}{2} p^T M^{-1} p. \tag{2.7}$$

Here, $\dot{q}$ is a vector that denotes velocity, and matrix $M \in \mathbb{R}^{n \times n}$ is the product of mass vector m and the identity matrix..

2.3.2 *Port-based Modelling and Dirac Structure*

Port-based modelling is based on the power-port interconnection of three types of elements in a system: static energy-dissipating elements, dynamical energy-storing elements, and static lossless energy-routing elements [15]. In physical systems, power ports provide an interface for the subsystems to interact with each other [2]. Moreover, each power-port consists of two conjugate variables named *flow f* and *effort e*, which are linked using a Dirac Structure. This Dirac structure is a geometric structure defined by the totality of the energy-routing elements and the interconnection topology of the system [15], and is therefore the key element underlying port-Hamiltonian systems as it ensures passivity of the system (these exact Dirac properties can be found in [2], [13] and [15]. According to [2][15], port-Hamiltonian systems are often represented using the structure below.

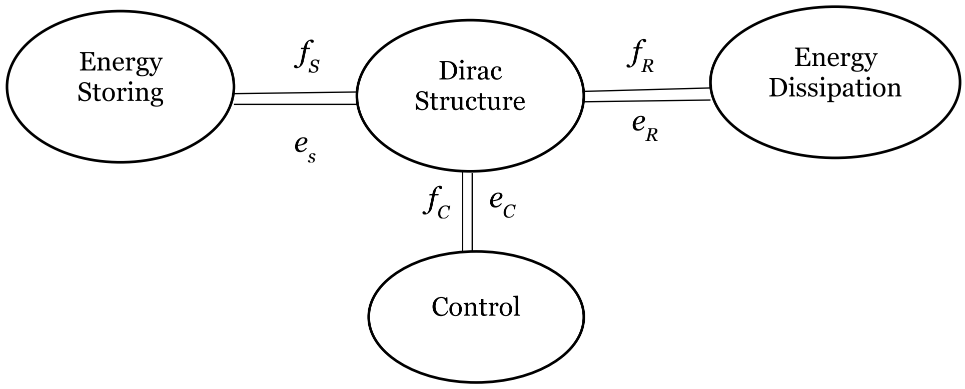

Figure 2.6: Port-Hamiltonian system with a Dirac structure.

2.3.3 Port-Hamiltonian System

As mentioned before, the interest of this research lies with the input-state-output form of port-Hamiltonian systems. The general formula for this [2][15] is given by

$$\dot{x} = \big(J(x) - R(x)\big)\frac{\partial H}{\partial x}(x) + g(x)u,$$

$$y = g^T(x)\frac{\partial H}{\partial x}(x). \tag{2.8}$$

Here, $\dot{x}$ represents the dynamics of the system using the Hamiltonian $\frac{\partial H}{\partial x}(x)$. Additionally, the matrix $J(x)$ contains information regarding the interconnection of subsystems within the system, and is always a skew-symmetric matrix [2][15] such that $J(x) = -J^T(x)$. Matrix $R(x)$ describes the dissipation of energy in the system, and is a positive semi-definite matrix such that $R(x) = R^T(x) \geq 0$ [2][15]. Last, $g(x)$ is the input matrix for u, the control input of the system. Note that port-variables (u, y) correspond to the control port in figure 2.6 [2][15].

2.4 Double-Integrator Models

The agents in the systems of this research will be modelled using double integrators, because it shows the second-order dynamic kinematics of the agents. In this case, the state of an agent i is given by $x_i = [q_i, v_i]^T$, where $q_i \in \mathbb{R}^{2\times1}$ denotes the relative position of agent i based on a global coordinate system, and $v_i \in \mathbb{R}^{2\times1}$ denotes its velocity and their dynamics are given by

$$\begin{cases} \dot{q}_i = v_i \\ \dot{v}_i = u_i. \end{cases} \tag{2.9}$$

This is an exemplary system in which the relation between the state, position and velocity of an agent is clearly visible by using a double integrator.

2.5 Lyapunov's Direct Method Stability Theorem

In this section, a brief introduction of the Lyapunov direct method stability theory will be introduced based on [23]. Take $x \in \mathbb{R}$ as the state vector of a system, and consider the following time-invariant system

$$\dot{x} = f(x), \tag{2.10}$$

where $f : \mathcal{D} \to \mathbb{R}^n$ is a locally Lipschitz map from a domain $\mathcal{D} \in \mathbb{R}^n$ into $\mathbb{R}^n$. Let x be an equilibrium point in the system and $\mathcal{D} \in \mathbb{R}^n$ be an open subset containing $x = 0$. Let $V : \mathcal{D} \to \mathbb{R}$ be a continuously differentiable function such that

$$V(0) = 0 \text{ and } V(x) > 0 \; in \; \mathcal{D} - \{0\}, \tag{2.11}$$

$$\dot{V}(x) \leq 0 \; in \; \mathcal{D}. \tag{2.12}$$

Then, $x = 0$ is stable, and if

$$\dot{V}(x) < 0 \; in \; \mathcal{D} - \{0\}, \tag{2.13}$$

then $x = 0$ is asymptotically stable.

A function $V(x)$ that is continuously differentiable and that satisfies (2.11) and (2.13) is called a Lyapunov function. It must be noted that many systems in fact fail to meet condition (2.13) due to the fact that $\dot{V}$ is only negative semi-definite $\dot{V}(x) \leq 0 \; in \; \mathcal{D} - \{0\}$. If so, LaSalle's invariance principle can be used to assess the stability of the system [2].

2.6 LaSalle's Invariance Principle

LaSalle's invariance principle is an extension of the Lyapunov stability theorem, and is stated as follows [2][23]:

Let $\Omega \subset D$ be a compact set that is positively invariant with respect to (2.10). Let $V : D \to \mathbb{R}$ be a continuously differentiable function such that $\dot{V}(x) \leq 0$ in Ω. Let E be the set of all points in Ω where $\dot{V}(x) = 0$. Let $\mathcal{M}$ be the largest invariant set in E. Then, every solution starting in Ω approaches $\mathcal{M}$ as $t \to \infty$.

3 Problem Formulation

3.1 Double Integrator in Input-State-Output port-Hamiltonian Form

Now that formation control, the double integrator model of the agents, and the input-state-output port-Hamiltonian systems have been introduced, it is time to write our double integrator modelled agents in the port-Hamiltonian form.

State $x_i = [q_i, p_i]^T$ of an agent i is modelled by its position $p_i \in \mathbb{R}^{2\times1}$ and momentum $p_i \in \mathbb{R}^{2\times1}$. Position and momentum are related by mass matrix $M \in \mathbb{R}^{2\times2}$, which is a mass vector m multiplied with an identity matrix I such that

$$p_i = M\dot{q}_i, \tag{3.1}$$

and their dynamics are given by

$$\begin{cases} \dot{q}_i = M^{-1}p_i \\ \dot{p}_i = u_i\,. \end{cases} \tag{3.2}$$

In order to write our agents in the input-state-output port-Hamiltonian form, we will first rewrite the Hamiltonian equation by substituting equations (2.7) and (3.1):

$$H_i = \frac{1}{2}{p_i}^T M^{-1} p_i. \tag{3.3}$$

Then, the system is written in the following way,

$$\begin{bmatrix} \dot{q}_i \\ \dot{p}_i \end{bmatrix} = \begin{bmatrix} 0 & I_2 \\ -I_2 & -D_i^r \end{bmatrix} \begin{bmatrix} \frac{\partial H_i}{\partial q_i}(p_i) \\ \frac{\partial H_i}{\partial p_i}(p_i) \end{bmatrix} + \begin{bmatrix} 0 \\ I_2 \end{bmatrix} u_i,$$

$$\tag{3.4}$$

$$y_i = \begin{bmatrix} 0 & I_2 \end{bmatrix} \begin{bmatrix} \frac{\partial H_i}{\partial q_i}(p_i) \\ \frac{\partial H_i}{\partial p_i}(p_i) \end{bmatrix},$$

where the interconnection matrix contains two nonzero elements, and the dissipative matrix only contains one non-zero element, namely $D^r \in \mathbb{R}^{N\times N}$, which is a damping vector multiplied with the identity matrix. This damping exists due to friction.

3.2 Control Objectives

In this section, the exact definitions of the formation control objectives are given for both displacement-, and distance based formation control.

3.2.1 Displacement-based Formation Control Objective

The goal of this research is to first achieve a desired formation in two- and three-agent systems using displacement-based formation control in input-state-output port-Hamiltonian form such that $z_j \to z_j^*$. Here, z_j denotes the edge between a pair of agents, and $i \in 1,2,3$ denotes the agent number (figure 3.1).

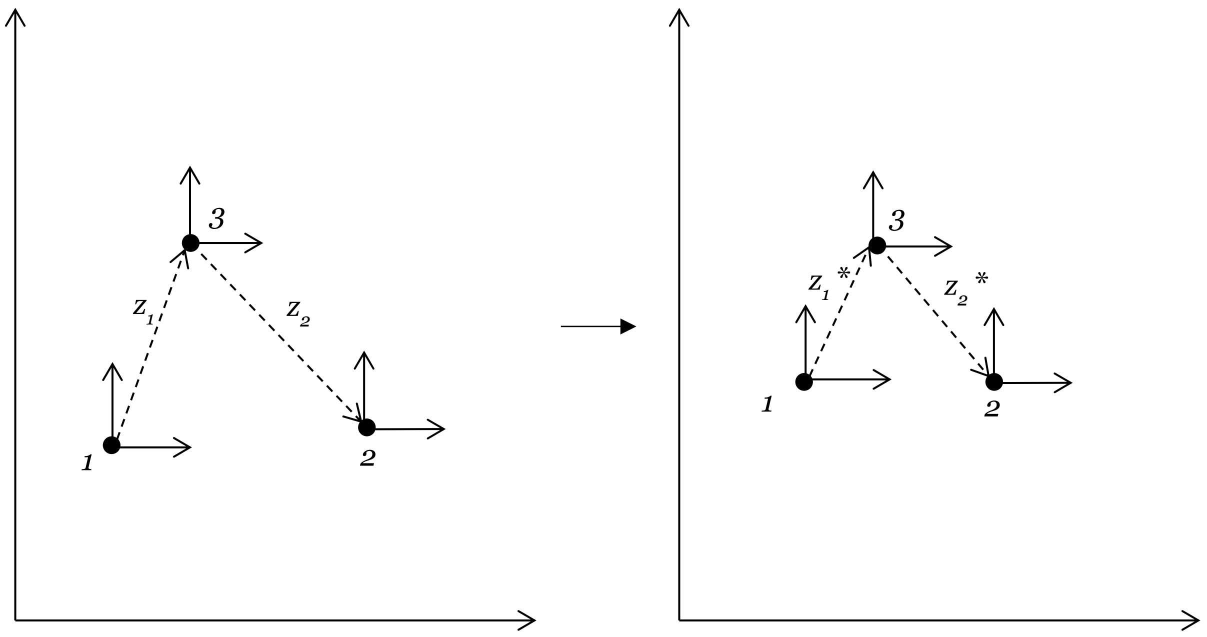

Figure 3.1: Displacement-based formation control with 3 agents

The incidence matrix B for this three-agent graph is given by

$$B = \begin{bmatrix} -1 & 0 \\ 1 & -1 \\ 0 & 1 \end{bmatrix}, \tag{3.5}$$

where the columns represent the number of edges (E), and the rows represent the number of agents (N).

3.2.2 Distance-based Formation Control Objective

Next, the goal is to achieve a desired distance d_j* between two agents $i \in 1,2,\ldots,N$ by designing a controller such that in systems with two or three agents, the distance converges such that $d_j \to d_j^*$ by using the port-Hamiltonian input-state-output form, and simulating the systems and their controller in Matlab to validate the research. Specifically, for a two-agent system, the objective is visualized in figure 3.2, and its incidence matrix B is given by

$$B = [-1 \quad 1]. \tag{3.6}$$

Note that the incidence matrix for a two-agent system is equal for both displacement and distance-based formation control.

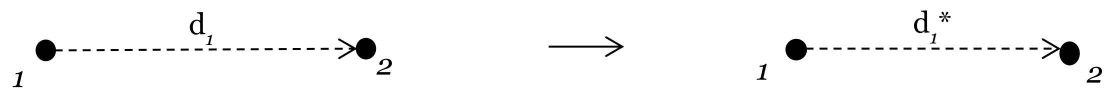

Figure 3.2: Two-agent distance-based formation control objective

For a three-agent system, the distance-based control objective is visualized in figure 3.3, and its incidence matrix is given by

$$B = \begin{bmatrix} -1 & 0 & 1 \\ 1 & -1 & 0 \\ 0 & 1 & -1 \end{bmatrix}. \tag{3.7}$$

As mentioned before, this incidence matrix for a three-agent system differs from the incidence matrix for a two-agent system due to the fact that one is an acyclic graph (displacement-based) and one is a cyclic graph (distance-based).

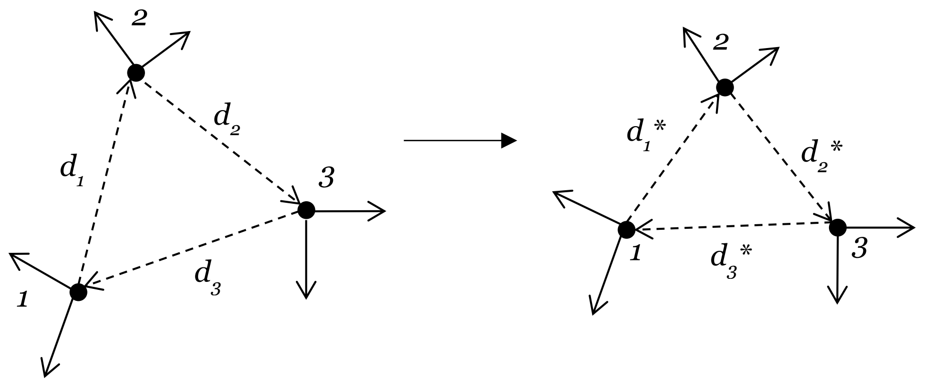

Figure 3.3: Three-agent distance-based formation control objective

4 Displacement-based Formation Control

In this section, displacement-based formation control in the input-state-output port-Hamiltonian form will be explored. In order to achieve the formation control goal as formulated in (2.3), a controller needs to be designed, after which it will be validated by proving its stability using Lyapunov's stability theorem.

4.1 Controller Design

The controller design in this section will be based on assigning virtual couplings between agents. These virtual couplings consist of a spring and a damper in parallel. That way, the spring ensures the formation control objective is reached, and the damper ensures a smooth transition by preventing overshoot and unnecessary oscillations. A schematic representation of the closed loop system is given below in figure 4.1.

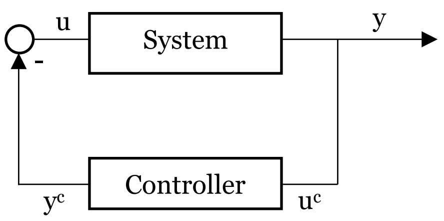

Figure 4.1: Schematic representation of closed loop system

The spring-damper controllers will be modelled using spring elongation z_j, input velocity u_j^c, and corresponding output force y_j^c [2][116]. Let $\bar{z}_j = z_j - z_j^* \in \mathbb{R}^{2\times 1}$, the dynamics are then given by

$$\dot{\bar{z}}_j = u_j^c, \tag{4.1}$$

$$y_j^c = \frac{\partial H_j^c}{\partial \bar{z}_j} + D_j^c u_j^c, \tag{4.2}$$

where $D_j^c \in \mathbb{R}^{E\times E}$ is a damping coefficient matrix, and H_{ij}^c the Hamiltonian that equals the stored potential energy in the spring and is given by

$$H_j^c = \frac{1}{2}\bar{z}_j{}^T K_j^c \bar{z}_j, \tag{4.3}$$

with spring constant matrix $K_j^c \in \mathbb{R}^{E\times E}$.

From the above equations, combined with the schematic representation of the closed loop system, the following control law can be written:

$$u_j = -k_j^c \bar{z}_j - d_j^c \dot{\bar{z}}_j. \tag{4.4}$$

Note that for the above equation, the formation control is related to the edge j, but the control objective of this research is to control the agents i rather than the edge. Hence, to transfer the control power from edge j to agent i, the incidence matrix defined in (3.5) or (3.6) can be used [2][16] for deriving the coupling equations such that

$$u = -(B \otimes I_2)y^c, \tag{4.5}$$

$$u^c = (B^T \otimes I_2)y. \tag{4.6}$$

Note that for both equations (4.5) and (4.6), y and y^c can be simply substituted with equations (3.4) and (4.2). Then, the final control law can be written such that

$$u = -(B \otimes I_2)K^c\bar{z} - (B \otimes I_2)D^c(B^T \otimes I_2)M^{-1}p \tag{4.7}$$

Theorem 4.1: Consider a multi-agent system modelled by double integrator dynamics, shown in (3.4). The underlying graph consists of a two or three-agent acyclic network as shown in figure 3.1. Under the proposed control law defined in (4.7), the group of agents achieve the control objective defined in (2.3), and are globally stable.

4.2 Closed-loop System in Input-State-Output port-Hamiltonian Form

Now that the dynamics of the system and the controller, and their relation has been defined, a new closed-loop system can be written in the input-state-output port-Hamiltonian form. Recall equations (3.2),(4.3) and (4.5), we then have that the initial state variable $\dot{p}$ becomes

$$\dot{p} = -D^r\frac{\partial H}{\partial p} - (B \otimes I_2)\left(\frac{\partial H^c}{\partial \bar{z}} + D^c u^c\right), \tag{4.8}$$

and can be derived further, substituting equations (3.4) and (4.6) into (4.8), such that

$$\dot{p} = -D^r\frac{\partial H}{\partial p} - (B \otimes I_2)\left(\frac{\partial H^c}{\partial \bar{z}} + D^c\left((B^T \otimes I_2)\frac{\partial H}{\partial p}\right)\right)$$

$$\dot{p} = -(B \otimes I_2)\frac{\partial H^c}{\partial \bar{z}} - \left(D^r + (B \otimes I_2)D^c(B^T \otimes I_2)\right)\frac{\partial H}{\partial p}. \tag{4.9}$$

As for the newly defined state variable $\dot{\bar{z}}$, the dynamics are given by combining equations (3.4),(4.1) and (4.6) such that

$$\dot{\bar{z}} = (B^T \otimes I_2)\frac{\partial H}{\partial p}. \tag{4.10}$$

The closed-loop Hamiltonian is then the sum of the two earlier defined Hamiltonian equations (3.3) and (4.3), and results in the following equation[2][16]:

$$H^{cl} = \frac{1}{2}p^TM^{-1}p + \frac{1}{2}\bar{z}^TK^c\bar{z}, \tag{4.11}$$

Finally, the new closed-loop system can be written as

$$\begin{bmatrix}\dot{q}\\ \dot{p}\\ \dot{\bar{z}}\end{bmatrix} = \begin{bmatrix}0 & 1 & 0\\ -1 & -D^r - (B \otimes I_2)D^c(B^T \otimes I_2) & -(B \otimes I_2)\\ 0 & (B^T \otimes I_2) & 0\end{bmatrix}\begin{bmatrix}\frac{\partial H^{cl}}{\partial q}(p,\bar{z})\\ \frac{\partial H^{cl}}{\partial p}(p,\bar{z})\\ \frac{\partial H^{cl}}{\partial \bar{z}}(p,\bar{z})\end{bmatrix},$$

$$y = \begin{bmatrix} 0 & I_2 & 0 \end{bmatrix} \begin{bmatrix} \frac{\partial H^{cl}}{\partial q}(p,\bar{z}) \\ \frac{\partial H^{cl}}{\partial p}(p,\bar{z}) \\ \frac{\partial H^{cl}}{\partial \bar{z}}(p,\bar{z}) \end{bmatrix}. \tag{4.12}$$

4.3 Stability of the System

In order to determine the stability of the system, Lyapunov's stability theorem will be used to prove theorem 4.1.

Proof: Take the closed-loop Hamiltonian $H^{cl}(p,\bar{z})$ (4.11) as the candidate Lyapunov function. The equilibrium of the formation is defined by $p \to 0$ and $z \to z^*$ (or $\bar{z} \to 0$) when $t \to \infty$. By taking (4.11) as the Lyapunov function, it can be proven that stability criteria $V(0) = 0$ (2.11) is satisfied in the following way:

$$H^{cl}(0,0) = \frac{1}{2}0M^{-1}0 + \frac{1}{2}0K^c0,$$

$$H^{cl}(0,0) = 0. \tag{4.13}$$

The other criteria for stability, $V(x) > 0 \; in \; \mathcal{R} - \{0\}$, is satisfied by the fact that both M and K^c are positive constants, and that momentum p and displacement z are squared. Therefore, the above mentioned constraint is satisfied for all values where $p \neq 0$ and $\bar{z} \neq 0$.

In order to test the following criteria $\dot{V}(x) \leq 0 \; in \; \mathcal{R}$ (2.12), the derivative of $H^{cl}(p,\bar{z})$ is written out below using the chain rule:

$$\dot{H}^{cl}(p,\bar{z}) = \frac{\partial H^{cl}}{\partial q}\dot{q} + \frac{\partial H^{cl}}{\partial p}\dot{p} + \frac{\partial H^{cl}}{\partial \bar{z}}\dot{\bar{z}}. \tag{4.14}$$

The derivative with respect to position q is zero, and both $\dot{p}$ and $\dot{\bar{z}}$ can be substituted from (4.12) resulting in the following simplified equation:

$$\dot{H}^{cl}(p,\bar{z}) = -\frac{\partial^T H^{cl}}{\partial p}\left(D^r + (B \otimes I_2)D^c(B^T \otimes I_2)\right)\frac{\partial H^{cl}}{\partial p}. \tag{4.15}$$

From this equation, it can be concluded that the minus sign, combined with the squared partial derivative of H^{cl}, ensures that $\dot{V}(x) \leq 0 \; in \; \mathcal{R}$. Then, using LaSalle's invariance principle, $\dot{H}^{cl}$ converges to the largest invariant set where $\dot{H}^{cl} = 0$. Subsequently, this results in $p = 0$ and $\dot{p} = 0$, which can be substituted in (4.12) such that

$$0 = -(B \otimes I_n)\frac{\partial H^c}{\partial \bar{z}} = -(B \otimes I_n)K^c\bar{z}. \tag{4.16}$$

In order to prove that the above equation holds, it must be proven that $-BK^c\bar{z} = 0$. This will be done by proving that the columns of BK^c are linearly independent. That way, it is proven that for (4.16) to hold, $\bar{z} \to 0$.

Linear independence can be proven by determining the kernel, and showing that it is equal to the zero vector $\{0\}$. The kernel can be determined by writing the matrices in Reduced Row Echelon Form (RREF), and determining the rank of the matrices. If the rank of a matrix is full (every column in the RREF matrix has a non-zero entry), the columns are linearly independent, and kernel is a zero vector. It must be noted that $\ker(BK^c) = \ker(B)$, because K^c is a positive diagonal matrix. As an example, a three-agent system will be taken to prove linear independence with incidence matrix $B \in \mathbb{R}^{3\times 2}$ and diagonal spring constant matrix $K^c \in \mathbb{R}^{2\times 2}$.

First, matrix BK^c will be put in RREF such that:

$$RREF\left(\begin{bmatrix} -k_1^c & 0 \\ k_1^c & -k_2^c \\ 0 & k_2^c \end{bmatrix}\right) = \begin{bmatrix} k_1^c & 0 \\ 0 & k_2^c \\ 0 & 0 \end{bmatrix}. \tag{4.17}$$

It can be seen that the rank of the RREF matrix is 2, meaning that the columns of this matrix are linearly independent. Subsequently, in order for equation (4.16) to hold, $\bar{z}$ must converge to zero. Hence, the control objective is achieved.

■

4.4 Simulations

The system was simulated in Matlab Simulink, version 2021b, with $N = 3$ agents, and $M = 2$ edges, and moving in an x-y plane. Each agent was considered to be a point-mass, and has a mass $m_i = 1\,kg$. The virtual couplings of edges j each have a spring constant $k_j^c = 0.3\,kgs^{-2}$ and a damping coefficient $d_j^c = 1\,kgs^{-2}$. The desired displacements z_j are set to $z_1 = [0.5, -2]$ and $z_2 = [-1{,}0]$, which is equal to the spring length.

Furthermore, the agents' initial conditions are given by $q_{1_0} = [1{,}1]^T$, $q_{2_0} = [2{,}2]^T$, $q_{3_0} = [3{,}1]^T$ in $[m]$, and $p_i = [0.5{,}0.5]^T$ $[kgms^{-1}]$ for all 3 agents i. This means that the initial velocity of all agents is set to $\dot{q}_i = 0.5$ $[ms^{-1}]$. The Simulink models and Matlab scripts can be found in Appendix A.

4.4.1 Simulation of system without friction

In this part, friction term d^r is set to zero. In the figures below, the results of the Matlab simulation can be seen.

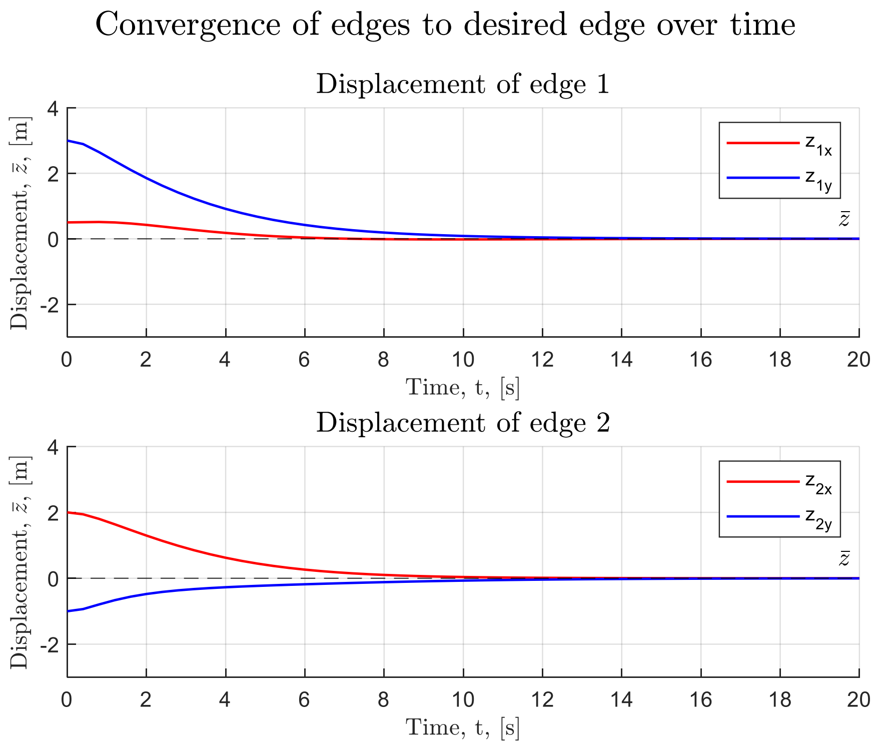

Figure 4.2: Edge convergence over time, which shows the control objective is achieved.

In figure 4.2, the convergence of both edges z_1 and z_2 is shown. It can be concluded that this plot shows that control objective (2.3) is achieved after roughly 10 seconds. Figure 4.3 shows the displacement of both edges as well, but this time it shows convergence to the values specified in the first part of section 4.4 (desired displacement values).

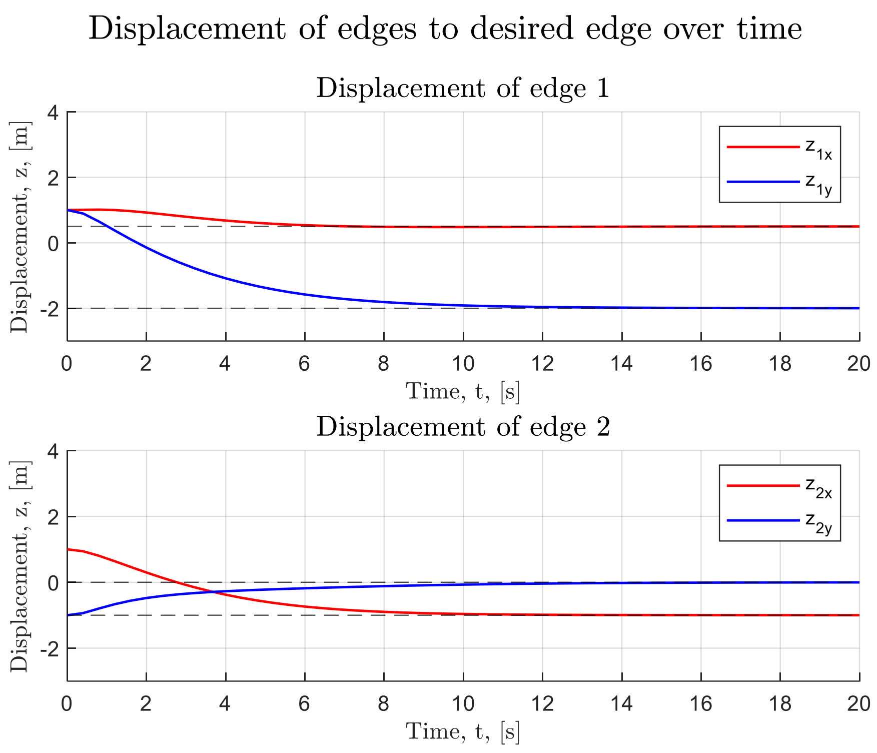

Figure 4.3: Edge displacement over time.

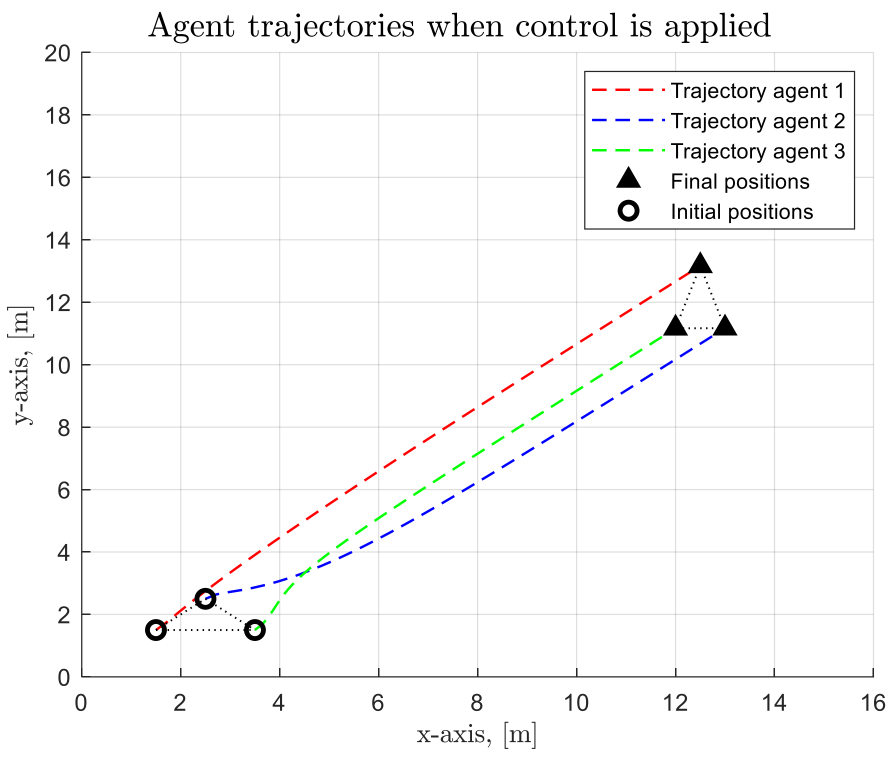

Figure 4.4: Trajectories of the agents after control is applied.

In order to visualize how the agents move in the system, figure 4.4 was created to show the trajectories of the agents. Here, it can be seen that the formation continues to travel at a certain velocity, even after the formation is initially formed. The Matlab simulation was made to be 20 seconds, Hence the final position of the agents in figure 4.4.

Last, figure 4.5 shows the velocity of each agent over time. After a few other simulations with different initial velocities of agents, an interesting observation arose.

Remark 4.1: The agents from system (4.12) return to their initial velocities as $t \to \infty$ when friction is neglected.

This can be attributed to the potential energy stored in the spring parts of the virtual couplings between agents. Naturally, the agents have an initial velocity. When the virtual coupling springs are not in equilibrium, the potential energy in the spring is not equal to zero. Subsequently, this affects the velocity of both agents connected by the edge. However, once the desired formation is achieved, the potential energy in the springs equal zero. Then, the agents return to their initial velocity, as there is no energy dissipated from the system (friction is neglected).

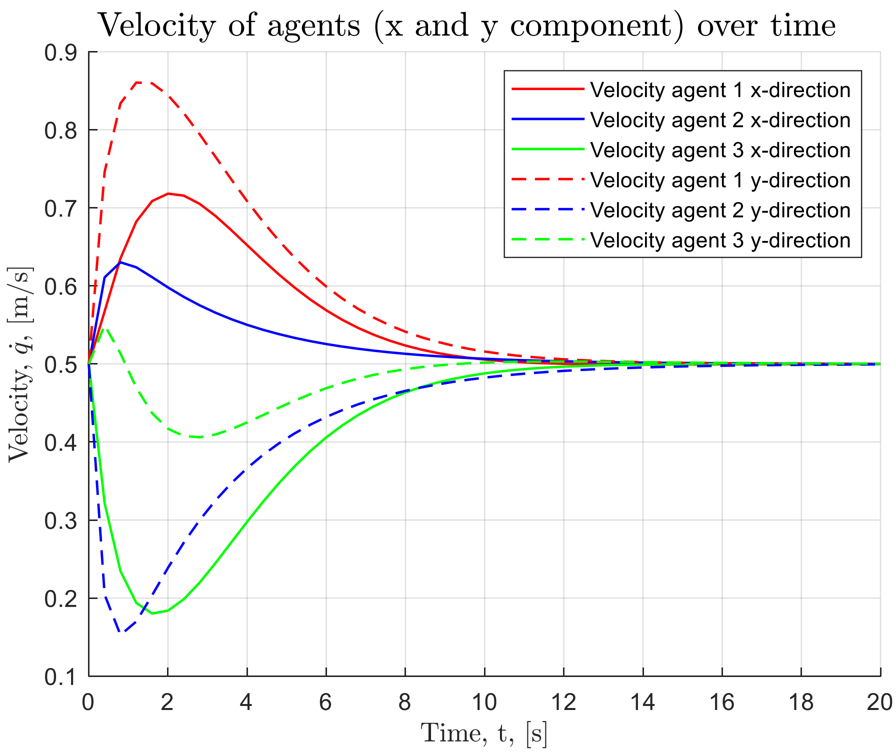

Figure 4.5: Velocity of agents.

4.4.2 *Simulation of system with friction*

For the next simulations, friction is considered. This means that friction coefficient d^r is set to 0.3 $[kgs^{-1}]$, and energy will dissipate from the system. This is both reflected in the time it takes for the final formation to form, and the final velocity of the agents. Figures 4.6 and 4.7 show the convergence and displacements of the edges again, the settling time is slightly longer than a system without friction (roughly 11 seconds).

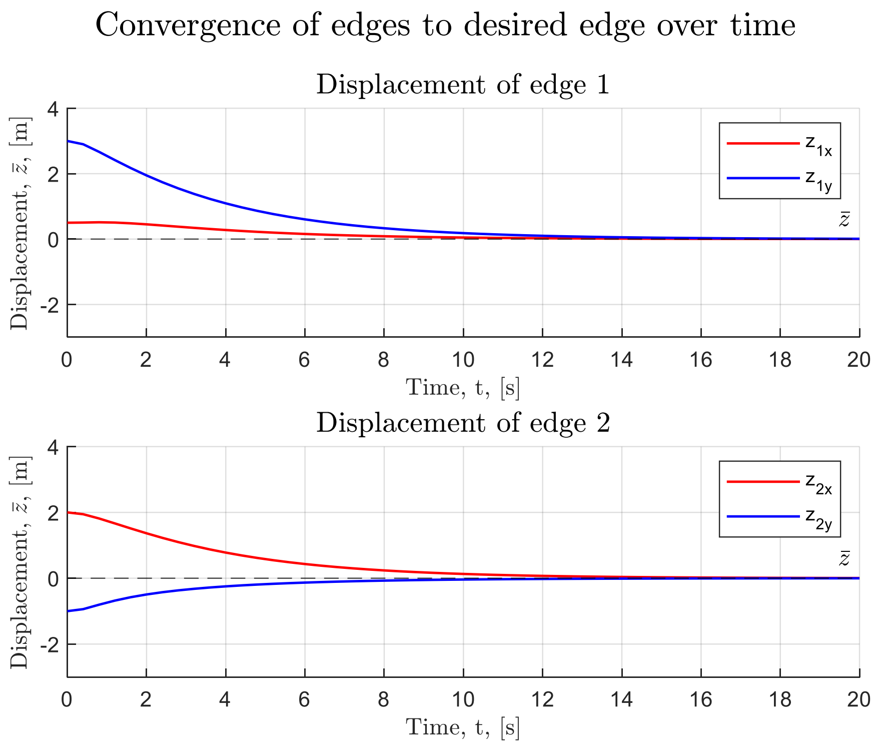

Figure 4.6: Edge convergence over time.

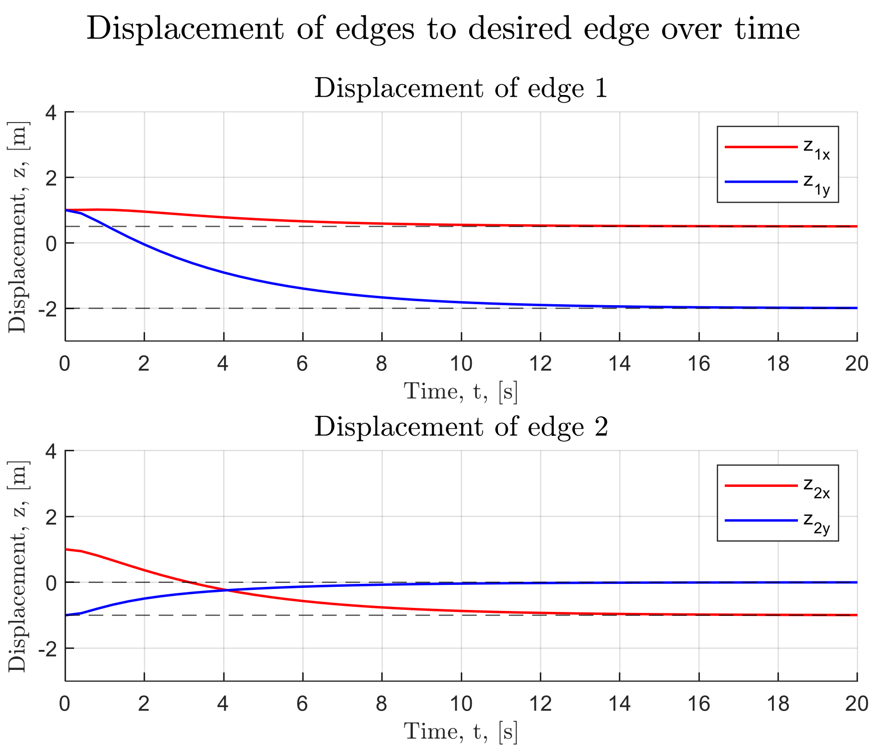

Figure 4.7: Edge displacement over time

The significant difference can be seen in the agent trajectory and velocity plots. In contrast to the system where friction is neglected, the agents come to a full stop relatively quickly once the final formation is reached.

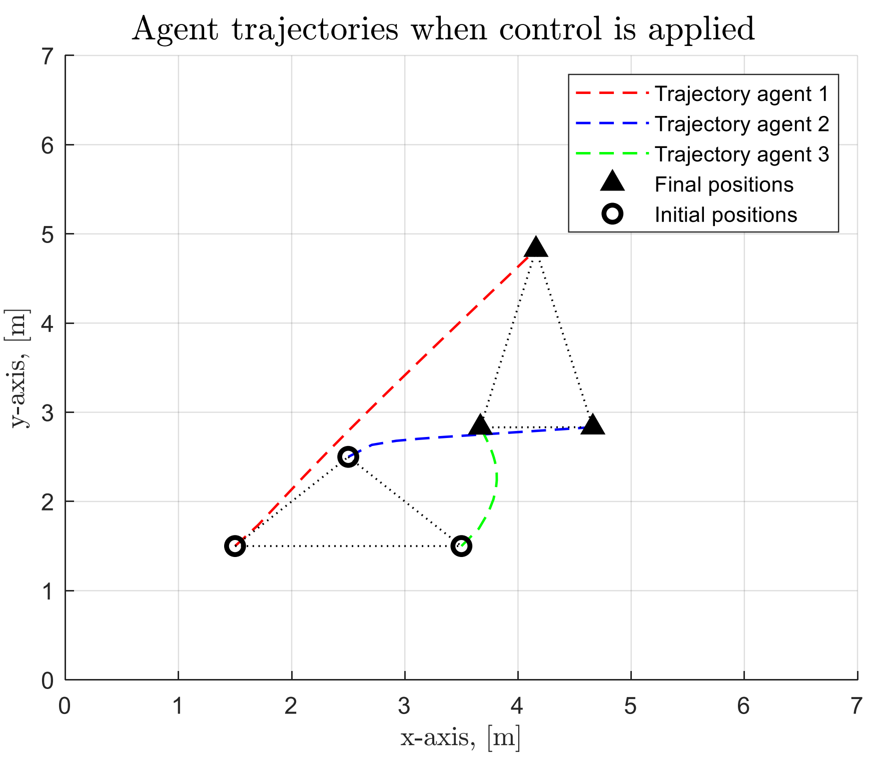

Figure 4.8: Trajectories of the agents after control is applied

Moreover, the velocity plot also shows that the agents velocities converge to zero instead of their initial given velocities. As mentioned before, this was to be expected due to energy leaving the system.

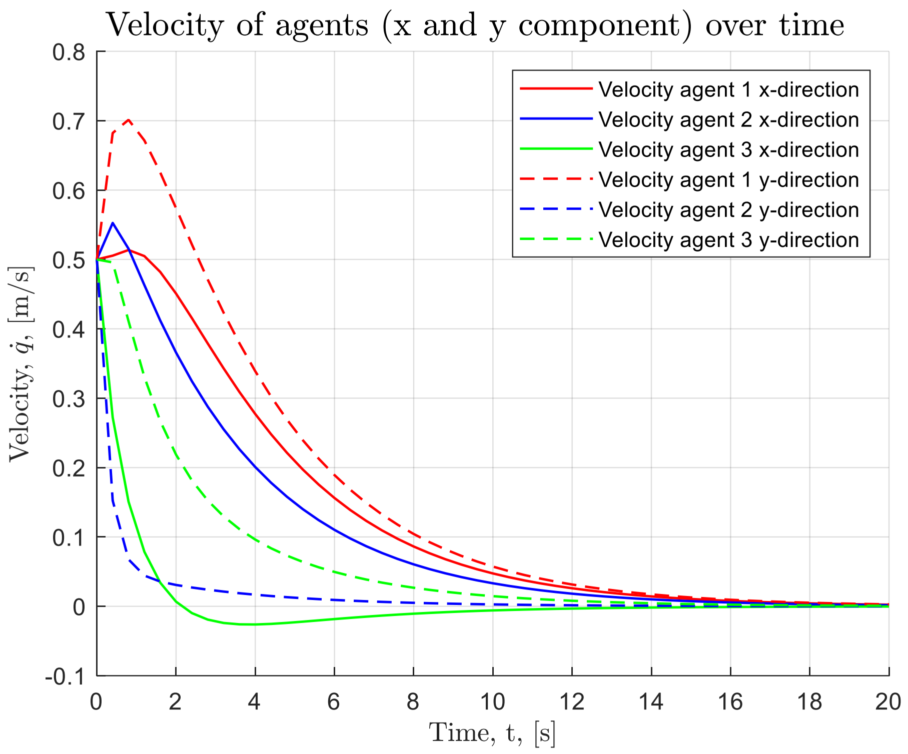

Figure 4.9: Velocity of agents

5 Distance-based Formation Control

For the distance-based formation control part of this research, there will be more challenges that need to be faced when designing and validating the controller. As mentioned before, this can partially be attributed to the fact that this type of formation control is non-linear. Additionally, the rank of incidence matrix B for a three-agent system is not full when put in RREF, making it harder to prove stability. Therefore, most of the section below will be based on research conducted by Li et al. [24], as that is also the only research up to date on distance-based control in port-Hamiltonian form.

For this part of the research, the same system will be used as defined in section 3.1. Additionally, the formal control objective for distance-based formation control was previously explained in section 3.2.2.

5.1 Controller Design

The closed loop system is again represented below in figure 5.1. For this part of the research, only a spring will be used for assigning virtual couplings between agents. This can be attributed to the fact that a damping term does not have an impact on the stability of the system, as distance d_j is a scalar compared to displacement z_j, which is a vector.

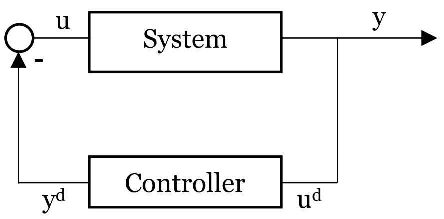

Figure 5.1: Schematic description of the system

The controller will be modelled using the dynamics of distance d_j, or $\dot{d}$, input velocity u_j^d, and corresponding output force y_j^d. The dynamics for the system pictured above, are given by

$$\dot{d} = \frac{\partial H^d}{\partial d} \frac{\partial d}{\partial z} \dot{z}, \tag{5.1}$$

$$y_j^d = \frac{\partial H_j^d}{\partial z_j}, \tag{5.2}$$

where H_j^d is the Hamiltonian. In order to force the formation to reach the desired distance requirements, as shown in figures 3.2 and 3.3, the desired distance-based Hamiltonian is given by

$$H_j^d = \frac{1}{2}\left(d_j^2 - d_j^{*2}\right). \tag{5.3}$$

From the above equations, combined with the schematic overview shown in figure 5.1, the following control law for edge j is proposed to be

$$u_j = -\frac{\partial H_j^d}{\partial z_j}. \tag{5.4}$$

Similar to displacement-based formation control, the power of control in the above described equation lies with the edge. This power can again be translated to power that lies with the agents by using the incidence matrix to derive the coupling equations such that

$$u = -(B \otimes I_2)y^d, \tag{5.5}$$

$$u^d = (B^T \otimes I_2)y. \tag{5.6}$$

Note that for both equations (5.5) and (5.6), y and y^d can be substituted with equations (3.4) and (5.2). The final control law is then given by

$$u = -(B \otimes I_2)z. \tag{5.7}$$

Theorem 5.1: Consider a multi-agent system modelled by double integrator dynamics, shown in (3.4). The underlying graph consists of a two or three-agent cyclic network as shown in figures 3.2 and 3.3. Under the proposed control law defined in (5.7), the group of agents achieve the control objective defined in (2.4). In addition, it is assumed that the initial position of the agents are in the neighborhood of the desired formation. Then, the system is locally stable.

5.2 Closed-loop system in input-state-output port-Hamiltonian form

Now that the dynamics of the controller, and its relation to the system has been defined, the closed-loop system can be written in input-state-output port-Hamiltonian form. Recall equations (3.4) and (5.2), then state variable $\dot{p}$ becomes

$$\dot{p} = -D^r \frac{\partial H}{\partial p} - (B \otimes I_2)\frac{\partial H^d}{\partial z}. \tag{5.8}$$

It must be noted that the objective of distance-based control is with respect to distance, but in the above equation, displacement z is used. This is because distance d and displacement z are naturally closely related. Hence, we have the following relation

$$\frac{\partial H^d}{\partial z} = \frac{\partial H^d}{\partial d}\frac{\partial d}{\partial z}. \tag{5.9}$$

From this, it can be concluded that $\frac{\partial d}{\partial z} = \frac{z}{d}$. Moreover, distance d is naturally a function of position q. Hence, another relation that is significant for the closed-loop system is the following:

$$\dot{d} = \frac{\partial d}{\partial q}\dot{q}. \tag{5.10}$$

Then, similar to displacement-based formation control, the closed-loop Hamiltonian for this system is simply $H^{cl} = H + H^d$ such that

$$H^{cl} = \frac{1}{2}p^T M^{-1} p + \frac{1}{2}\left(d^2 - d^{*2}\right). \tag{5.11}$$

Finally, the closed-loop system can be written as

$$\begin{bmatrix}\dot{q}\\ \dot{p}\\ \dot{d}\end{bmatrix} = \begin{bmatrix} 0 & I_2 & 0 \\ -I_2 & -D^r & -\frac{z}{d}(B \otimes I_2) \\ 0 & \frac{z^T}{d}(B^T \otimes I_2) & 0 \end{bmatrix} \begin{bmatrix} \frac{\partial H^{cl}}{\partial q}(p,d) \\ \frac{\partial H^{cl}}{\partial p}(p,d) \\ \frac{\partial H^{cl}}{\partial d}(p,d) \end{bmatrix}, \tag{5.12}$$

$$y = \begin{bmatrix} 0 & I_2 & 0 \end{bmatrix} \begin{bmatrix} \frac{\partial H^{cl}}{\partial q}(p,d) \\ \frac{\partial H^{cl}}{\partial p}(p,d) \\ \frac{\partial H^{cl}}{\partial d}(p,d) \end{bmatrix}.$$

5.3 Stability of the system

Similar to displacement-based formation control, the stability of the controller will be tested using the Lyapunov stability theorem. However, when taking the closed loop Hamiltonian H^{cl} (5.11) as the candidate Lyapunov function, it becomes clear quickly that this equation does not satisfy the stability requirements. Hence, a new state will be formulated in order to prove that the system is stable.

The new state will be defined using a new Hamiltonian given by

$$H_j^d = \frac{1}{4}\left(e_j^d\right)^2, \tag{5.13}$$

where

$$e_j^d = d_j^2 - d_j^{*2}. \tag{5.14}$$

Using the same control law u_j (5.4) and coupling dynamics (5.5) and (5.6), the interconnection between the controller and the original system is defined as

$$u^d = -(B^T \otimes I_d)\frac{\partial H^d}{\partial z} = -z^T(B^T \otimes I_2)e^d. \tag{5.15}$$

The new Hamiltonian of the closed-loop system is then given by $H^{cl} = H + H^d$ such that

$$H^{cl} = \frac{1}{2}p^T M^{-1} p + \frac{1}{4}(e^d)^2. \tag{5.16}$$

The new state equation for $\dot{p}$ can be obtained by combining equations (5.8),(5.9), (5.10) and (5.14) and is given by

$$\dot{p} = -D^r\frac{\partial H}{\partial p} - z(B \otimes I_2)\frac{\partial H^d}{\partial e^d}, \tag{5.17}$$

and the new state equation for $\dot{e}^d$ is given by

$$\dot{e}^d = z^T (B^T \otimes I_2) \frac{\partial H^{cl}}{\partial p}. \tag{5.18}$$

The new closed-loop system is then

$$\begin{bmatrix} \dot{q} \\ \dot{p} \\ \dot{e}^d \end{bmatrix} = \begin{bmatrix} 0 & I_2 & 0 \\ -I_2 & -D^r & -(B \otimes I_2) z \\ 0 & z^T (B^T \otimes I_2) & 0 \end{bmatrix} \begin{bmatrix} \frac{\partial H^{cl}}{\partial q}(p, e^d) \\ \frac{\partial H^{cl}}{\partial p}(p, e^d) \\ \frac{\partial H^{cl}}{\partial e^d}(p, e^d) \end{bmatrix}, \tag{5.19}$$

$$y = \begin{bmatrix} 0 & I_2 & 0 \end{bmatrix} \begin{bmatrix} \frac{\partial H^{cl}}{\partial q}(p, d) \\ \frac{\partial H^{cl}}{\partial p}(p, d) \\ \frac{\partial H^{cl}}{\partial e^d}(p, d) \end{bmatrix}.$$

Now, the proof for theorem 5.1 will follow.

Proof: Take closed loop Hamiltonian equation H^{cl} (5.16) as the candidate Lyapunov function. The equilibrium for the formation is defined by $d \to d^*$ when $t \to \infty$. It follows that $H^{cl} \geq 0$, because the both terms are squared. Moreover, it can be seen that $H^{cl}(0,0) = 0$, satisfying stability constraint (2.11).

As for the time derivative of H^{cl}, using the chain rule it is given by

$$\dot{H}^{cl}(p, e^d) = \frac{\partial H^{cl}}{\partial q} \dot{q} + \frac{\partial H^{cl}}{\partial p} \dot{p} + \frac{\partial H^{cl}}{\partial e^d} \dot{e}^d \tag{5.20}$$

where $\dot{q}$, $\dot{p}$ and $\dot{e}^d$ can be substituted from (5.17). It follows that

$$\dot{H}^{cl}(p, e^d) = -\frac{\partial H^{cl}}{\partial p} D^r \frac{\partial H^{cl}}{\partial p}, \tag{5.21}$$

which satisfies (2.12). Then, invoking LaSalle's Invariance Principle, the system converges to the largest invariant where $\dot{H}^{cl} = 0$. Subsequently, $p = 0$ and $\dot{p} = 0$, which can be substituted in (5.19) such that

$$0 = -(B \otimes I_d) z \frac{\partial H^d}{\partial e^d}. \tag{5.22}$$

By calculating $\frac{\partial H^d}{\partial e^d}$ and substituting it into (5.22), the following equation can be obtained:

$$0 = -R(z) e^d, \tag{5.23}$$

where $R(z)$ is the rigidity matrix [24] defined as

$$R(z) = (B \otimes I_d) z. \tag{5.24}$$

As mentioned before, the graphs in this research are considered to be minimally and infinitesimally rigid. Subsequently, according to lemma 1 from [25], matrix $R(z)R(z)^T$ is positive definite. A condition for positive definiteness is that all columns of the matrix in question are linearly independent. Hence, in order for (5.23) to hold, e^d is forced to zero, which concludes the proof.

■

5.4 Simulations

In contrast to the simulations for displacement-based formation control, no Matlab Simulink was used. Instead, only a Matlab script was used which can be found in Appendix B.

For these simulations, a network with $N = 3$ agents and $E = 3$ edges is used, which can move in an x-y plane. Each agent has a mass $m_i = 1\ [kg]$. Regarding the virtual couplings, there is no spring constant and no damping coefficient. Moreover, similar to the displacement-based setup, the initial positions are set to $q_{1_0} = [0,1]^T$, $q_{2_0} = [2,2]^T$, $q_{3_0} = [3,1]^T$ in $[m]$, and $p_i = [0.5,0.5]^T\ [kgms^{-1}]$ for all 3 agents i. In turn, this means the initial velocity of all agents is $\dot{q}_i = 0.5\ [ms^{-1}]$. Last, all three desired distances, or spring lengths, are set to $1.5\ [m]$.

For these simulations, the friction coefficient d^r is set to $1.5\ [kgs^{-1}]$.

Remark 5.1: The agents in system (5.17) will never achieve a steady formation when friction in the system is neglected and no damping term is included in the controller.

This can be attributed to the fact that a spring coupling between agents will continue to oscillate when there is no damping term. Hence, the virtual couplings between agents will not converge to their equilibrium. A lack of a damping term could be compensated by adding in a friction term to the system. Hence, the friction is taken into account in the following simulations. It must be noted that when velocity tracking is applied to the system, which was also researched in [16] and [24].

Figure 5.2 shows the convergence of the distance errors as defined in (5.12). It can be noted that after roughly 7 seconds, the final formation is formed.

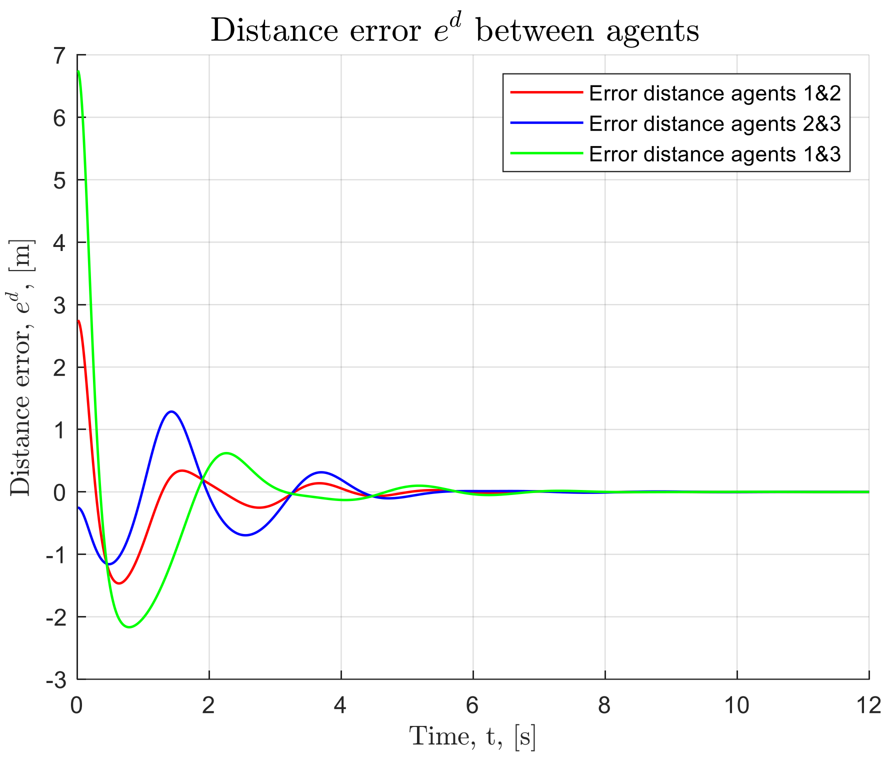

Figure 5.2: Distance error convergence over time, which shows the control objective is achieved.

In figure 5.3, the distance between agents over time is shown. As all desired distances were set to 1.5, the figure shows that the distances converge to this value.

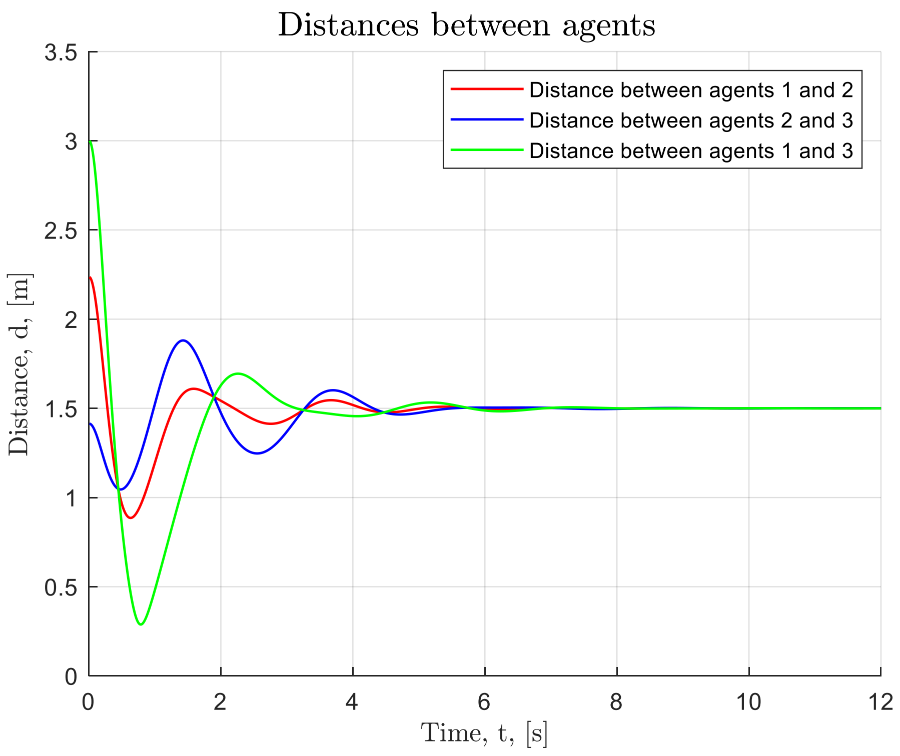

Figure 5.3: Distances between agents over time.

In order to see how the positions of the agents change over time, another agent trajectory graph was created, and can be seen in figure 5.4. This figure shows that the agents eventually achieve their desired inter-distances. The initial formation, and final formation are also clearly visualized with black dotted and dash-dotted lines.

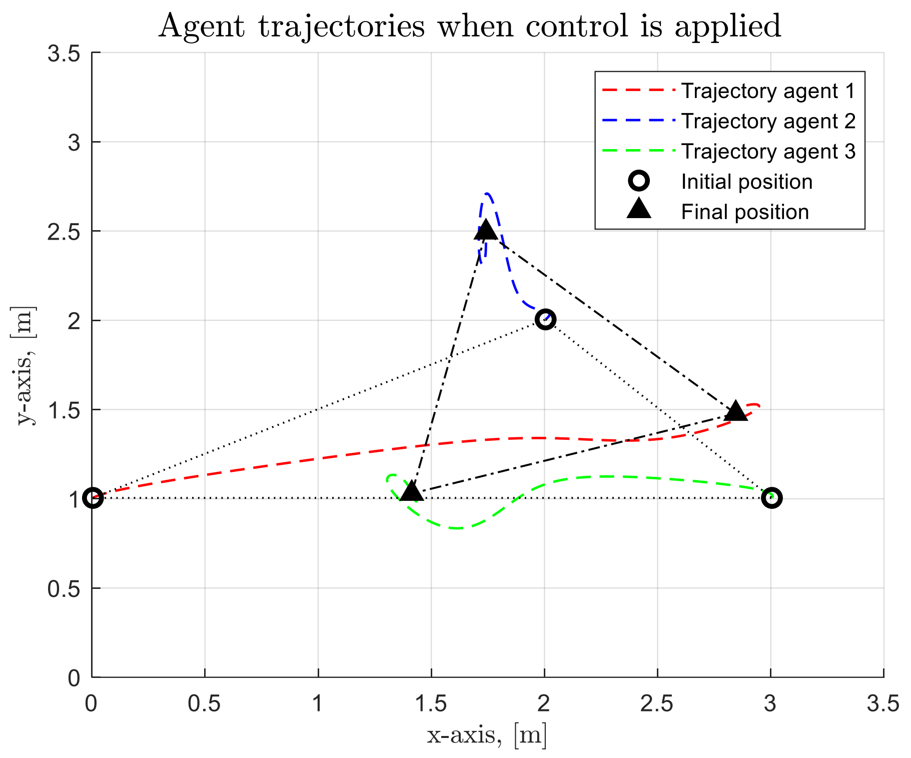

Figure 5.4: Trajectories of agents after control is applied

The last figure, figure 5.5, shows the velocities of the agents over time. As expected, due to friction the velocities converge to zero.

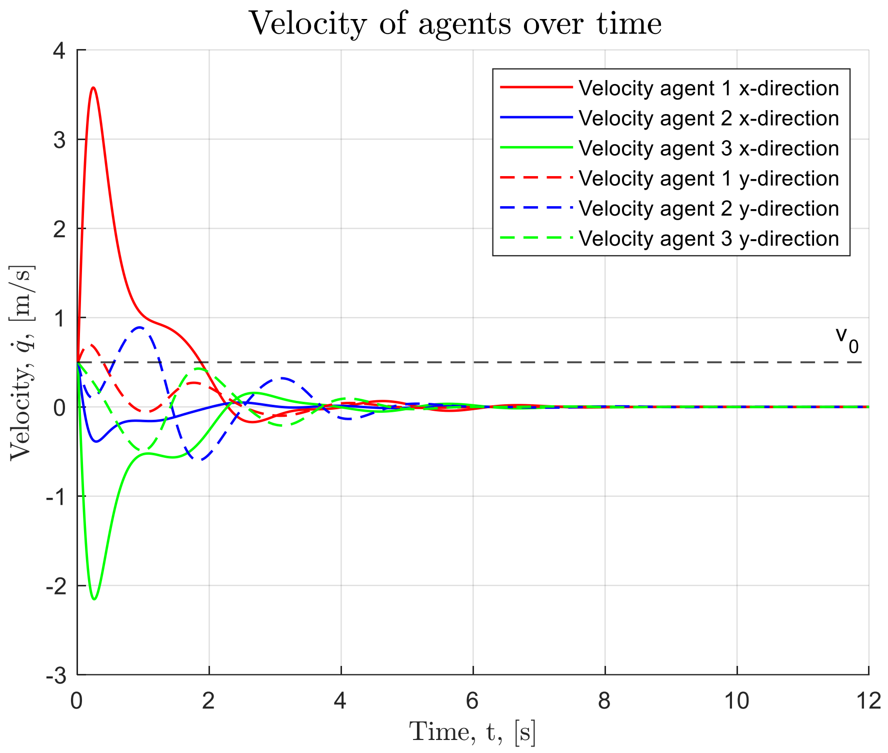

Figure 5.5: Velocity of agents

6 Conclusion and Future Research

6.1 Conclusion

In this research, two different control laws were proposed in order to achieve a desired formation for displacement- and distance-based formation control. This was done in the port-Hamiltonian framework, as this provides interesting insights into the behaviour and interconnections of the systems.

For the displacement-based setup, the control law is based on a virtual coupling that consists of a spring and a damper in parallel. It was shown using simulations that this control law ensures that the control objective is reached. Further, it was proven that this control law can ensure this achievement for any system consisting of two or three agents in a 2-dimensional space. In addition, using Lyapunov's Stability Theorem, the proposed control law was proven to be stable in create a closed-loop system.

Regarding the distance-based control formation, it can be concluded that theorem 5.1 is proven to be true. The figures in section 5.4 show that the control objective as defined in equation (2.4) is achieved. Additionally, using the Lyapunov Stability Theorem and the LaSalle Invariance Principle, local stability was proven.

6.2 Future Research

Based on the research conducted for this Bachelor Integration Project, some interesting ideas for future research suggestions came to mind.

First, a somewhat obvious suggestion would be to explore systems with more than three agents. Due to the fact that only two and three agent systems were considered in this research, all graphs in this research are considered to be rigid. When adding more agents to the system, graph rigidity becomes an extra factor that will need to be taken into account. In addition, the relation between proving stability of a formation controlled system and rigidity is interesting, as [24] made an interesting discovery.

Another interesting research would be to research formation control with two and three agents in a 3-dimensional space. This would make the formation control problems more complicated, but this is necessary in order to practically apply formation control.

For this research, some graphs were made of the trajectories that the agents followed once control was applied. It is interesting to see what path the agents choose in order to obtain the desired formation, because not always the position closest to the agents initial position is chosen with displacement-based formation control. With this regard, research could be done to improve its efficiency by applying agent trajectory tracking to a formation control setup.

Last, the systems used in this research did not include disturbances. When formation control is applied in the physical world, some problems may arise, such as agents encountering obstacles. As this was not researched in this paper, the model could also not be tested under disturbing circumstances. Hence, the model's robustness could not be tested. Due to the fact that incorporating disturbances in the system makes the research immediately very complicated, it would make an interesting topic for future research.

Appendix A. Displacement-based Formation Control

A.1 Simulink Models

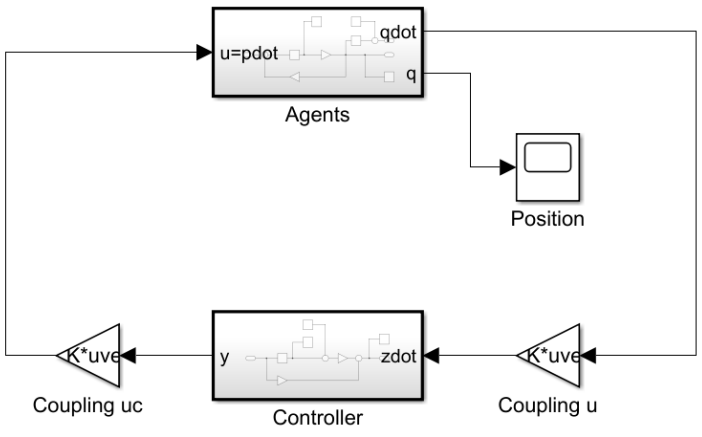

Figure 7.1: Simulink model representation of figure 4.1.

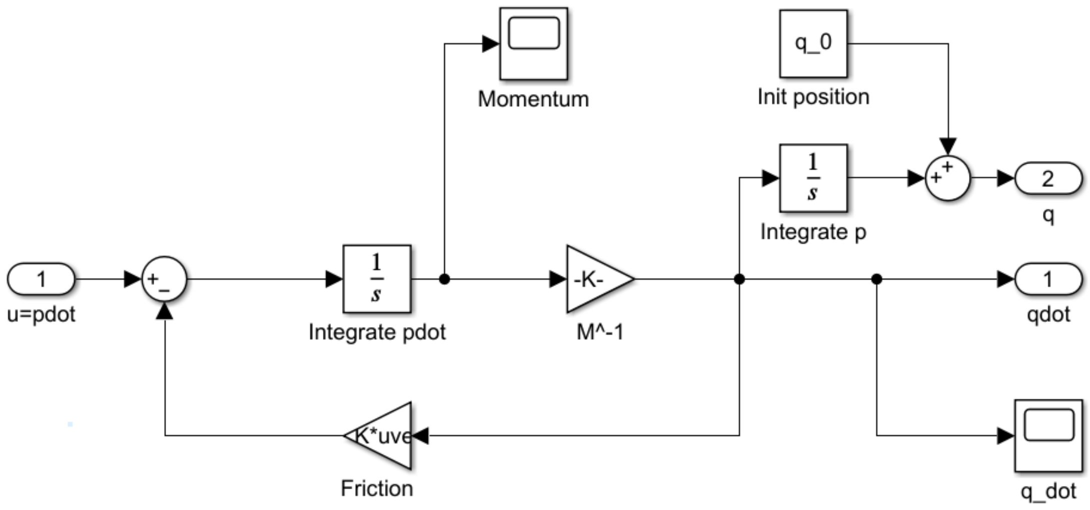

Figure 7.2: Simulink model of Agents.

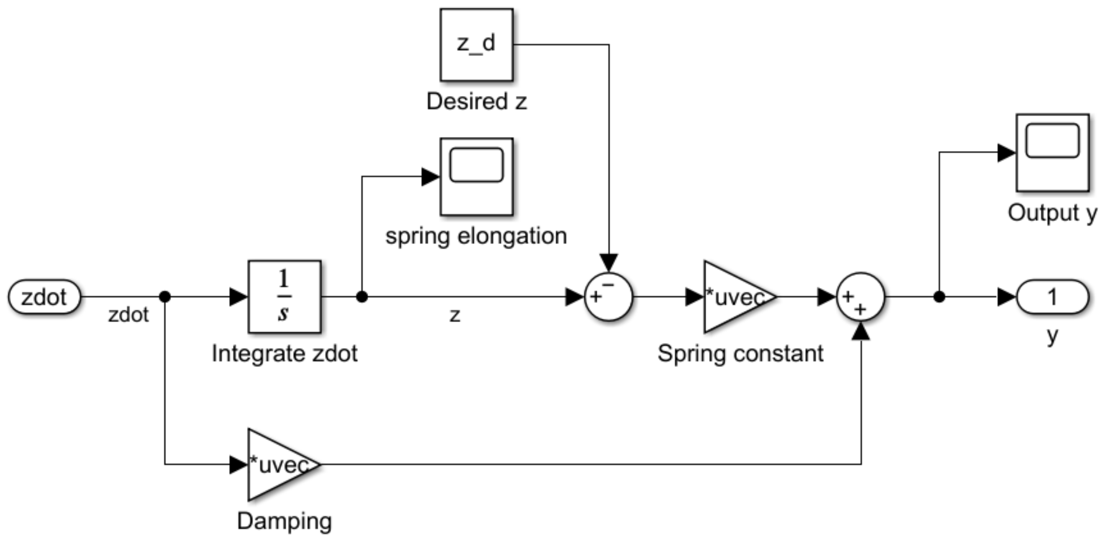

Figure 7.3: Simulink model of Controller

A.2 Matlab Script

```
% Script for displacement-based formation control
% Ba IP project
% J.M. Oeben
% s4101839

clc
clear
close all

% Number of agents
n=3

% Define constants
M = eye(n);                         % mass matrix
B = [-1,0;1,-1;0,1];                % Incidence matrix B
E = 2;                              % nr of edges
z_d1 = [0.5,-2]; z_d2 = [-1,0]; %desired displacement
z_d = [z_d1,z_d2];

kronB = kron(B,eye(2)); % Kronecker product

% Agents
M_a = 1.*ones(2*n,1); % agent mass
D_r = 0.*eye(2*n)';  % linear damping
K = 0.3.*eye(2*E);     % spring constant matrix
q_10=[1 1]; q_20=[2 2]; q_30=[3 1];
q_0 = [q_10'; q_20'; q_30'];     %initial position agents
p_0 = 0.5.*ones(1,2*n);          %initial momentum agents

%Virtual couplings
D_f = 1.*eye(2*E);      % virtual damping constant
p_d = 1*ones(2*n,1);    % Desired momentum

z_0 = kronB'*q_0;       % Initialising z_0
```

```
% Setting up simulation
simtime = 20;
sim('sim_attempt',simtime)

% Retrieving data from simulink model
sim_t = position.time();
sim_q_a = position.signals.values();
sim_z = spring_elongation.signals.values();
sim_v = velocity.signals.values();

% Z bar convergence plot
figure(1)
sgtitle("Convergence of edges to desired edge over time",...
    'interpreter','LaTex')
subplot(2,1,1)
hold on
    plot(sim_t,sim_z(:,1)-z_d(1),'r','LineWidth',1)     % z1x data
    plot(sim_t,sim_z(:,2)-z_d(2),'b','LineWidth',1)     % z1y data
    yline(0,'k--','$$\bar{z}$$','interpreter','LaTex')  % ref line
hold off
xlim([0,20]);ylim([-3,4]);
grid on
ylabel("Displacement, $$\bar{z}$$, [m]",'interpreter','LaTex')
xlabel("Time, t, [s]",'interpreter','LaTex')
legend("z_1_x","z_1_y")
title("Displacement of edge 1",'FontSize',12,'interpreter','LaTex')
subplot(2,1,2)
hold on; grid on
    plot(sim_t,sim_z(:,3)-z_d(3),'r','LineWidth',1)     % z2x data
    plot(sim_t,sim_z(:,4)-z_d(4),'b','LineWidth',1)   % z2y data
    yline(0,'k--','$$\bar{z}$$','interpreter','LaTex')  % ref line
xlim([0,20]);ylim([-3,4]);
ylabel("Displacement, $$\bar{z}$$, [m]",'interpreter','LaTex')
xlabel("Time, t, [s]",'interpreter','LaTex')
title("Displacement of edge 2",'FontSize',12,'interpreter','LaTex')
legend("z_2_x","z_2_y")

% Agent trajectory plot
q_a = length(sim_q_a);  % Determining last index of position vector
figure(2)
sgtitle("Agent trajectories when control is applied",...
    'interpreter','LaTex')
hold on; grid on
plot(sim_q_a(:,1),sim_q_a(:,2),'r--','LineWidth',1) % agent 1 data
plot(sim_q_a(:,3),sim_q_a(:,4),'b--','LineWidth',1) % agent 2 data
plot(sim_q_a(:,5),sim_q_a(:,6),'g--','LineWidth',1) % agent 3 data
%   Marking begin & end points
plot(sim_q_a(q_a,1),sim_q_a(q_a,2),...
    '^','color','k','MarkerFaceColor','k')
plot(sim_q_a(1,1),sim_q_a(1,2),...
    'o','color','k','MarkerSize',7)
plot(sim_q_a(q_a,3),sim_q_a(q_a,4),...
    '^','color','k','MarkerFaceColor','k')
plot(sim_q_a(q_a,5),sim_q_a(q_a,6),...
    '^','color','k','MarkerFaceColor','k')
plot(sim_q_a(1,3),sim_q_a(1,4),...
    'o','color','k','MarkerSize',7)
plot(sim_q_a(1,5),sim_q_a(1,6),...
    'o','color','k','MarkerSize',7)
xlim([0,16]);ylim([0,20]);
xlabel("x-axis, [m]",'interpreter','LaTex');
ylabel("y-axis, [m]",'interpreter','LaTex');
%   Making tirangle shapes in plot
plot([sim_q_a(1,1) sim_q_a(1,3)],[sim_q_a(1,2) sim_q_a(1,4)],...
    'k:','LineWidth',0.8)
plot([sim_q_a(1,1),sim_q_a(1,5)],[sim_q_a(1,2),sim_q_a(1,6)],...
    'k:','LineWidth',0.8)
plot([sim_q_a(1,3),sim_q_a(1,5)],[sim_q_a(1,4),sim_q_a(1,6)],...
    'k:','LineWidth',0.8)
plot([sim_q_a(q_a,1) sim_q_a(q_a,3)],[sim_q_a(q_a,2) sim_q_a(q_a,4)],...
    'k:','LineWidth',0.8)
plot([sim_q_a(q_a,1),sim_q_a(q_a,5)],[sim_q_a(q_a,2),sim_q_a(q_a,6)],...
    'k:','LineWidth',0.8)
plot([sim_q_a(q_a,3),sim_q_a(q_a,5)],[sim_q_a(q_a,4),sim_q_a(q_a,6)],...
    'k:','LineWidth',0.8)
legend("Trajectory agent 1","Trajectory agent 2",...
    "Trajectory agent 3","Final positions","Initial positions")

% Agent velocity plot
figure(3)
grid on
hold on
plot(sim_t,sim_v(:,1),'r','LineWidth',1)
plot(sim_t,sim_v(:,3),'b','LineWidth',1)
plot(sim_t,sim_v(:,5),'g','LineWidth',1)
plot(sim_t,sim_v(:,2),'r--','LineWidth',1)
plot(sim_t,sim_v(:,4),'b--','LineWidth',1)
plot(sim_t,sim_v(:,6),'g--','LineWidth',1)
hold off
legend("Velocity agent 1 x-direction","Velocity agent 2 x-direction",...
    "Velocity agent 3 x-direction","Velocity agent 1 y-direction",...
    "Velocity agent 2 y-direction","Velocity agent 3 y-direction")
ylabel('Velocity, $$\dot{q}$$, [m/s]','Interpreter','latex')
xlabel("Time, t, [s]",'interpreter','LaTex')
sgtitle("Velocity of agents (x and y component) over time",...
    'interpreter','LaTex')

% Edge displacement plot
figure(4)
sgtitle("Displacement of edges to desired edge over time", ...
    'interpreter','LaTex')
subplot(2,1,1)
hold on
plot(sim_t,sim_z(:,1),'r','LineWidth',1)
plot(sim_t,sim_z(:,2),'b','LineWidth',1)
hold off
yline([0.5,-2],'k--')
xlim([0,20]);ylim([-3,4]);
grid on
ylabel("Displacement, z, [m]",'interpreter','LaTex')
xlabel("Time, t, [s]",'interpreter','LaTex')
legend("z_1_x","z_1_y")
title("Displacement of edge 1",'FontSize',12)
subplot(2,1,2)
hold on; grid on
plot(sim_t,sim_z(:,3),'r','LineWidth',1)
plot(sim_t,sim_z(:,4),'b','LineWidth',1)
```

```
yline([0,-1],'k--')
xlim([0,20]);ylim([-3,4]);
ylabel("Displacement, z, [m]",'interpreter','LaTex')
xlabel("Time, t, [s]",'interpreter','LaTex')
title("Displacement of edge 2",'FontSize',12)
legend("z_2_x","z_2_y")
```

Appendix B. Distance-based Formation Control

B.1. Matlab Script

```
% Script for distance-based formation control
% Ba IP project
% J.M. Oeben
% s4101839


clear
clc
close all

N = 3; %number of agents

% Define constants
B = [-1 0 1; 1 -1 0; 0 1 -1];
E = 3;
kronB = kron(B,eye(2));
M_a = 1*ones(2*N,1);      % agent mass
D_r = 1.5.*eye(2*N);      % linear damping

% Agents
q_10=[0 1]; q_20=[2 2]; q_30=[3 1];
q_0 = [q_10'; q_20'; q_30']; % initial position agents
p_0 = 0.5.*ones(2*N,1);      % initial momentum agents
v_0 = p_0./ M_a;             % initial velocity

% Desired distances
d_r = [1.5; 1.5; 1.5];    % virtual nominal spring length position

% Setting up loop simulation
t_step_sim = 0.01;  % Timestep simulation
end_time = 20;      % End time simulation
a = zeros(2*N,1);   % Initialising a
v = 0.5.*ones(2*N,1);   % Initialising v
q = q_0;            % Initialising q

% Calculating q and v values
for i = 1 : end_time / t_step_sim
    % physical state update
    v = v + a * t_step_sim;
    q = q + v * t_step_sim;
    z = kronB'*q;    % Calculating distances between agents

    dist = [sqrt((q(3)-q(1))^2+(q(4)-q(2))^2);
            sqrt((q(5)-q(3))^2+(q(6)-q(4))^2);
            sqrt((q(5)-q(1))^2+(q(6)-q(2))^2)];
    e_z = [(q(3)-q(1))^2+(q(4)-q(2))^2;
           (q(5)-q(3))^2+(q(6)-q(4))^2;
           (q(5)-q(1))^2+(q(6)-q(2))^2];
    % Error of distances between agents as defined in report
    e_d = e_z - (d_r).^2;
    a1 = -kronB*(blkdiag(z(1:2,:)',z(3:4,:)',z(5:6,:)')')*e_d;
    a2 = -D_r*v;
    a = a1+a2;

    % Storing data in plot format
```

```matlab
    dis_plot(i,1:N)=dist;
    error_plot(i,1:N)=e_d;

    % Saving positions in seperate vectors
    ag1x(i)=q(1);
    ag1y(i)=q(2);
    ag2x(i)=q(3);
    ag2y(i)=q(4);
    ag3x(i)=q(5);
    ag3y(i)=q(6);

    % Saving velocities in seperate vectors
    v1x(i)=v(1);
    v1y(i)=v(2);
    v2x(i)=v(3);
    v2y(i)=v(4);
    v3x(i)=v(5);
    v3y(i)=v(6);
end

% Define again for plots
sim_t = 0.01:0.01:20;

% Agent trajectory plot
figure(1)
sgtitle("Agent trajectories when control is applied",'interpreter','LaTex')
hold on
grid on
plot(ag1x,ag1y,'r--','LineWidth',1)
plot(ag2x,ag2y,'b--','LineWidth',1)
plot(ag3x,ag3y,'g--','LineWidth',1)
plot(ag1x(1),ag1y(1),...
    'o','color','k','MarkerSize',7)
plot(ag1x(end),ag1y(end),...
    '^','color','k','MarkerFaceColor','k')
plot(ag2x(1),ag2y(1),...
    'o','color','k','MarkerSize',7)
plot(ag2x(end),ag2y(end),...
    '^','color','k','MarkerFaceColor','k')
plot(ag3x(1),ag3y(1),...
    'o','color','k','MarkerSize',7)
plot(ag3x(end),ag3y(end),...
    '^','color','k','MarkerFaceColor','k')
xlim([0,3.5]);ylim([0,3.5])
xlabel("x-axis, [m]",'interpreter','LaTex');ylabel("y-axis,
[m]",'interpreter','LaTex');
%   Making tirangle shapes in plot
plot([ag1x(1) ag2x(1)],[ag1y(1) ag2y(1)],...
    'k:','LineWidth',0.8)
plot([ag1x(1) ag3x(1)],[ag1y(1) ag3y(1)],...
    'k:','LineWidth',0.8)
plot([ag2x(1) ag3x(1)],[ag2y(1) ag3y(1)],...
    'k:','LineWidth',0.8)
plot([ag1x(end) ag2x(end)],[ag1y(end) ag2y(end)],...
    'k-.','LineWidth',0.8)
plot([ag1x(end) ag3x(end)],[ag1y(end) ag3y(end)],...
    'k-.','LineWidth',0.8)
plot([ag2x(end) ag3x(end)],[ag2y(end) ag3y(end)],...
    'k-.','LineWidth',0.8)
```

```
legend("Trajectory agent 1","Trajectory agent 2",...
    "Trajectory agent 3","Initial position","Final position")

% Distance plot
figure(2)
hold on
sgtitle("Distances between agents",'interpreter','LaTex')
plot(sim_t,dis_plot(:,1),'r','LineWidth',1)
plot(sim_t,dis_plot(:,2),'b','LineWidth',1)
plot(sim_t,dis_plot(:,3),'g','LineWidth',1)
grid on
legend("Distance between agents 1 and 2","Distance between " + ...
    "agents 2 and 3","Distance between agents 1 and 3")
ylabel("Distance, d, [m]",'interpreter','LaTex')
xlabel("Time, t, [s]",'interpreter','LaTex')
xlim([0,12]);ylim([0,3.5])

% Distance error plot
figure(3)
hold on
grid on
sgtitle("Distance error $$e^{d}$$ between agents",'interpreter','LaTex')
plot(sim_t,error_plot(:,1),'r','LineWidth',1)
plot(sim_t,error_plot(:,2),'b','LineWidth',1)
plot(sim_t,error_plot(:,3),'g','LineWidth',1)
legend("Error distance agents 1&2","Error distance agents "+...
    "2&3","Error distance agents 1&3")
ylabel("Distance error, $$e^{d}$$, [m]",'interpreter','LaTex')
xlabel("Time, t, [s]",'interpreter','LaTex')
xlim([0,12]);ylim([-3,7])

% Velocity plot
figure(4)
hold on
grid on
sgtitle("Velocity of agents over time",'interpreter','LaTex')
plot(sim_t,v1x,'r','LineWidth',1)
plot(sim_t,v2x,'b','LineWidth',1)
plot(sim_t,v3x,'g','LineWidth',1)
plot(sim_t,v1y,'r--','LineWidth',1)
plot(sim_t,v2y,'b--','LineWidth',1)
plot(sim_t,v3y,'g--','LineWidth',1)
xlim([0,12]);
ylabel('Velocity, $$\dot{q}$$, [m/s]','Interpreter','latex')
xlabel("Time, t, [s]",'interpreter','LaTex')
yline(0.5,'k--','v_0','LineWidth',0.8)
legend("Velocity agent 1 x-direction","Velocity agent 2 x-direction",...
    "Velocity agent 3 x-direction","Velocity agent 1 y-direction",...
    "Velocity agent 2 y-direction","Velocity agent 3 y-direction")
```

Bibliography

[1] Kang S.-M *et al.* (2014) "Distance-Based Formation Control with a Single Moving Leader," *Proceedings of the American Control Conference*, 305-310, pp. 305–310. doi: 10.1109/ACC.2014.6858587.

[2] Vos, E. (2015) "Formation Control in the Port-Hamiltonian Framework," Nieuw Archief voor Wiskunde, 16(2), pp. 127–129

[3] Anderson, B.D.O., Fidan, B., Yu, C., Walle, D. (2008). UAV Formation Control: Theory and Application. In: Blondel, V.D., Boyd, S.P., Kimura, H. (eds) Recent Advances in Learning and Control. Lecture Notes in Control and Information Sciences, vol 371. Springer, London. https://doi.org/10.1007/978-1-84800-155-8_2

[4] Javanmardi, N., Yaghmaei, A. and Yazdanpanah, M. J. (2020) "Spacecraft Formation Flying in the Port-Hamiltonian Framework," *Nonlinear Dynamics : An International Journal of Nonlinear Dynamics and Chaos in Engineering Systems*, 99(4), pp. 2765–2783. doi: 10.1007/s11071-019-05445-0.

[5] Barogh, S. A. and Werner, H. (2017) "Cooperative Source Seeking with Distance-Based Formation Control and Single-Integrator Agents," *IFAC PapersOnLine*, 50(1), pp. 7911–7916. doi: 10.1016/j.ifacol.2017.08.759.

[6] Ahn, H.-S. (2019) *Formation control : approaches for distributed agents*. Cham: Springer (Studies in systems, decision and control, volume 205). (Accessed: April 13, 2022).

[7] Oh, K.-K. and Ahn, H.-S. (2014) "Distance-Based Undirected Formations of Single-Integrator and Double-Integrator Modeled Agents in N-Dimensional Space," *International Journal of Robust and Nonlinear Control*, 24(12), pp. 1809–1820. doi: 10.1002/rnc.2967.

[8] Park M.-C *et al.* (2018) "Distance-Based Control of K_n Formations in General Space with Almost Global Convergence," *IEEE Transactions on Automatic Control*, 63(8), pp. 2678–2685. doi: 10.1109/TAC.2017.2776524.

[9] Zou, Y., Wen, C. and Guan, M. (2020) "Adaptive Estimator-Based Formation Maneuvering Control of Nonholonomic Mobile Robots," International Journal of Adaptive Control and Signal Processing, 34(2), pp. 199–209. doi: 10.1002/acs.3078.

[10] González-Sierra, J., Dzul, A. and Martínez, E. (2022) "Formation Control of Distance and Orientation Based-Model of an Omnidirectional Robot and a Quadrotor Uav," Robotics and Autonomous Systems, 147. doi: 10.1016/j.robot.2021.103921.

[11] Zhao, Y. *et al.* (2021) "Formation of Multi-Agent Systems with Desired Orientation: A Distance-Based Control Approach," *Nonlinear Dynamics : An International Journal of Nonlinear Dynamics and Chaos in Engineering Systems*, 106(4), pp. 3351–3361. doi: 10.1007/s11071-021-06948-5.

[12] Xiao, Z., Fan, X. (2021) "Bearing-Based Formation Control of Multi-Agents with Dual-Leaders Structure" *33rd Chinese Control and Decision Conference (ccdc)*: *IEEE*, pp. 5589–5595. doi: 10.1109/CCDC52312.2021.9602036.

[13] A. van der Schaft and D. Jeltsema. (2014) Port-Hamiltonian Systems Theory: An Introductory Overview. *Foundations and Trends R in Systems and Control*, vol. 1, no. 2-3, pp. 173–378

[14] Zhang M et al. (2018) "Pid Passivity-Based Control of Port-Hamiltonian Systems," *IEEE Transactions on Automatic Control*, 63(4), pp. 1032–1044. doi: 10.1109/TAC.2017.2732283.

[15] van der Schaft, A. (2020) "Port-Hamiltonian Modeling for Control," *Annual Review of Control, Robotics, and Autonomous Systems*, 3(1), pp. 393–416. doi: 10.1146/annurev-control-081219-092250.

[16] Vos, E., van der Schaft, A. J. and Scherpen, J. M. A. (2016) "Formation Control and Velocity Tracking for a Group of Nonholonomic Wheeled Robots," *IEEE Transactions on Automatic Control*, 61(9). doi: 10.1109/TAC.2015.2504547.

[17] Zheng, Y. *et al.* (2019) "Application of Floquet Theory to Formation Flying in Elliptic Orbits Via a Hamiltonian Structure-Preserving Control," *Aerospace Science and Technology*, 95. doi: 10.1016/j.ast.2019.105508.

[18] Yang, T., Yu, S. and Yan, Y. (2019) "Formation Control of Multiple Underwater Vehicles Subject to Communication Faults and Uncertainties," *Applied Ocean Research*, 82, pp. 109–116. doi: 10.1016/j.apor.2018.10.024.

[19] Wu Z, Sun J and Wang X (2018) "Distance-Based Formation Tracking Control of Multi-Agent Systems with Double-Integrator Dynamics," *Chinese Physics B*, 27(6). doi: 10.1088/1674-1056/27/6/060202

[20] Mehdifar, F. *et al.* (2020) "Prescribed Performance Distance-Based Formation Control of Multi-Agent Systems," *Automatica*, 119. doi: 10.1016/j.automatica.2020.109086.

[21] Oh, K.-K., Park, M.-C. and Ahn, H.-S. (2015) "A Survey of Multi-Agent Formation Control," *Automatica*, 53, pp. 424–440. doi: 10.1016/j.automatica.2014.10.022.

[22] Arroyo Rodriguez, V. *et al.* (2020) "A Data-Driven Method Based on Quadratic Programming for Distance-Based Formation Control of Euler-Lagrange Systems," *IEEE control systems letters*, 5(1), pp. 313–318.

[23] Khalil, H. K. (2000) *Nonlinear systems*. 3Rd ed., International edn. Upper Saddle River, N.J.: Pearson Education.

[24] Li, N. *et al.* (2022) "A passivity approach in port-Hamiltonian form for formation control and velocity tracking", *ECC 2022*

[25] Sun, Z. (2018) Cooperative coordination and formation control for multi-agent systems. Cham, Switzerland: Springer (Springer theses). doi: 10.1007/978-3-319-74265-6