FLOCKING OF MECANUM WHEELED ROBOT CONSENSUS

by

Xianfeng Lu

A Thesis

Submitted to the Faculty of Purdue University

In Partial Fulfillment of the Requirements for the degree of

Master of Science in Mechanical Engineering

School of Mechanical Engineering

West Lafayette, Indiana

December 2019

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF COMMITTEE APPROVAL

Dr. Jianghai Hu, Co-chair

School of Electrical and Computer Engineering

Dr. George T. C. Chiu, Co-chair

School of Mechanical Engineering

Dr. Peter H. Meckl

School of Mechanical Engineering

Approved by:

Dr. Nicole L. Key

To my family, friends, and whoever cheered me in this countryside life.

ACKNOWLEDGMENTS

I have received much precious support while working on this thesis. First of all, I would like to express my sincere gratitude to my major professor, Prof. Jianghai Hu. He guided me throughout this research and patiently answered many questions from me. Without the help and encouragement from Prof. Hu, the ship of my research might become Titanic at any minute. I am also grateful to Prof. Chiu for agreeing to be my co-chair and Prof. Meckl for serving on my advisory committee.

Finally, I can never forget to thank my parents for their unconditional love, support, and encouragement. Of course, there are many other friends, colleagues who have supported me. I would like to thank you all!

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

This thesis applies flocking algorithms for the distributed consensus control of a multi-agent system composed of four-Mecanum-wheeled robots. The working mechanism of flocking is an artificial potential field consisting of attractive/repulsive forces and velocity alignment. The potential function of the attractive and repulsive force is introduced to control the connected distance among agents in the network. A consensus is a group of robots in a communication network to achieve common goals, which are the agreement of position and heading angle in this thesis. The main contribution of this thesis is our proposed feasible methods to achieve consensus control for general multi-agent systems of four-Mecanum-wheeled robots.

With the fast development of information technology and the growing demand for data exchange around the world, the sensors and actuators of agents become more complicated and require more resources. Local communication among agents reduces the need for high material costs and lengthy installation time. This thesis established a controllable model of four-Mecanum-wheeled robots in a local communication network. An assumption is that all robots can obtain information on the relative position and heading angle difference between themselves and their neighbors. A few robots with installed GPS are directly connected to the central host. Our flocking methods under the assumed communication conditions adjust the velocities of robots by controlling the speed of Mecanum wheels.

This thesis simulated the proposed leader-following flocking algorithms for cases of connected formation and snake formation with different numbers of leaders. The simulation results regarding the position consensus and heading angles consensus are provided to illustrate the robustness of the proposed algorithms.

CHAPTER I: INTRODUCTION

1.1 Background and Problem Statement

With the fast development and broad application of cooperative automatic robots, the information exchanging process, including giving instructions and receiving feedbacks, gradually becomes complicated and time-consuming. In comparison to central host controlling subordinate agents, the benefits with less operational energy and higher functional efficiency can be achieved from the implementation of local information communication of the multi-agent network.

To increase the efficiency of signal communication among robots and reduce the computational workload of the central host, a group of omnidirectional mobile robots equipped with Mecanum wheels is an example of the multi-agent network application. The Mecanum wheels as a kind of omnidirectional wheels are widely used in terms of their superior performance on dexterity and driving ability compared with other wheels (Cox, 1990). The omnidirectional mobile robots are often designed to move and carry objects in industry. The robots are usually wanted to converge to their optimal positions with same heading angle, which in turn is position consensus and heading angle consensus.

The flocking algorithm is based on the study of the flocking phenomenon happening in the natural world. Reynolds (1987) indicated three heuristic rules for natural biological flocks/swarms: 1) Separation: avoid collision with nearby groupmates; 2) Alignment: match the velocity of group mates in moderate distances; 3) Cohesion: stay close to remote group mates. In a multi-agent system composed of many local interacting agents, the flocking algorithm allows agents to control their behaviors based on neighbors' reactions adaptively.

Most of the current researches concentrated on developing mathematical flocking algorithms for consensus of a multi-agent system. This thesis studied flocking algorithms in a kinematic model under physical constraints.

1.2 Study Objective and Research Setting

There are many different purposes of consensus by flocking algorithms, including to head to the same velocities, to arrive at the desired same area, or to generate a typical movement pattern. Under the assumption that the initial interactive network of the multi-agents is not all connected, the flocking algorithms in this thesis help a virtual leader or multiple virtual leaders to collect follower agents and create a connected graph to keep a common movement pattern.

The hypothetical scenario in this thesis has been made that a set of Mecanum wheeled robots are randomly located on the ground. These robots start with different initial velocities in a designated area that has a specific size. Each robot has an individual microcontroller connected to a local network and a compass measurement device installed for information exchange and data computation. The fundamental goal of this research is to develop an optimized flocking algorithm to help this set of Mecanum wheeled robots meet position and heading angle consensuses in a finite timing period.

An incomplete and connected graph system composed of n-homogeneous agents represents the set of robots in free space. A beacon agent is capable of acquiring global information with more advanced technology, such as Global Positioning System (GPS) and Global Cellular Network. The remaining robots with cheaper devices are defined as blind agents. Network sensing range of blind agents is limited in a smaller region, and blind agents have null measurements of local relative positional data. By using angle of arrival (AoA) measurement, the position coordinate of a blind agent can be computed by the data collection from beacon agents. Under the assumption of

individual compass measurement device installation on each agent, the heading angle is measured locally on its own. Its microcontroller computes the consensus of heading angle for each agent with the heading angle data collected from neighboring agents in local network sensing range.

The consensus was analyzed under a time-varying fashion with a leader or a few leaders. The leader has a predetermined trajectory guiding the rest of the agents without knowing the full knowledge of the travel plan to reach the desired location. In advance, leader-follower consensuses without the formation and with snake formation were studied in this thesis to observe the behavior. Formation control of multiple mobile robots using a consensus scheme in terms of sensing capability is subdivided as position, displacement, and distance-based control method as shown in (Gulzar, Hussain, Javed, Munir, & Asif, 2018). Based on the limited sensing range of the blind agents, the flocking algorithm utilizes the displacement-based control method.

1.3 Literature Review

The concept of the multi-agent system (MAS) was widely known in the late twentieth century and inspired by the collective characteristics among human nature and the observation of animal behaviors. Schelling in 1971 presented a multi-agents model of human beings who are intended to facilitate the formation of consensus in a shared social network (Schelling, 1971), and Reynolds in 1987 summarized three rules of bird flocking phenomenon based on a group of biology agents (Reynolds, 1987). Recently, the flocking algorithm has been studied mainly with interest in consensus problems starting with the work of Olfati-Saber and Murray (2007).

In this section, the related work will be reviewed. The consensus control of the multi-agent system is overviewed and followed by the flocking algorithm theories and implementation.

1.3.1 Consensus Control

A consensus algorithm (or protocol) is an interaction rule that specifies the information exchange between an agent and all its neighbors on the network. As a distributed solution to multi-agent coordination, consensus, or agreement problems have been studied extensively in the literature. Convergence to a common value is called the consensus or agreement problem in the literature (Ren, Beard, & M., 2005). The consensus control for a multi-agent system is referred to reach an agreement upon certain quantities of interest. The fundamental concept is to generate a general state for all the agents in the network under the influence of the neighborhood environment.

The existing dynamic control strategies for the consensus problem are mainly studied on simple, stable dynamic systems with either single-integrator (Yong, Guangming, & Huiyang, 2012) or double-integrator under preset time (Ren & Beard, 2008). Under different systems dynamic, consensus control approaches have been massively developed. A study led by Ferrari-Trecate (2009) proposed an MPC solution for consensus problem under a single-integrator dynamic with bounded inputs and a double-integrator dynamic. A control strategy, which does not require continuous monitoring of the neighbor' states, for multi-agent coordination with event-based broadcasting was analyzed under three scenarios: Networks of single-integrator agents with and without communication delays, and networks of double-integrator agents (Seyboth & Johansson, 2013). Most of the present researches for different dynamic models do not account for physical input constraints, which in many cases have to be included in the problem formulation due to actuator and sensor limitations.

Control architectures of a multi-agent system in a shared network consists of many sensors and actuator nodes. Due to beneficial characters for multi-agent coordination with limited resources, the consensus control approaches are widely studied with multi-vehicle, aircraft, and other autonomous agents. For example, the formation method of unmanned aerial vehicles (UAVs)

based on behavior was considered to reduce the requirements on wireless data update rate and enhance the ability of obstacle avoidance of UAVs (Cai, Sun, & Wu, 2012). A group of spacecraft utilized formation control laws to maintain attitude alignment in either deep space or earth orbit. (Lawton & Beard, 2002) A distributed strategy for mobile autonomous agents was developed to solve the rendezvous problem, which is concerned with the collective behavior of a group of $n > 1$ mobile autonomous agents, labelled 1 through n, which can all move in the plane. (Lin, Morse, & Anderson).

In cooperative control problems, the states of a group of agents shall be influenced by each other to converge to the final common goal. If this final universal value is not prescribed, the consensus algorithm is called leaderless consensus because there is no group reference trajectory known by the team. In the case when a virtual or physical agent is designated as a leader for the team to design the reference trajectory which can be totally or partially known by the members of the group, the consensus algorithm is called leader-following consensus (Djaidja & Wu, 2015). Leaderless consensus and leader-following consensus (Gu & Wang, 2009) have been extensively studied under a directed network topology (Meng, Ren, Cao, & You, 2010), networked Euler–Lagrange systems (Ren W. , 2009), switching network topology (Su & Huang, 2011), fixed directed communication network topology (Kim, Park, & Choi, 2014), etc.

1.3.2 Flocking

For a long time, biologists have been interested in a particular swarm behavior with living beings in the wild (Okubo, 1986) (Warburton, 1991) (Breder Jr, 1954). This swarming behavior is first brought up as flocking in 1987 by Reynolds with three heuristic rules of bird flocking (Reynolds, 1987). "Flocking" is the collective motion of many self-propelled entities and is a collective animal behavior exhibited by many living beings such as birds, fish, bacteria, and insects (O'Loan & Evans, 1999).

Based on the animal behaviors in nature, the control law is discovered as a local artificial potential field of attractive/repulsive and alignment forces based on the connection distance (Grünbaum & Okubo, 1994). For the multi-agent system, flocking with repulsive reaction prevents the impacts to the obstacle existing in the environment or avoids collisions among autonomous agents by a decentralized navigation function (Dimarogonas & Kyriakopoulos, 2005). In the case of mobile agents with limited sensing capability in a consistent data exchange network, a decentralized motion controller aligns agent velocity vectors and regulates inter-agent distances to maintain existing network links (Zavlanos, Tanner, Jadbabaie, & Pappas, 2009). Alignment force is the primary factor for an aggregate motion along a prevailing heading direction, and in another term, it is velocity matching, which was also studied (Jadbabaie, Lin, & Morse, 2003). In 1995, Vicsek et al. proposed an evolutionary model based on local information feedback to mimic the emergence of self-organized motion in systems of particles with biologically motivated interactions. It is also a particular case in which all agents only follow an alignment rule with the same moving speed. (Vicsek, Czirók, Ben-Jacob, Cohen, & Shochet, 1995).

Inspired by Reynolds' computer animation model of bird flocks, many flocking algorithms have been proposed and facilitated based on flocking control laws. In the peer to peer network, the transferred information must be chosen wisely due to limited sensor availability. Existing flocking algorithms rely on information about both relative position and relative velocity among neighboring agents (Lawton & Beard, 2002) (Wen, Duan, Li, & Chen, 2012). Nonetheless, flocking algorithms were also researched with only relative position information (Lizarralde & Wen, 1996) (Wang, Wang, & Hu, 2013). In the work of Tanner et al., the measured position/velocity among neighboring agents for flocking was discussed under two different control topologies: a static topology of the control interconnections and a dynamic communication network. (Tanner, Jadbabaie, & Pappas). A static topology mostly referred to the leaderless control law, where the measured objects remain unchanged in the network. The dynamic communication

network has time-variant network nodes and is massively studied with a single or multiple virtual leader (Wang, Shi, & Chu, 2005) (Su H. X., 2008) and leader-follower flocking (Zhou, Wu, Yu, Small, & Lu, 2012) (Luo, Shaobao, & Guan, 2010).

The engineering applications of flocking include massively distributed sensations using mobile sensor networks in an environment, self-assembly of connected mobile networks, automated parallel delivery of payloads, and performing military missions such as reconnaissance, surveillance, and combat using cooperative unmanned aerial vehicles (UAVs) (Reza, 2006). For example, Crowther proposed a set of control laws to generate true flocking behavior in that the flight UAV density is increased, and the flock members converge on a common heading by applying the cohesion and alignment rules (Crowther, 2003). Controllers to an artificial potential based approach in the flocking control for multi-agent systems were also implemented on a team of nonholonomic mobile robots with a double integrator model and mass point model (Li & Jiang, 2008).

CHAPTER II: MECANUM WHEELED ROBOT

The purpose of using Mecanum wheels in this project is mainly because of its omnidirectional feature, which is an ability to move instantaneously in any direction, from any configuration. According to Bengt Erland Ilon, who invented this type of the wheel in 1972 (United States Patent No. US3876255A), to drive the vehicle, it is previously known to dispose individually drivable rolls or the like along the opposite sides of the vehicle. These rolls are driven to be rotated with the aid of the driving assembly of the vehicle, and in order to thereby obtain the required grip on the base, each roll has an exterior flange, spirally positioned around the roll in the longitudinal direction thereof as shown in Figure 1.

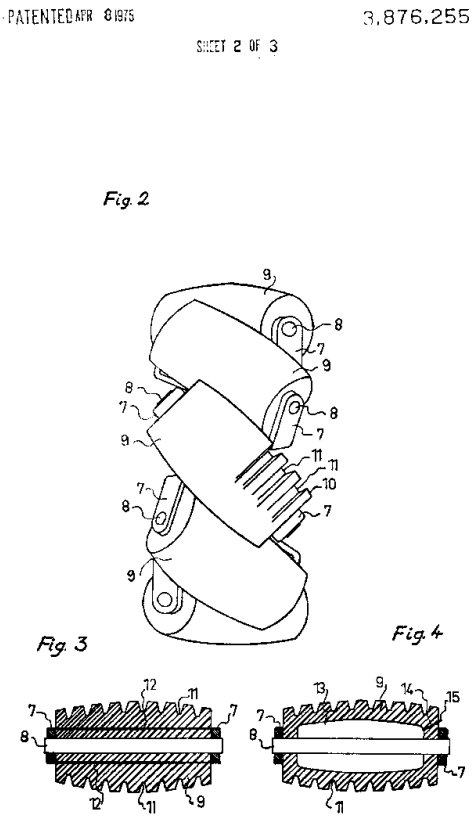

Figure 1: Mecanum Wheel Side Demonstration (United States Patent No. US3876255A, 1972)

For regular wheels, each turning of the robot needs a space room larger than itself, while the Mecanum wheels are more maneuverable and can achieve turning direction without any planar movements. The heading angle of a robot with regular wheels is the moving direction, which makes adjusting the heading angle accompany movement adjustment. Due to the omnidirectionality, the heading angle of a robot with Mecanum wheels can be controlled separately with the planar velocities. The difference among regular wheels and Mecanum wheels makes the heading angle of a robot with Mecanum wheels to achieve heading angle consensus much faster and more convenient than with regular wheels. Therefore, the Mecanum wheels were used in this project.

2.1 Kinematic Analysis

The model of a Mecanum wheeled robot is presented in Figure 2. Given a collective system of n-identical autonomous mobile robots whose own equation of motion is given in Equation (2.1), let (x, y, β) denote the 2D planar position location and heading angle of the robot. The velocity in Cartesian coordinate is described as $V_c = [\nu_x\ \nu_y\ \omega_z]^T \epsilon\ \mathbb{R}^3$ where ν_x and ν_y are the linear velocities along x-axis and y-axis, and ω_z is the angular velocity about the z-axis.

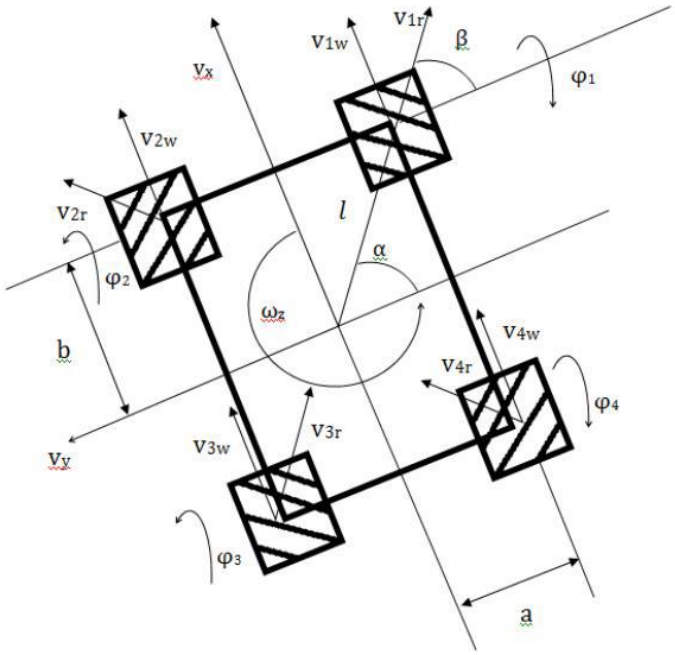

Figure 2: Configuration of the Omnidirectional platform. (A. F. M. Fuad, 2017)

The four-wheel angular velocities are described as $\Phi = [\varphi_1\ \varphi_2\ \varphi_3\ \varphi_4]^T \epsilon \mathbb{R}^4$, with rollers angled at $\delta = 45^\circ$ to the wheel axis, taken as clockwise looking outwards from the mounted frame, as shown in Figure 3.

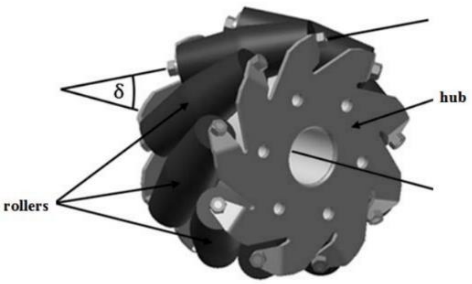

Figure 3: Mecanum Wheel

The resulting kinematic equation relating the angular velocities of wheels and the Cartesian velocities of robot platform (A. F. M. Fuad, 2017) is:

$$\begin{bmatrix}\varphi_1\\ \varphi_2\\ \varphi_3\\ \varphi_4\end{bmatrix} = -(\sqrt{2}/\mathrm{r})\begin{bmatrix}-\sqrt{2}/2 & \sqrt{2}/2 & lsin(\frac{\pi}{4}+\alpha)\\ -\sqrt{2}/2 & -\sqrt{2}/2 & lsin(\frac{\pi}{4}+\alpha)\\ \sqrt{2}/2 & -\sqrt{2}/2 & lsin(\frac{\pi}{4}+\alpha)\\ \sqrt{2}/2 & \sqrt{2}/2 & lsin(\frac{\pi}{4}+\alpha)\end{bmatrix}\begin{bmatrix}\cos\theta & sin\theta & 0\\ -sin\theta & \cos\theta & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}\nu_x\\ \nu_y\\ \omega_z\end{bmatrix} \quad (2.1)$$

where r is the radius of the wheel, l is the distance from the center of robot to the center of wheel coordinate, α is the angle between the geometric center of the platform and the wheel center, θ is the angle between the robot frame and inertial frame.

The Jacobian matrix for converting velocity coordinates is defined as:

$$J_0 = \begin{bmatrix}-\sqrt{2}/2 & \sqrt{2}/2 & lsin(\frac{\pi}{4}+\alpha)\\ -\sqrt{2}/2 & -\sqrt{2}/2 & lsin(\frac{\pi}{4}+\alpha)\\ \sqrt{2}/2 & -\sqrt{2}/2 & lsin(\frac{\pi}{4}+\alpha)\\ \sqrt{2}/2 & \sqrt{2}/2 & lsin(\frac{\pi}{4}+\alpha)\end{bmatrix}\begin{bmatrix}\cos\theta & sin\theta & 0\\ -sin\theta & \cos\theta & 0\\ 0 & 0 & 1\end{bmatrix} \quad (2.2)$$

Since the number of the robot's degrees of freedom (DoF) is larger than the number of the wheel speed inputs, the pseudo-inverse of J_0 is:

$$J^{+} = (J^T J)^{-1} J^T \quad (2.3)$$

Therefore, the final equation can be presented as:

$$\mathrm{V}_c = -\left(\frac{\mathrm{r}}{\sqrt{2}}\right) J^{+} * \Phi \quad (2.4)$$

2.2 Control System Representation

For the i^{th} robot in the multi-agent system, the linear position coordinates, and angular heading angle are given as a vector:

$$\mathrm{q}_i = \begin{bmatrix}\mathrm{x}_i\\ \mathrm{y}_i\\ \beta_i\end{bmatrix} \epsilon R^2 \quad (2.5)$$

For the motion control of a robot, a vector containing the linear velocity in cartesian coordinates and the angular velocity of its heading is as follows:

$$\dot{\mathrm{q}}_i = \mathrm{V}_c = \begin{bmatrix} \nu_x \\ \nu_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} \dot{\mathrm{x}}_i \\ \dot{\mathrm{y}}_i \\ \dot{\beta}_i \end{bmatrix} \tag{2.6}$$

The control input is the velocities of the four Mecanum wheels (φ):

$$u_i = \Phi_i = \begin{bmatrix} \varphi_1 \\ \varphi_2 \\ \varphi_3 \\ \varphi_4 \end{bmatrix} \tag{2.7}$$

The discrete-time domain is a more practical way to update the states of the i^{th} robot. With a sampling period of T, the state function is:

$$\mathrm{q}_i(k+1) = \mathrm{q}_i(k) + \dot{\mathrm{q}}_i(k) * T \tag{2.8}$$

Combining Equation (2.4), Equation (2.6) and Equation (2.7), the discrete-time system is transferred into a discrete state-space system with control input as:

$$\mathrm{q}_i(k+1) = A\mathrm{q}_i(k) + \mathrm{Bu}_i(k) \tag{2.9}$$

where A is an identity matrix, B is $-\left(\frac{\mathrm{r}}{\sqrt{2}}\right) J^+ * \mathrm{T}$ as calculated in section 2.1.

Depending on the distinctive characteristics of Mecanum wheels, the speeds of wheels manipulate the linear velocity and angular rotation of the robot platform. Therefore, each wheel needs an individual feedback controller to reach the desired speed with a designed control performance.

2.3. Mecanum Wheel Controller Design

Considering that each Mecanum wheel is attached with a DC motor, the problem becomes controlling the input voltage of the DC motor to minimize the frictional disturbance. The model

of an individual DC motor is shown below (Control Tutorials for MATLAB and Simulink - Motor Speed: System Modeling, 2019):

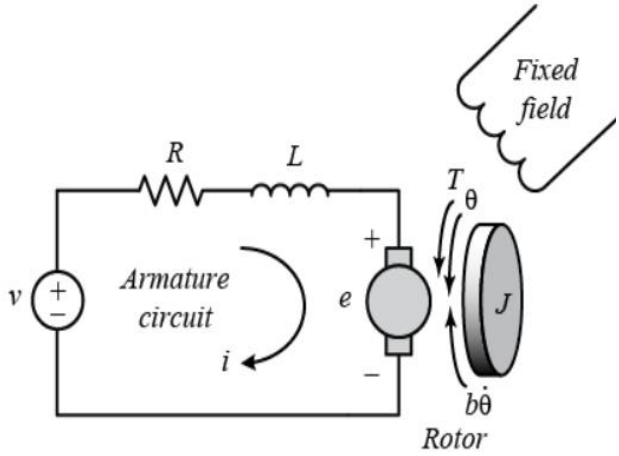

Figure 4: DC model circuit

v: circuit voltage

R: resistance

L: inductance

e: back emf

i: current

T: torque

b: viscous motor friction constant

J: moment of inertia of the rotor and wheel

θ: rotating angle

θ̇: angular speed

The governing equations are:

$$\mathrm{J}\ddot{\theta} + \mathrm{b}\theta = Ki \tag{2.10}$$

$$\mathrm{L}\frac{\mathrm{di}}{dt} + Ri = V - K\theta \tag{2.11}$$

The transfer function derived from the above equations are:

$$\mathrm{s}(\mathrm{Js} + \mathrm{b})\Theta(\mathrm{s}) = KI(s) \tag{2.12}$$

$$(\mathrm{Ls} + \mathrm{R})I(s) = V(s) - Ks\Theta(\mathrm{s}) \tag{2.13}$$

The state-space representation is:

$$\frac{\mathrm{d}}{dt}\begin{bmatrix}\dot{\theta}\\ i\end{bmatrix} = \begin{bmatrix}-\frac{\mathrm{b}}{J} & \frac{\mathrm{K}}{J}\\ -\frac{\mathrm{K}}{L} & -\frac{\mathrm{R}}{L}\end{bmatrix}\begin{bmatrix}\dot{\theta}\\ i\end{bmatrix} + \begin{bmatrix}0\\ \frac{1}{L}\end{bmatrix}V \tag{2.14}$$

$$y = \begin{bmatrix}1 & 0\end{bmatrix}\begin{bmatrix}\dot{\theta}\\ i\end{bmatrix} \tag{2.15}$$

As sampled at a frequency of 1000 Hz, a discrete feedback control system with the input of a reference voltage and the output of wheel velocity is demonstrated in Figure 5. The state x is

composed of the wheel velocity and the motor current. At each sampling time, the states of the plant will be updated based on the feedback of the difference between the output and the reference input.

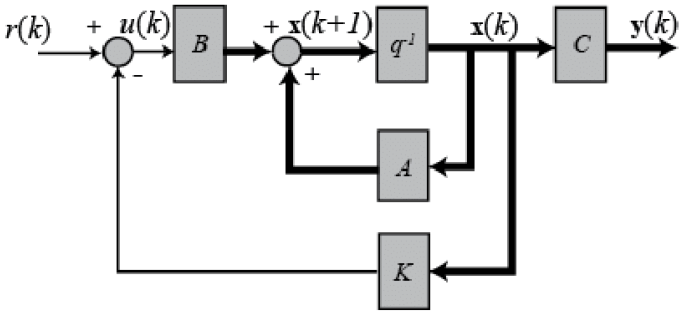

Figure 5: Discrete Feedback Control Diagram

$$\mathrm{x(k+1)} = Ax(k) + Br(k) \tag{2.16}$$

$$y(k) = Cx\,(k) \tag{2.17}$$

$$\mathrm{A} = \begin{bmatrix} -\frac{\mathrm{b}}{J} & \frac{\mathrm{K}}{J} \\ -\frac{\mathrm{K}}{L} & -\frac{\mathrm{R}}{L} \end{bmatrix} \tag{2.18}$$

$$B = \begin{bmatrix} 0 \\ \frac{1}{L} \end{bmatrix} \tag{2.19}$$

$$C = [1 \quad 0] \tag{2.20}$$

By utilizing the Simulink PID transfer function tuning tool in the feedback control loop block diagram shown in Figure 6, a discrete-time PID controller was designed with under 0.01 seconds settling time, which gives the fastest sampling frequency of 100 Hz for states update in flocking algorithm.

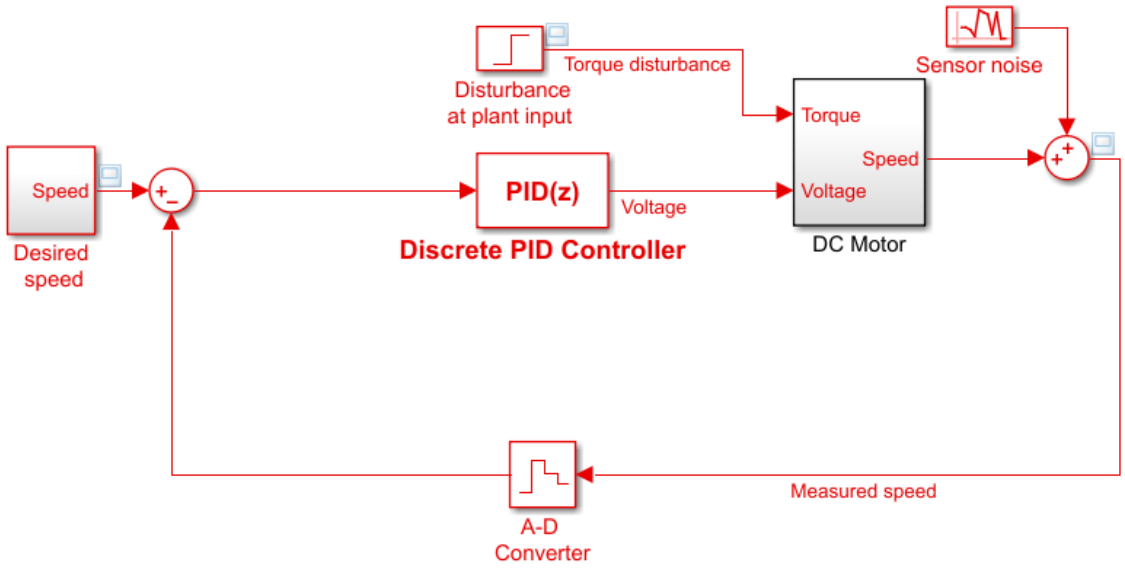

Figure 6: DC motor Simulink Display with Disturbance

The simulated response of a feedback control loop output with a disturbance input is shown below. From the resulting graph, the feedback controller is robust in eliminating the torque disturbance added into the DC motor. The settling time of the output response successfully meets the desired control performance.

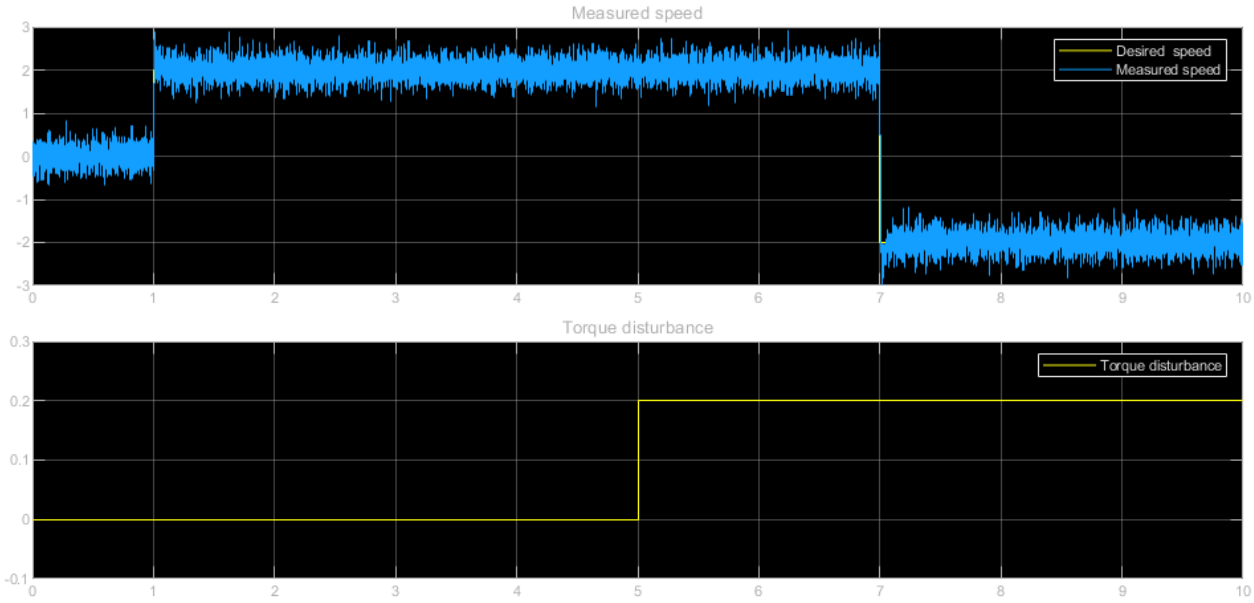

Figure 7: Wheel PID Controller Result with Disturbance Present

The algorithm shown below implements a method for the PID control of the individual wheel related to the platform velocity.

Algorithm 1 Wheel Speed PID Control

function WHEEL CONTROL(v_d, v_p)

▷ v_d = desired platform speed, v_p = current platform speed

$v_d \rightarrow W_d = \begin{bmatrix} w_{d,1} \\ w_{d,2} \\ w_{d,3} \\ w_{d,4} \end{bmatrix}$ ▷ W_d = four wheel desired speed

$v_p \rightarrow W_p = \begin{bmatrix} w_{p,1} \\ w_{p,2} \\ w_{p,3} \\ w_{p,4} \end{bmatrix}$ ▷ W_p = four wheel current speed

for *each wheel* **do**

PID control tuning(w_d, w_p)

end

end function

CHAPTER III. NETWORK LOCALIZATION

Based on graph theory, a network of agents equipped with omnidirectional range sensors can be viewed as a graph, with nodes corresponding to the agents and edges to the interaction (Mesbahi & Egerstedt, 2010). To reduce cost and increase feasibility, the network with only limited nodes knowing their locations is preferred while others measure the relative distance between nodes to determine location coordinates. The process of computing the locations of the nodes is called network localization (Aspnes, et al., 2006). Based on the assumption of initial random positions, the local agents without their position knowledge must be measured by beacon agents who have access to global information.

The beacon agents connect to the central host and are equipped with a global localization system to indicate the arbitrary locations. In a closed area, the beacon agents become lidars of other agents who do not have knowledge of arbitrary locations. Local agents without location sensibility are defined as blind agents in this thesis. The arbitrary location coordinate information is transferred from beacon agents to the blind agents so the blind agents can keep track of their location coordinates.

The angle of arrival measurements (AoA) is a method of network localization. A few beacon agents can provide a location estimation of a blind agent based on a combination of each angle measurement. AoA measurements require the time difference of arrival (TDoA) between individual elements of the antenna array. AoA estimation is a process that determines the direction of arrival of a received signal by processing the signal impinging on an antenna array. Orientation, defined as a fixed direction against which the AoAs are measured, is represented in degrees in a clockwise direction from the North (Rong & Mihail, 2006). In the most straightforward system, only two beacon agents are needed as long as they are not aligned with the measured agents. The

minimum system including AoA measurement contains two beacon agents and one blind agent. For the purpose of saving energy and increasing efficiency, the number of blind agents should be controlled in a reasonable range.

It is crucial that the beacon agents are not all aligned with blind agents, which will cause an incorrect position measurement since the location difference matrix rank will be reduced to be lower than the number of beacon agents. To decrease the chance of alignments among beacon agents and blind position agents, we choose the number of beacon agents according to the size of the agents' set.

As shown in Figure 8, $\vec{x} = \begin{bmatrix} x \\ y \end{bmatrix}$ is the position of blind agent, and $\vec{s}_k = \begin{bmatrix} x_k \\ y_k \end{bmatrix}$ is the position of the k^{th} beacon agent. Due to the measurement error existing in reality, we set the $\vec{\hat{\theta}}$ to be the measured angle of arrival from beacon agents, while $\vec{\theta}$ is the real angle of arrival. Similarly, there is a difference between the measured displacement, $\vec{\hat{r}}$, between a beacon agent and a blind agent and real displacement, $\vec{r}$.

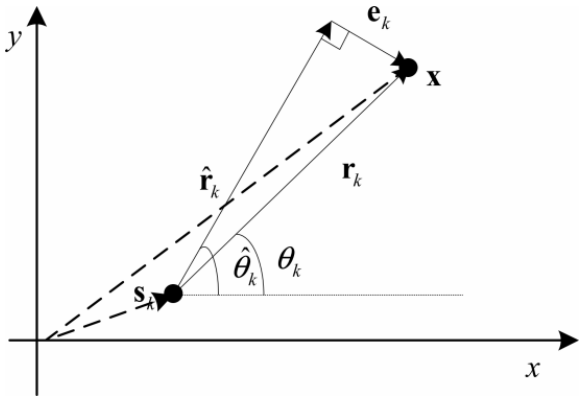

Figure 8: Relationship between the measured bearing vector and target position (Yan, Chen, Ottoy, Cox, & Strycker, 2018)

The measured direction angle from a blind agent to the k^{th} beacon agent is as

$$\hat{\theta}_k = \theta_k + \varepsilon_k \tag{3.1}$$

where ε_k is a uniformly distributed measuring angle error and θ_k is the actual direction angle given by:

$$\theta_k = \frac{y-y_k}{x-x_k} \tag{3.2}$$

The displacement then becomes:

$$\overrightarrow{r_k} = \|\overrightarrow{r_k}\| \begin{bmatrix} \cos(\hat{\theta}_k) \\ \sin(\hat{\theta}_k) \end{bmatrix} \tag{3.3}$$

The position of the blind agent can be derived as

$$\vec{x} = \overrightarrow{s_k} + \overrightarrow{r_k} \tag{3.4}$$

To minimize the measurement error, we used the least square approximation to approach the solution of the blind agent's location coordinate. Considering the least square estimator for the measurement error, we defined the orthogonal error vector by algebra as:

$$\overrightarrow{e_k} = \overrightarrow{r_k} - \overrightarrow{\hat{r}_k} \tag{3.5}$$

$$\overrightarrow{e_k} = \|\overrightarrow{r_k}\| \sin(\varepsilon_k) * \vec{a}_k \tag{3.6}$$

where $\overrightarrow{r_k}$ and $\overrightarrow{\hat{r}_k}$ is the actual and measured distance between the blind agent and k^{th} beacon agent, $\varepsilon_k = \hat{\theta}_k - \theta_k$ is assumed to be the Gaussian bearing noise, and $\vec{a}_k = \begin{bmatrix} \sin(\hat{\theta}_k) \\ -\cos(\hat{\theta}_k) \end{bmatrix}$ is the unit vector orthogonal to $\overrightarrow{\hat{r}_k} = \|\overrightarrow{\hat{r}_k}\| \begin{bmatrix} \cos(\hat{\theta}_k) \\ \sin(\hat{\theta}_k) \end{bmatrix}$.

Plugging the least square estimator to the position equation, we get:

$$\vec{x} = \overrightarrow{s_k} + \overrightarrow{\hat{r}_k} + \overrightarrow{e_k} \tag{3.7}$$

Since there are three unknowns, $\vec{x} = \begin{bmatrix} x \\ y \end{bmatrix}$ and $\left\|\overrightarrow{\hat{r}_k}\right\|$, besides $\overrightarrow{e_k}$ in the above equation, we need to lower the rank of the unknowns in the new position equation. By multiplying Equation (3.4) with $\vec{a}_k{}^T = \begin{bmatrix} \sin(\hat{\theta}_k) \\ -\cos(\hat{\theta}_k) \end{bmatrix}^T$, the $\overrightarrow{\hat{r}_k}$ is canceled. Now we get an equation as:

$$\overrightarrow{a_k}^T * \vec{x} = \overrightarrow{a_k}^T * \overrightarrow{s_k} + \overrightarrow{a_k}^T * \overrightarrow{e_k} \tag{3.8}$$

However, the vector multiplication reduced the rank of the equation by 1. To find the location coordinates for the blind agent, we need at least an additional AoA measurement from the beacon agent. The equation becomes:

$$\begin{pmatrix} \overrightarrow{a_1}^T \\ \vdots \\ \overrightarrow{a_k}^T \end{pmatrix} * \vec{x} = \begin{pmatrix} \overrightarrow{a_1}^T * \overrightarrow{s_1} \\ \vdots \\ \overrightarrow{a_k}^T * \overrightarrow{s_k} \end{pmatrix} + \begin{pmatrix} \overrightarrow{a_1}^T \\ \vdots \\ \overrightarrow{a_k}^T \end{pmatrix} * \begin{pmatrix} \overrightarrow{e_1} \\ \vdots \\ \overrightarrow{e_k} \end{pmatrix} \tag{3.9}$$

Assuming a linear relationship represents the equation:

$$\vec{A} * \vec{x} = \vec{b} + \vec{\epsilon} \tag{3.10}$$

where

$$\vec{A} = \begin{pmatrix} \overrightarrow{a_1}^T \\ \vdots \\ \overrightarrow{a_k}^T \end{pmatrix} \tag{3.11}$$

$$\vec{b} = \begin{pmatrix} \overrightarrow{a_1}^T * \overrightarrow{s_1} \\ \vdots \\ \overrightarrow{a_k}^T * \overrightarrow{s_k} \end{pmatrix} \tag{3.12}$$

$$\vec{\epsilon} = \begin{pmatrix} \overrightarrow{a_1}^T \\ \vdots \\ \overrightarrow{a_k}^T \end{pmatrix} * \begin{pmatrix} \overrightarrow{e_1} \\ \vdots \\ \overrightarrow{e_k} \end{pmatrix} \tag{3.13}$$

Because of measuring angle errors existence, there is often no exact solution for $\vec{x}$ among all measurements from two beacon agents. The solution of an unknown position becomes a null space unless all measurements are perfect. However, in real operations, it is not possible to neglect all

the measurement errors during position calculations. To minimize the noise from the measurements, we used the least squares approximations. The squared length for any $\vec{\epsilon}$ in the position equation is:

$$\|\vec{\epsilon}\|^2 = \left\|\vec{A} * \vec{x} - \vec{b}\right\|^2 \tag{3.14}$$

By minimizing the squared length of $\left\|\vec{A} * \vec{x} - \vec{b}\right\|^2$ with the smallest possible error, the estimated position coordinate of the blind agent can be found as:

$$\vec{A} * \vec{x}_{est} = \vec{b} \tag{3.15}$$

where:

$$\vec{A} = \begin{pmatrix} \sin(\hat{\theta}_1) & -\cos(\hat{\theta}_1) \\ \vdots & \vdots \\ \sin(\hat{\theta}_k) & -\cos(\hat{\theta}_k) \end{pmatrix} \tag{3.16}$$

$$\vec{b} = \begin{pmatrix} x_1 * \sin(\hat{\theta}_1) - y_1 * \cos(\hat{\theta}_1) \\ \vdots \\ x_k * \sin(\hat{\theta}_k) - y_k * \cos(\hat{\theta}_k) \end{pmatrix} \tag{3.17}$$

Algorithm 2 AoA Measurements

1: **function** AoA Calculation$(\theta_1, ..., \theta_k, x_1, ..., x_k, y_1, ..., y_k)$
▷ k = # of beacon agents

2: $A = \begin{bmatrix} sin(\theta_1) & -cos(\theta_1) \\ \vdots & \vdots \\ sin(\theta_k) & -cos(\theta_k) \end{bmatrix}$

3: $B = \begin{bmatrix} x_1 * sin(\theta_1) - y1 * cos(\theta_1) \\ \vdots \\ x_1 * sin(\theta_k) - y1 * cos(\theta_k) \end{bmatrix}$

4: $x = \begin{bmatrix} x_u \\ y_u \end{bmatrix} = B/A$ ▷ x = location of the unknown agent

5: **return** x

6: **end function**

Based on the measured angles from beacon agents to the blind position agent, the coordinate of a blind agent can be calculated and interpreted using the calculation algorithm shown above. The average location accuracy was evaluated for different network node density under the consideration of randomly placed beacon agents, as shown in Figure 9. All results are normalized with respect to the measured error range. The performance improves along with the number of beacons in the same swarm by the function of percentage, and higher network node density has better effects on measuring the location of unknown agents.

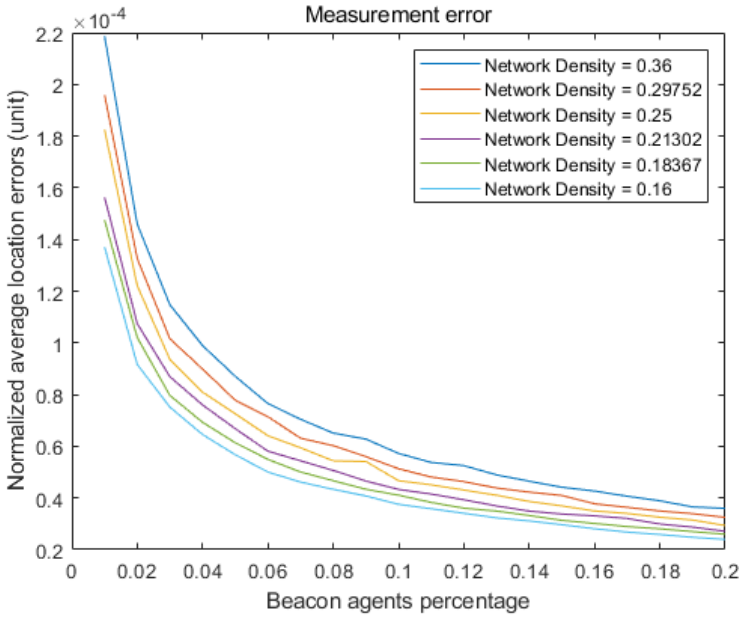

Figure 9: Average accuracy as a function of the # of beacon agents for different node density

CHAPTER IV. LOCAL COMMUNICATION / INFORMATION EXCHANGE

In the case of information exchange through local sensing, vehicles may be equipped with sensors that only have a limited field of view, which may result in unidirectional information exchange topologies (Ren & Atkins, Distributed multi-vehicle coordinated control via local information exchange, 2007). It is the same case in this research, assuming that each agent has an individual microcontroller on board. The position information of each regular agent is measured by the beacon agents, and is calculated in the central host computer, and then sent to the blind position agent itself. The position consensus is then calculated by each microcontroller after all locations of detectable neighbors are collected. The wheel motor commands for the next corresponding site are calculated onboard and directly given to itself.

The omnidirectional feature of the Mecanum wheeled robot allows the central platform to move separately with the rotation of the heading angle. Therefore, the heading angle direction cannot be determined simply by the path of an agent's movement. The heading angle consensus requires bilateral information exchange among neighborhood agents. For heading angle alignments of agents with neighbors, transmitting the information about heading angles is essential. Unfortunately, the heading angle cannot be detected by sensors installed on beacon agents due to the omnidirectional feature. Therefore, transmitting the information of heading angles is vital for heading angle alignments of agents to leaders. Assuming an individual compass measurement device for the heading angle is locally installed on each agent, each agent can achieve consensus via the information exchange of a narrow local network.

When the follower agents are out of the sensing range of the leader agent, a local consensus is generated based on the heading directions of neighborhood agents. Within the sensing range of the leader agent, follower agents should mimic the movement of a leader agent.

The system for providing targeted internet information to mobile agents allows that local agent includes a short-range transmitter to distribute information pointers to the portable information terminal and a mechanism for transferring data into the transmitter (Washington, DC: U.S. Patent and Trademark Office. Patent No. 6,219,696, 2001). Imaging the case of a leader carrying a transmitter that broadcasts the heading angle command, the neighboring agents will be able to receive the control command about heading angle through the local network when the distance requirement is satisfied.

Gossip Algorithms are designed for informational communication for mobile agents. As indicated by Mr. Liu and his co-workers (Liu, Guan, Li, Zhang, & Xiao, 2012), Broadcast Gossip Algorithms is to broadcast agent's own state to neighbors via quantized communication, while Randomized Gossip Algorithms distribute the computational burden and in which a node communicates with a randomly chosen neighbor (Boyd, Ghosh, Prabhakar, & Shah, 2006). In a network environment of nodes that are randomly distributed in a closed area, Gossip Algorithm builds connections between nodes to create a connected graph. In the branch of mathematics called graph theory, where a graph is a collection of nodes called vertices, and line segments between those vertices are called edges.

In this thesis, an incomplete and connected graph is established for a multi-agents' consensus communication network. An incomplete graph is not necessary to have an edge between every single pair of vertices in the graph. In a connected graph, every vertex in the graph can be connected to every other vertex in the graph through a series of edges, called a path. Figure 9 is

the initial state of agents with a free formation that is not a connected graph, while the leader is not connected to any agent.

Figure 10 shows the initial stage of free formation. Agents are grouped as clusters and are not connected to any of the leaders. The difference between the two final formations is the way of path arrangement. Figure 11 shows the connected formation graph with 36 vertices; each has a degree higher than or equal to 1. The vertex 0 represents the leader of flocking with connected formation, and the rest of the vertices has connected to the leader without any restriction of the edge. From Figure 12, the vertices have two degrees to be connected except the leader and the end agent of the snake. In the flocking algorithm with snake formation, the followers form a particular sequence to keep a snake movement formation. For example, the first agent, followed by the leader, becomes vertex 1, and the second agent connected to the leader is vertex 2. Each agent becomes a part of the network graph and links to the leader by adding itself to the end of the line in this graph until all agents are connected to the leader. Under the case of multiple leaders, the edges of the connected graph vary along with time. When two agents from separate snakes are within sensible ranges, the edges from two snake break out and reconnect to form a new snake with a selected leader.

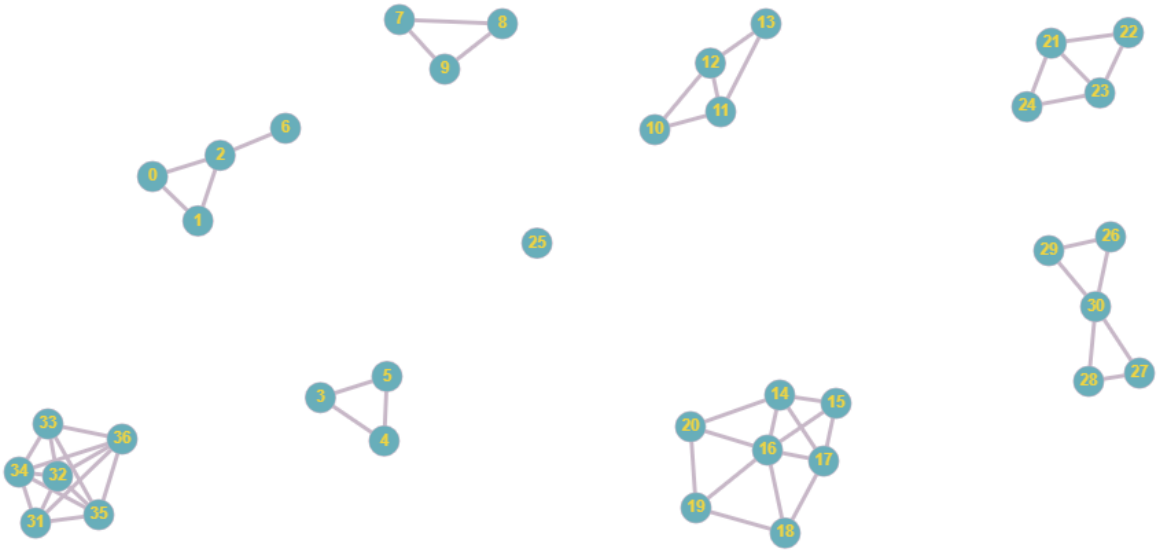

Figure 10: Layout of free formation

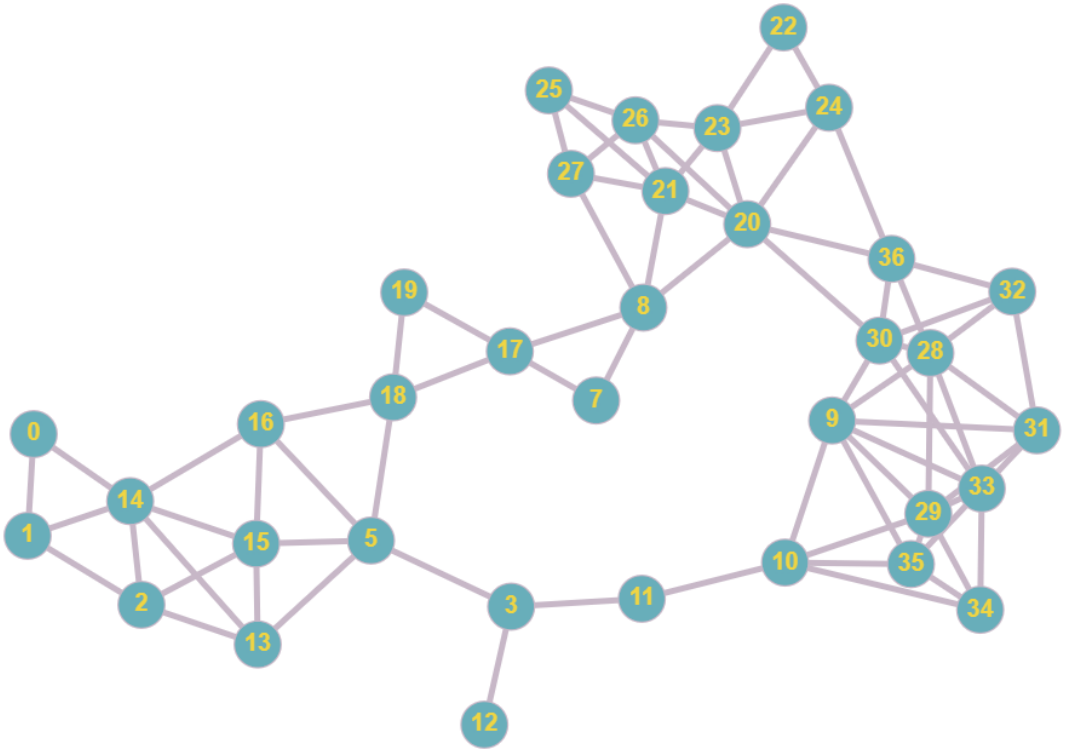

Figure 11: Layout of connected formation

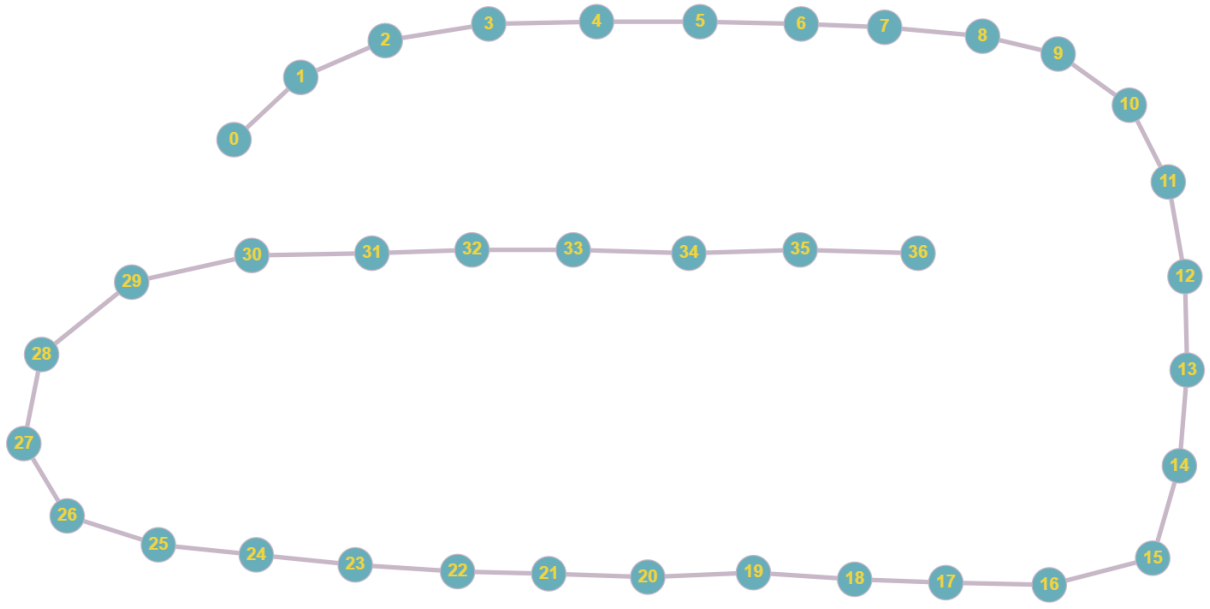

Figure 12: Layout of snake formation

CHAPTER V. FLOCKING ALGORITHM

According to the paper published in 1987 by Craig Reynolds (1987), the flocking phenomenon contains 1) separation, 2) alignment, 3) cohesion. The separation is defined as that two objects remain with absolute displacement to avoid collision between two agents when the velocity directions are directing to each other. The alignment requires that the velocities of all objects are pointing towards the same direction. The cohesion is trying to minimize the displacement among two objects.

In this research, the agents should achieve position consensus and heading angle consensus by flocking algorithm. The flocking algorithm is working as an adjustment for the control input. As stated in section 2.1, the control input is the speed of wheels, which is convertible with the velocity of the agent platform.

5.1 Flocking Measurement Analysis

For each iteration, the position displacement between an i^{th} agent and an j^{th} agent is updated in the i^{th} on-board microcontroller shown below:

$$d_{ij} = \begin{bmatrix} \mathrm{x}_j \\ \mathrm{y}_j \end{bmatrix} - \begin{bmatrix} \mathrm{x}_i \\ \mathrm{y}_i \end{bmatrix} \tag{5.1}$$

The range displacement is the key to the comparison of sensing range:

$$r_{ij}{}^2 = (\mathrm{x}_j - \mathrm{x}_i)^2 + (\mathrm{y}_j - \mathrm{y}_i)^2 \tag{5.2}$$

When the range displacement of an agent meets the satisfaction of the sensing range, both movements of separation and cohesion are determined by two factors. The first decision weighting number controls a constant static displacement, and the second decision weighting number controls the speed based on the displacement among agents.

5.1.1 Separation: Collision Avoidance

Each agent speed adjusted based on the displacement between itself and neighbors:

$$\begin{bmatrix} v_{i,\mathrm{x}}(k) \\ v_{i,\mathrm{x}}(k) \end{bmatrix} = \begin{bmatrix} v_{i,\mathrm{x}}(k-1) - ca * \frac{d_{ij,x}(k-1)}{r_{ij}(k-1)^2} \\ v_{i,\mathrm{x}}(k-1) - ca * \frac{d_{ij,y}(k-1)}{r_{ij}(k-1)^2} \end{bmatrix} \tag{5.3}$$

ca: collision avoidance parameter related to the displacement between agents. The velocity increases with a larger ca setting or more significant displacement with neighbors. When agents are close to each other, they will generate a repulsing force that prevents collisions.

5.1.2 Alignment, heading to the same direction

Each agent has a speed matching constant as

$$\begin{bmatrix} v_{i,\mathrm{x}}(k) \\ v_{i,\mathrm{x}}(k) \end{bmatrix} = \begin{bmatrix} v_{i,\mathrm{x}}(k-1) + \mathrm{M}_x \\ v_{i,\mathrm{x}}(k-1) + \mathrm{M}_y \end{bmatrix} \tag{5.4}$$

M: velocity adjustment constant.

5.1.3 Cohesion, move close to other agents

$$\begin{bmatrix} v_{i,\mathrm{x}}(\mathrm{k}) \\ v_{i,\mathrm{x}}(k) \end{bmatrix} = \begin{bmatrix} v_{i,\mathrm{x}}(k-1) + fc * d_{ij,x}(k-1) \\ v_{i,\mathrm{x}}(k-1) + fc * d_{ij,y}(k-1) \end{bmatrix} \tag{5.5}$$

fc: flock centering parameter related to the displacement between agents.

For heading angle is the same as the position shown below:

$$\omega_{i,\beta}(\mathrm{k}) = \omega_{i,\beta}(\mathrm{k}-1) + fc_{\beta} * (\beta_j(\mathrm{k}-1) - \beta_i(\mathrm{k}-1)) \tag{5.6}$$

fc_{β}: flock centering parameter related to the displacement between agents for heading angle.

The position of an i^{th} agent will be updated depending on the updated velocity:

$$q_i(\mathrm{k}+1) = q_i(\mathrm{k}) + T * \begin{bmatrix} v_{i,\mathrm{x}}(k) \\ v_{i,\mathrm{y}}(k) \\ \omega_{i,\beta}(k) \end{bmatrix}, q_i = \begin{bmatrix} \mathrm{x}_i \\ \mathrm{y}_i \\ \beta_i \end{bmatrix} \tag{5.7}$$

5.2 Flocking Algorithm Procedures

The multi-agent network initially consists of random distributed mobile agents, predesignated leading agents, and selected beacon agents in a closed area. Beacon agents measure the angle of arrival to other blind agents and calculate the location of each blind agent. The flocking algorithm with free formation is shown below with inputs of its states and neighbor's states and output of the desired speed for itself to wheels.

Algorithm 3 Flocking Algorithm with Free Formation

procedure FREE FORMATION($q_i, \dot{q}_i, q_j$)

$$\triangleright\ q_i = \text{its own states} = \begin{bmatrix} x_i \\ y_i \\ \beta_i \end{bmatrix}$$

$$\triangleright\ q_j = \text{neighbor agent's states} = \begin{bmatrix} x_j \\ y_j \\ \beta_j \end{bmatrix}$$

for *each neighbor* j **do**

$$d = q_j - q_i = \begin{bmatrix} \Delta x_j \\ \Delta y_j \\ \Delta \beta_j \end{bmatrix}$$

$$r_j = \sqrt{(\Delta x_j)^2 + (\Delta y_j)^2}$$

$$\dot{q}_{i,desired} = \begin{bmatrix} \dot{x}_{i,d} \\ \dot{y}_{i,d} \\ \dot{\beta}_{i,d} \end{bmatrix} = \begin{bmatrix} \dot{x}_i - ca \times \frac{\Delta x_j}{r_j^2} + M_x + fc \times \Delta x_j \\ \dot{y}_i - ca \times \frac{\Delta y_j}{r_j^2} + M_y + fc \times \Delta y_j \\ \dot{\beta}_i + fc_\beta \times \Delta \beta_j \end{bmatrix}$$

end

Wheel Control($\dot{q}_{i,desired}, \dot{q}_i$)

end procedure

The difference between connected formation and snake formation is the target object of separation and cohesion. Connected formation achieves cohesion with all neighbor agents, while the snake formation follows one agent in the snake. The following flocking algorithm with snake formation demonstrates the case of multiple leaders.

Algorithm 4 Flocking Algorithm with Snake Formation

```
procedure SNAKE FORMATION(q_i, q̇_i, q_j)
    if i^th agent contacted with a leader then
        if connect with j^th agent contacted with a different leader then
            Merge two snakes to one snake
        else
            Keep follow the original target
        end
    else
        if connect with j^th agent contacted with a leader then
            Merge to this snake and follow j^th agent
            Cohesion with only j^th agent previous location
            Seperation with all other agents
        else
            Free Formation(q_i, q̇_i, q_j)
        end
    end
end procedure
```

CHAPTER VI. SIMULATION RESULT

In this section, simulation studies are carried out to demonstrate the effectiveness of the proposed flocking algorithm. In a predesigned area, one or a few leading agents go through a searching function to collect all isolated agents on the map. Beacon agents are set to measure the location of blind agents dynamically. Two approaches of flocking: constrained flocking in free-space and the presence of movement shape formation consensus were tested for the consensus effectiveness and efficiency. Both algorithms embody all three rules of Reynolds and peer-to-peer network architectures. The first algorithm implies no need for a specific movement shape pattern. The constrained flocking is based on incomplete but connected graph nodes with one/multiple leading agents with a predesigned trajectory moving path. The position consensus goal for this algorithm is to achieve decentralized formation.

Due to the fact that the heading angle is not aligned with the movement of the agent, cohesion becomes the major part of the flocking phenomenon taken into consideration of heading angle consensus. No collision of the heading angle needs to be avoided. The heading angle consensus is based on the local broadcast-type communication and the decentralized control method. By controlling the velocity of each agent platform, the position of each agent is updated after a specified time interval. The sampling rate of the position data transmitting and receiving should allow the speed of wheels to meet the steady-state after the completion of the wheels' feedback control loop. The agents are continually detecting their neighbors and reacting correspondingly if under their sensing range. The effectiveness of the proposed control strategies is demonstrated through simulation results. A comprehensive analysis is provided in the following subsections for the result of heading angle consensus and position consensus.

6.1 Flocking with Connected Formation

Without a specific formation requirement, a group of agents will eventually generate a free cluster led by leader agents. The free cluster has no movement sequence constraint for follower agents. Initially, the agents were randomly distributed on the map with four beacon agents and three leading agents, as shown in Figure 13. From Figure 14, agents with a sensing range of 5 then gathered around to the center of the sensible region and formed a free cluster. The consensus is expected to be achieved after the leading agents finish searching the entire map. The consensus is that agents are gathered into one cluster and led by the leading agents, as shown in Figure 15.

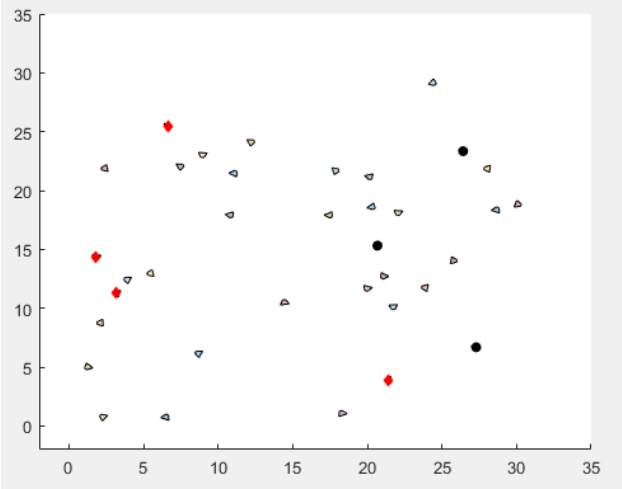

Figure 13: Initial state (red dot: beacon agents; black dot: leading agents)

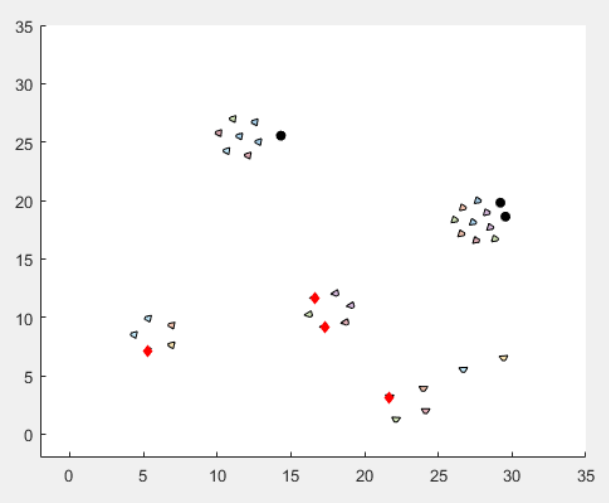

Figure 14: Free clusters state

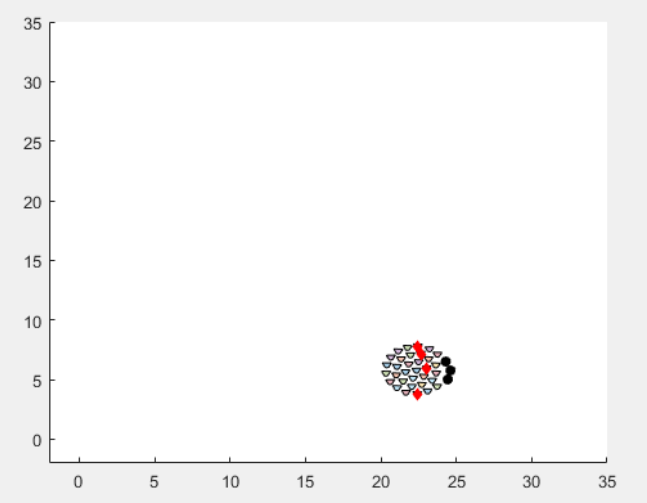

Figure 15: Final state with leading agents

The displacements and heading angle difference among all the agents were varied with time, and the result is shown below. The displacement is the summation of all displacements from one agent to all other agents. The difference of heading angles is the summation of all differences among all agents. In a timely frame of 20 seconds, the difference of heading angle initially is reasonably significant but decreased quickly under the effect of heading angle cohesion. Due to the sensing range limitation, the difference of heading angles remained unchanged until two clusters merged at a time of 1.79 seconds. The displacement at the initial state among all the agents gradually decreased due to the cohesion rule of flocking, but the majority of the agents were not in contact with leading agents. At 1.79 seconds, the leading agents started to do searching function as the displacement was growing larger. At 3.71 seconds, the cause of the peak is the turning of leading agents instead of amalgamation of two clusters. As indicated by the sudden drops of the heading angle difference among all agents at the time of 1.68, 4.71. 7.09, 9.77, and 13.66 seconds, there are at least 5 cluster unities. From Figure 16, the group of agents achieved the final state around the time of 13.65 seconds.

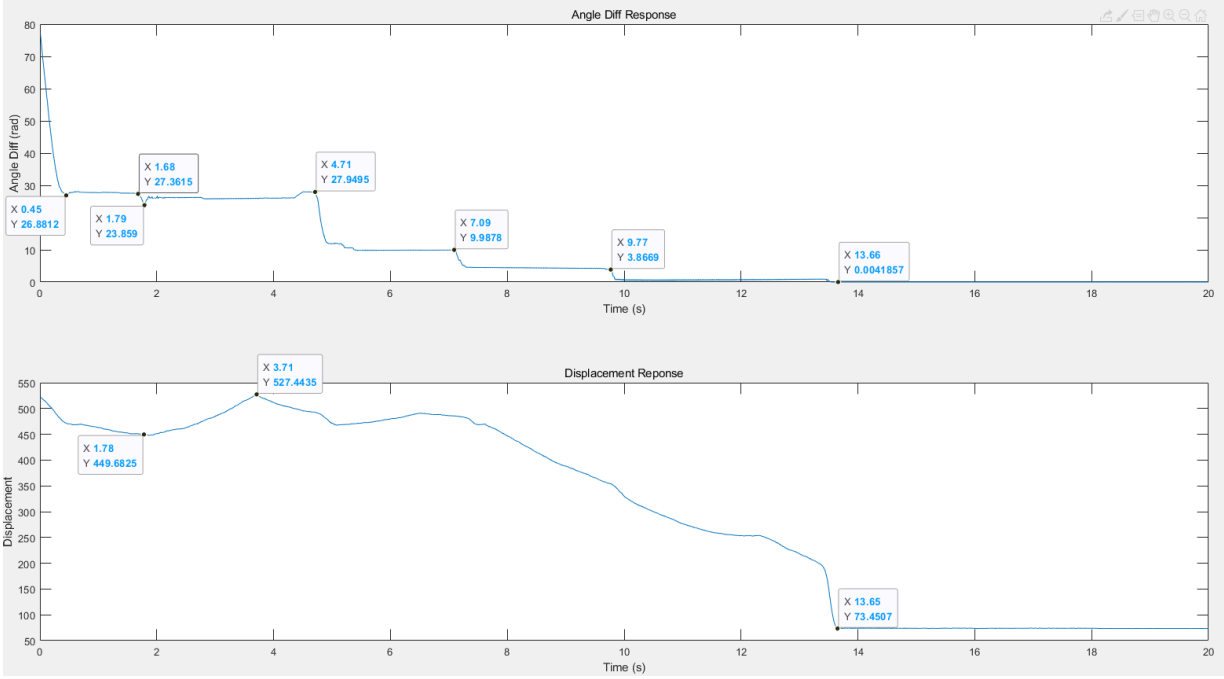

Figure 16: Simulation result of connected formation (single test)

6.2 Flocking with Snake Formation

With a specific formation requirement, a group of agents will eventually generate a snake shape movement led by leader agents. Similar to no formation, the agents were initially randomly distributed on the map and then gathered to a free cluster without communication to leaders shown in Figure 13. In the case of a single leader, a leading agent searched the whole map and collected isolated agents to form a snake movement formation demonstrated in Figure 17. When an isolated agent was in a direct or indirect connection with the leader, this agent will merge with the snake and become a member of followers. Assigning sequence order to each follower agent allowed the existence of snake shape in movement. In the case of multiple leaders, when two snakes led by leading agents are within each other's sensing range, one of the leaders became a follower and joined the other snake sequence. The number of leaders decremented during encounters until only one snake formation existed on the map shown in Figure 18. When the leader agent reached the desired location coordinate, the snake spun around a leader until all agents are in a designated range of leaders to form a position consensus shown in Figures 19 and 20. The consensus is expected to be achieved after the leading agents finish searching the entire map, and followers started to spin around the final leader agent.

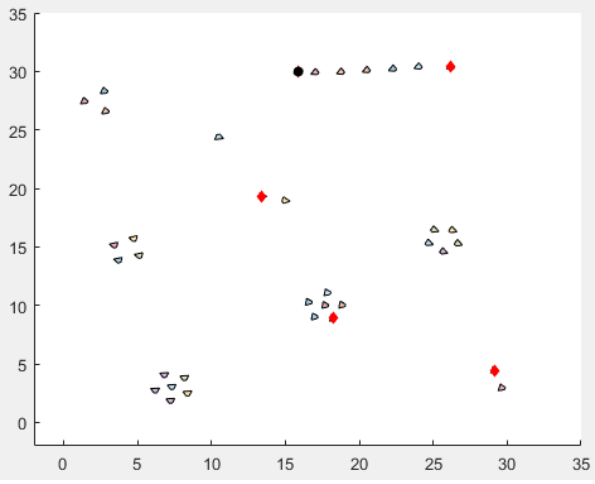

Figure 17: Snake formation searching state

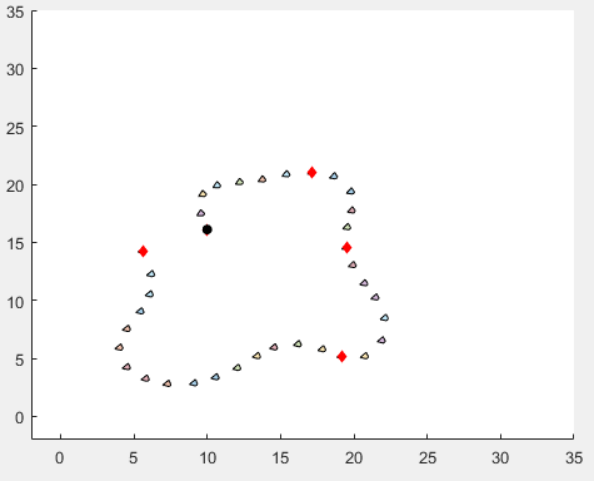

Figure 18: Single snake formation in the map

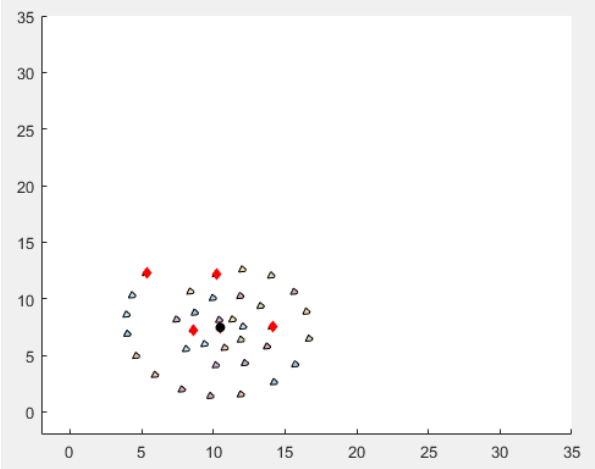

Figure 19: Followers spun around a leading agent

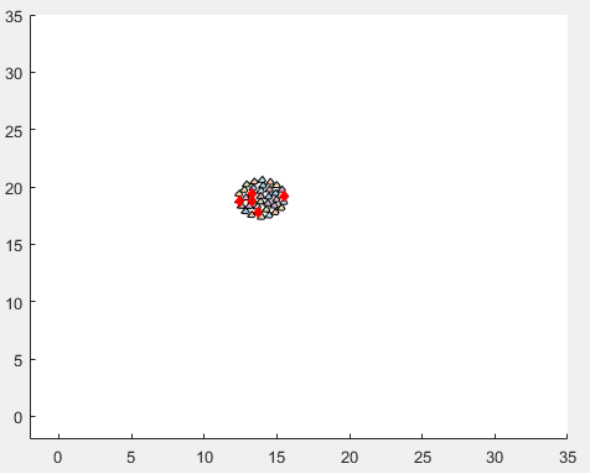

Figure 20: Position consensus with snake formation

The simulation results of displacements and heading angle difference among all the agents were similar to connected movement formation. In the same time frame, the difference of heading angle started at a considerable value but decreased while the leading agents are collecting isolated agents. From the observation of angle difference among all agents, the free agents started to gather together into free clusters until time of 0.42 seconds, and then it was stable until the leading agent went into the sensing range of other agents. While all agents are in connection with a leading agent, the heading angles remain specific differences among all agents until the disappearance of snake formation due to the limited connected nodes of one node in the communication network.

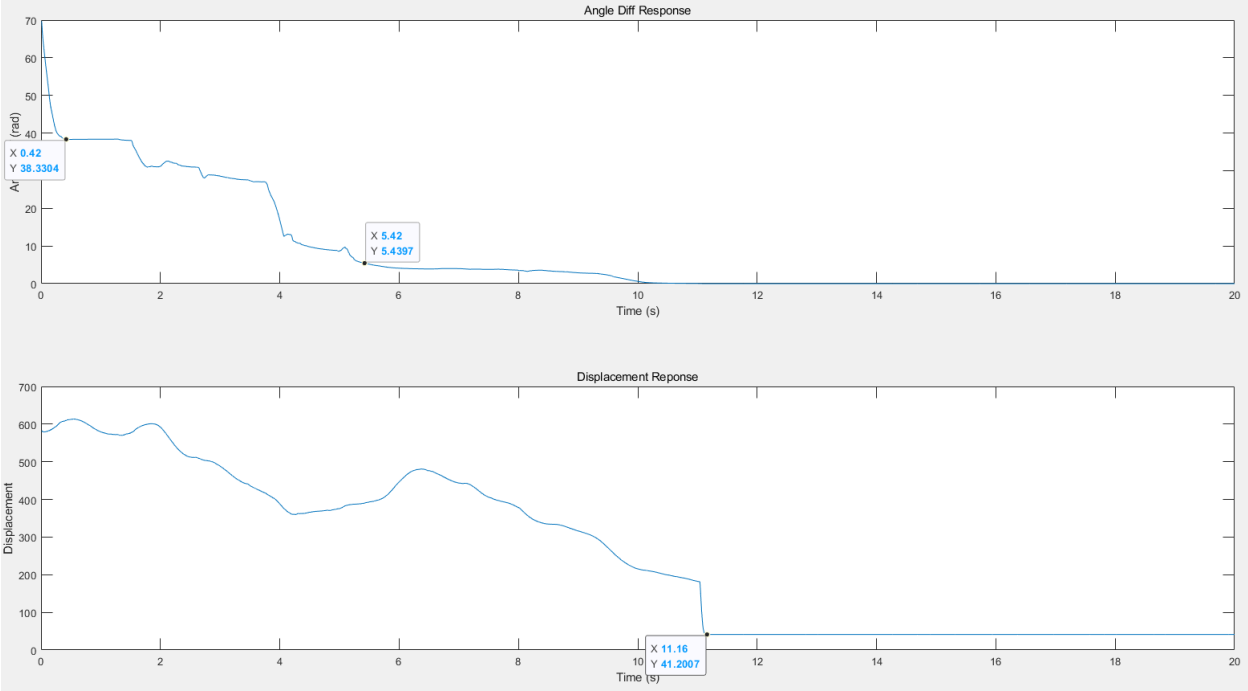

Figure 21: Simulation result of snake formation (single test)

6.3 Formation Comparison

Two simulations (Figures 22 and 23) from different methods with multiple-leaders were compared in this section. Under the same condition of five leaders, six beacon agents and a sensing range of 5 units. Heading angle consensus is slower than the connected formation, but the position consensus is faster than the connected formation. Overall, snake formation is a more efficient way to collect agents than connected formation but less stable with adjusting the heading angle to meet the consensus.

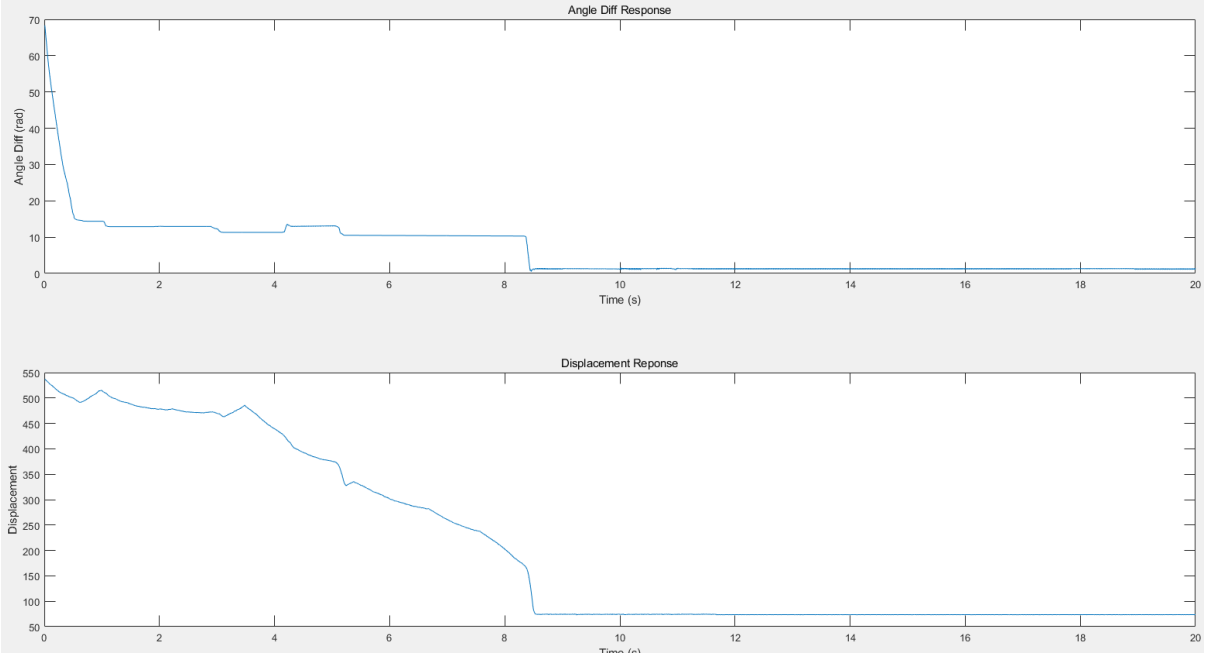

Figure 22: Simulation result of connected formation (comparison)

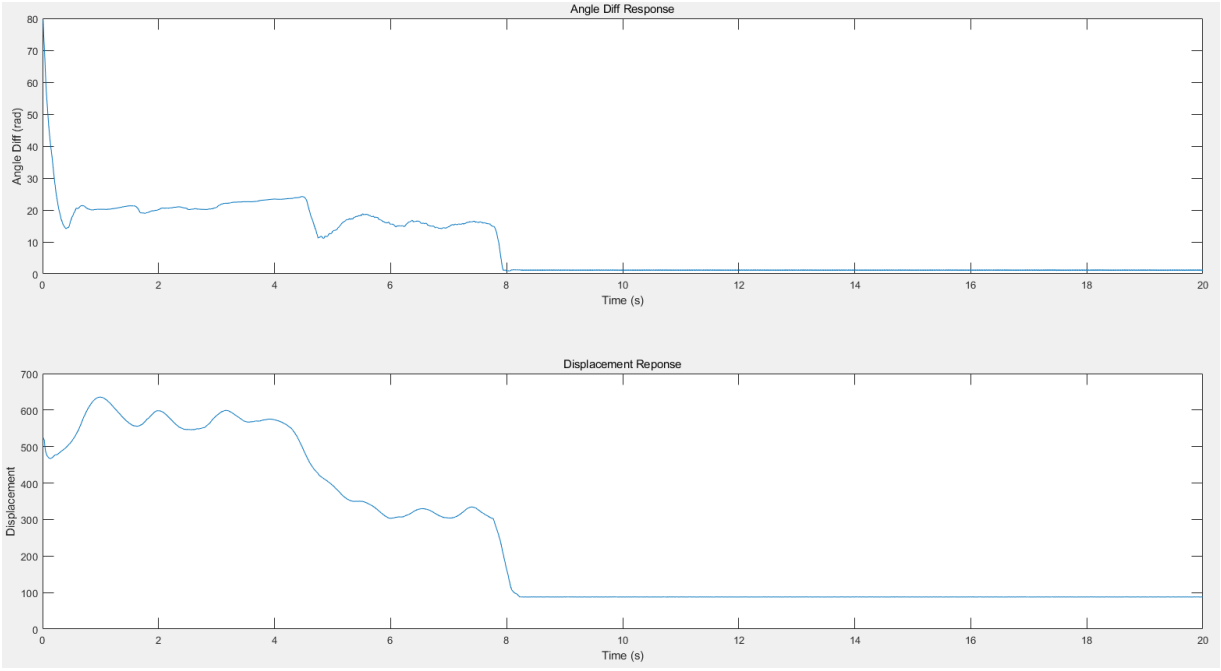

Figure 23: Simulation result of snake formation (comparison)

CHAPTER VII. SUMMARY AND FUTURE WORK

7.1 Summary

In this thesis, we proposed two flocking algorithms that used three rules and artificial potential force concept. These two flocking algorithms are for Mecanum wheeled robots' position and heading angle consensus. We also generated detailed simulation results of connected formation and snake formation to verify the robustness of algorithms.

At first, we extended a kinematic model for a four-Mecanum-wheeled robot to get a relationship between wheel speeds and robot platform velocity. By obtaining a developed relationship from Fuad in 2017 (Modeling and simulation for heavy-duty mecanum wheel platform using model predictive control.), we defined a Jacobian matrix and applied pseudo-inverse theory to get an inverse relationship between individual wheel speeds and robot cartesian position velocities and angular velocity about the heading angle direction. This relationship is the base of the discrete plant system in the feedback control loop of the robot moving speed. Assuming the Mecanum wheels are attached to DC motors, the stability control problem became controlling the rotational speed of DC motors. Under the establishment of the DC motor circuit state-space representation, a discrete PID controller was designed by utilizing the Simulink PID transfer function tuning tool.

Next, we discussed three possible system connectivities among all the agents by proposing three graph formations. The free formation is the initial state where agents are randomly distributed on the field without connection to the virtual leader. Connected formation and snake formation are the final states where the follower agents have connectivity with the virtual leader.

We generated the least square calculation method for an existing technology called Angle of Arrival to find a position for an unknown agent. The existing AoA used the TDoA between

individual elements of the antenna array to get an AoA measurement. With multiple AoA measurements acquired from beacon agents, the calculation of the location coordinate of the agent applied the least squared method to realize network localization. Due to the omnidirectional characteristic of the Mecanum wheel, a requirement of heading angle consensus is the local communication network among agents. Information exchange is built upon the Gossip Algorithm, which allows agents to parallel communicate with each other and react accordingly.

Finally, we developed flocking algorithms with connected and snake formation based on the Reynolds' three rules of flocking algorithm: separation, alignment, and cohesion. The simulation results in this thesis showed the robustness of these two flocking algorithms meet the position and heading angle consensus requirements. In addition, the simulation results showed that the connected formation allows follower agents to have a much faster response in finding the heading angle consensus once the follower agent has connectivity with the virtual leader. The snake formation has better performance on collecting follower agents and achieve position consensus.

7.2 Future Work

The kinematic model of a four-Mecanum-wheeled robot can be extended to include the motor and physical components, which increases the model complexity. Besides, fuzzy logic can be implemented into the PID controller to gain a fuzzy-PID controller with a higher efficient and more stable performance. Fuzzy logic helps in tuning the PID controller parameters with fuzzification, rule-base, and defuzzification processing. Faster response to meet the robots' velocities allows a lower sampling period of the flocking algorithm and a more robust consensus result.

Furthermore, other methods of AoA localization can be used to locate the position coordinates of robots, such as time of flight (ToF), time of arrival (ToA), and received signal strength indication

(RSSI). RSSI is less dependent than ToF, ToA, and TDoA on the synchronized timers, which induce smaller timing errors. However, the RSSI is not as robust as TDoA to noise since the signal fades with the propagation distance between transmitter and receiver. Currently, the heading angle is assumed to be measured by a compass measurement device on each robot. It would save more energy and equipment budgets by applying the central measurement method. Moreover, a Kalman Filter can be implemented with signal processing to get a more accurate measurement. Kalman Filter uses a series of measurements observed over a certain period and produces an estimate with highly statistically accuracy.

There is much room for improvement in the flocking algorithms provided to achieve position and heading angle consensus. The formations of the flocking algorithms can be explored to be more flexible movement shapes, such as triangular or circular. The searching function will be accordingly varied with the follower formations to simultaneously increase the converge rate of the position consensus and heading angle consensus.

REFERENCE

A. F. M. Fuad, I. A. (2017). Modeling and simulation for heavy-duty mecanum wheel platform using model predictive control. *Conference Series: Materials Science and Engineering. Vol. 184, No. 1*, p. p. 012050. IOP Publishing.

Aspnes, J., Eren, T., Goldenber, D. K., Morse, A. S., Whiteley, W., Yang, Y. R., . . . Belhumeur, P. N. (2006). A theory of network localization. *IEEE Transactions on Mobile Computing*, 5(12), 1663-1678.

Boyd, S., Ghosh, A., Prabhakar, B., & Shah, D. (2006). Randomized gossip algorithms. *IEEE/ACM Transactions on Networking (TON), 14*(SI), pp. 2508-2530.

Breder Jr, C. M. (1954). Equations descriptive of fish schools and other animal aggregations. *Ecology, 35*(3), pp. 361-370.

Cai, D., Sun, J., & Wu, S. (2012). UAVs formation flight control based on behavior and virtual structure. *Asian Simulation Conference* (pp. 429-438). Berlin, Heidelberg: Springer.

Control Tutorials for MATLAB and Simulink - Motor Speed: System Modeling. (2019, 10 11). Retrieved from MATLAB® 9.2: http://ctms.engin.umich.edu/CTMS/index.php?example=MotorSpeed&section=SystemModeling

Cox, I. J. (1990). *Autonomous Robot Vehicles.* New York: Springer.

Crowther, B. (2003). Flocking of autonomous unmanned air vehicles. *The Aeronautical Journal, 107*(1069), pp. 99-109.

Dimarogonas, D. V., & Kyriakopoulos, K. J. (2005). Formation control and collision avoidance for multi-agent systems and a connection between formation infeasibility and flocking behavior. *Proceedings of the 44th IEEE Conference on Decision and Control* (pp. 84-89). IEEE.

Djaidja, S., & Wu, Q. (2015). Leader-following consensus for single-integrator multi-agent systems with multiplicative noises in directed topologies. *International Journal of Systems Science, 46*(15), pp. 2788-2798.

Ferrari-Trecate, G., Galbusera, L., Marciandi, M. P., & Scattolini, R. (2009). Model predictive control schemes for consensus in multi-agent systems with single-and double-integrator dynamics. *IEEE Transactions on Automatic Control, 54*(11), 2560-2572.

Grünbaum, D., & Okubo, A. (1994). Modelling social animal aggregations. In *Frontiers in mathematical biology.* (pp. 296-325). Berlin, Heidelberg: Springer.

Gu, D., & Wang, Z. (2009). Leader–follower flocking: algorithms and experiments. *IEEE Transactions on Control Systems Technology , 17.5*, 1211-1219.

Gulzar, M. M., Hussain, R. S., Javed, M. Y., Munir, U., & Asif, H. (2018). Multi-agent cooperative control consensus: A comparative review. *Electronics, 7.2*, 22.

Ilon, B. E. (1972). *United States Patent No. US3876255A.*

Jadbabaie, A., Lin, J., & Morse, A. S. (2003). Coordination of groups of mobile autonomous agents using nearest neighbor rules. *Departmental Papers (ESE)*, 29.

Kim, J. M., Park, J. B., & Choi, Y. H. (2014). Leaderless and leader-following consensus for heterogeneous multi-agent systems with random link failures. *IET Control Theory & Applications, 8*(1), 51-60.

Kwang-Kyo, O., Park, M.-C., & Ahn, H.-S. (2015). A survey of multi-agent formation control. *Automatica, 53,* 424-440.

Lawton, J. R., & Beard, R. W. (2002). Synchronized multiple spacecraft rotations. *Automatica, 38*(8), 1359-1364.

Li, Q., & Jiang, Z.-P. (2008). Flocking of decentralized multi-agent systems with application to nonholonomic multi-robots. *IFAC Proceedings Volumes, 41*(2), pp. 9344-9349.

Lin, J., Morse, A. S., & Anderson, B. D. (2003, December). The multi-agent rendezvous problem. *42nd IEEE International Conference on Decision and Control (IEEE Cat. No. 03CH37475). 2*, pp. 1508-1513. IEEE.

Liu, Z.-W., Guan, Z.-H., Li, T., Zhang, X.-H., & Xiao, J.-W. (2012). Quantized consensus of multi-agent systems via broadcast gossip algorithms. *Asian Journal of Control, 14*(6), pp. 1634-1642.

Lizarralde, F., & Wen, J. T. (1996). Attitude control without angular velocity measurement: A passivity approach. *IEEE transactions on Automatic Control, 41*(3), 468-472.

Luo, X., Shaobao, L., & Guan, X. (2010). Flocking algorithm with multi-target tracking for multi-agent systems. *Pattern Recognition Letters, 31*(9), pp. 800-805.

Meng, Z., Ren, W., Cao, Y., & You, Z. (2010). Leaderless and leader-following consensus with communication and input delays under a directed network topology. *41*(1), 75-88.

Mesbahi, M., & Egerstedt, M. (2010). Graph theoretic methods in multiagent networks. In M. Mesbahi, & M. Egerstedt, *Graph theoretic methods in multiagent networks* (p. Vol. 33). Princeton University Press.

Netjinda, N., Achalakul, T., & Sirinaovakul, B. (2015). Particle Swarm Optimization inspired by starling flock behavior. *Applied Soft Computing, 35*, 411-422.

Okubo, A. (1986). Dynamical aspects of animal grouping: swarms, schools, flocks, and herds. *Advances in biophysics, 22*, 1-94.

Olfati-Saber, R., Fax, J. A., & Murray, R. M. (2007). Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE, 95*(1), pp. 215-233.

O'Loan, O. J., & Evans, M. R. (1999). Alternating steady state in one-dimensional flocking. *Journal of Physics A: Mathematical and General, 32.8*, L99.

Ren, W. (2009). Distributed leaderless consensus algorithms for networked Euler–Lagrange systems. *International Journal of Control, 11*, pp. 2137-2149.

Ren, W., & Atkins, E. (2007). Distributed multi-vehicle coordinated control via local information exchange. *International Journal of Robust and Nonlinear Control: IFAC-Affiliated Journal, 17*(10-11), pp. 1002-1033.

Ren, W., & Beard, R. W. (2008). Consensus algorithms for double-integrator dynamics. In *Distributed Consensus in Multi-vehicle Cooperative Control: Theory and Applications* (pp. 77-104).

Ren, W., Beard, R. W., & M., A. E. (2005). A survey of consensus problems in multi-agent coordination. *Proceedings of the 2005, American Control Conference, 2005.* (pp. 1859-1864). IEEE.

Reynolds, C. W. (1987). *Flocks, herds and schools: A distributed behavioral model.* New York, NY, USA: ACM.

Reza, O.-S. (2006). Flocking for multi-agent dynamic systems: Algorithms and theory. *IEEE Transactions on automatic control, 51.3*, 401-420.

Rong, P., & Mihail, L. S. (2006, 9). Angle of arrival localization for wireless sensor networks. *2006 3rd annual IEEE communications society on sensor and ad hoc communications and networks, 1*, pp. 374-382.

Schelling, T. C. (1971). Dynamic models of segregation. *Journal of mathematical sociology, 1*(2), pp. 143-186.

Seyboth, G. S., & Johansson, K. H. (2013). Event-based broadcasting for multi-agent average consensus. *Automatica, 49*(1), 245-252.

Shi, Y., & Eberhart, R. C. (2001). Fuzzy adaptive particle swarm optimization. *Proceedings of the 2001 congress on evolutionary computation (IEEE Cat. No. 01TH8546), 1*, 101-106.

Su, H. X. (2008). "Flocking in multi-agent systems with multiple virtual leaders. *Asian Journal of control, 10*(2), pp. 238-245.

Su, Y., & Huang, J. (2011). Stability of a class of linear switching systems with applications to two consensus problems. *IEEE Transactions on Automatic Control, 57*(6), 1420-1430.

Tanner, H. G., Jadbabaie, A., & Pappas, G. J. (2003, December). Stable flocking of mobile agents, Part I: Fixed topology. *42nd IEEE International Conference on Decision and Control (IEEE Cat. No. 03CH37475). 2*, pp. 2010-2015. IEEE.

Vicsek, T., Czirók, A., Ben-Jacob, E., Cohen, I., & Shochet, O. (1995). Novel type of phase transition in a system of self-driven particles. *Physical review letters, 75*(6), 1226.

Wang, L., Shi, H., & Chu, T. (2005, June). Flocking control of groups of mobile autonomous agents via local feedback. *Proceedings of the 2005 IEEE International Symposium on, Mediterrean Conference on Control and Automation Intelligent Control, 2005.*, pp. 441-446.

Wang, L., Wang, X., & Hu, X. (2013). Connectivity preserving flocking without velocity measurement. *Asian Journal of Control, 15*(2), pp. 521-532.

Warburton, K. a. (1991). Tendency-distance models of social cohesion in animal groups. *Journal of theoretical biology, 150*(4), pp. 473-488.

Wen, G., Duan, Z., Li, Z., & Chen, G. (2012). Flocking of multi-agent dynamical systems with intermittent nonlinear velocity measurements. *International Journal of Robust and Nonlinear Control, 22*(16), pp. 1790-1805.

Wynblatt, M., & Hsu, A. (2001). *Washington, DC: U.S. Patent and Trademark Office. Patent No. 6,219,696.*

Yan, Q., Chen, J., Ottoy, G., Cox, B., & Strycker, L. D. (2018, March). An accurate AOA localization method based on unreliable sensor detection. *2018 IEEE Sensors Applications Symposium (SAS)*, pp. 1-6.

Yong, C., Guangming, X., & Huiyang, L. (2012). Reaching consensus at a preset time: Single-integrator dynamics case. *Proceedings of the 31st Chinese Control Conference.* (pp. 6220-6225). IEEE.

Zavlanos, M. M., Tanner, H. G., Jadbabaie, A., & Pappas, G. J. (2009). Hybrid control for connectivity preserving flocking. *IEEE Transactions on Automatic Control, 54*(12), 2869-2875.

Zhan, J., & Li, X. (2012). Flocking of multi-agent systems via model predictive control based on position-only measurements. *IEEE Transactions on Industrial Informatics, 9.1*, 377-385.

Zhou, J., Wu, X., Yu, W., Small, M., & Lu, J. (2012). Flocking of multi-agent dynamical systems based on pseudo-leader mechanism. *Systems & Control Letters, 61*(1), pp. 195-202.