ROBUST TIME-VARYING FORMATION CONTROL OF HETEROGENEOUS MULTI-AGENT SYSTEMS USING NOVEL POTENTIAL FIELD AVOIDANCE

by

Ryan Adderson

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy

at

Dalhousie University
Halifax, Nova Scotia
August 2024

Table of Contents

List of Tables

List of Figures

Abstract

This thesis investigates time-varying formation algorithms for teams of heterogeneous multi-agent systems consisting of ground based two-wheel mobile robots and aerial quadrotors while developing novel obstacle avoidance algorithms to improve the navigation capabilities of the agents. Sliding mode controllers are developed based on designs from literature, with some refinements for their specific applications. These are tested for a variety of cases in order to ensure they will be sufficient for the control of agents in formation. The obstacle avoidance algorithm developed is an improvement on the artificial potential field method. By using additional information about the position of the agent relative to an obstacle, a more informed obstacle avoidance protocol is developed, which reduces the impact of the repulsive fields of the obstacles. This is done primarily based on factors such as the heading of the agent relative to the obstacle. Two methods for time-varying formation are then developed which use the previously developed controller for navigation, and the obstacle avoidance to prevent collisions. With these algorithms, agents navigate through an environment by making collective changes in their formation based on the information about the environment gathered by leader agents. Simulated and experimental testing validates the effectiveness of these methods.

List of Abbreviations and Symbols

Abbreviations

APF	Artificial Potential Field
DOF	Degrees of Freedom
ETC	Event Triggered Control
GPMP2	Gaussian Process Motion Planning 2
GRA	Group Role Assignment
IMU	Interial Measurement Unit
MAP	Maximum A Posteriori
MAS	Multi-Agent System
PID	Porprotional Integral & Differential
ROS	Robot Operating System
SMC	Sliding Mode Control
TSMC	Terminal Sliding Mode Control
UAV	Unpiloted/Unmanned Aerial Vehicle
UGV	Unpiloted/Unmanned Ground Vehicle
2WMR	Two Wheel Mobile Robot

Symbols

A	Adjacency Matrix
a_{ij}	Communication between agents
C_θ	Cosine of theta
$\vec{d_G}$	Distance to goal
e	Error
$L(e\|r)$	Likelihood
$\|\|\vec{X}\|\|$	Magnitude of Vector
$\prod$	Product
$P(r)$	Prior

s	Sliding Surface
S_θ	Sine of theta
sign()	Signum Function
$\sum$	Summation
X^T	Transpose of Matrix X
$\vec{U}_A$	Attractive Field
$\vec{U}_R$	Repulsive Field
u_x	Control Input

Acknowledgements

This thesis represents the culmination of five years of research work, and over those years I have received tremendous support from those around me. I would first like to thank my supervisor, Dr. Ya-Jun Pan, who provided invaluable assistance throughout my doctoral studies. Without her insights and guidance, this work would not have been possible.

This work was also aided by my thesis committee, Dr. Darrel Doman and Dr. Vincent Sieben. Both provided valuable feedback during our annual meetings which helped to refine this thesis into the final product.

I have also had the opportunity to work with a number of fantastic people in the Advanced Controls and Mechatronics Lab. Dr. Henghua Shen, Dr. Behzad Akbari, and Lucas Wan all co-authored papers with me, and all provided unique insights to the types of problems I have worked with which truly elevated what I was able to accomplish during my studies. Dr. Zipeng Huang also offered a great deal of valuable information about the control of the Crazyflie quadrotor.

I would also like to thank my parents, Tom and Tracy Adderson, for providing a truly incredible support structure. I would have never been able to make it to this point without the years of support they have provided. I would also like to thank my brother, Michael Adderson, as well as the rest of my family and friends for everything they have done for me over the years.

Chapter 1

Introduction and Background

1.1 Research Motivation and Objectives

As teams of robots become more embedded into various elements of life, it becomes increasingly important to develop robust methods of control. This includes interactions between robots of the same type, but also of different types. An example of the former would be autonomous vehicles communicating with each other in streets, and an example of the latter would be aerial robots providing an overview and investigation of an environment, and then directing robots on the ground what their objectives should be. Agents with a variety of different capabilities can work together to accomplish complex tasks, and so novel approaches are needed to optimize these agents.

Many avenues of research have emerged within the realm of robotics to help address this growing need. Algorithms and techniques to enable teams of agents to collaborate to achieve specific goals have been seeing a prominent surge of interest. A particularly significant avenue is formation control, in which a group of agents arrange themselves into a specific configuration.

The objective of this work is to develop robust control algorithms for heterogeneous multi-agent systems (MASs) which implement obstacle avoidance protocols and can produce time-varying formations. The investigated systems use teams of robots, with the heterogeneity coming from the use of unpiloted/unmanned ground vehicles (UGV), in the form of two wheel mobile robots (2WMRs), and unpiloted/unmanned aerial vehicles (UAV), in the form of quadrotors.

With these systems it is desirable to have agents which can achieve a formation, but also for the formation to be able to change. "Time-Varying Formation" is a term which is applied to a variety of different applications, from constant formations moving in space to dynamically swapping between multiple different formations. Here, the emphasis is on having teams which are capable of dynamically navigating an

environment and making the adjustments to the formation which are needed to allow that to happen.

There are a number of potential applications of this work which serve as the primary motivations for research in this field. Search and rescue operations can see tremendous benefits from having teams of agents engaged in the search operations, and can communicate with potential rescuers (whether these be people or additional agents). This can apply to sea-based search and rescue as well as land-based. Military applications for teams of agents are also plentiful, as these agents can be used for reconnaissance in mapping environments.

The scope of this thesis will cover algorithms for the purpose of formation control, as well as a novel obstacle avoidance algorithm based on sensor information and two approaches to time-varying formation. The controllers are specifically designed for 2WMRs and quadrotors. The obstacle avoidance algorithm uses sensor data to adjust the trajectory of the agent. Signal filtering is considered outside the scope of this work, and so sensor data is assumed to be ideal.

1.2 Thesis Contributions

The primary contributions of the thesis are as follows:

1. Two sliding mode control (SMC) protocols have been developed for each of 2WMRs and quadrotors. These are designed to allow for agents to communicate with one another for the purpose of achieving consensus or formation. These use a position-based approach to formation in which the formation is defined by relative positions. A simple classical SMC is developed for both agent types which provides a simplified approach to control. A terminal SMC approach is also implemented for both agent types. For the 2WMR, this controller modifies an existing approach which could properly track curvilinear motion and makes it capable of tracking rectilinear motion. These controllers are shown to be resilient to disturbances.

2. A novel artificial potential field (APF) algorithm for obstacle avoidance has also been developed. A classic APF uses the proximity of an agent to an obstacle, and creates a “repulsive field” in the controller which directs the agent away

from the obstacle if it is sufficiently close. The new algorithm developed also accounts for the agent's heading relative to the obstacle, the time spent in the potential field, and relative velocities between the agent and obstacle to produce smoother, more efficient trajectories.

3. Two approaches to time-varying formation have also been developed and tested. The first of these uses a centralized distance-based methodology to adjust the size of a formation in order to navigate through an unknown environment. This methodology is also implemented with two other processes. One uses a previously developed motion planning algorithm for path-planning, and the other uses a center of formation estimation process to decentralize the process.

4. The second approach to time-varying formation utilizes multiple formations, and based on the environment, the agents can change between these. In addition, formations are designed so agents will be able to operate along a continuous spectrum of formations between the set of formations chosen.

5. These core elements are tested in simulations and experiments to validate their effectiveness. Each builds on the previous work, adding new elements to produce a more effective controller which is capable of navigating more complex environments.

1.3 Literature Review

The literature review is broken into four key areas: multi-agent systems, consensus formation control, obstacle and collision avoidance, and time-varying formation.

1.3.1 Multi-Agent Systems

Research into multi-agent systems (MASs) has been seeing a surge of studies due to an increase in the demand for teams of robots to be able to perform tasks autonomously in collaboration with one another [1] [2]. There are a large number of applications for industrial [3], military [4], and entertainment use [5] which contribute to this growing field of research. A key distinction in the classification of MASs is whether they are homogeneous (all agents are the same), or heterogeneous (containing different types

of agents). [6] presents a framework for conceptualizing the heterogeneity of a system. Here it is considered in terms of complexity (how many different types of agents are present in the system) and disparity (how different those agents are from one another). A number of key challenges remain in research. [7] notes that formation control of multi-agent systems remains a key area for future developments, particularly for cases with heterogeneity.

Communciation between agents is a significant factor in MASs. Communication is conventionally conceptualized using graph theory, with edges and vertices representing agents and communication [8] [9]. The structures of communication can lead to a variety of different approaches, but a commonly deployed approach uses a leader-follower dynamic where one or more agents serve take on the role of leaders and other agents will follow. This can reduce computational costs and allow for more distributed processes. Communication constraints such as time delays are also a common concern in consensus protocols and this has seen significant attention in research [10] - [12].

2WMRs are a specific class of agents which have gained notable popularity due to the extensive research into their models and control in MASs. Developing algorithms for these agents navigating environments in formations is thus an essential avenue of research to maximize the utility of these robots and engage them in cooperative operations [13]. A number of models have been developed for these agents [14].

Quadrotors are another type of agents which have been significantly studied in recent years. The numerous potential applications, such as operation in city environments and carrying loads as a team, have made them very desirable for researchers and industry [15] - [18]. These systems have a well developed history of modelling [19]. Quadrotors present a number of challenges due to being underactuated systems and having to deal with ground effects caused by the air from their propellers contacting the ground and create additional disturbances [20]. Fault tolerant control is thus a major area of research [21], as actuator faults can cause catastrophic failure.

Path-planning algorithms are the major point of focus for MASs, and have been implemented for a variety of cases over the years [22] [23]. They are popular with teams of agents for helping to avoid obstacles and planning the paths of multiple agents navigating an environment. However, optimized path-planning is computationally expensive, and becomes increasingly so as more agents are added to a

MAS [24]. Limiting the number of agents receiving optimized trajectories, and relying on leader-follower dyanmics for the remaining agents provides a novel avenue to gain much of the advantage of optimized control while dramatically reducing the computational complexity of the problem [25].

[26] notes that both local and global optimization problems remain for MASs, and that continued development of these areas is necessary. [27] emphasizes the variety of potential applications for MASs, including how they can be implemented for improvements in other fields such as game theory. Many of these problems may be investigated with neural networks and reinforcement learning, however conventional approaches remain popular as well.

Agents within a system can also be assigned specific "roles". These can be as simple as specific tasks to be completed, or in processes like Group Role Assignment (GRA) the role can be much more robust, encompassing trajectories and interactions with other agents [28]. This opens up avenues in Role Based Collaboration which allows for agents to interact with each other through different methodologies [29].

1.3.2 Consensus and Formation

A prominent focus of research in MASs is in consensus and formation. Consensus refers to agents reaching an agreement with one another about the current state of the system. Formation refers to a more specific state focused on agents positions being aligned with one another with some form of offset. For physical systems, formation has gained notable attention, and it has been applied to teams of underwater vehicles [30], ground robots [31], as well as aerial robots [32].

The communication topologies of agents is a significant factor in consensus and formation. Directed topologies are commonly utilized for leader-follower dynamics [33], but can also be utilized in fully distributed cases [34]. In these cases, the leader sends information to the follower, but not the reverse. Switching topologies are also deployed for a number of cases [35]- [37]. In these cases the topology will be time-varying.

Formation control in heterogeneous systems has received prominent research in recent years. The premise of more advanced functionality offered by having teams of robots with different capabilities working together has seen a number of novel

developments. This has included underwater vehicles interacting with surface vehicles to relay information [39] and coordination between ground and air vehicles [40]. Approaches using visual information have also been demonstrated [41].

Event-triggered control (ETC) has been proposed as an effective approach to reduce the communication load for multi-agent system and the communication between agents is needed when necessary. ETC has been implemented in many cases because it can be used to add fault tolerance to a system [42]. It has also been implemented in heterogeneous cases and been shown to be effective [43]- [46].

A common method of control for formations is SMC. The variable structure nature of SMC makes it useful for developing robust controllers [47] - [49]. In particular, it is often implemented when dealing with disturbances. Applications with Kalman filters have also been used [50]. Super-twisting SMC is also commonly applied to quadrotors with notable results [51] - [53].

Formation control can be implemented in a number of different ways. In position-based formation, agents move to a specific location within a global system [54]. In distance-based formation, agents move to positions which are a relative distance away from neighbouring agents within a communication topology [55]. In bearing-based formation, agents maintain positions such that the angles between positions of agents are constant [56].

1.3.3 Time Varying Formation

Time varying formation is used to refer to a variety of different approaches to formation. The simplest examples would be a formation where the positions of the agents varies in time, regardless of whether or not the desired formation of agents, relative to one another, varies [57]. Another approach covered under the term is cases wherein the relative positions of agents change as the formation moves with some type of scaling factor on an initial formation [58]. Finally, time-varying formation can also be used to refer to cases where multiple distinct formations employed, and the agents transition between these formations. Processes using ETC have also been developed [60].

In [59], a time-varying formation is deployed for a team of AUVs in which they track a moving virutal leader. These types of processes are commonly deployed

in path-planning algorithms, as the virtual leader effectively serves as a guide for the leader agents by dictating their path. A similar process was used in [61] to develop a time-varying formation algorithm in which clusters of agents form separate formations.

Affine Formation Control has also been deployed in many applications such as the control of quadrotors [63] [64]. This process uses a "stress matrix" which dictates how the formation will change, and has been shown to be able to effectively respond to disturbances [65]. Time-varying formation can also be applied to cases of bipartite formation in which two separate formations are deployed [66] [67].

Bearing-based time-varying formations have also received notable attention [68] - [71]. While most approaches to formation tend to define formations in terms of relative distances, these approaches do so in terms of relative angles. This can allow for seamless changes in the size of the formation, particular formations with multiple leaders. However, these approaches can be somewhat rigid in terms of the allowable formations.

1.3.4 Obstacle and Collision Avoidance

In most cases, the environment the agents are in will have obstacles, and so having the capability for agents to avoid both obstacles and each other is paramount to ensuring their use in real scenarios. Avoidance can be conceptualized as being broken into two main types: reactive and deliberative [72].

A reactive method will take specific actions based on the sensor input. These types of methods are ideal for operation in a dynamic environment, as it allows for quick responses to sudden changes. Deliberative methods, on the other hand, have a planning phase once data has been collected about obstacles. This typically means additional computational power is applied in order to develop a more optimized trajectory. This is ideal for navigating complex, but relatively static environments. Hybrid approaches, which combine elements of deliberative and reactive methods, are also seeing attention. [73] uses a deliberative approach for lane management of cars, but also maintains a reactive approach for sudden oncoming vehicles.

Within this work, artificial potential fields are used. This is a reactive method in which the proximity to obstacles is used to create a "repulsive force" in the controller,

guiding agents away from the obstacles. This original conceptualiziation remains relevant today, and a number of avenues have been pursued attempting to improve on the concept. Many modern applications include mobile robots [74], robotic arms [75], and mobile manipulator [76].

APF can also be deployed for formations. Examples include [77] which used a standard APF for a team of 3 mobile robots, [78] deployed a dynamic APF for use in simulations of a team of 4 agents, [79] utilizes an APF algorithm for a simulated team of 6 quadrotors in a complex environment, and [80] implements an APF approach for a team of autonomous underwater vehicles. The flexibility of this approach allows it to be used for a variety of types of teams. They have also been deployed for both local [81] and global [82] path planning.

Artificial potential fields have seen a number of novel approaches due to the flexible nature of the processes. Some of these include the implementation of decision trees [83], adaptive control laws [84], rapidly-exploring random trees for pre-planning [85], and A-Star obstacle avoidance [86]. A notable example is the implementation of an algorithm which adjusts the magnitude of the field based on the heading of an agent relative to obstacles [87]. A similar process was developed in the work of this thesis.

Another notable area of research is optimization-based path planning which has the obstacle avoidance built-in. Gaussian Process Motion Planning 2 (GPMP2) is a prominent example of such a path planner and has been deployed in a variety of applications [88] - [90]. In GPMP2, path planning is treated as a problem of probabilistic inference. The goal of the algorithm is to find the maximum a posteriori (MAP) estimate of the path. These algorithms have been applied to a variety of cases such as surface vehicles [91] and multi-manipulator cooperative robotics [92]

1.4 Background

The background will cover core mathematical ideas from literature which are necessary to fully understand the subsequent problem formulations of this thesis.

1.4.1 Graph Theory

Graph theory is commonly used in many MAS applications as a means of expressing the communication topology between agents. Here we will consider a case where the

vertices of the graph, V, are the agents within the system. We will also consider the edges of the graph, E, to be the communication between those agents.

Now, we can construct an adjacency matrix, A. This will be an $n \times n$ matrix, where n is the number of agents in the system. For each point in the matrix we can say the value, a_{ij}, is defined as:

$$\begin{cases} a_{ij} = 0, & i = j, \\ a_{ij} = 0, & \text{No Communication}, \\ a_{ij} = 1, & \text{Communication}, \end{cases} \tag{1.1}$$

If the values for a_{ij} and a_{ji} are both equal to 1, then we can say the communication is bidirectional. If one of these values is 1 and the other is 0, then the communication is unidirectional. All values along the main diagonal, where $i = j$ are set to 0, as an agent does not communicate with itself.

An agent is considered "reachable" if it can be reached from a given point by means of the communication connections. A globally reachable agent can be reached from any other agent. It has been shown for a globally reachable system, convergence to consensus is guaranteed. This serves as the backbone of communication between agents.

A simple case can be considered to express the basic concepts. Fig.1.1 shows a three vertex case with three edges. The edge between vertices 1 and 2 is birectional, while the edges between vertices 1 and 3, as well as 2 and 3, are both unidirectional. This indicates that Agent 1 sends signals to Agents 2 and 3, Agent 2 sends signals to Agent 3, and Agent 3 sends signals to Agent 1.

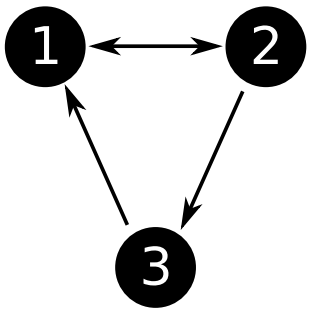

Figure 1.1: A visualization of communication topology using graph theory.

1.4.2 Sliding Mode Control

Sliding Mode Control (SMC) is a variable structure control protocol which uses multiple different (though similar) controllers, and switches between them as necessary to converge towards a desired result. Typically this desired result will be causing multiple errors to converge towards zero.

For these controllers, a "sliding surface" is developed. The surface is typically denoted as s. This surface is typically defined by linear combination of errors. For example, in the figure below the position and velocity errors of an imaginary system are plotted, along with a sliding surface. The objective then is to converge towards the center, where the errors will all be zero.

In SMC there are two main phases: the "reaching phase", and the "slding phase". Fig.1.2 shows a generalization of how SMC is designed to work. First, in the reaching phase, the function converges to a point where the value of s trends to 0. Then, in the sliding phase, the errors move along the $s = 0$ line until converging to 0.

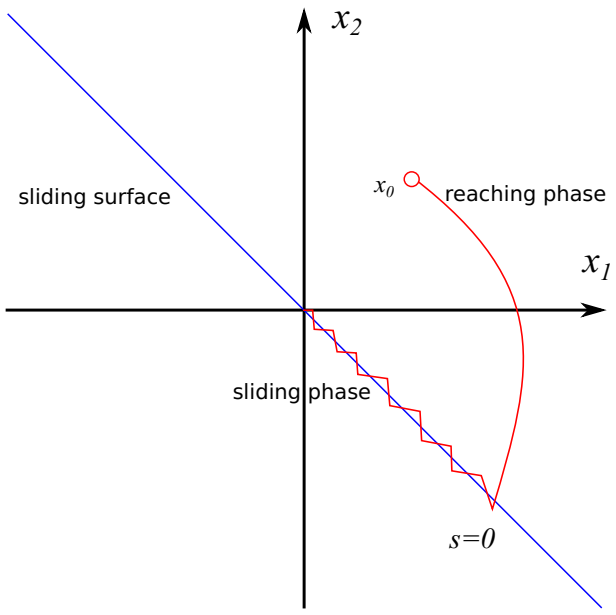

Figure 1.2: A visualization of the basics of sliding mode control.

The line which occurs in Fig.1.2 indicates a line where $s = x_1 + x_2$. More complex functions than this are typically used, however it is a useful visualization. The errors will broadly trend to be equal and opposite, and then the switching function will gradually trend the errors towards 0. Conventionally in SMC this swapping function

uses a signum function. This is defined as:

$$\text{sign}(x) = \begin{cases} 1 & x > 0, \\ 0 & x = 0, \\ -1 & x < 0, \end{cases} \tag{1.2}$$

However, this can lead to chattering within the system as it constantly oscillates between the two cases. To mitigate this, an alternative function is used in place of the signum function. The new function is:

Conventionally in sliding mode control a swapping function is used in the form of a signum function. This is defined as:

$$\eta \frac{s}{s+\beta}, \tag{1.3}$$

This will produce a similar result, but with some smoothing which can help to reduce the chattering in the system.

1.4.3 Artificial Potential Field

The artificial potential field (APF) method is a common approach to collision and obstacle avoidance. At its core, it has two main elements: an attractive field at the goal point, and a repulsive field around each obstacle in an environment. This approach can be deployed either as a global approach, where it is often used for path planning, or as a local approach, where it is simply used for avoidance of obstacles in the immediate surroundings. The ease of implementation makes it a popular choice, especially for 2D cases such as mobile robots. In the typical formulation, the attractive field is defined as:

$$\vec{U}_A = k_A ||\vec{d_G}||^2, \tag{1.4}$$

where k_A is a gain which can be tuned for the attractive field, and $||\vec{d_G}||$ is the distance to the goal point. This field is used to guide an agent from its current position to the goal point. This is squared, which leads to massive attractive fields at a distance, and a very soft field near the goal point. Many other variants have been implemented, but this remains the most popular.

For the robots being used, it is worth highlighting this large potential field has a fairly low limit, as their maximum velocities are quite low. This means at most distances, provided there is no obstacle in the way, the agents will travel at or near their maximum velocities. The repuslive field is then defined as:

$$U_R = k_R \left(\frac{1}{||P_o - P_a||} - \frac{1}{P_r} \right)^2, \tag{1.5}$$

where k_R is a gain which can be tuned for the repulsive field, $||P_o - P_a||$ is the magnitude of the distance between the obstacle and the agent, and P_r is an additional protective distance between the obstacle and agent.

A simple visual representation can be seen in Fig.1.4. In this case, there is a global attractive field which will pull an agent from a starting point of (0,0) towards the goal point. However, at approximately (12,12) is an obstacle. This creates a local repulsive field. This acts in the area of a circle with a 4 meter radius from the center of the obstacle. The black line roughly traces out a path that an agent would travel along based on this configuration of obstacle and goal point. The forces from these two fields are interpreted by the agent, and applied to the controller which directs it through the environment.

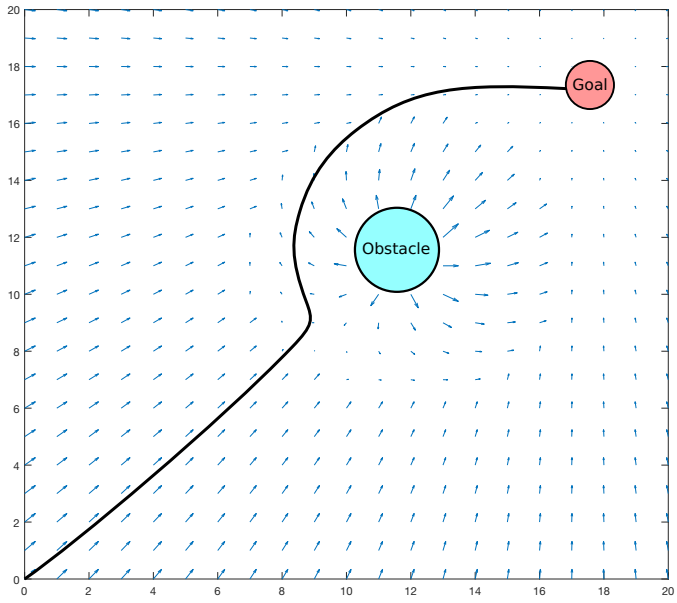

Figure 1.3: A visual representation of an agent's path resulting from APF.

Depending on the exact configuration of the obstacle, and the initial position of the ball, it would be possible for the ball to get stuck on the slopes around the obstacle instead of making it to the final goal. This is referred to as a "local minimum" and is a common problem with APF avoidance algorithms, particularly in more complex environments. We can conceptualize a local minimum by looking at a case with three obstacles. Here, a set of arrows are plotted indicating the direction which the potential fields would "push" an agent.

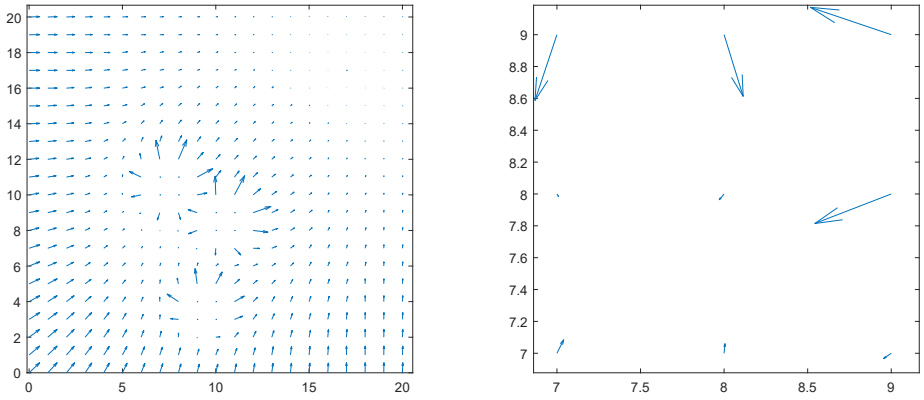

Figure 1.4: An example of a local minimum in a case with 3 obstacles.

For this case, at around the point (8,8) the agent is repelled backwards, and the forces in all directions will roughly converge to that point. Depending on the initial conditions of the agent, it is then possible for it to become trapped by the potential fields of the different obstacles. Even if the agent does not become completely trapped however, it can still often take more time than would otherwise be necessary as the agent works around an obstacle. As mentioned, chattering problems in APF algorithms are fairly common.

This approach can also be implemented for quadrotors in 3D-space. In general, these cases tend to use a cylindrical field around the quadrotor itself, and then create the forces based on the location of obstacles in that field.

Chapter 2

Experimental Setup

This chapter will discuss the hardware and software used in the simulated and experimental tests which were completed as part of the thesis.

2.1 Mobile Robots

The mobile robots used in this work are the Turtlebot3. This robot uses a very customizable framework in order to allow a variety of sensors to be added to the robot. The Turtlebot3 is a 2WMR. It has two independently powered wheels, as well as a ball bearing which contacts the ground for stability. The robot has two inputs: the power to each wheel. However, for the purpose of control, it is often more meaningful to consider the robot in terms of its linear and angular velocities. As such, a layer exists between the robot and the controller which converts a linear velocity signal and an angular velocity signal into the powers for the two wheels.

The 2WMRs are non-holonomic. In a holonomic system, it is possible to determine the state of the system based on a set of position changes within the system, regardless of when they occurred or the order they occurred in. For a 2WMR, the forward motion of the robot will have a different effect on the position of the agent depending on the current heading of the agent, meaning the prior actions will impact the effect of the current actions. This leads to additional complexities in control compared to a holonomic system where steps can be issued in whatever order is convenient.

Within the model there are two coordinate systems to consider. The first is the global axes of x and y, which dictate the position and motion of the agent within a globally defined area. In addition, a local coordinate system, x_l and y_l is also used. These axes are relevant for the general control of the system, as linear motion can be considered to occur along the x_l axis. The x_l-y_l plane exists at an angle θ to the global plane. This means θ is the heading of the robot relative to the global plane. Fig.2.1 shows these two coordinate systems for the 2WMR used in this work.

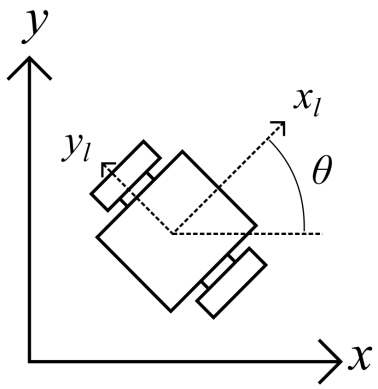

Figure 2.1: The two coordinate systems used for the 2WMR.

The Turtlebot3 comes in two primary form factors: the "Burger" (Fig.2.2(a)) and the "Waffle" (Fig.2.2(b)). The default configuration of the Burger is a 4-stage platform, while the Waffle is a 3-stage platform, but with twice the width and length as the Burger. These two are similarly controlled, however the maximum velocities of the two vehicles differ slightly. The Burger has a maximum linear velocity of 0.22 m/s and a maximum angular velocity of 2.7 rad/s. The Waffle has a maximum linear velocity of 0.26 m/s and a maximum angular velocity of 1.8 rad/s. The primary difference here is the angular velocity, which is notably lower for the Waffle as a result of the wheels being much further apart, requiring more revolutions of the tires in order to complete a turn. For all testing involving both types of agents, the lower thresholds were set as maximums to ensure all agents would have the same basic dynamics. This means the maximum linear velocity is 0.22 m/s and the maximum angular velocity is 1.8 rad/s.

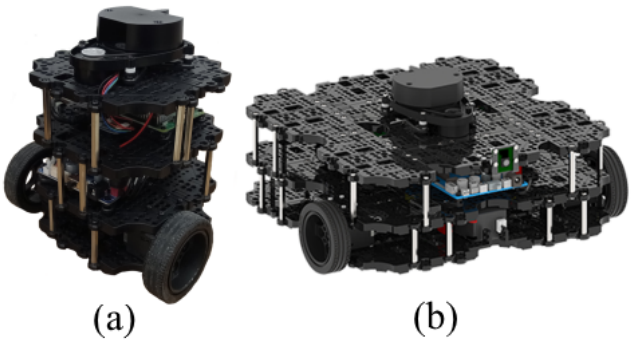

Figure 2.2: (a) The Turtlebot3 Burger and (b) The Turtlebot3 Waffle.

The Turtlebot3 uses the LDS-01 360 degree LiDAR sensor to detect its environment. This rotates at a rate of 5 rev/s, meaning the measurement at each angle is updated every 0.2 seconds. The sensor provides readings in the range from 120 to 3500 mm. The LDS-01 has a 1 degree angular resolution, meaning it provides 360 separate measurements for each revolution. They are also able to track velocity and changes in position using an inertial measurement unit (IMU).

The standard kinematic model of the mobile robots is based on Fig.2.1. The linear velocities $\dot{x}$ and $\dot{y}$, and the angular velocity $\dot{\theta}$ are then calculated as:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} u_l \cos(\theta) \\ u_l \sin(\theta) \\ u_\theta \end{bmatrix}, \tag{2.1}$$

where u_l is the linear velocity control input and u_θ is the angular velocity control input. As previously noted, the system is non-holonomic, and so we see the linear velocities are dependent on the current angle, θ of the agent.

From this, one control scheme which is deployed is referred to as a "hand-position model". With this, a position directly in front of the mobile robot, called the "hand-position," is controlled. This point will be a distance L from the centroid along the positive x_l axis. This is shown in Fig.2.3 with the hand-position denoted in red.

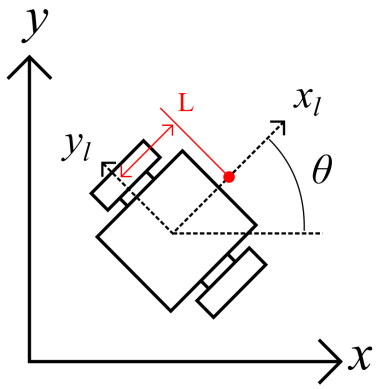

Figure 2.3: The hand-position model for a 2WMR.

From here we define two new control inputs u_x and u_y. These define the x and y motion of the hand-position. We can then convert directly between the hand-position control inputs and the kinematic model control inputs by saying:

$$\begin{bmatrix} u_l \\ u_\alpha \end{bmatrix} = \begin{bmatrix} \cos(\alpha) & \sin(\alpha) \\ -\frac{1}{L}\sin(\alpha) & \frac{1}{L}\cos(\alpha) \end{bmatrix} \begin{bmatrix} u_x \\ u_y \end{bmatrix}. \tag{2.2}$$

This mapping is used for the classic SMC deployed in Chapter 4.2. The subsequent TSMC controller in Chapter 4.3 uses the conventional model.

2.2 Quadrotors

For the quadrotors, the Crazyflie 2.1 was used. This lightweight quadrotor has a sizeable hobbyist community established, as well as packages which can be developed from. The Crazyflie 2.1 weighs 27 grams, and does not have an onboard controller. As a result, it is controlled using the CrazyRadio, an antennae that connects by USB to a computer or laptop. One CrazyRadio can send signals to up to two different Crazyflie's at once. Both are pictured in Fig.2.4.

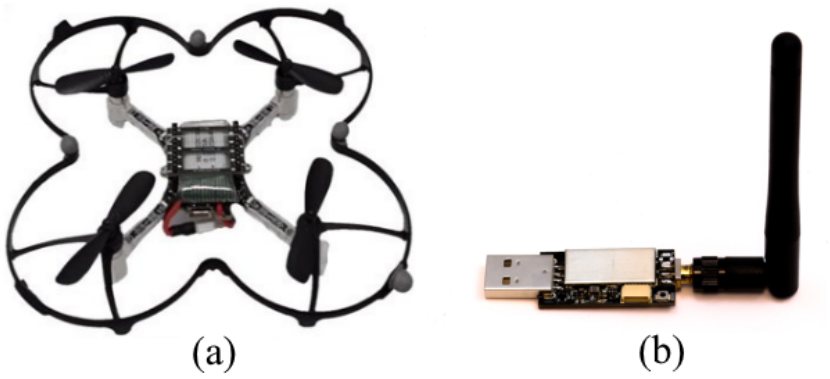

Figure 2.4: The Crazyflie 2.1 and the CrazyRadio antenna.

The general quadrotor model considers 6 degrees of freedom (DOFs): the linear acceleration of the agent in the x, y, and z directions, and the rotational acceleration of the agent about the θ, ϕ, and ψ directions. Each propeller has a thrust T_i, where i corresponds to the number of the propeller. The basic formulation of the model is presented in Fig.2.5.

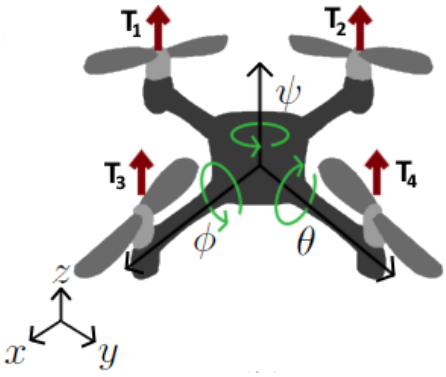

Figure 2.5: The general model of a quadrotor.

From this, the standard model of the quadrotor is defined as:

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \\ \ddot{\theta} \\ \ddot{\phi} \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} \frac{T}{m_q}(C_\psi S_\theta C_\phi + S_\psi S_\phi) \\ \frac{T}{m_q}(S_\psi S_\theta C_\phi + C_\psi S_\phi) \\ \frac{T}{m_q} C_\theta C_\phi - g \\ \dot{\phi}\dot{\psi}(\frac{J_y - J_z}{J_x}) + \frac{l}{J_x}\tilde{\tau}_\theta \\ \dot{\theta}\dot{\psi}(\frac{J_z - J_x}{J_y}) + \frac{l}{J_y}\tilde{\tau}_\phi \\ \dot{\theta}\dot{\phi}(\frac{J_x - J_y}{J_z}) + \frac{l}{J_z}\tilde{\tau}_\psi \end{bmatrix}, \tag{2.3}$$

where S and C are used to refer to the sine and cosine functions of the angle specified in the subscript. T is the total thrust of the motors, m_q is the mass of the UAV, g is the acceleration due to gravity, τ_θ, τ_ψ, and τ_ϕ are the torques about each axis, and J_x, J_y, and J_z are the polar moment of inertia.

From [93], a linearized model for the Crazyflie 2.1 has been developed and tested. This is defined as:

$$\dot{\mathbf{x}}_l = A_l \mathbf{x}_l + B_l \mathbf{u}_l, \tag{2.4}$$

where

$$\mathbf{x}_l = \begin{bmatrix} x & \dot{x} & y & \dot{y} & z & \dot{z} & \psi & \theta & \phi \end{bmatrix}^T,$$

and the matrices A_l, B_l, and the inputs $\mathbf{u}_l$ are defined as:

$$
\begin{aligned}
A_l &= \begin{bmatrix}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 9.8 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -9.8 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -5.018 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -5.083
\end{bmatrix}, \\
B_l &= \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 5.233 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5.516
\end{bmatrix}^T, \\
\mathbf{u}_l &= \begin{bmatrix} \Sigma T - m_q g & \tilde{\tau}_\psi & \tilde{\tau}_\theta & \tilde{\tau}_\phi \end{bmatrix}^T.
\end{aligned}
$$

This linearization is a reasonable assumption when the rotation angles are small (less than 10 degrees) during operation of the Crazyflie quadrotor. The simplified quadrotor model for the x-direction is given as

$$
\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & g \\ 0 & 0 & -5.018 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \theta \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 5.233 \end{bmatrix} u. \tag{2.5}
$$

For the x-direction case in 2.5, $u = \tilde{\tau}_\theta$, and for the y-direction $u = \tilde{\tau}_\phi$. These dynamics are the same in the y-direction, so a generalized setup is used wherein: $\mathbf{x} = \begin{bmatrix} x & \dot{x} & \theta \end{bmatrix}^T$, $\mathbf{x} = \begin{bmatrix} y & \dot{y} & \phi \end{bmatrix}^T$. To generalize, a matrix such that $\mathbf{x} = \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}^T$ is used. Using the dynamic model for $\mathbf{x}$, the equations are defined such that

$$
\begin{cases}
\dot{x}_1 = x_2 \\
\dot{x}_2 = 9.8 x_3 \\
\dot{x}_3 = c_A x_3 + c_B u,
\end{cases} \tag{2.6}
$$

where c_A and c_B are the values associated with the angular velocity of the specified angle given in A_l and B_l for the given DOF.

2.3 ROS

Robot Operating System (ROS) is a Linux based middleware that is used for a variety of applications in the control of robots. It serves as a means of communicating between a computer and robots in the field. Additionally, simulation platforms have been developed which allow for easy transition from simulation to practical experiments. ROS is built on a framework of nodes, and allows for individual programs to publish information to nodes, as well as subscribe to nodes to retrieve information.

The basic structure of ROS has four main elements: the ROS Master, publishers, subscribers, and topics. The ROS Master is a central processor which manages all the ongoing operations. Topics contain information about the system. The information in a topic is provided by a publisher, and the information is then read by a subscriber. A simple framework can then be presented in Fig.2.6.

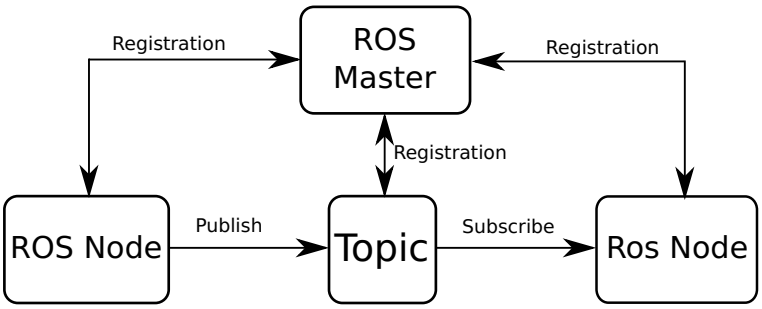

Figure 2.6: ROS communication between agents and software.

For experiments or simulations, we then have three main elements: ROS, the agents, and the control programs. Here, the agents and the programs both act as both publishers and subscribers. The agents will publish position, velocity, and sensor data, while the programs will publish control values to dictate how the agents should move. Conversely, the agents will subscribe to these control values and the programs will subscribe to the relevant data from the agents. We can present this framework in Fig.2.7.

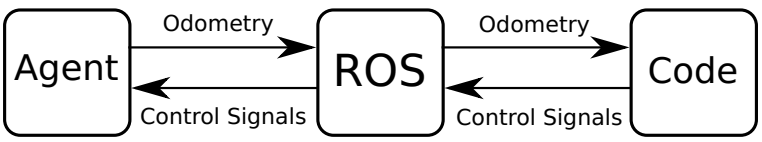

Figure 2.7: ROS communication between agents and software.

Within ROS there are visualization tools such as RQT Graph which are able to express topics and what nodes are publishing and subscribing to them. An example of the output from this visualization is presented in Fig.2.8 with a single Turtlebot interacting with a program. The Turtlebot here is named "tb3_0" and has three topics, "joint_states", "cmd_vel", and "odom". "odom" refers to the current position and velocity information for the agent, which is passed along to a node called "Leader_Controls". This node collects this data and processes it to provide control signals for a leader agent. The calculated control inputs are then sent to "cmd_vel" where they are then passed along to the simulation environment, Gazebo. From there, Gazebo updates the odometry as well as the joint states of the robot.

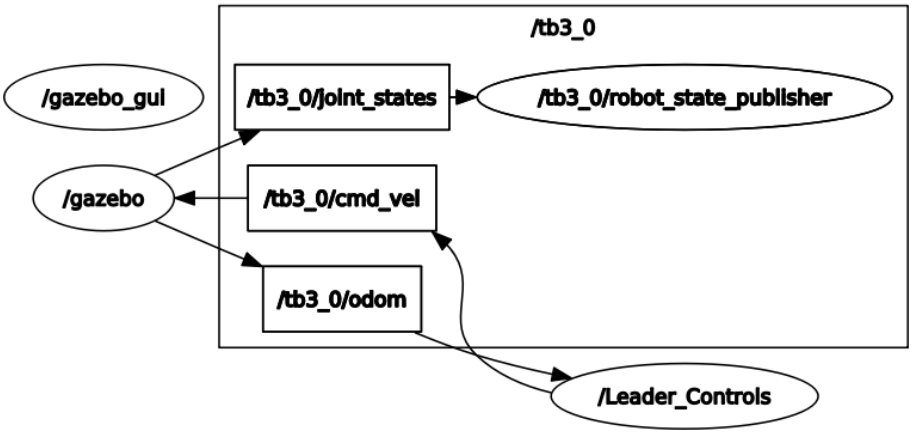

Figure 2.8: ROS communication as demonstrated by RQT Graph

All Turtlebot simulations conducted in this thesis were completed using the ROS distribution "Melodic Morenia". This is the ROS version which is used for Ubuntu 18.04 and had long term service running until 2023. This is part of ROS1, which recieved its last version in 2020, with ROS2 being the new focus for the community.

2.4 Simulation Environment

The primary simulation platform used was Gazebo. This allows the use of existing models within a 3D environment, as well as the creation of obstacles. This is used for the Turtlebot3 mobile robots as these already have a model designed for the space. An example of 4 Turtlebots in Gazebo is shown in Fig.2.9.

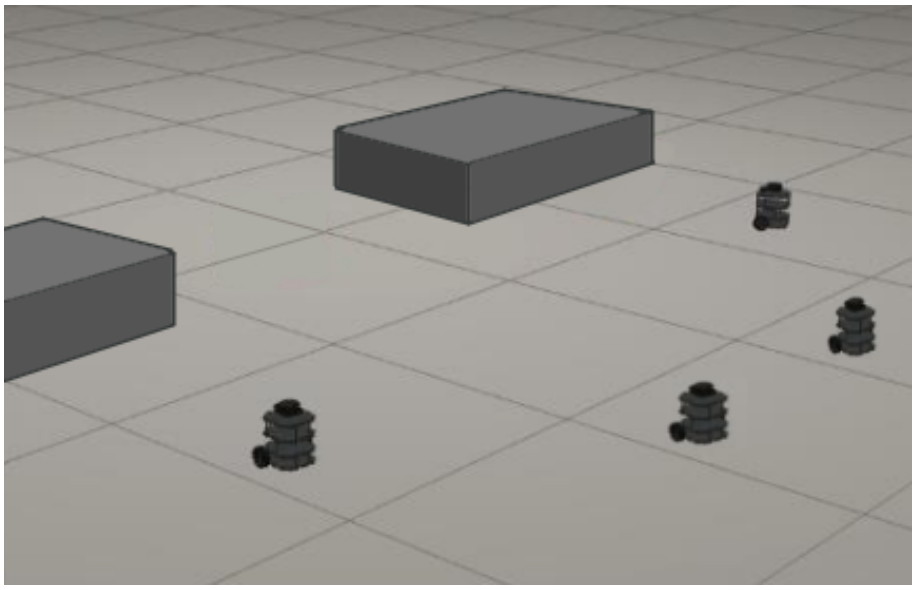

Figure 2.9: Four Turtlebot3 agents in a gazebo environment.

For the quadrotors, a MATLAB simulator is used instead. There was not a sufficiently accurate model of the Crazyflie 2.1 available for Gazebo, and so the mathematical model from [93] is used instead. This is a workable approach, however it does mean that it becomes necessary to run both programs at the same time in order to run heterogeneous simulations. MATLAB has ROS integration within it, so for a heterogeneous system with both agent types, MATLAB would subscribe and publish to nodes as necessary. Additionally, as the Crazyflie does not have any sensing capability, it was not a concern that it was not in the Gazebo environment, as it would not have been directly interacting with other agents and obstacles regardless.

2.5 Motion Tracking Cameras

Since the Crazyflie has no onboard odometry, a motion capture system is used to track the position of the agent in a fixed space. These provide a very precise position estimate and can be used with multiple robots simultaneously. Typical results will have mean errors of less than 1 mm.

A series of up to 8 cameras are used to track the motions of the agents within a set space. At least 3 cameras must be able to see each of the markers for sufficiently optimized tracking, although with a greater number of cameras a more accurate estimate of the position can be acheived.

The system needs to be initialized using a wand with 3 markers on it. This is waved through the environment, with the cameras identifying where each marker is in its frame of view. Once a sufficient number of data points have been collected, an algorithm calculates the relative positions of the cameras. Once this has been completed, a small triangle is placed at the desired origin, and this is used to align the axes of the space being used. Results from the wanding are shown below in Fig.2.10(a) and the ground plane alignment is shown in Fig.2.10(b). Each line from the wanding represents a data sample, and in this case each camera has between 2000 and 3000 samples. The ground plane tracking shows a 3D environment, with each camera being represented by a pyramid shape, and the reference tracker being the three orange dots in the center.

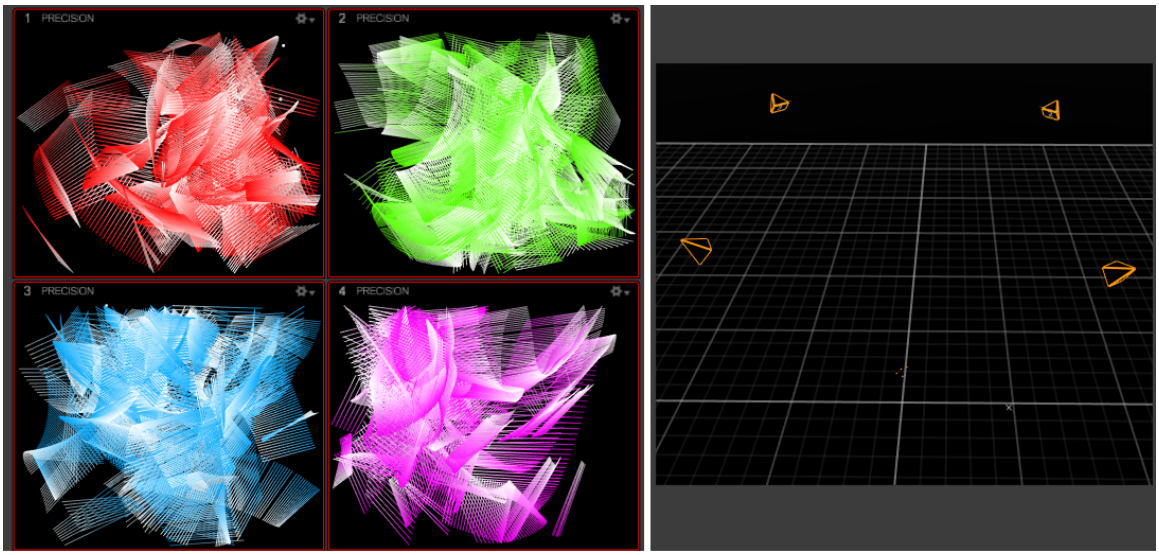

Figure 2.10: (a) A demonstration of wanding and (b) tracking the ground plane in OptiTrack.

The mo-cap software can then output the positions of objects with markers on them. A greyscale image showing a single agent captured by four cameras is shown in Fig.2.11(a) while the "Precision Mode" which tracks the markers is shown in Fig.2.11(b). Highly reflective material should not enter the testspace during setup or experiment to produce optimal results, as it will interfere with the ability of the cameras to see the tracking markers.

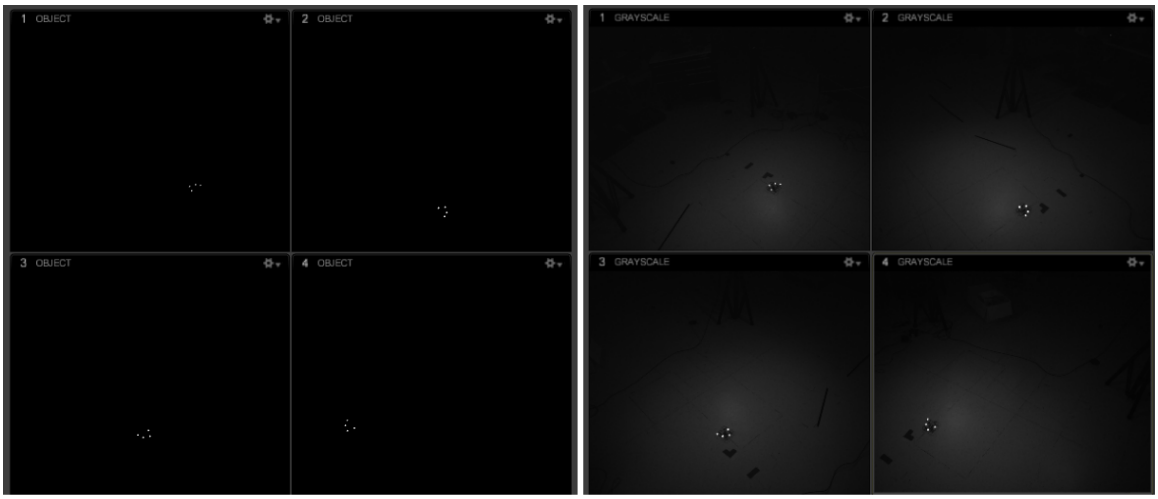

Figure 2.11: (a) The cameras on Precision Mode and (b) Greyscale Mode

Once this is completed, a set of markers are highlighted within the Optitrack software, and are identified as a "Rigid Body". In doing so, these markers are treated as a single object at a point which occurs at the average value of the markers. Fig.2.12 shows an example of a set of markers treated as a rigid body by the software.

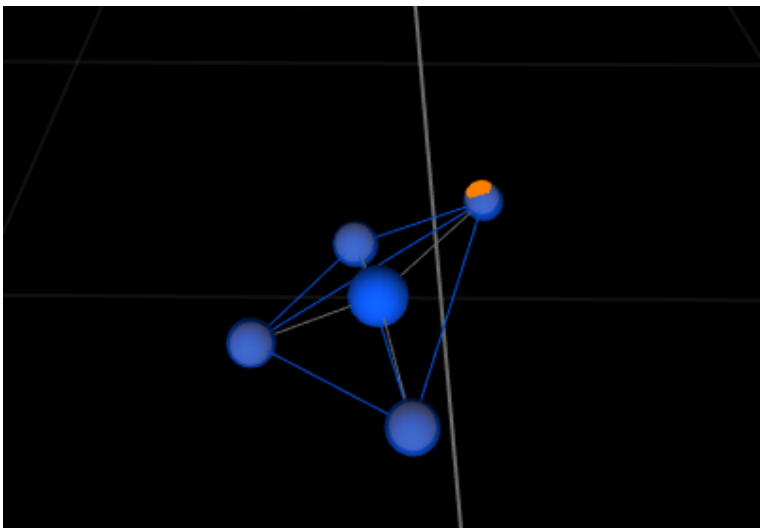

Figure 2.12: An example of a rigid body in Optitrack.

This information is then broadcast by OptiTrack. ROS recieves this information, and it can then be sent to agents and controllers. Two separate computers are required for this process, as the mo-cap software runs in Windows, but ROS is designed for Ubuntu. The communication for this is completed over a wireless network.

2.6 Flight Space

The experimental test space is roughly 5 meters wide and 5 meters deep, however this is reduced to about 4 meters by 4 meters when the curtain is drawn for UAV use. The UGVs travel slow and do not have serious capacity to injure, but the Crazyflies are capable of suddenly changing direction, and if there is an error this can result in injuries, particularly to the eye. Safety goggles are required in the flight space while quadrotors are present.

The actual usable test space then becomes smaller as a result of the cameras and computer necessary to run all the motion capture, as well as the limitations of the visible range of the cameras themselves. Fig.2.13 shows the test space with the curtains drawn and four Turtlebots set up with a series of obstacles.

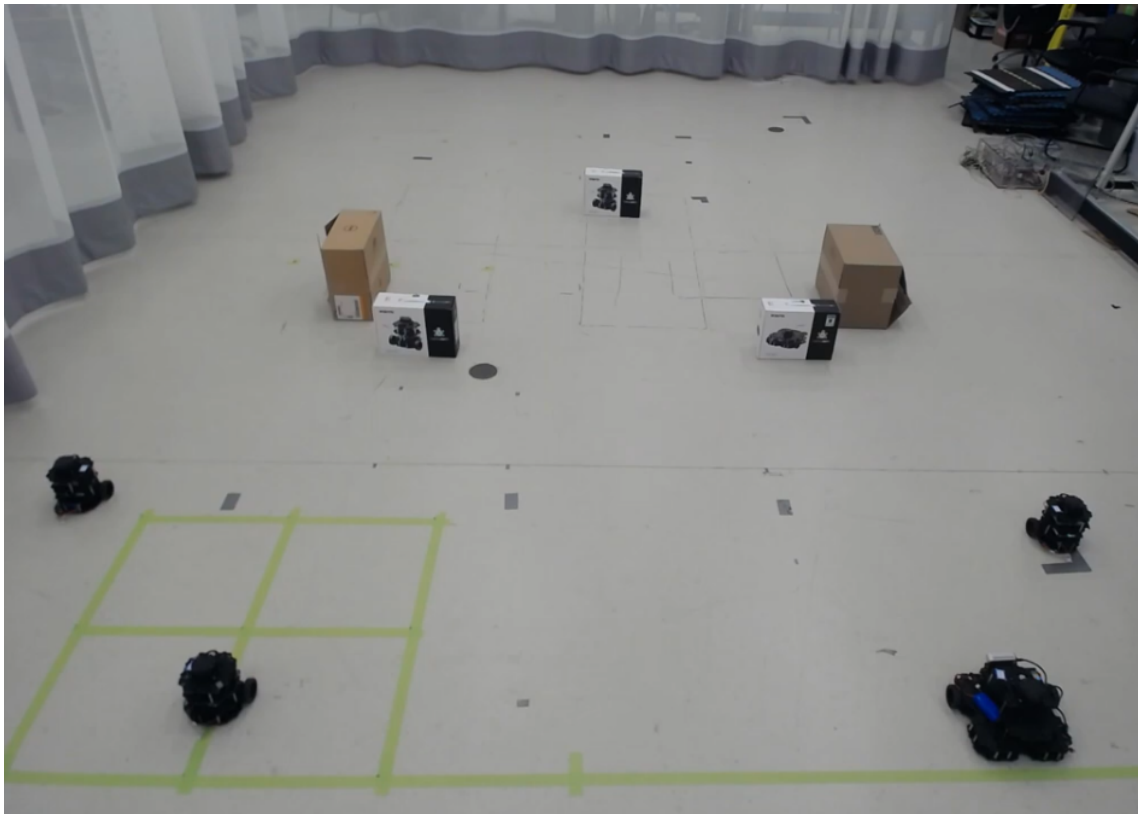

Figure 2.13: The flight space used for experimental testing.

Chapter 3

Formation Control

The first major research objective was developing controllers for the quadrotors and the mobile robots. These are designed to be functional for teams of agents operating in an empty environment.

3.1 Objective and Assumptions

This chapter presents two controllers for quadrotor UAVs and 2WMR UGVs. The first of these is the development of a simple SMC controller to control the agents, however a subsequent TSMC controller was developed which proved to be more robust. The first goal of these controllers is for individual agents to move to a designated point based on the current errors of the agent.

From this, the end goal of the controllers is to develop heterogeneous formations based on errors derived from position information of neighbouring agents within a communication framework. This will allow for multiple agents to achieve a formation and then move as a group. This chapter will present the two controllers, as well as their stability proofs, then present a number of simulations validating their robustness.

Assumption 3.1 *There are no communication constraints on the agents. Agents are able to communicate without packet losses, time delays, corrupted data, or other constraints.*

Assumption 3.2 *There are no ground effects present on the quadrotors. Ground effects are caused by the winds from the propellers contacting the ground, wall, or ceiling and being reflected back at the quadrotor. These effects are ignored in the controller.*

Assumption 3.3 *The angles of the quadrotor agents must remain small during flight, allowing the assumptions of the linear model to be sufficiently accurate.*

Assumption 3.4 *The vertical control of the quadrotor is not considered in the controllers. This is handled instead using a separate PID controller which is not part of this work. Within the linearized model, z and $\dot{z}$ do not interact with the other factors, so as long as the angles remain small, this approximation will hold.*

3.2 Classical Sliding Mode Control

Two SMCs are developed in this chapter: one for the 2WMR and one for the quadrotor. These will both use information from neighbouring agents in order to achieve consensus and formation. The controller for the quadrotor is presented first, followed by the controller for the mobile robot. Both agents use linearized models in this derivation.

Note that for these controllers the x- and y-directions are nearly identical subsystems, and so the controllers will be presented for one direction. This controller is then separately applied to the control of each direction.

3.2.1 Quadrotor Controller

The controllers for the x- and y-direction sub-systems of the quadrotor are defined as

$$u_x = \frac{1}{\gamma}(-c_1\dot{e}_x - c_2 g\dot{e}_{\dot{x}} - c_3\dot{e}_\theta + \gamma\dot{x}_{\dot{\theta},d} - \eta\frac{s_x}{|s_x| + \beta}), \tag{3.1}$$

$$u_y = \frac{1}{\gamma}(-c_1\dot{e}_y - c_2 g\dot{e}_{\dot{y}} - c_3\dot{e}_\psi + \gamma\dot{x}_{\dot{\psi},d} - \eta\frac{s_y}{|s_y| + \beta}), \tag{3.2}$$

where the values $\dot{x}_{\dot{\psi},d}$ and $\dot{x}_{\dot{\theta},d}$ refer to the desired values of the angular velocities, and the variables c_1, c_2, c_3, η and β are design variables. Let $c_1/c_2 = c_3$. γ is the sum of values a_{ij} for values of j ranging from 1 to m. The smoothing function

$$\eta\frac{s_x}{|s_x| + \beta},$$

is used in place of the conventional signum function. Fig.3.1 plots this for values of s_x when $\eta = 1$ and $\beta = .001$.

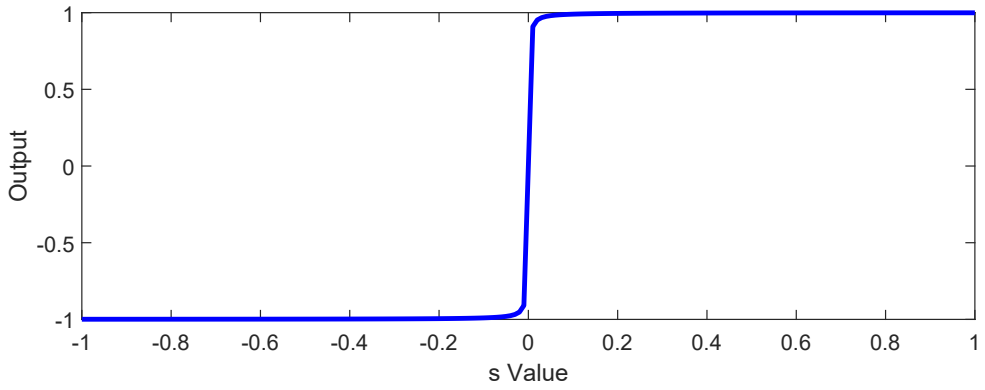

Figure 3.1: The output of the smoothing function based on the value of s.

Depending on the update time of the system, the β value can be adjusted to facilitate this smoothing. The larger the time between updates, the larger β should be. η can also be tuned to increase or decrease the effect of the switching function.

In the stability proof, only the x-subsystem will be demonstrated, as the proof for the y-subsystem will be similar. The sliding surface is defined as

$$s = c_1 e_{x_1} + c_2 e_{x_2} + c_3 e_{x_3} + e_{x_4}, \tag{3.3}$$

where the errors are defined as

$$\begin{cases} e_{x_1} & = \sum\limits_{\substack{j=1 \\ j\neq i}}^{m} a_{ij}(x_{1,i} - (x_{1,j} + \delta_{ij})) \\ & \quad + \sum\limits_{\substack{j=m+1 \\ j\neq i}}^{n} a_{ij}(x_{1,i} - (x_{1,j} + \delta_{ij})) \\ e_{x_2} & = \sum\limits_{\substack{j=1 \\ j\neq i}}^{m} a_{ij}(x_{2,i} - x_{2,j}) \\ e_{x_3} & = \sum\limits_{\substack{j=1 \\ j\neq i}}^{m} a_{ij}(x_{3,i} - x_{3,j}) \\ e_{x_4} & = \sum\limits_{\substack{j=1 \\ j\neq i}}^{m} a_{ij}(x_{4,i} - x_{4,j}). \end{cases} \tag{3.4}$$

Here the values $a_{i,j}$ are from Equation 1.1. Note that for e_{x_1}, the values δ_{ij} refer to the distance between the ith and jth agents in the desired formation. As the communication is undirected, a desired distance between the ith agent and the jth

agent of δ_{ij} will have a complementary distance of $-\delta_{ij}$ between the ith agent and the jth agent. Agents 1 to m are quadrotors, and $m+1$ to n are 2WMR.

Then, the time-derivative of s is

$$\dot{s} = c_1\dot{e}_{x_1} + c_2\dot{e}_{x_2} + c_3\dot{e}_{x_3} + \dot{e}_{x_4}, \tag{3.5}$$

where

$$\begin{aligned} \dot{e}_{x_4} &= \sum_{\substack{j=1 \\ j\neq i}}^{m} a_{ij}(\dot{x}_{4,i} - \dot{x}_{4,j}) \\ &= \sum_{\substack{j=1 \\ j\neq i}}^{m} a_{ij}(u - \dot{x}_{4,j}) \\ &= \gamma u - \sum_{\substack{j=1 \\ j\neq i}}^{m} a_{ij}(\dot{x}_{4,j}). \end{aligned} \tag{3.6}$$

Using the controllers (3.1) and (3.2), and from (3.4) we get

$$\dot{x}_4 = -c_1\dot{e}_{x_1} - c_2 g\dot{e}_{x_2} - c_3\dot{e}_{x_3} + \dot{x}_{4,d} - \eta\frac{s}{|s|+\beta}. \tag{3.7}$$

Moving the terms of $-c_1\dot{e}_{x_1} - c_2 g\dot{e}_{x_2} - c_3\dot{e}_{x_3}$ to the left-hand side, we have

$$c_1\dot{e}_{x_1} + c_2 g\dot{e}_{x_2} + c_3\dot{e}_{x_3} + \gamma\dot{x}_4 - \dot{x}_{4,d} = -\eta\frac{s}{|s|+\beta}. \tag{3.8}$$

Now the left-hand side is equal to $\dot{s}_x$ as in (3.5), that is,

$$\dot{s} = -\eta\frac{s}{|s|+\beta}. \tag{3.9}$$

Choosing the Lyapunov function candidate as $V_x = \frac{1}{2}s_x^2$. Taking the time-derivative of V_x gives

$$\dot{V}_x = s\dot{s} = -s\eta\frac{s}{|s|+\beta} = -\eta\frac{s^2}{|s|+\beta} \leq 0. \tag{3.10}$$

This proves that $s \to 0$ can be ultimately achieved in finite time.

However, $s \to 0$ does not necessarily mean that $e_{x_1} \to 0$ and $e_{x_3} \to 0$. When $s = 0$, from (3.3), we have

$$\dot{x}_4 = -c_1\dot{e}_{x_1} - c_2 g\dot{e}_{x_2} - c_3\dot{e}_{x_3} + \dot{x}_{4,d}, \tag{3.11}$$

that is,

$$\dot{e}_{x_4} = -c_1\dot{e}_{x_1} - c_2 g\dot{e}_{x_2} - c_3\dot{e}_{x_3}. \tag{3.12}$$

From the model of the Crazyflie we can say that $\ddot{e}_{x_1} = ge_{x_3}$. Taking the time derivative and applying this leads to

$$\begin{aligned} \dddot{e}_{x_4} &= -c_1\dddot{e}_{x_1} - c_2 g\dddot{e}_{x_2} - c_3\dddot{e}_{x_3}, & (3.13)\\ \dddot{e}_{x_4} + c_3\dddot{e}_{x_3} &= -c_1 g e_{x_3} - c_2 g\dot{e}_{x_3}, & (3.14)\\ \dddot{e}_{x_4} + c_3\dddot{e}_{x_3} &= -c_2 g(\frac{c_1}{c_2}e_{x_3} - \dot{e}_{x_3}). & (3.15) \end{aligned}$$

As $c_1/c_2 = c_3$, we have

$$\dddot{e}_{x_4} + c_3\dddot{e}_{x_3} = -c_2 g(c_3 e_{x_3} - \dot{e}_{x_3}). \tag{3.16}$$

Let $f(t) = c_3 e_{x_3} + \dot{e}_{x_3}$, then $\ddot{f}(t) = -c_2 g f(t)$. We can state that $f(t) = 0$. From (3.3), we have

$$\begin{aligned} c_1 e_{x_1} + c_2 e_{x_2} + c_3 e_{x_3} + e_{x_4} &= 0,\\ c_1 e_{x_1} + c_2 e_{x_2} &= -c_3 e_{x_3} - e_{x_4}. \end{aligned} \tag{3.17}$$

Therefore, as $f(t) = c_3 e_{x_3} + e_{x_4} = 0$ holds, we have

$$\begin{cases} c_1 e_{x_1} + c_2 e_{x_2} = 0 \\ c_3 e_{x_3} + e_{x_4} = 0, \end{cases} \tag{3.18}$$

or

$$\begin{cases} e_{x_2} = -\frac{c_1}{c_2} e_{x_1} = -c_3 e_{x_1} \\ e_{x_4} = -c_3 e_{x_3}. \end{cases} \tag{3.19}$$

Then we construct another Lyapunov function candidate as

$$V_{e,x} = \frac{1}{2}e_{x_1}^2 + \frac{1}{2}e_{x_3}^2.$$

Hence the derivative of $V_{e,x}$ is

$$\begin{aligned} \dot{V}_{e,x} &= e_{x_1}\dot{e}_{x_1} + e_{x_3}\dot{e}_{x_3}\\ &= e_{x_1}(-c_3 e_{x_1}) + e_{x_3}(-c_3 e_{x_3})\\ &= -c_3(e_{x_1}^2 + e_{x_3}^2) \leq 0. \end{aligned} \tag{3.20}$$

As a result, $e_{x_1} \to 0$ and $e_{x_3} \to 0$ can be achieved.

3.2.2 Mobile Robot Controller

For the mobile robot, a hand position model is used

$$\begin{bmatrix} \dot{h}_x \\ \dot{h}_y \end{bmatrix} = \begin{bmatrix} u_x \\ u_y \end{bmatrix}. \tag{3.21}$$

The x- and y-direction control inputs u_x and u_y are designed as

$$u_x = \sum_{j=1}^{n} \dot{x}_j - \eta \frac{s_x}{|s_x| + \beta}, \tag{3.22}$$

$$u_y = \sum_{j=1}^{n} \dot{y}_j - \eta \frac{s_y}{|s_y| + \beta}. \tag{3.23}$$

For the controllers in each direction, such that $x_j = h_{x,j}$ and $y_j = h_{y,j}$. Using the dynamic model for x

$$\dot{x_j} = u. \tag{3.24}$$

The sliding surface is then constructed as

$$s_m = e_x, \tag{3.25}$$

where s_m is an arbitrary sliding surface which can represent either the sliding surface for the x-direction or y-direction. The error values for this case is defined as

$$\begin{aligned} e_x \quad &= \sum_{j=1}^{m} a_{ij}(x_i - x_j + \delta_{ij}) \\ &+ \sum_{j=m+1}^{n} a_{ij}(x_i - x_j + \delta_{ij}). \end{aligned} \tag{3.26}$$

Then, the time-derivative of s_m is

$$\dot{s}_m \quad = \quad \dot{e}_x = \gamma \dot{x} - \sum_{j=1}^{n} \dot{x}_j. \tag{3.27}$$

From (3.22) and (3.26), we get

$$\dot{x} = \frac{1}{\gamma}(\sum_{j=1}^{n} \dot{x}_j - \eta \frac{s_m}{|s_m| + \beta}). \tag{3.28}$$

By moving the terms of $\dot{x}_d$ to the left-hand side, we have

$$\gamma\dot{x} - \sum_{j=1}^{n} \dot{x}_j = -\eta \frac{s_m}{|s_m| + \beta}. \tag{3.29}$$

Now the left-hand side is equal to $\dot{s}_x$ as in (3.27), that is,

$$\dot{s}_m = -\eta \frac{s_m}{|s_m| + \beta}. \tag{3.30}$$

Choose the Lyapunov function candidate as $V_x = \frac{1}{2}s_x^2$. Taking the time-derivative of V_x gives

$$\dot{V}_x = s_m \dot{s}_m = -s_m \eta \frac{s_m}{|s_m| + \beta} = -\eta \frac{s_m^2}{|s_m| + \beta} \leq 0. \tag{3.31}$$

This proves that $s_m \to 0$ can be ultimately achieved.

For the mobile robot, since $s_m = e_x$, as the separate values $s_x, s_y \to 0$, the errors $e_x, e_y \to 0$.

3.2.3 Quadrotor Simulations

A simulation is presented in which a quadrotor tracks a spiral trajectory. This will serve as a test of the ability of the agent to follow a goal point, and adjust in two dimensions. Fig.3.2 plots the results of this test.

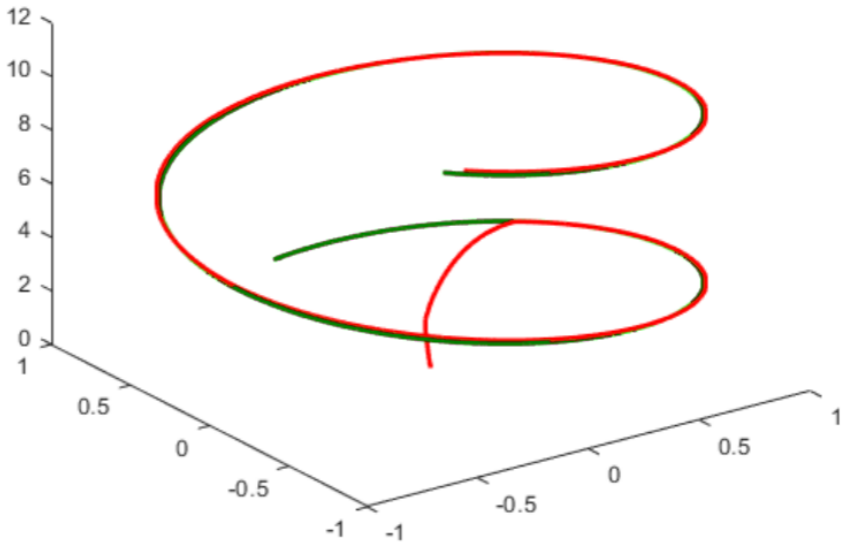

Figure 3.2: A quadrotor tracking a spiral with SMC.

It can be seen that the SMC does an effective job of tracking the overall trajectory. Starting from a position of (0,0,0), it is able to quickly meet the spiral, and does not have any significant errors after this point. There are still some minor discrepancies between the goal and the agent, but this is an acceptable level of performance.

3.2.4 Mobile Robot Simulations

For the mobile robots, a simulation was completed in which a mobile robot tracks to a specific position. From an initial position of (0,0) while facing in the direction of the positive x-axis, the agent moves toward a position of (0,2). The result of this simulation is presented in Fig.3.3. The agent has a smooth trajectory and is able to reach the goal point without any problems.

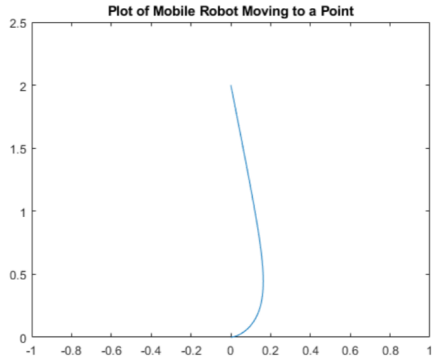

Figure 3.3: A mobile robot tracking to a desired position.

3.2.5 MAS Simulations

A series of consensus and formation simulations are conducted to verify the effectiveness of the controller. Simulations were conducted using MATLAB and Simulink. , as shown in Fig.3.4, is applied for all simulations. This topology includes a virtual leader (indicated in Fig.3.4 with VL) which will identify the desired point of consensus, or the center of the desired formation, depending on the simulation. This is not itself a physical agent. The agents consist of three UGVs and three UAVs. For the design constants, they are selected as $\eta = 0.1$, $\beta = 0.001$, and $c_1 = c_2 = c_3 = 1$.

To begin, a consensus simulation is conducted with the six agents being placed at random locations in an environment. The virtual leader in this case has a value of 0 for the x-direction. These results were used to initially validate the controller, and

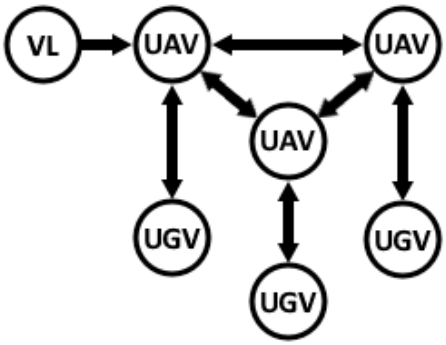

Figure 3.4: Communication topology for six agent simulation, including virtual leader.

would be for a formation in which all values of $\delta_{ij} = 0$. As shown in Fig.3.5, with the sliding mode controller implemented, the agents converged to a single point in approximately 50 seconds.

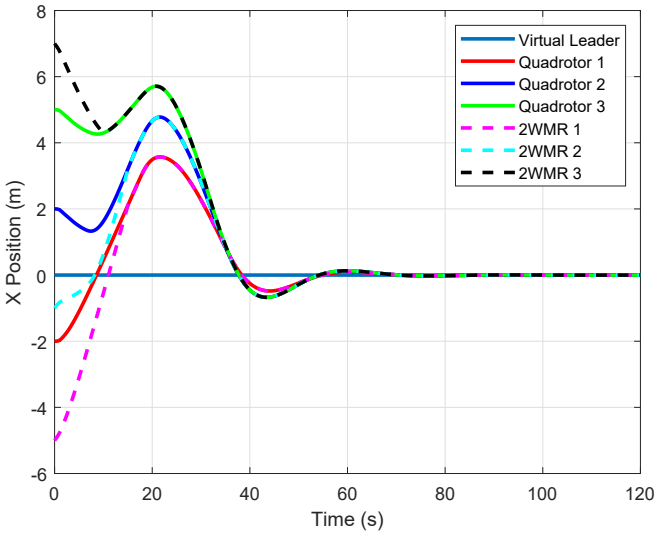

Figure 3.5: Consensus of six agents, showing the motion in the x-direction

Following this result, a formation was attempted, with each mobile robot following a specific quadrotor, and the three quadrotors forming a triangle. The formation is also set up by way of the virtual leader such that Quadrotor 1 (indicated in red) will reach a position of (-5,-10) in the X-Y plane. The convergence time for the six agents to achieve the desired formation in the desired position is approximately 65 seconds. The results of this simulation are shown in Fig.3.6.

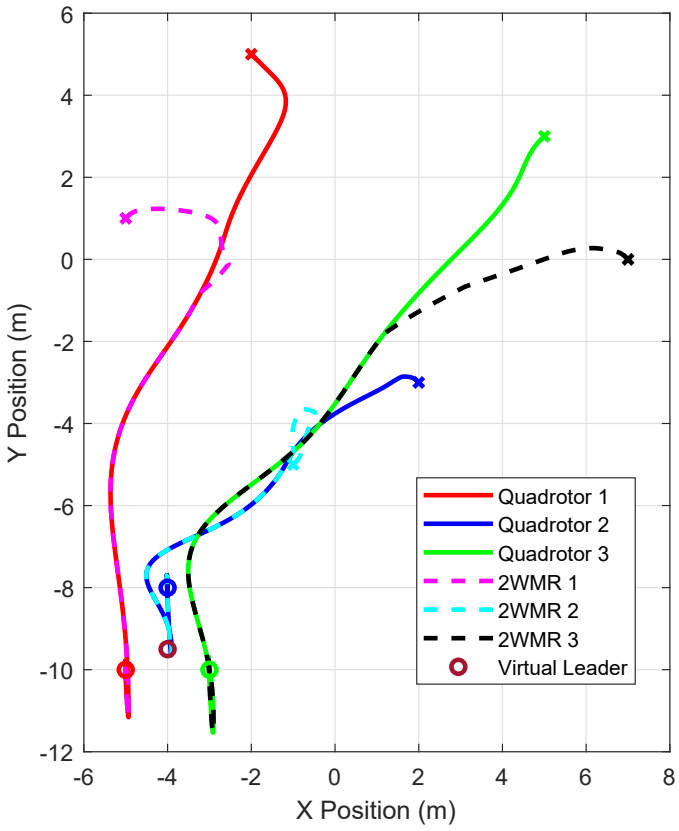

Figure 3.6: Six agents making triangle formation: X's indicate starting position, O's indicate final position

A number of factors were also tracked for this result and are presented in Fig.3.7. These numbers are specific to the three drones in the simulation. The control inputs in both directions are presented first, indicating that there is some chattering in these results. It mostly occurs at the beginning, but this is notable because it can lead to instabilities in some cases. Here the chattering quickly converges, but it is worth keeping an eye on in future testing just in case.

The formation errors also show that the agents converge fairly sensibly to their desired positions within the formation. There is some minor overshooting in the results, but this is not uncommon when agents are all trying to match up with one another. It is not overly pronounced, and so is unlikely to be a major problem. The agents all converge to their desired positions.

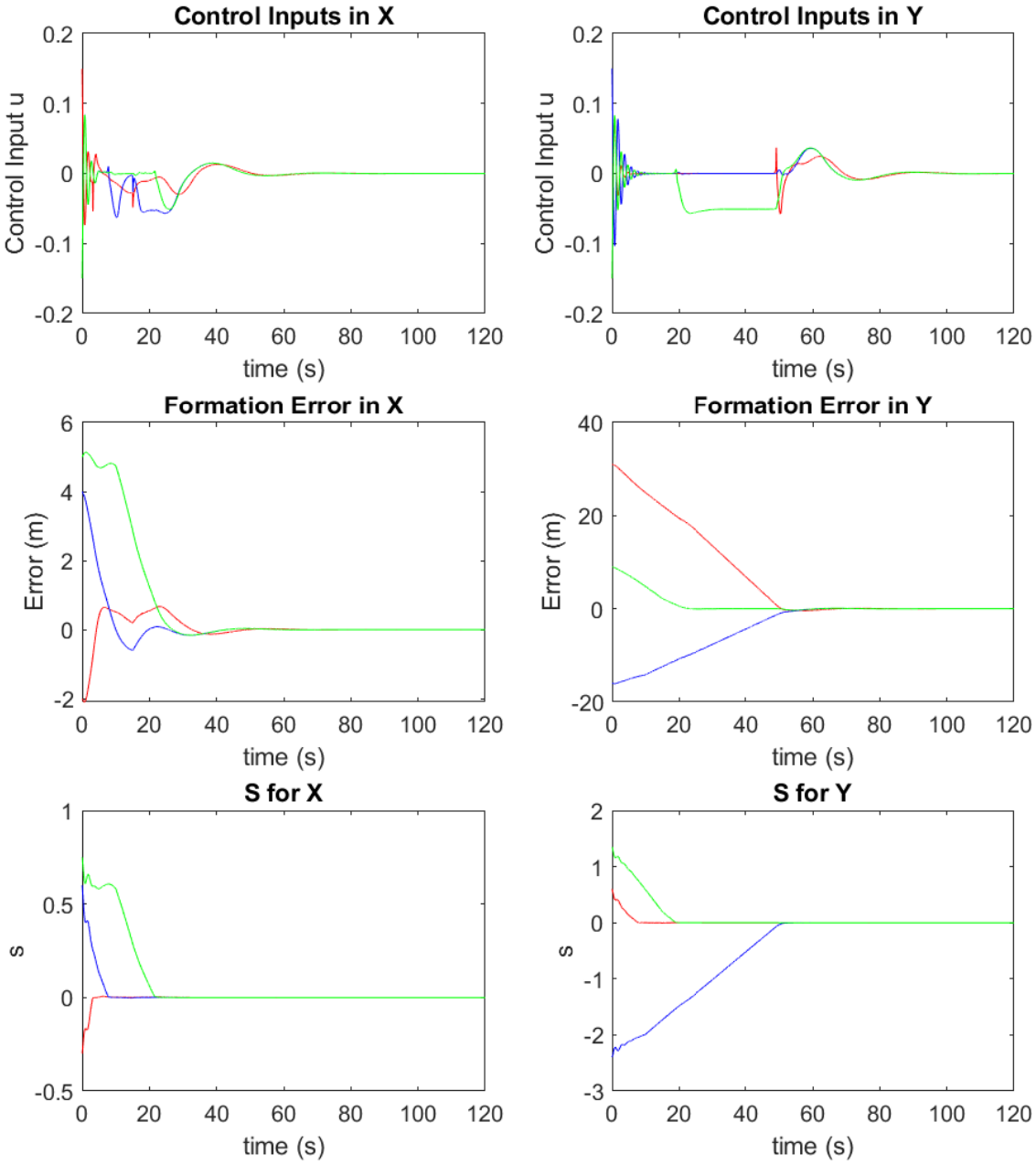

Figure 3.7: Six agents making triangle formation: X's indicate starting position, O's indicate final position

For the sliding surface, it can be seen that the values converge to approximately zero. Of significance, once the agent reaches 0, it does not deviate from this. This is fairly typical of a system without disturbances, but if disturbances were to be added

in these could cause the system to be pushed off of the $s = 0$ line.

As a verification that the linear model remains valid, the roll and pitch angles were measured to observe that they did not exceed the 10 degree limit required to maintain the validity of the linear model. As shown in Fig.3.8, these angles are well short of the requirements, indicating that a more aggressive controller can be implemented, allowing for faster convergence. The roll and pitch angles for Quadrotors 2 and 3 have a similar range.

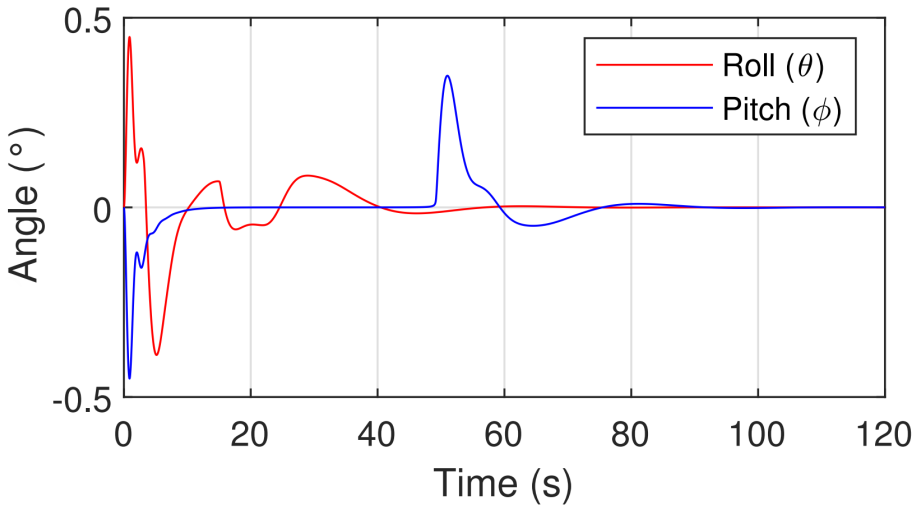

Figure 3.8: Roll and pitch angles of Quadrotor 1 over the course of a simulation

As shown in Fig.3.9 (a)-(c), a dynamic formation is also tested, wherein the agents move to a triangular formation as before, but the center point of the formation moves along a line $x = y/2$ at a rate of approximately 0.2 m/s. Most of the agents are simply trying to maintain a formation with one another, with only the red agent receiving position information from the virtual leader. This leads to it being a bit stretched out from the remainder of the formation in the early moments of the formation. However, by 60 seconds, the agents are largely in their triangular formation. After approximately 90 seconds, the agents reach the desired formation and the desired position. In Fig.3.9 (d)-(f) a disturbance of a constant force of 1 N is applied to the x-direction of the three drones. It is shown that the formation can able be formulated with little transient reponses which shows the robustness of the proposed controller.

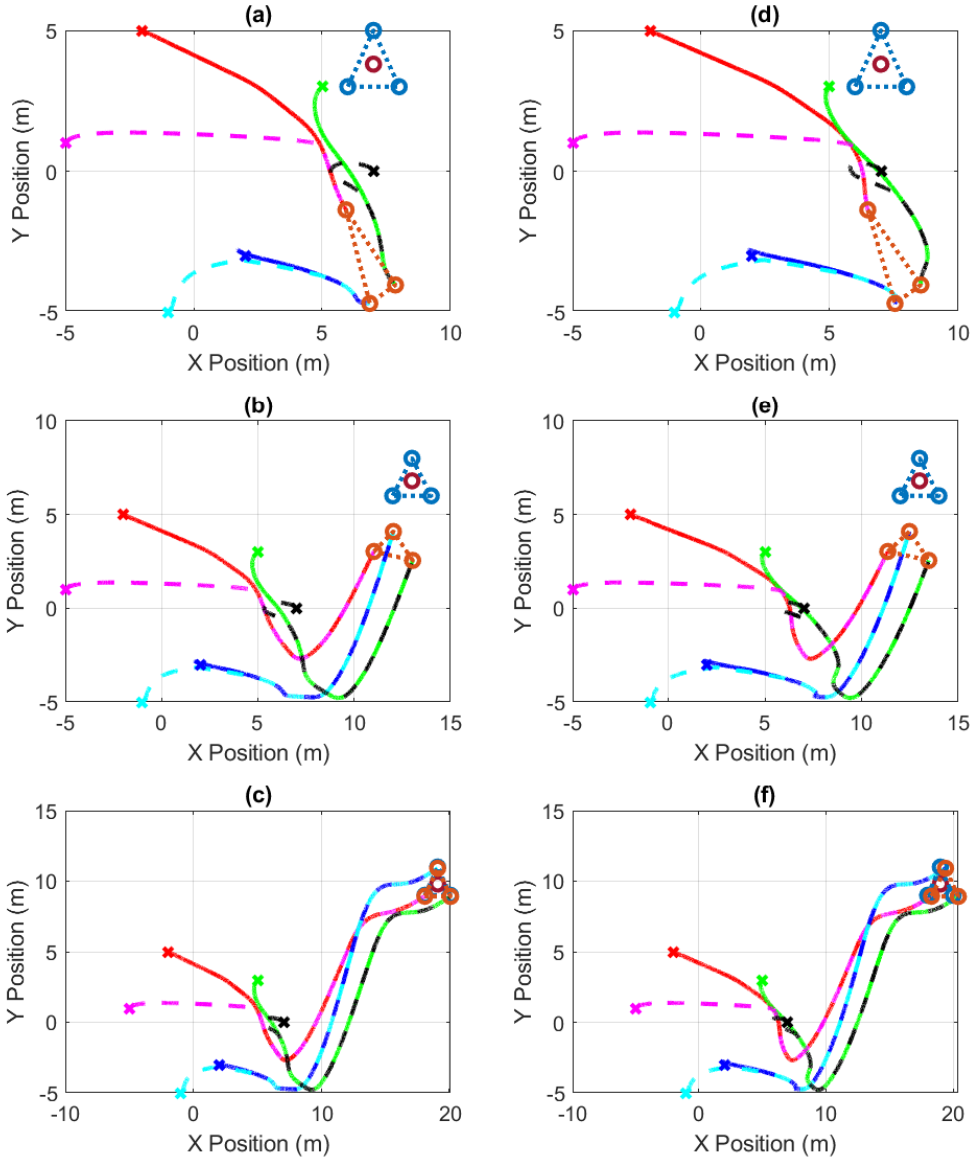

Figure 3.9: A formation with a dynamic virtual leader after (a) 30 seconds, (b) 60 seconds, and (c) 90 seconds. (d), (e), and (f) repeat this with a disturbance applied.

Fig.3.10 shows the average position error for the six agents with respect to their desired position within the formation over time. Initially the errors behave somewhat erratically, as the driving factors are the errors with one another. This is because of the limited communication with the virtual leader. At about $t = 40$ s, the errors between agents are approaching zero, and so the dominant factor becomes the error

with the virtual leader, causing the agents to move towards the virtual leader.

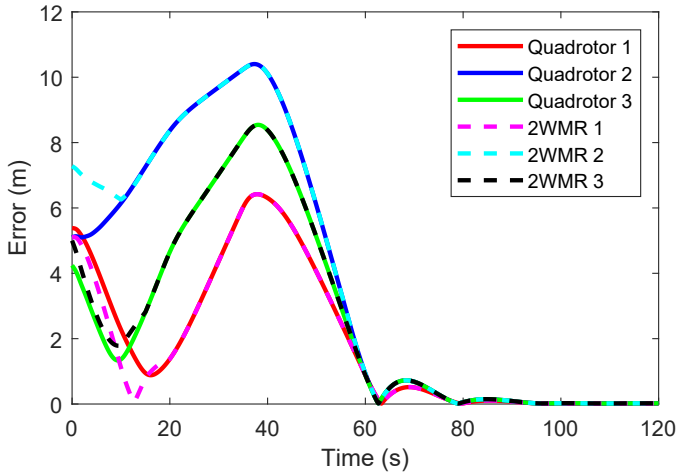

Figure 3.10: Formation errors relative to the virtual leader

3.3 Terminal Sliding Mode Control

3.3.1 Quadrotor Controller

The control inputs will be defined as

$$u_x = -\frac{1}{c_B}(c_1\dot{e}_x + c_2\dot{e}_{\dot{x}} + c_A\theta + c_3\dot{e}_x^{q/p} - \eta\frac{s}{|s|+\delta}), \tag{3.32}$$

$$u_y = -\frac{1}{c_B}(c_1\dot{e}_y + c_2\dot{e}_{\dot{y}} + c_A\phi + c_3\dot{e}_y^{q/p} - \eta\frac{s}{|s|+\delta}). \tag{3.33}$$

The inputs and dynamics for the x- and y-directions are similar, and so a generalized derivation will be completed that is true for both. The function $\eta\frac{s}{|s|+\delta}$ is used as a smoothing function which can be used in place of a standard switching surface. This is used to help eliminate the chattering problem found in many SMC formulations. η and δ are design constants which can be tuned depending on the scenario.

For the quadrotor, the sliding surface will be designed as a fast terminal sliding mode controller such that

$$s = c_1 e_{x_1} + c_2 e_{x_2} + e_{x_3} + c_3|e_{x_1}|^{q/p}\text{sgn}(e_{x_1}), \tag{3.34}$$

where

$$\begin{aligned} e_x &= \sum_{\substack{j=1 \\ j\neq i}}^{m} a_{ij}(x_i - (x_j + \delta_{ij})) \\ &+ \sum_{\substack{j=m+1 \\ j\neq i}}^{n} a_{ij}(x_i - (x_j + \delta_{ij})), \\ e_{\dot{x}} &= \dot{x}_i - 0, \\ e_\theta &= \theta_i - 0. \end{aligned}$$

For the final formation, the desired desired velocity and angular position are both 0, as the desired final condition is a stable formation. δ_{ij} denotes the distance between agent i and agent j in the desired formation, and a_{ij} is from the adjacency matrix.

Taking the time-derivative of s

$$\dot{s} = c_1\dot{e}_{x_1} + c_2\dot{e}_{x_2} + \dot{e}_{x_3} + c_3|\dot{e}_{x_1}|^{q/p}\text{sgn}(\dot{e}_{x_1}), \tag{3.35}$$

where

$$\begin{aligned} \dot{e}_{x_3} &= \dot{x}_3 - 0 \\ &= c_A x_3 + c_B u. \end{aligned} \tag{3.36}$$

Substituting (8) into (7) we find that

$$\dot{s} = c_1\dot{e_x} + c_2\dot{e}_{\dot{x}} + c_A x_3 + c_B u + c_3|\dot{e}_x|^{q/p}\text{sgn}(\dot{e}_x), \tag{3.37}$$

and by rearranging (10) it can be found that

$$\dot{s} = -\eta\frac{s}{|s| + \delta}. \tag{3.38}$$

From this, a Lyapunov function can be defined such that

$$V = \frac{1}{2}s^2. \tag{3.39}$$

Taking the derivative of (12) and using (11), we have

$$\dot{V} = s\dot{s} = -s\eta\frac{s}{|s| + \delta} = -\eta\frac{s^2}{|s| + \delta} < 0. \tag{3.40}$$

This shows that the value of $\dot{V}$ will be less than or equal to 0 for all time, which means that s will converge to 0 over time.

3.3.2 Mobile Robot Controller

The controller for the mobile robots is adapted from a controller covered in [94]. Testing the original controller showed that it was unable to properly track rectilinear motion, and so modifications were made to the controller such that it would be able to properly track this type of motion in addition to the curvilinear motion it was already shown to track. Two separate sliding surfaces are developed. The first is a fundamental sliding surface used to control angular velocity input, and the second is a fast terminal sliding surface used to control the linear velocity input.

The control inputs will be defined as

$$\begin{cases} u_\alpha = \omega_d + \lambda\omega_p + \eta\frac{s_\alpha}{|s_\alpha|+\delta} \\ u_l = \omega_d x_e + \omega_d y_e + v_d - \alpha z - \beta z^{q/p}, \end{cases} \tag{3.41}$$

where u_α is the input for the angular velocity input and u_l is the linear velocity input.

For the first sliding surface

$$s_1 = \alpha_e + \lambda\alpha_p. \tag{3.42}$$

In this case, α_e is the error between the current angle and the desired angle, λ is a design constant, and α_p is the error between the current heading and the heading towards the goal point as shown.

Note that in this case, the heading towards the goal point and the desired heading are not necessarily equal. The two error values can be calculated as

$$\begin{cases} e_{\alpha_e} & = \alpha_c - \alpha_d, \\ e_{\alpha_p} & = \tan^{-1}\frac{y_d - y_c}{x_d - x_c}. \end{cases} \tag{3.43}$$

Then $\dot{s}$ is

$$\dot{s} = \omega_d - \omega_c + \gamma\omega_p. \tag{3.44}$$

Since ω_c is the control input for the mobile robot, it can be found that $\omega_c = u_2$. Therefore

$$\begin{aligned} \dot{s}_\alpha &= \omega_d - \omega_c + \gamma\omega_p \\ &= \omega_d - u + \gamma\omega_p \\ &= \omega_d - (\omega_d + \gamma\omega_p + \eta\frac{s_\alpha}{|s_\alpha| + \delta}) + \gamma\omega_p \\ &= -\eta\frac{s_\alpha}{|s_\alpha| + \delta}. \end{aligned} \tag{3.45}$$

Using a Lyapunov function of $V = \frac{1}{2}s^2$, then the derivative will become

$$\dot{V} = s_\alpha \dot{s}_\alpha = s\eta \frac{s_\alpha}{|s_\alpha| + \delta} = \eta \frac{s_\alpha^2}{|s_\alpha| + \delta} \leq 0. \tag{3.46}$$

Additionally

$$s_l = \dot{z} + \alpha z + \beta z^{q/p}, \tag{3.47}$$

where

$$\dot{x}_e = \omega_d y_e - v + v_d, \tag{3.48}$$

$$\dot{y}_e = -\omega_d x_e, \tag{3.49}$$

and

$$z = y_e - x_e. \tag{3.50}$$

From this, it can be found that

$$\begin{aligned} s_l &= \dot{z} + \alpha z + \beta z^{q/p}, \\ s_l &= \dot{y_e} - \dot{x_e} + \alpha z + \beta z^{q/p}, \\ s_l &= -\omega_d x_e - (\omega_d y_e - u_l + v_d) + \alpha z + \beta z^{q/p}. \end{aligned} \tag{3.51}$$

Substituting u_l from (14) to (24) yields

$$s_l = 0.$$

A Lyapunov function is chosen such that

$$V = 1/2 s_\alpha^2 + 1/2 s_l^2. \tag{3.52}$$

Therefore

$$\dot{V} = s_\alpha \dot{s}_\alpha + s_l \dot{s}_l.$$

Since $s_l = 0$, it can then be found that

$$\dot{V} = s_\alpha (-\eta \frac{s_\alpha}{|s_\alpha| + \delta}) = (-\eta \frac{s_\alpha^2}{|s_\alpha| + \delta}) < 0.$$

As such, the value of s will tend to zero in finite time.

3.3.3 Quadrotor Simulation

The spiral trajectory is repeated with the newly developed controller. Overall, the basic behaviour is quite similar, but there is a clear improvement from the TSMC over the conventional SMC. The errors have clearly been reduced in comparison, and the error at the end of the test is smaller than it was in the case of the SMC. Fig.3.11 plots this result.

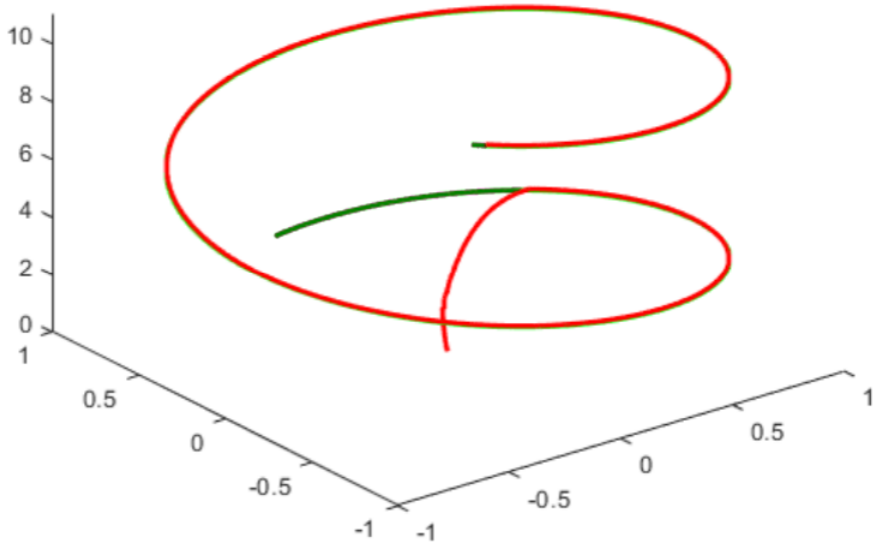

Figure 3.11: A quadrotor tracking a spiral trajectory with TSMC.

3.3.4 Mobile Robot Simulations

For the mobile robots, a one agent simulation was conducted in order to validate that the agents were behaving as intended. In this simulation the TSMC algorithm tracks along the line $y = x$ with the agent starting from the origin with an initial heading along the positive x-axis. The controller is shown to effectively track this motion after an initial error.

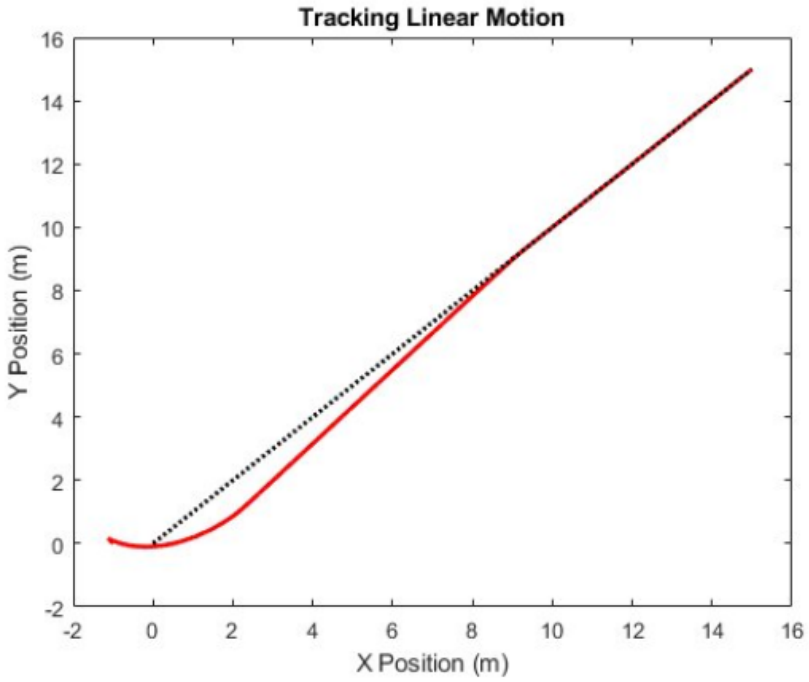

Figure 3.12: A mobile robot tracking to a desired goal point.

3.3.5 MAS Simulations

A series of consensus and formation simulations were conducted to verify the effectiveness of the controller. Simulations were conducted using MATLAB and Simulink. The topology for this simulation will be the same as it was for the SMC case. The agents again consist of three UGVs and three UAVs. For the design constants, they are selected as $\eta = 0.1$, $\beta = 0.001$, and $c_1 = c_2 = c_3 = 1$.

To begin, a consensus simulation is conducted with the six agents being placed at random locations in an environment. The virtual leader in this case has a value of 0 for the x-direction. These results were used to initially validate the controller, and would be for a formation in which all values of $\delta_{ij} = 0$. As shown in Fig.3.13, with the sliding mode controller implemented, the agents converged to a single point in approximately 30 seconds. This demonstrates an improved convergence compared with the classical sliding mode controller.

Following this result, a formation was attempted, with each mobile robot following a specific quadrotor, and the three quadrotors forming a triangle. The formation is also set up by way of the virtual leader such that Quadrotor 1 (indicated in red) will

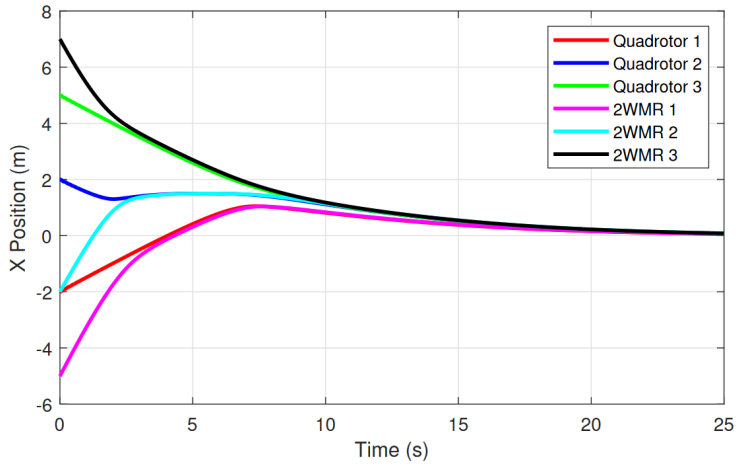

Figure 3.13: Consensus of six agents, showing the motion in the x-direction

reach a position of (-5,-10) in the X-Y plane. The convergence time for the six agents to achieve the desired formation in the desired position is approximately 65 seconds. The results of this simulation are shown in Fig.3.14.

This approach was also tested with an applied disturbance. In this case, as the agents approach the final goal points, a series of wind gusts effect the quadrotors. These are all in different directions, and the system must respond to them in order to avoid breaking the formation. The robustness of the TSMC allows for the agents to make it back to the goal points in spite of the disturbance. There is some notable chattering that can be seen in some of the movements (especially for Quadrotor 1).

Fig.3.16 shows the average position error for the six agents with respect to their desired position, as dictated by their neighbours, within the formation over time. Initially the errors behave somewhat erratically, as the driving factors are the errors with one another. This is because of the limited communication with the virtual leader. At about $t = 40$ s, the errors between agents are approaching zero, and so the dominant factor becomes the error with the virtual leader, causing the agents to move towards the virtual leader.

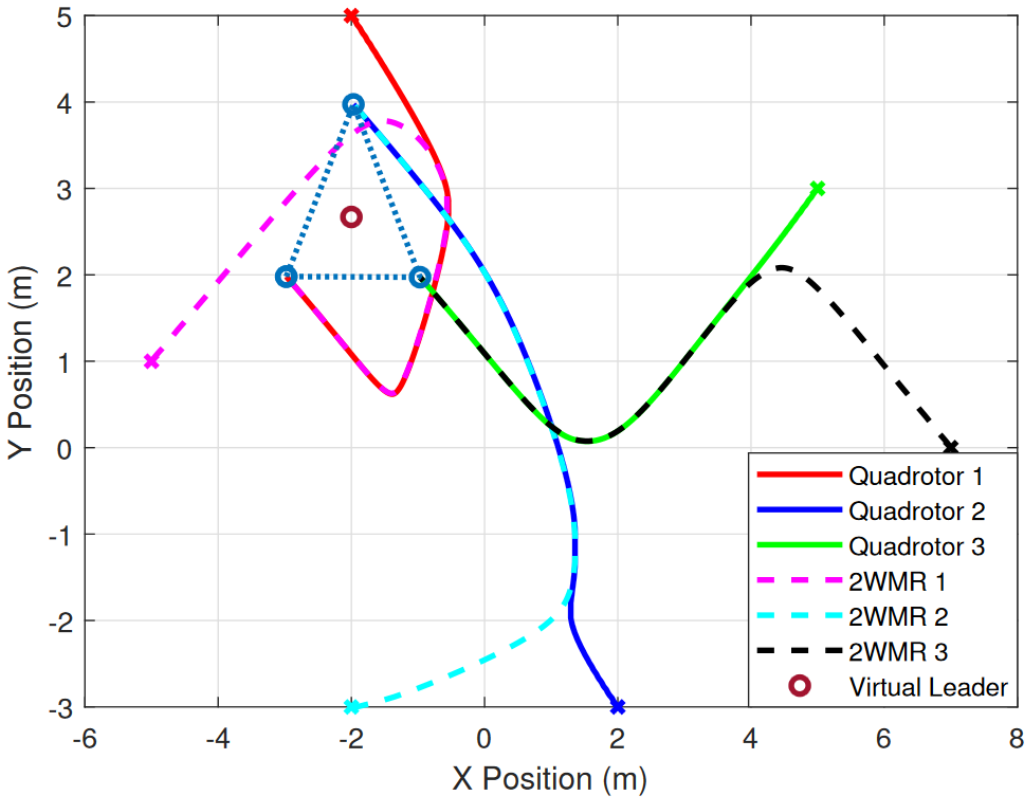

Figure 3.14: Six agents making triangle formation: X's indicate starting position, O's indicate final position.

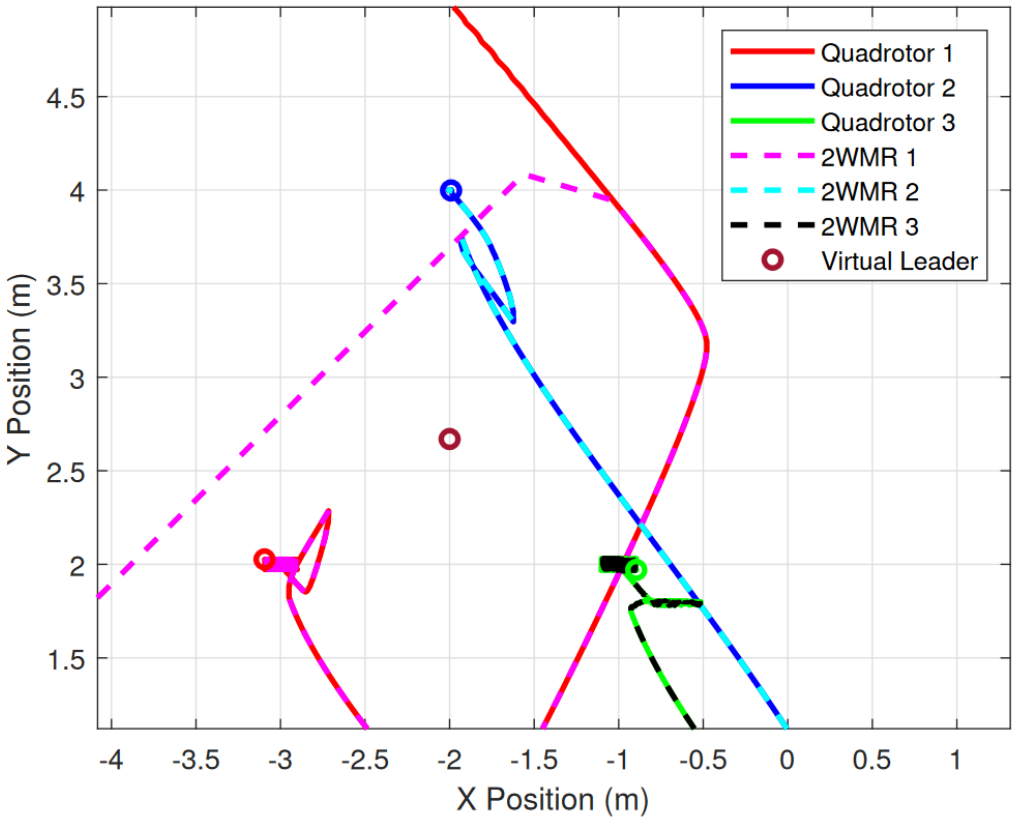

Figure 3.15: A zoom in showing the effect of the disturbances at the goal points.

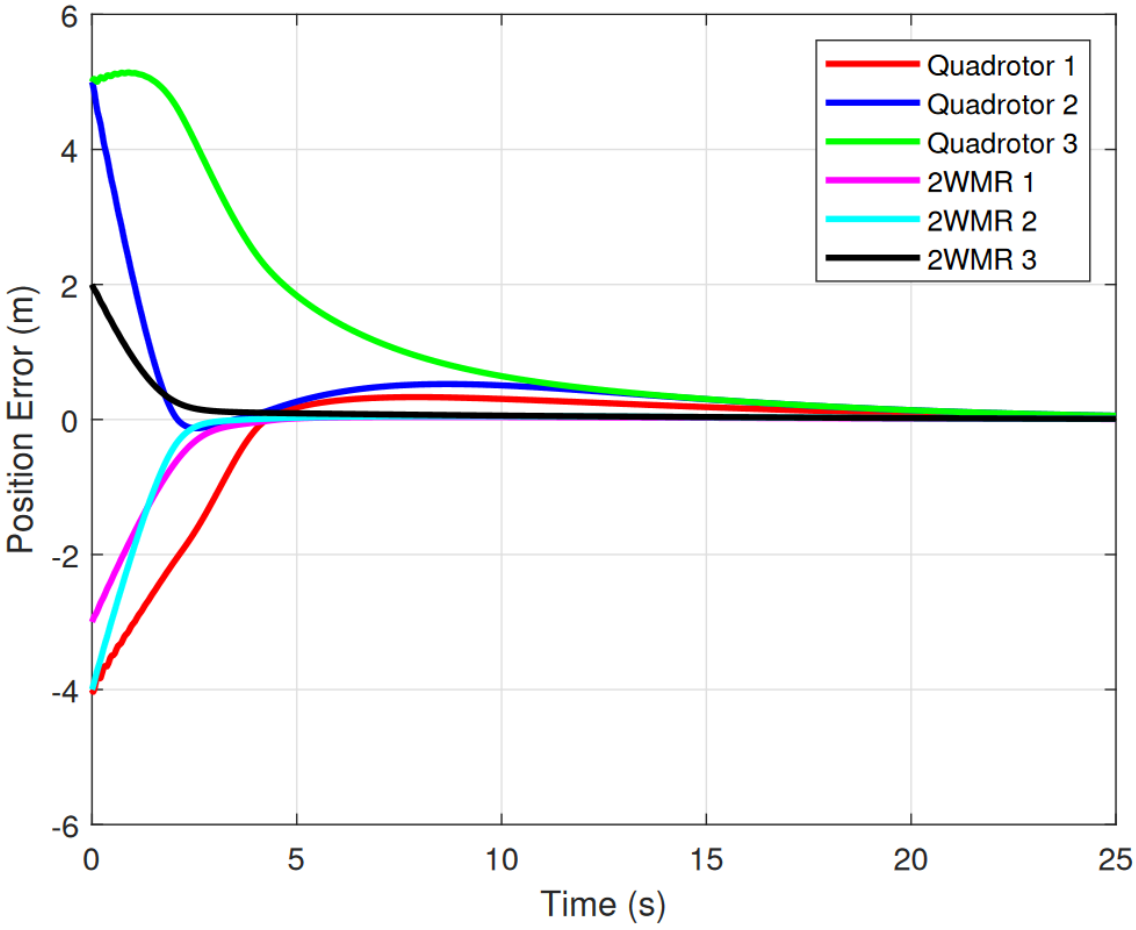

Figure 3.16: Formation errors relative to the signals from neighbouring agents.

3.4 Conclusion

In this chapter two pairs of controllers were tested. A conventional SMC approach was applied to both quadrotors and 2WMRs. This was tested and validated through a series of single agent and multi-agent simulations. A TSMC approach was also applied to both types of agents. A new set of simulations was devised to test this approach. Both sets of controllers showed robust responses to disturbances.

Chapter 4

A Novel Approach to Artificial Potential Field Based Obstacle Avoidance

Both of the SMC approaches developed can guide agents to specific goal points, however they do not have a mechanism for avoiding obstacles that might impede the trajectories. In this chapter, a novel approach to obstacle avoidance is developed which uses a further development of the classic APF approach to obstacle avoidance. This is done by using additional information to better optimize the algorithm.

4.1 Objective and Assumptions

The purpose of this novel APF algorithm is to provide a robust and efficient approach to local obstacle avoidance for a team of agents.

Assumption 4.1 *The agent has a higher level path planning algorithm that manages tracking towards the goal point. This process solely provides a repulsive field around obstacles. That is to say, the novel algorithm developed here is primarily used for the purpose of local obstacle avoidance, not global path planning.*

Assumption 4.2 *The signals from the LiDAR sensor can be assumed to be reasonably accurate without the need for filtering and there will not be significant noise in the signals of the LiDAR sensors.*

Assumption 4.3 *The environment can be treated as 2D. The LiDAR sensors used are only able to detect obstacles along a single plane, and so all obstacles will be assumed to be consistent along the z-axis.*

The objective of this chapter is to develop an improved version of the APF algorithm which will enhance performance, primarily by decreasing both the total travel required to navigate to a goal point and the time to reach the goal point. This algorithm can quickly respond to new information from sensor data, making this a reactive approach to obstacle avoidance.

4.2 Novel Artificial Potential Field for Obstacle Avoidance

For the purpose of obstacle and collision avoidance, it is desirable to have a system that can respond quickly and appropriately to sudden changes in the surroundings. The avoidance protocol being developed here is a local avoidance system. This means that it does not consider the full environment, but only the current immediate surroundings of the robot. It does not require a full map, simply the sensor data.

Artificial potential fields are commonly implemented, but have a number of limitations which lead to sub-optimal performance. As an example, a repulsive field acts around an entire obstacle equally in all directions. However, if the agent has already moved past the obstacle and is moving in the opposite direction of the obstacle, there is no longer any concern about a collision. In this case however, there will still be a repulsive force acting on the agent until it is completely outside of the field.

Chattering on the edge of a repulsive field is also a common problem when dealing with APF algorithms. In these cases, the binary nature of the potential field leads to cases where the agent will constantly alternate between being pushed out by the repulsive field, and then being pulled back in by the attractive field or other controller.

As such, a new factor based approach was developed for obstacle avoidance. In this method, a series of factors are developed which can be used to create a more optimized obstacle avoidance protocol. This approach to obstacle avoidance can be conceptualized as

$$U_{R,new} = \prod_{i=1}^{n} K_i U_{R,old}. \tag{4.1}$$

Here, $U_{R,old}$ is the repulsive "force" from a classical APF approach, while $U_{R,new}$ is the newly developed repulsion. K_i is then the series of factors that are developed, and each subsequently multiplied by the original repulsion in order to generate a new repulsive force. Depending on the exact needs of a system, some of these may be turned off (which would mean setting them to 1), and tuning may be required. The avoidance factors that will be considered in this work are as follows:

- **Heading Factor**: The heading factor is based on the direction of the agent's current linear velocity. Depending on the direction relative to obstacles, it will adjust the magnitude of the repulsion. This factor is represented by k_H.

- **Velocity Factor**: If the magnitude of the force is changing, the velocity factor will allow the system to respond quicker. This would primarily be used as a means of detecting incoming collisions from dynamic obstacles. This factor is represented by k_D.

- **Time Factor**: The agent should initially have no repulsive force acting on it when it enters the field, and from there the magnitude of the force should gradually ramp up to the desired value. This factor is represented by k_t.

In addition to these three factors, there is an additional consideration for a **Shift Factor** which operates as a separate function from $U_{r,new}$. The concept for this factor is that if an agent is heading directly towards an obstacle, then the end result will always be to turn away from it. Here, the shift factor is designed to begin that process earlier in order to reduce the amount that the potential field is used altogether. This is significant because when heading directly towards an obstacle, the repulsive field will naturally push back on the agent, but does not provide a lateral force, which can lead to an increased chance of reaching a local minimum.

The process will be considered using two distinct formulations. First, an obstacle-based approach will be developed. In this context, a set of cylindrical obstacles are known to the agent. With this formulation, the agents will respond in a perfectly idealized manner where they are able to identify their exact distance to each obstacle at all times. However, this is not always a realistic and representative approach, and so a second technique is developed.

In the second case, an angle-based approach is developed. For the agents in an unknown environment, they will use feedback from a 2D LiDAR sensor to create the repulsive fields instead of using known obstacle positions. This allows for a robust response in an unknown environment.

Either of these two formulations can work depending on the application. The former is best deployed for cases where the environment is already known, and the latter for when it is not. The simulations and experiments conducted will be completed using the angle-based approach. However, since this means that each gain is calculated 360 times, it is not used for visualizing the gains. Instead, the obstacle-based approach will be used to give a rough approximation of these values.

In both formulations, the controllers for the mobile robots are designed such that

$$\begin{cases} u_\alpha = \omega_d + \lambda\omega_p + \eta\frac{s_\alpha}{|s_\alpha|+\delta} + U_{R,\alpha} + F_S, \\ u_l = \omega_d x_e + \omega_d y_e + v_d - \alpha z - \beta z^{q/p} + U_{R,l}. \end{cases} \tag{4.2}$$

Here we have the two values $U_{R,\alpha}$ and $U_{R,l}$ which are the repulsive fields converted into an angular and linear function, as well as F_S which accounts for the shift factor.

4.2.1 Obstacle Based Approach

From Chapter 2, we can say that the repulsive field is designed as

$$U_{R,old} = \begin{cases} \omega_R(\frac{1}{||P_o - P_a||} - \frac{1}{P_r})^2 & ||P_o - P_a|| \leq P_r, \\ 0 & ||P_o - P_a|| > P_r. \end{cases}$$

Here, $||P_o - P_a||$ is the magnitude of the distance between and obstacle and the agent. If this value is greater than P_r, the radius of the potential field, then the repulsive field goes to 0, otherwise it is present as indicated, with the field growing rapidly in strength as the agent approaches the obstacle. ω_R is a control variable. Based on the direction of the force, a vector can be created in which

$$\vec{U}_{R,j} = (U_{R,j}, \theta + \pi). \tag{4.3}$$

Here $\vec{U}_{R,j}$ indicates a vector composed of a magnitude $U_{R,j}$ which is the repulsive force created by obstacle j, and an angle $\theta + \pi$, as the force will act in the opposite direction of the heading between the obstacle and agent. This is then combined with the various factors to produce a final result.

For the time factor, the objective is to ensure the field does not immediately provide a massive jolt to the agent. This should be controlled based on the maximum velocity of the agents and the size of the repulsive fields. As an example, a fast moving agent using smaller repulsive fields would likely benefit from a reduced effect of this factor, if not turning it off entirely. This new factor, k_T is as

$$k_T = \min\{at, 1\}, \tag{4.4}$$

where a is a control variable, and t is the time within the potential field. This causes the field to build up from the time it enters until returning to what would

be seen normally within the field. Typically, when first entering the potential field, the potential for imminent collision is not high, and so a large response may not be immediately necessary. The value of a can be adjusted to either make this process slower (by decreasing a) or faster (by increasing a).

The heading factor accounts for the direction the agent is facing relative to the obstacle. If the agent is heading in the exact opposite direction of the obstacle, for example, then there is no need for the field to do anything to the agent. As such, the idea is that if the agent is facing directly towards the obstacle, the APF should behave normally. However, as it begins to turn away, the force should be reduced, as the likelihood of collision will begin to drop. Fig.4.1 demonstrates this by showing an obstacle and how its angle is measured relative to an agent.

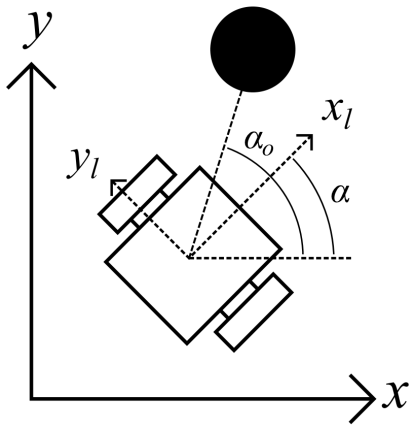

Figure 4.1: Visualing the heading angle and absolute angle to an obstacle.

This shows that the heading of the agent is found from a frame of reference using the x-axis as the line of 0 angle. Anything parallel to this will likewise have an angle of zero. The agent's heading relative to this reference is α and the angle of the line between the center of the agent and the obstacle taken relative to the reference is α_o. We can then say that the angle between the heading of the agent and the obstacle is θ. From this, the heading is then calculated as

$$\theta = \alpha_0 - \alpha. \tag{4.5}$$

This can then be applied to the heading factor and is applied to the heading factor as follows

$$k_H = \max(H\cos(\theta), c), \tag{4.6}$$

where θ is the angle between the current heading and the obstacle, and c is a control variable which can be adjusted in order to reach a stability point, and H is a gain which can be used to increase or decrease the magnitude of the heading factor for a specific case. Ideally c and H are designed so the agent can travel perpendicular to a wall with a repulsive field such that it will not get so close as to be dangerous, but will not need to be completely pushed out of the field.

Two plots are considered to demonstrate the broad tendencies of the controller. Both have an H value of 1. In Fig.4.2(a) the value of c is set to 0, and in Fig.4.2(b) the value is set to 0.25. Here we see that at an angle of 0 rad, the gain should return a value of 1, indicating no change in the magnitude of the potential field. As the angle grows in either direction, the magnitude of the potential field will be decreased, until it reaches the c value, at which point it becomes a constant. A c value of 0 is used in the first case because with a lower value, the gain would become negative and cause the repulsive field to become an attractive field. This is undesirable, and so c should never be lower than 0. A higher value may be used to ensure that there is always some level of repulsive field in order to maintain some distance from the obstacle. $c = 0.25$ is also shown to demonstrate that different minimum values can be selected to increase the repulsion. This can be increased up to $c = 1$, at which point the gain would stop having an impact at all.

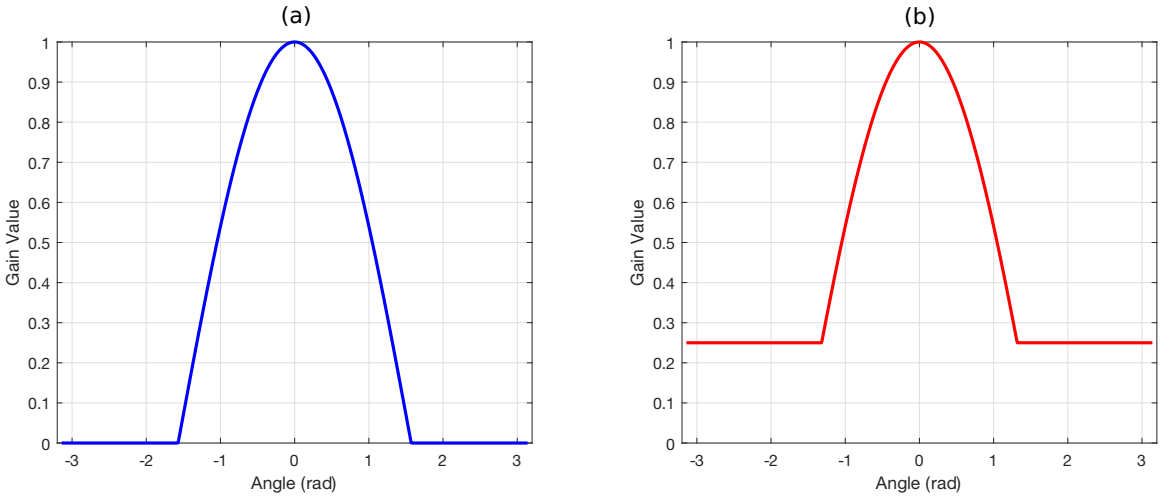

Figure 4.2: Visualing the heading angle and absolute angle to obstacle, with c=0 (a) and c=0.25(b)

Another factor which is considered is the changes in velocity of the agent relative

to the surrounding obstacles. This will allow the system to respond in the case of a dynamic obstacle. The difference between the velocities is considered as

$$\Delta V = V_o - V_a. \tag{4.7}$$

This difference will have a magnitude and direction, and is then employed in the velocity factor as

$$k_D = \max(D\frac{d}{dt}(||\Delta V||) + 1, 1), \tag{4.8}$$

where D is a control gain, and is set to $D = 2$. $||\Delta V||$ is the measured relative velocity of the agent relative to an obstacle. The positive ΔV indicates the two agents are heading towards one another, and the negative ΔV indicates that they are heading away from each other. In the event that the two are moving towards each other, then this factor will create a repulsion. If they are moving away from each other, there will not be a repulsion present in the system, as the value will become 1.

As there may be multiple separate obstacles which are identified simultaneously, multiple forces are summed up from these obstacles. Therefore, it can be stated that

$$U_R = \sum_{i=1}^{n} F_{R,i}, \tag{4.9}$$

for a set of n obstacles.

The shift factor is also applied to the controller, but only to the angular velocity in the form of

$$F_S = \begin{cases} \alpha_o - \alpha, & k_H \geq k_S, \\ 0, & k_H < k_S, \end{cases} \tag{4.10}$$

where α_o is the heading towards the obstacle. This is used to account for a tendency in the APF algorithms to respond the least effective in the worst case scenarios. Considering a case where an obstacle is directly between the agent and the goal point, the agent will head directly towards the obstacle such as what is seen in Fig.4.3. In this case, the local minimum is caused because of equal but opposite forces in the x-direction. However, any imbalance in the y-direction can cause the agent to be removed from the local minimum. As such, this scenario where the obstacle sits directly between the goal and the agent is one which should be actively avoided.

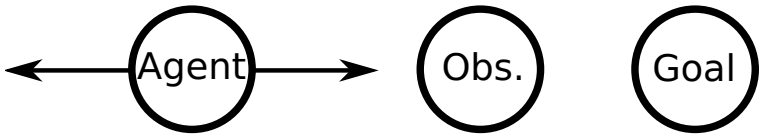

Figure 4.3: Visualizing a basic case of a local minimum.

Using the shift factor, once the agent is within the repulsive field, it will be pushed in the opposite direction, but this will not lead to the agent experiencing a push to the sides, which is what it needs in these cases in order to move around the obstacle. Otherwise it may simply end up at a local minimum. This is designed to only be effectively while the agent is facing in the general direction of the obstacle, and can be tuned as needed.

4.2.2 Angle Based Approach

For the angle based approach there are some foundational differences in how the agents interact with the environment that leads to some minor differences in their general behaviour. This approach is primarily designed to allow for quick interpretation of the environment without requiring an additional layer of software which would convert from a set of LiDAR to an obstacle. Instead, each angle measurement is treated as an independent obstacle which is re-evaluated each time the LiDAR sensor completed a full revolution.

It is important to note that there are two different ways to interpret the angle. The first is from a global perspective, and the second is from a local perspective. In both cases the angle measurements are taken centered about the LiDAR sensor, however in the local coordinate system, 0 degree will be directly ahead of the agent instead of being in a specified, fixed direction.

The primary difference in this method then is that the larger an obstacle is from a radial perspective, the greater influence it will have on the resulting repulsive fields. Note that since this is from a radial perspective, it does not necessarily mean the larger obstacle, as a small obstacle near the agent will appear larger, as indicated in Fig.4.4. Here the square is the Turtlebot, and the two circles represent obstacles.

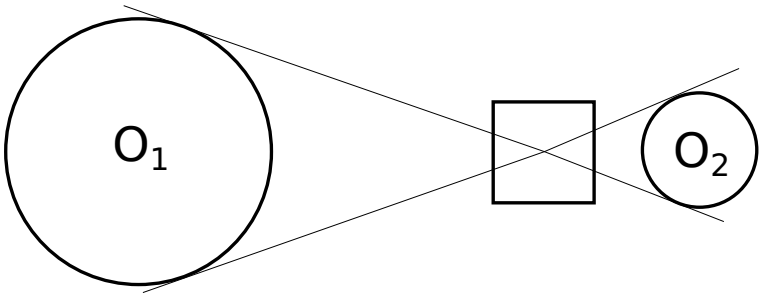

Figure 4.4: Visualing of how size of obstacles are dependent on their size and proximity to the agent.

In Fig.4.4, Obstacle 1 is 80 units in diameter, and is seen as about 39 degrees by the LiDAR sensor. Obstacle 2, on the other, is 35 units in diameter and is seen as about 45 degrees by the LiDAR sensor. By summing over the radial size of obstacles, large obstacles are generally avoided, and as an obstacle gets closer, it has another mechanism to increase the impact of the repulsive force.

As previously, we can define

$$U_{R,old} = \begin{cases} \omega_R(\frac{1}{P(\theta,t)} - \frac{1}{P_r})^2, & P(\theta,t), \leq P_r, \\ 0, & P(\theta,t) > P_r, \end{cases}$$

$$U_{R,new}(\theta) = \prod_{i=1}^{n} K_i(\theta) U_{r,old}. \tag{4.11}$$

The significant difference here is that the measurements here are taken based on the measurement from the LiDAR sensor at any given angle. $P(\theta, t)$ is the distance measurement of a specific angle at a specific time. Using the most recent value, this is used to approximate the

$$\vec{U}_R(\theta) = (U_R(\theta), \theta + \pi). \tag{4.12}$$

This is largely the same formulation as previously, but instead of the vector being based on an obstacle j, it is instead based on an angle measurement θ. From this point, the values of the various k values are all computed as they normally would be, however this is done at each angle instead of being done separately for each obstacle.

For the time factor, the idea is to ensure the field does not immediately provide a massive jolt to the agent. This should be controlled based on the maximum velocity

of the agents and the size of the repulsive fields. This new factor, k_T is as

$$k_T(\theta) = \min\{at, 1\}. \tag{4.13}$$

The heading factor accounts for the direction the agent is travelling relative to the obstacle. If the agent is pointing in the exact opposite direction of the obstacle, for example, then there is no need for the field to do anything to the agent. We can say that the heading of the robot is then defined as

$$\theta = \alpha + \begin{cases} 0, & V \geq 0, \\ \pi, & V < 0, \end{cases} \tag{4.14}$$

This indicates that the "heading" of the agent refers not to the forward-facing direction of the robot itself, but to the current direction of linear motion based of the agent. If the robot is moving in reverse, it is necessary that the heading factor be adjusted to account for the new direction of movement to ensure that obstacles can be properly avoided.

As such, the idea is that if the agent is travelling directly towards the obstacle, the APF should behave normally. However, as it begins to turn away, the force should be reduced, as the likelihood of collision will be reduced. This is represented by

$$k_H(\theta) = \max\{H\cos(\alpha_0 - \alpha) + c, c\}, \tag{4.15}$$

where α_0 is the relative angular position of the obstacles on a global coordinate system from the position of the agent, and α is the heading of the agent. c is a control variable which can be adjusted in order to reach a stability point, and H is a gain which can be used to increase or decrease the magnitude of the heading factor for a specific case. Ideally c and H are designed so the agent can travel perpendicular to a wall with a repulsive field such that it will not get so close as to be dangerous, but will not need to be completely pushed out of the field. For this paper, $H = 1$ and $c = 0.25$ are selected.

Another factor which is considered is the changes in velocity of the agent relative to the surrounding obstacles. For this, we use the following

$$k_D(\theta) = D\frac{d}{dt}(V(\theta, t_c) - V(\theta, t_{c-1})) + 1. \tag{4.16}$$

D is a control gain, and is set to $D = 2$. $V_a - V_o$ is the measured relative velocity of the agent relative to an obstacle.

This is then summed up for all angles to produce

$$U_R = \sum_{\theta=-\pi}^{\pi} \vec{F}_R(\theta), \tag{4.17}$$

for the set of angles scanned. For larger obstacles, it is likely that they will be detected multiple times by the LiDAR sensor. In this case each separate detection is added to the sum, as an obstacle which takes up a large part of the area that the LiDAR sensor can see is usually best avoided. An example case for this would be a wall, which will be seen in Simulation 4.

The shift factor is still applied in the same manner, but is modified to account for it being relative to the measured angle and not to a specific obstacle

$$F_{S,\theta} = \begin{cases} \theta - \alpha & k_H \geq k_S. \\ 0 & k_H < k_S, \end{cases} \tag{4.18}$$

Other agents within the system can also act as obstacles. The range of the APF in this case, meaning it will only occur if a collision is imminent. This is workable in the application due to the low maximum velocity of the agents. Since they are limited in how quickly they can travel, they do not need a large amount of time to respond to obstacles. Depending on how fast agents move, the size of the potential field would need to be calibrated for more optimal results.

This formulation can be used for both global and local obstacle avoidance, however it is used in this work for local obstacle avoidance. This approach reduces the impact of limitations in a conventional APF approach such as the local minimum, however these are still possible under the right circumstances. In complex environments, this novel APF will not be guaranteed to be able to navigate environments, and so it is best employed in local obstacle avoidance in parallel with a more robust path planning algorithm.

4.3 Simulations

4.3.1 Tuning Gains

A thorough gain tuning process was not conducted for this approach. By evaluating the impact of each gain, the novel APF can be designed to be useful for a variety of applications. However, while the gains selected may be useful in a variety of cases, the optimal gains for any given case can be quite varied.

Heading Factor

The heading factor was the primary factor considered in the original plan, and so it is expected that it should be the largest factor in the effectiveness of the process. As such, tuning for this case is especially significant. The heading factor reduces the effect of the potential field, and has a range from a minimum value of c to a maximum value of 1. The piece wise function is broken into two separate functions. First is a cosine wave based on the heading, and the second is a constant value. Whichever of these two values will be larger is the one that is used in the calculations.

The value that will be tuned here is the value c. This acts as the minimum gain. If it is set to 0, then an agent not facing an obstacle will experience 0 repulsion from it. This may be useful, but can also run into problems of getting too close to obstacles, and errors in readings can lead to collisions. As such, some repulsion is still useful to help create a buffer. With this in mind, a number of values of c will be tested in order to see what kind of effect will be present in multiple different cases.

Using a simple environment with a single obstacle, a series of values from $c = 0$ to $c = 1$ with a step of 0.05 were tested to see the impact of this tuning. Plotted in Fig.4.5 is these 21 results. As the value increases, each test gets pushed further away from the obstacle. At the closest we have the $c = 0$ case, which stops receiving any repulsion once it is travelling parallel with the tangent line of the cylinder. For this case, the agent turns back in towards the obstacle a small amount. The other cases do not see this trend happening as there continues to be a repulsive force acting on the agents.

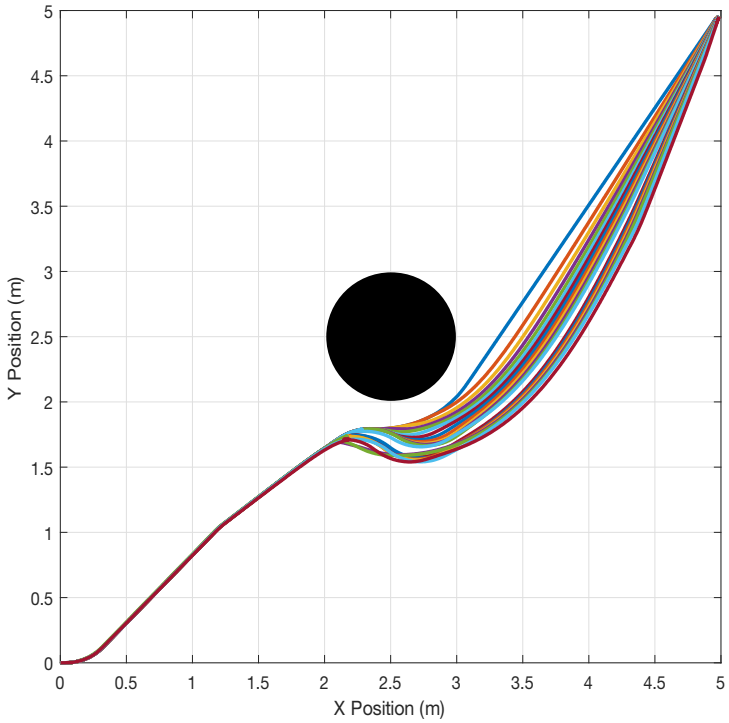

Figure 4.5: Simulations of 21 different c values with $0 < c < 1$ for the heading factor.

The unsurprising result here is that the lower the c value, the closer the agent will remain to the obstacle. This will shorten the total distance travelled, but the closer the agent gets to the obstacle the less room there is for error in the controller. Depending on the application it may be desirable to provide a greater amount of security and the cost of less optimal pathing. To get a more comprehensive idea of the effect of increasing the value of c, the distance travelled for each case is plotted in Fig.4.6.

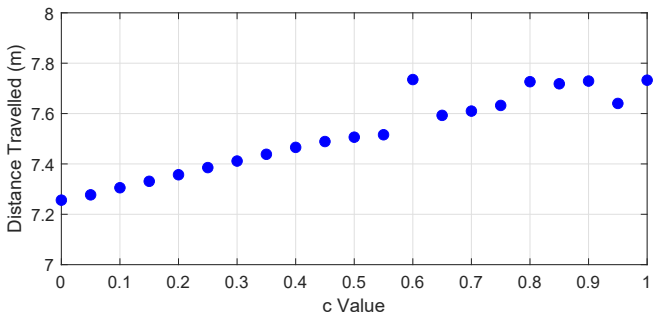

Figure 4.6: Distance travelled for each simulation.

From this, a fairly linear trend emerges wherein the larger the value of c the greater the distance travelled. This could theoretically also continue if c was able to be a negative number, however this would lead to the obstacle having an attractive force if the agent is facing away, which is very undesirable. The data is not perfectly smooth, which serves as a reminder that this is not a problem with simple binary answers. The optimal control gains will be dependent on the specific case being investigated, and so depending on the case different gains may be beneficial. A similar trend was observed with the time to reach the goal.

Another relevant factor is the presence of chattering when applying a controller. This is something that this controller is attempting to reduce, and so a couple of tests were conducted to see how c might impact the results. These were done in an environment with 3 obstacles to present more opportunities for chattering to manifest in the data. The primary focus here is on the angular movement. Generally with APF avoidance agents will pivot back and forth as a result of varying fields. This would manifest in the results as sudden swings in the angular velocity input for the agent. To see if chattering is a concern, the derivative of the discrete angular velocity data is plotted for $c = 0$, $c = 0.1$, and $c = 0.2$ in Fig.4.7.

There is some minor chattering present in all three cases, but it is much more prominent in the case of $c = 0$. Between $c = 0.1$ and $c = 0.2$ there is not much of a difference. This emphasizes that while $c = 0$ might provide a more optimal trajectory, there are still advantages to providing a small repulsive field even when the agent is not directly facing the obstacle. From this, it can be seen that depending on the environment, different values of c may be preferable. Further testing in the current environment at higher c values did not appear to cause any prominent shifts beyond what is demonstrated here.

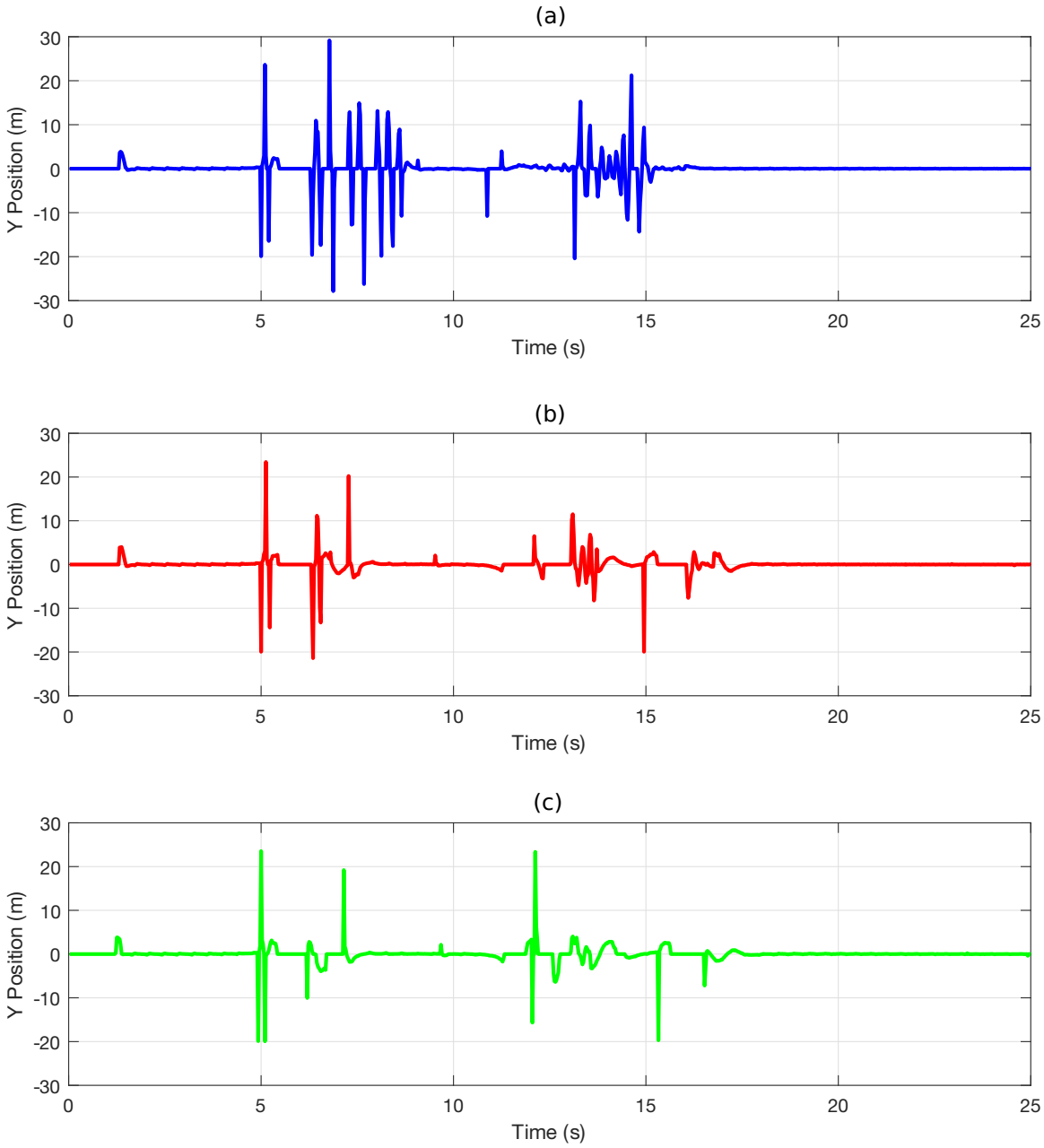

Figure 4.7: Plots for the angular acceleration when $c = 0$ (a), $c = 0.1$ (b) and $c = 0.2$ (c)

Differential Factor

Values ranging from 1 to 100 were tested for the differential factor. This is primarily focused on evaluating how the system responds to changes in the environment. Values for D tested were 1, 5, 10 and 20, and in all cases the system had a very similar

trajectory. When zoomed out, the four appear to overlap, as in Fig.4.8(a). However, zooming in Fig.4.8(b) there are minor variations between the tests. The larger values of D see the agent pushed back a bit. as they advance towards the obstacle.

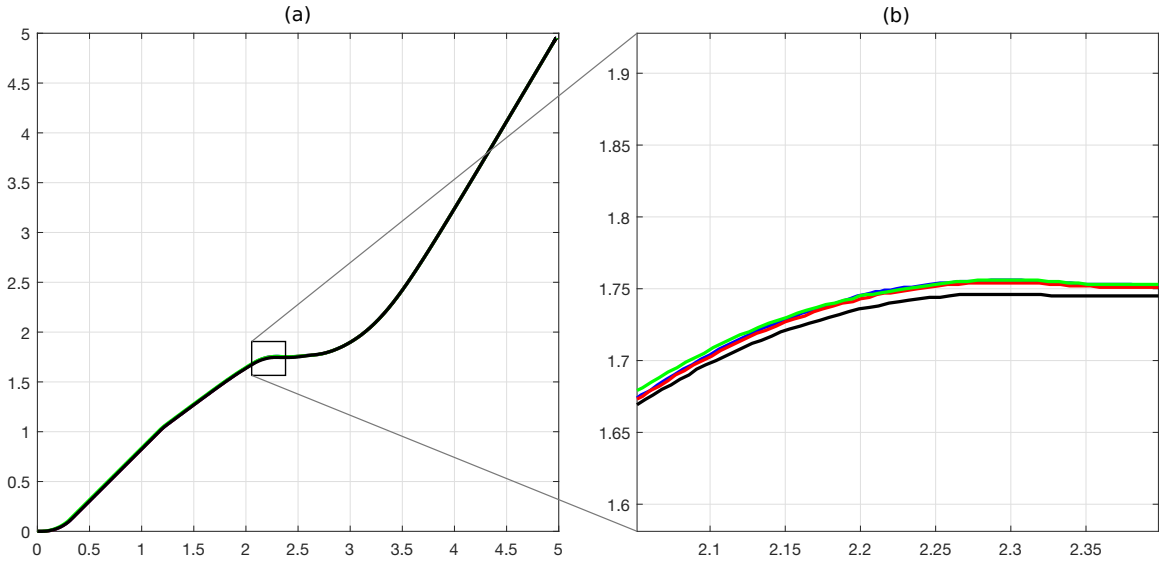

Figure 4.8: Comparison of different values for D in differential gain.

There are some minor variations in the results here, but overall the impact is quite small. As the environment is static, this is not surprising. Were it to be tested in a more dynamic environment it would likely show a much more valuable role in the obstacle avoidance process. A D value of 1 is used in subsequent tests.

Time Factor

The time factor allows for a gradual ramping up of the repulsive force. A series of time constants ranging from 0.5 to 100 are used, with the results indicated in Fig.4.9. At a value of 100, the system will reach it is neutral value in 0.01 seconds, meaning that it will happen almost instantaneously from the perspective of the controller. At 0.5, the process will take 2 seconds to ramp up. The differences between the results are fairly minor, but there is a pronounced change in the approach to the obstacle and how early these agents will begin to diverge from the original trajectory, which is to be expected. However, beyond this the behaviour is fairly consistent.

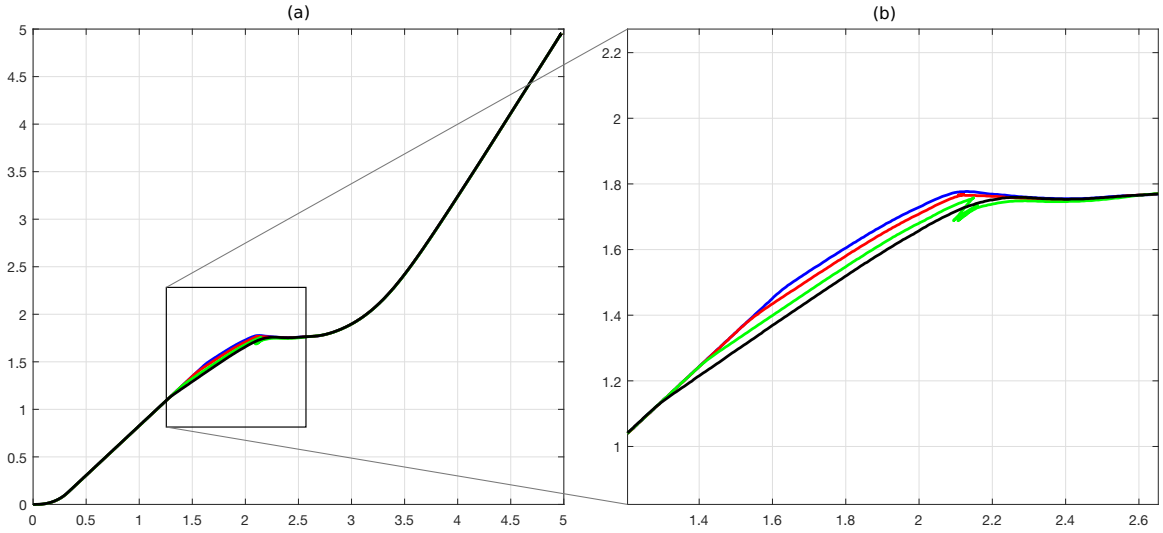

Figure 4.9: Comparison of different values for a in time gain.

The broad response of the system is that decreasing the value of a, and therefore increasing the impact of the time factor, will cause the agent to begin adjusting course from the obstacle later, but that the overall trajectory will be largely unaffected. There is a stuttering seen in one of the cases, however this appears to be a result of very particular geometry and timing, as small changes to the value of a around this do not appear to present a similar case.

Shift Factor

The shift factor was also tested, using values from 0 to 1 for K_S. The results of this test are plotted in Fig.4.10. For the 6 cases tested, the general differences are the angle that they adjust the trajectory by when they enter into the repulsive field. But as the shift factor stops having an impact, the agents will all follow very similar trajectories. There is some minor variations in the position values, but the results fairly clearly indicate that the long term results are not likely to be heavily impacted by the shift factor, only the immediate concerns of a collision.

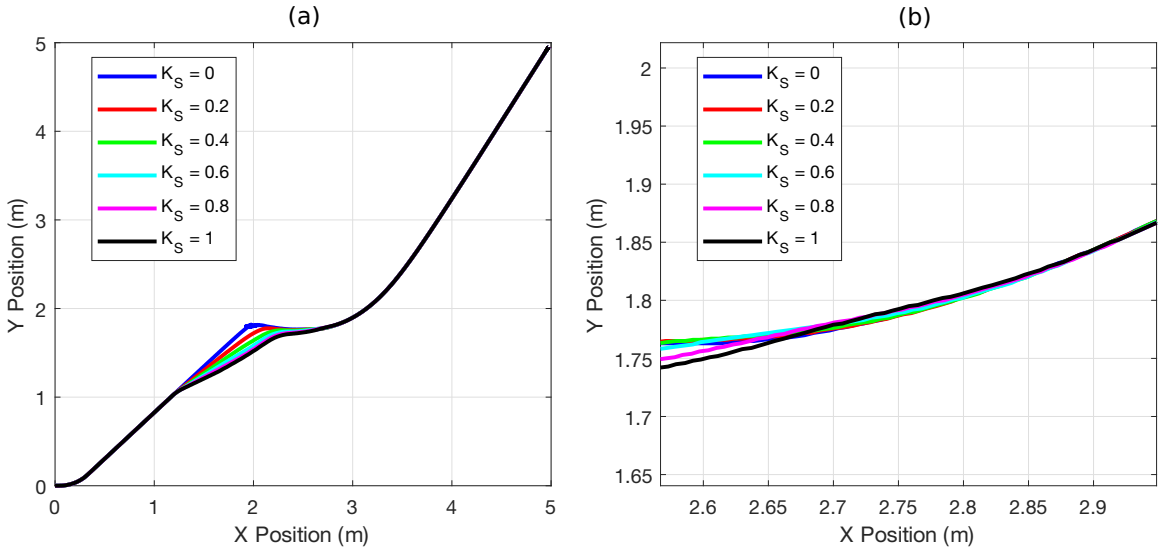

Figure 4.10: Comparison of different values for K_S for the shift factor.

It can be seen that there is still some stuttering at lower values of K_S. This comes as a result of the very small lateral repulsion compared to the much greater repulsion that applies in the backward direction. The primary benefit of the shift factor then comes from aiding the mobile robot by ensuring that there will be a non-trivial lateral repulsion to create an angular velocity.

From these experiments, there are a few generalities that can be taken away. First off, the angle factor appears to have the largest impact on the overall trajectories. By changing the value of the c constant, large shifts in the overall trends of the trajectory are observed. The shift factor and time factor both provide notable, but less significant effects on the trajectory. These tended to impact the early interactions with the obstacle, however the end results would broadly converge to similar trajectories. The differential factor had the least significance, though in an environment with dynamic obstacles it is likely that it would be much more pronounced. Depending on the specific make-up of the environment the impact of each factor may vary.

4.3.2 Final Tests

Four simulations were run to demonstrate the improvements of the obstacle avoidance protocol. These use a variety of different environments to show the approach is robust, and compare it with a classical APF. Each simulation starts from a position (0 m, 0

m)and with a goal at position (5 m, 5 m).

In the comparisons against the classical APF, the classic approach has been tuned to provide a reasonable approximation of the best results in both time to the goal point and total distance travelled. The size of the repulsive field is kept constant between the two APF approaches for the sake of comparison.

The first simulation is avoiding a single obstacle which the agent is approaching nearly head on. In Fig.4.11, the novel APF begins adjusting course earlier as a result of the shift factor, and is then able to smoothly get around the obstacle. As a result of the heading factor, once it is pointing away from the obstacle there is only a minor repulsive force, and so the novel APF is not pushed nearly as far away as the classic APF is, instead taking a much better optimized trajectory towards the goal point.

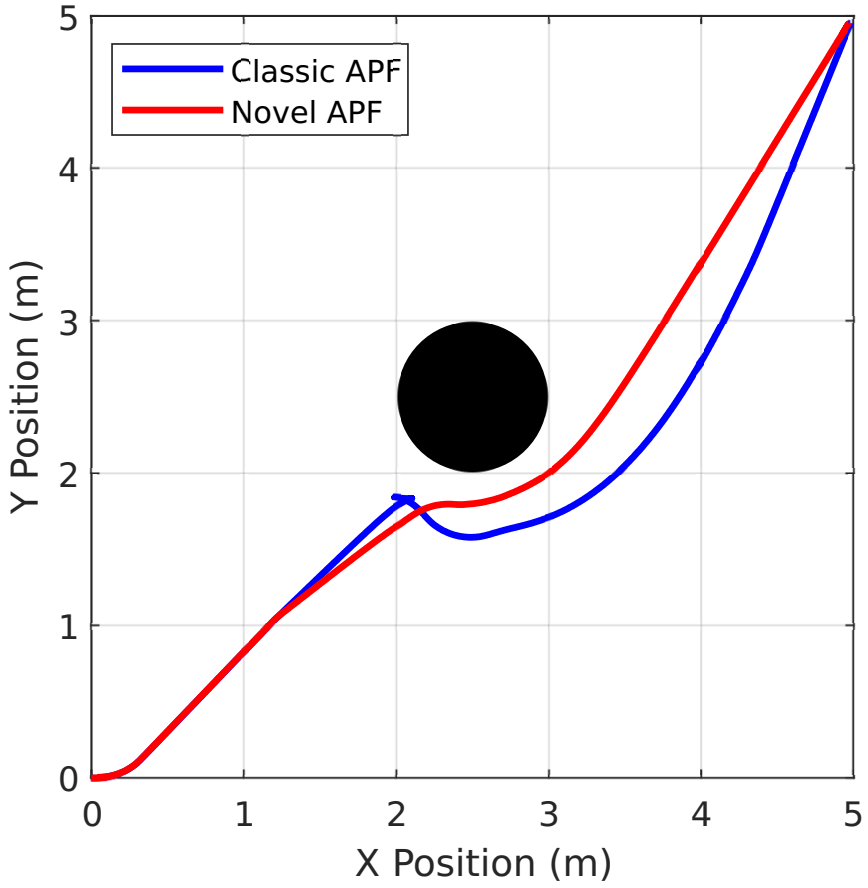

Figure 4.11: Simulations 1: Novel and Classic APF with a single obstacle.

The classic APF, on the other hand, displays some minor chattering just below (2

m, 2 m) where it bounces back and forth as it rotates to move around the obstacle, before taking the longer route. This is a result of the potential field having a small lateral force and large backwards force. The agent does not receive a sufficiently large control input to lead to a large change in the angular velocity, and so it turns slowly while the forward/backwards motion chatters. Once it has gotten out of this chattering near a local minima, the system is much smoother, but the deviation is far larger. This leads to both a longer time to reach the goal and a larger travel distance.

Within this test the proximity of the Novel APF is tracked over time and plotted in Fig.4.12. This shows the agent comes within 9.6cm of the obstacle once the size of the Turtlebot itself is accounted for. Depending on the desired safety factor, a different value of c may be desirable to increase the tendency to be pushed away from the obstacle.

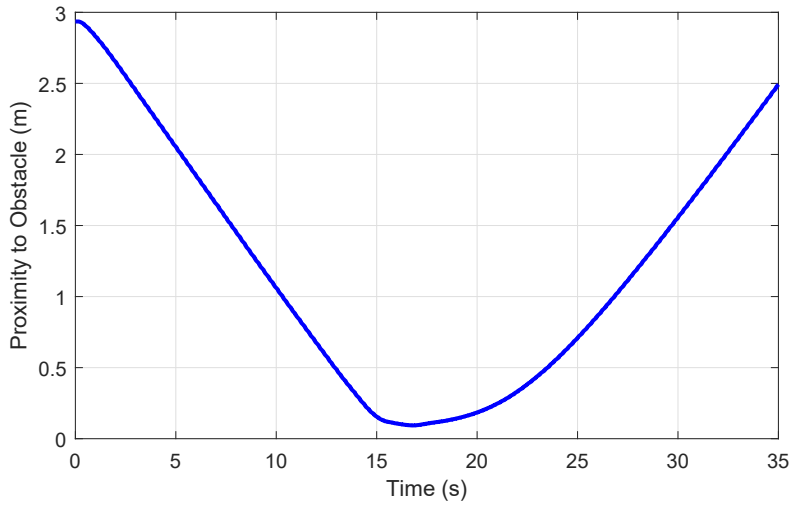

Figure 4.12: Proximity of the novel APF to the obstacle over time.

Additionally, it is worth looking at the magnitudes of each of the factors over the duration of this simulation. This is done using the obstacle approach, which produces the same basic result. Fig.4.13 plots the three factors of $U_{R,new}$ over the duration of the simulation. The heading factor is seen to immediately reach a value near 1 as it enters the repulsive field, as the agent is facing towards the obstacle. The field slowly declines from this point, until a time of approximately 13 seconds, at which it begins a sustained decline, eventually stabilizing at the minimum value. There is

some minor choppiness on the way down, but this generally shows the effectiveness of this methodology.

The time factor increases over the course of about 1 second as planned, and the velocity factor has some small increases in the initial seconds in the repulsive field, but generally maintains a value of 1 at all times.

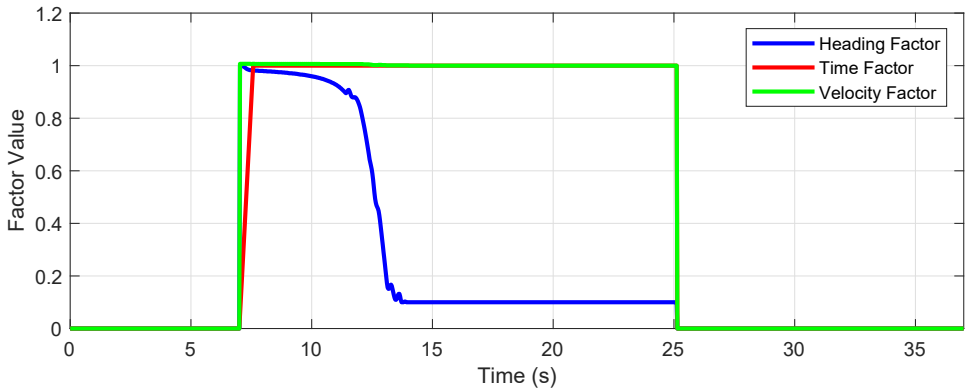

Figure 4.13: Value of each gain during the simulation using the obstacle based method.

The second simulation has a similar environment to the first simulation, but with the obstacle shifted 0.5 m to the left in the x-direction. The results of both simulated tests are included in Fig.4.14. The objective of this simulation is to demonstrate a key utility of the novel APF. The obstacle is not going to be in the direct path of the agent while it is heading to the goal point, and so it does not cause a prominent change in the trajectory of the agent. While it does pass through the potential field of the obstacle, the heading factor drives the repulsive force to be quite low, meaning only a minor deviation is seen. As the agent changes its heading away from the obstacle, this will lead to a further reduction in the effect of the repulsive force, and so the agent will follow a similar trajectory to what it would have.

In contrast, the classic APF approach sees a large deviation in its trajectory even though it would not have contacted the obstacle. It is somewhat reduced from the first simulation, but this is a good example of the suboptimal performance which can be expected from a classic APF. It is still effective at avoiding the obstacle, but the excessive movement leads to a longer path distance and a longer time to reach the goalpoint. It also can lead to problems when dealing with more than one obstacle,

as each obstacle will continue to have an impact on the agent even when a collision is unlikely.

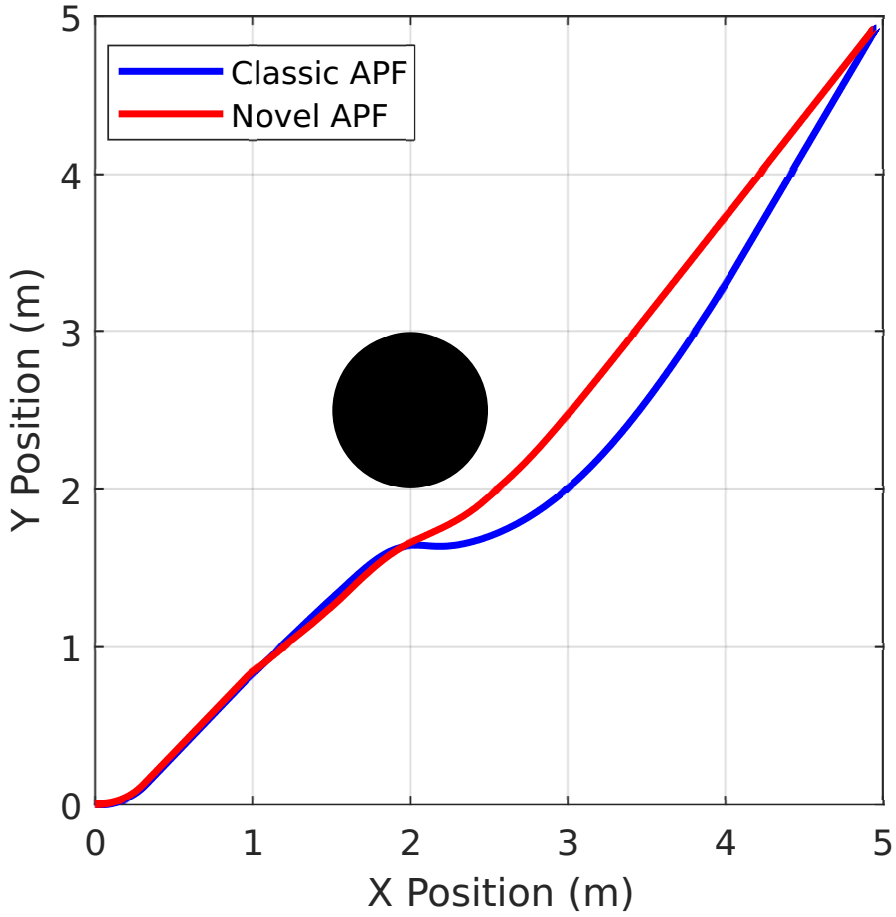

Figure 4.14: Simulations 2: Novel and Classic APF with a single obstacle, offset from Simulation 1.

Simulation 3 presents an example with three obstacles to demonstrate the utility of the novel APF, as in Fig.4.15. The agents must navigate around them. They have been specifically placed so the APF should naturally funnel the agents towards the center, where they will be experiencing the fields of all three agents. The agents pass along the right side of the first obstacle, and then ideally should pass through the small gap between the pair of obstacles behind it. These obstacles combine to provide plenty of opportunities for local minimum to occur.

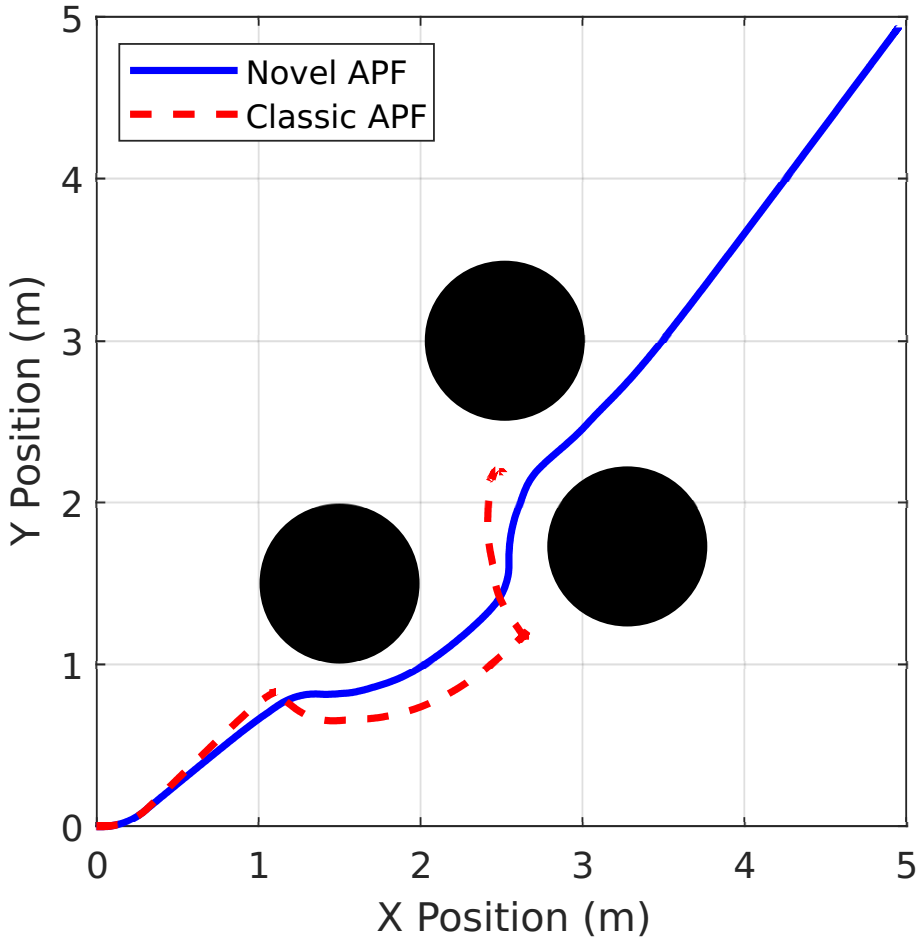

Figure 4.15: Simulation 3: Novel and Classic APF with multiple obstacles.

It can be seen that the classic APF approach reaches a local minimum here, where the various repulsive forces in combination with the controller all balance out, causing the agent to become trapped. However, in the novel APF, the repulsive forces are notably lower as a result of the obstacles not being directly ahead of the agent. While facing towards either of the last two obstacles, the novel APF will cause the heading of the agent to shift towards the gap between the agents, which causes a reduced repulsive forces.

To demonstrate this, the approximated heading factors can be plotted. In Fig.4.16 the obstacle approach is used to approximate the overall gain that each obstacle would be applying to the agent as it moves through the environment. The agent has fairly distinct phases in which each obstacle has a strong impact, with a small crossover between Obstacles 2 and 3. Since the obstacle that the agent is facing has

the dominant repulsion, it is naturally pushed away from that agent, but once it is pushed sufficiently far away it is not encouraged to continue turning until it faces the next. It can be seen that as the force from the third obstacle (in green) has been reduced to the minimum value for an agent within the potential field, the value does not climb again. It does not push back and forth between obstacles, but is able to reach a balance point.

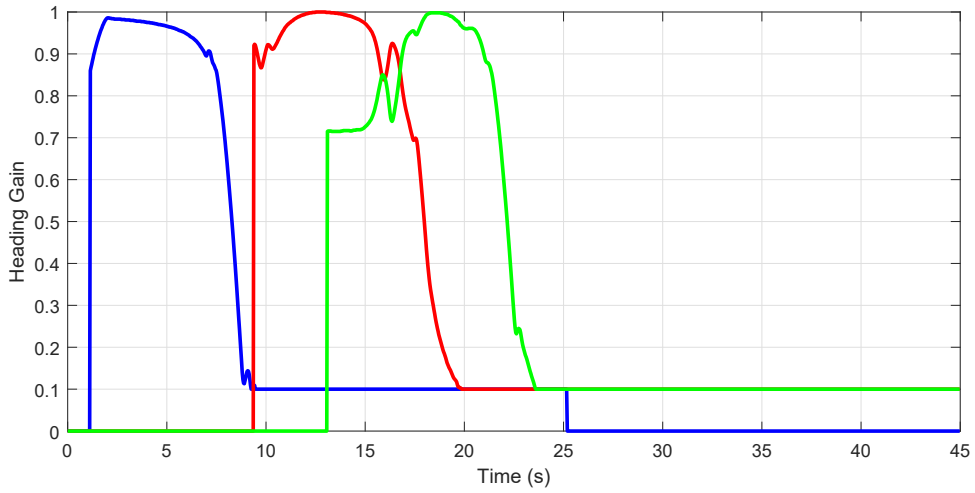

Figure 4.16: The heading gain from each of the three obstacles.

It is also worth noting that the heading factor for each agent tends to rise and fall fairly quickly, with minimal oscillations. Even with multiple obstacles pushing on the agents, the result is still generally quite smooth. This shows that the

The results of these three simulations are compared below in Table 4.1. Here it is demonstrated how the classic and novel APF algorithms compare in distance travelled, as well as in time to reach the goal point. The environment being explored is 5 m $\times$ 5 m, and the agent starts at a point (0, 0). The goal is to get within 5 m in the x- and y-directions of the goal at (5 m, 5 m) and will be considered "reached" when the agent is within 0.07 m of the goal point. that the shortest path would be a direct line to (4.95 m, 4.95 m) in each direction. The line between the start and this nearest part of the goal is 7.00 m in length. As such, distances travelled should be considered relative to this absolute minimum (although with obstacles the true minimum would be larger). The maximum velocity applied to the Turtlebots is 0.22

m/s, meaning that the minimum time to the goalpoint would then be 31.8 s, though this ignores that the agent will need to accelerate.

Table 4.1: Results of Novel and Classic APF simulations for distance travelled and time to objective.

Simulation	Dist. (C)	Dist. (N)	Time (C)	Time (N)
1	8.04 m	7.36 m	50.2 s	35.9 s
2	7.78 m	7.18 m	40.9 s	34.8 s
3	-	7.65 m	-	38.5 s

The results show a dramatic improvement in both the distance travelled and the time to reach the goal point. The time could be improved somewhat, as the sliding mode controller used slows down as it approaches the goal, but this small time loss is something that we consider acceptable. Note that there is no distance or time measurement for the classic APF in the third simulation as it was unable to complete the course.

Simulation 4 (Fig.4.17) was conducted with a much more complex environment. The enviornment contains a wall which blocks much of the path that the agent would naturally want to take, which leads to the agent being naturally guided to follow a less direct path in the beginning of the trajectory. From here, the agent gradually approaches the wall until a sufficiently large force causes it to align parallel with the wall, tracking along this way until passing the corner. Once past the wall, the agent is able to navigate around two cylindrical obstacles, generally getting close to them, and then travelling parallel to their sides until a direct path to the goal becomes available. The affect of the potential field is mostly minimal unless circumstances occur which could lead to a potential for a collision, which leads to the smooth results seen in Fig.4.17.

This demonstrates the effectiveness of the algorithm at navigating through complex environments. There are still limitations present in the algorithm however. It is not adept at handling maze-like environments with prominent dead-ends. It is not designed for this type of global path planning, but rather for local obstacle avoidance. These results have shown it is exceptionally capable of avoiding obstacles along its path.

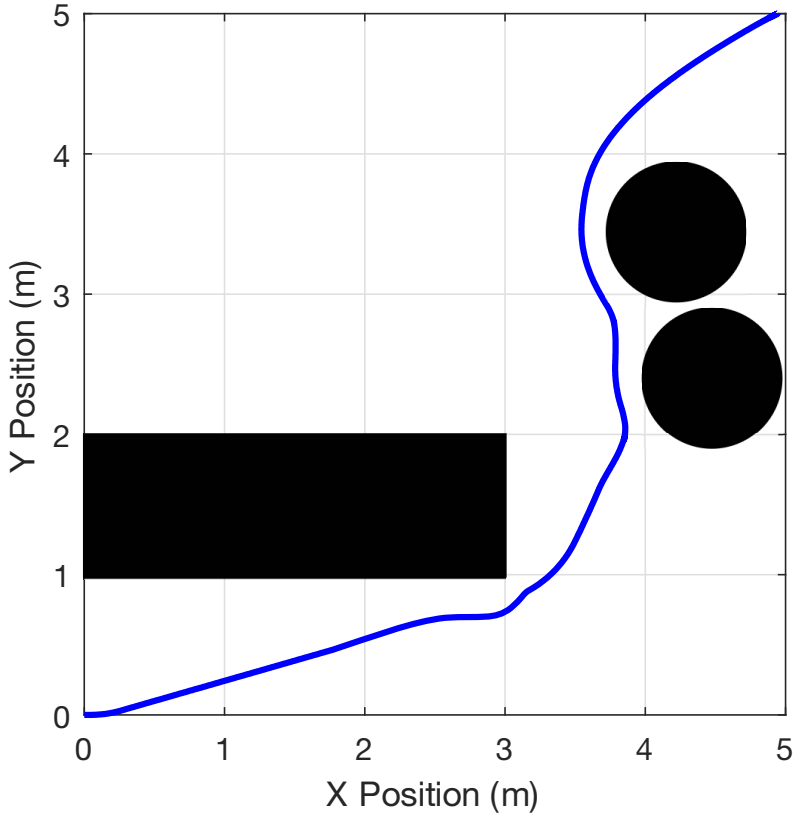

Figure 4.17: Simulation 4: Novel APF in more complex environment.

4.4 Experimental Results

In order to validate the simulated results, a series of experimental tests were conducted on both the novel APF and the classical APF in order to determine how they will behave on real hardware. The primary distinguishing difference will be the behaviour of the LiDAR sensors, as the simulated sensors in Gazebo will behave perfectly. A picture of the second experimental test is shown below, with three obstacles that the agent must navigate through.

Figure 4.18: An example of an experimental setup with a Turtlebot and three obstacles.

The first experiment recreates the first simulation; demonstrating the effectiveness of the novel APF and avoiding a single obstacle in the agent's path. A large garbage can was found to serve as the obstacle, although it is still a bit smaller than the obstacle used in simulation. The results are largely the same as what was seen by the simulation. Fig.4.19 presents the results of this simulation.

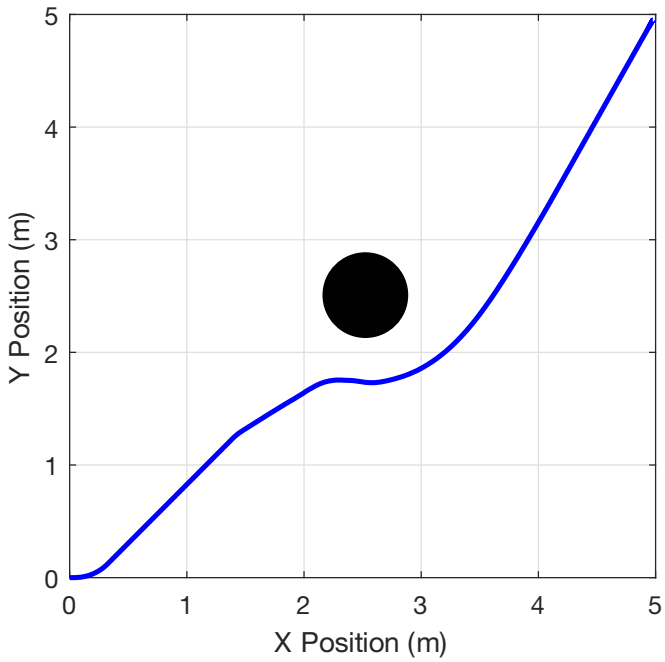

Figure 4.19: Experiment 1: One agent navigating around one obstacle.

The behaviour in this simulation very closely mirrors what was seen in the simulated testing. The obstacle is a bit smaller, so the agent gets closer before reacting, but otherwise this is a very precise recreation of the existing tests.

A second experiment was conducted that used three obstacles for the agent to navigate through, comparable to the third simulation. Here, the obstacles are buckets filled with sand, and are notably smaller than the 1 meter diameter cylinders used in the simulation. However, it still serves as a good demonstration of the utility of the avoidance protocol. Fig.4.20 presents the results of this simulation.

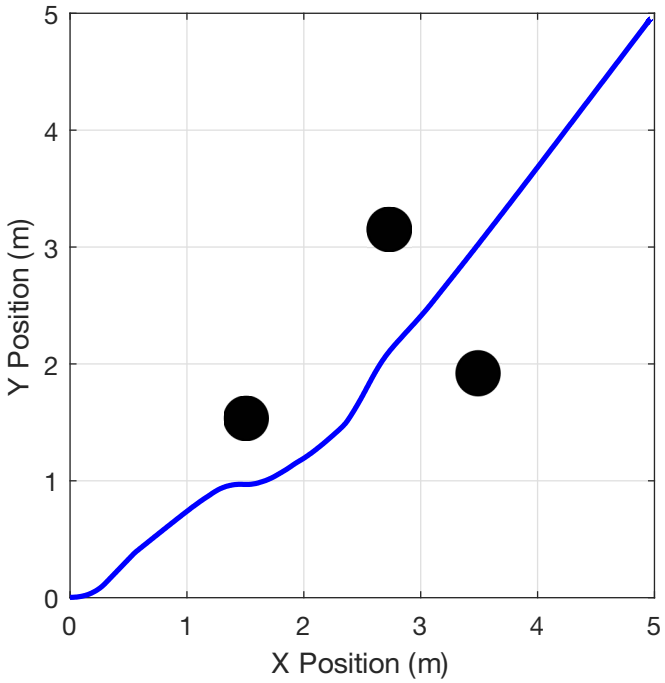

Figure 4.20: Experiment 2: One agent navigating through three obstacles.

Here it can be seen that the agent is able to smoothly navigate between the obstacles without making any prominent detours. There is a good amount of clearance provided, and there is no significant chattering present in the result. After this result was collected, a new simulation was produced which recreated the environment in order to create a direct comparison between the simulated and experimental testing. This is shown in Fig.4.21.

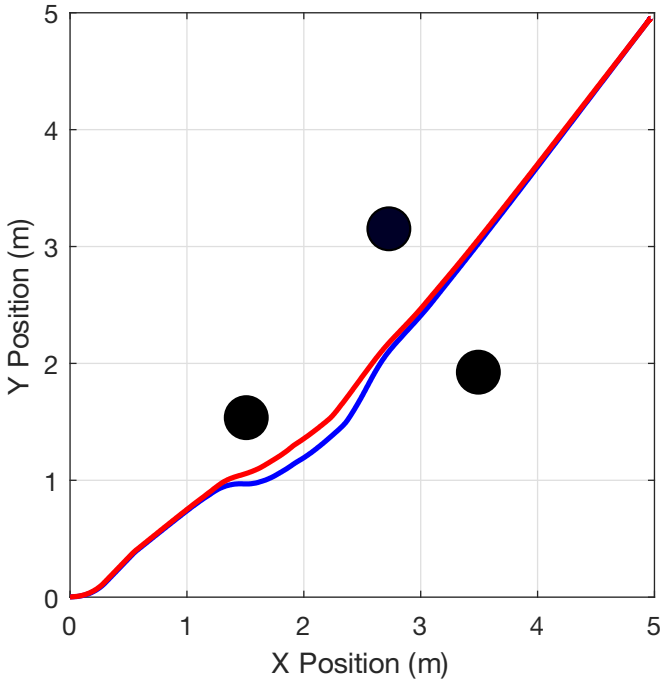

Figure 4.21: Comparing simulated (red) and experimental (blue) results

The performance of these two tests is quite similar, however the simulation appears to be a bit less effected by the repulsive field than the experimental case, leading to a deviation of about 10 cm in the two trajectories. They do approximately converge back together as they approach the goal point, which makes sense as the greater push applied to the agent in the experimental case will cause a greater clockwise rotation around the first obstacle, and a greater counter clockwise rotation around the second obstacle.

4.5 Conclusion

In this chapter a novel APF style obstacle avoidance algorithm has been developed. The primary advantage of this approach to obstacle avoidance is the agent is able to more effectively respond to an obstacle, with optimizations to reduce the distance travelled and the time to reach the goal point. The adjustments also produce smoother trajectories with reduced chattering in the control signals. It has been shown that it can be deployed for avoiding obstacles in an unknown environment.

Chapter 5

Formation Shaping with Dynamic Leader Selection

Within this chapter a method of time-varying formation will be formulated, discussed, and evaluated across multiple different simulated and experimental tests. This approach has the formation adjust the size of its shape in order to navigate an environment based on the obstacles within the environment.

The agents use the previously developed TSMC to move from the initial positions to the goal point and use the local obstacle avoidance to handle obstacles that may get too close. The formation shaping is applied in addition to provide an additional avenue to navigate environments with tighter constraints. By allowing for a dynamic formation, all of the agents can make adjustments at once in order to simplify processes.

This process uses a leader-follower dynamic with multiple leaders which evaluate the environment for potential obstacles which could require avoidance. However, the onboard sensors do not distinguish between obstacles and other agents. This creates a potential problem, as leaders could see agents ahead of them as potential obstacles depending on the configuration of the formation.

In order to manage this concern, a dynamic leader selection protocol is implemented which allows the MAS to select a set of leaders based on the desired formation and the goal point. By doing this, the leader agents will always be at the head of the formation, which should minimize the risk of seeing other agents as forward obstacles. As the system can respond rapidly from the sensor information and there is no change to the overall path planning, this is considered a reactive approach to obstacle avoidance.

5.1 Objective and Assumptions

The control objective is to have the overall MAS achieve a pre-determined formation from arbitrary positions in finite time using a fast TSMC approach. The formation

will navigate an environment with a series of obstacles, changing its shape in response to sensor data. The work here is conducted with a homogeneous team of Turtlebot3 agents.

Assumption 5.1 *The agents can be assumed to exist in a 2D plane. As the agents are ground based and the environments would be buildings where the floors are reasonably flat, everything can be approximated to a 2D environment. It should be possible to scale up the basic processes to 3D, however this was outside the scope of the testing for this chapter.*

Assumption 5.2 *Environments are designed such that it is possible for the formation to navigate through the gaps by changing the shape of the formation. The robots have physical size, and in theory environments could be constructed which are non-navigable due to this physical limitation. Cases which cannot be navigated are thus not tested.*

Assumption 5.3 *All obstacles are consistent in the Z-direction. The LiDAR sensor can only detect obstacles within a single plane, and so it is assumed that there will not be obstacles that it cannot detect.*

The overall objective of the formation shaping is demonstrated in an illustrative example presented in Fig.5.1. The agents approach a set of obstacles and the formation is too large to pass through the gap between them. In order to do so, the formation is reduced in size. This is done through a shift in the distances between agents. The distances between the two leader agents at the front of the formations are reduced, and the distances between leaders and followers can also be scaled to shrink the total formation.

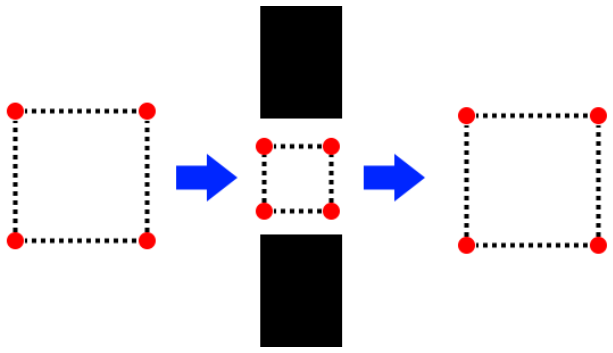

Figure 5.1: A simple visual demonstration of the objective of the mission.

5.1.1 Leader Selection

The leader selection strategy proposed in this paper is completed by determining the position of center of the formation, and then calculating the bearing to the current goal point. The desired positions of the agents within the formation are then used to determine which agents should act as leaders. The agents whose desired positions would be closest to the goal are selected. Fig.5.2 shows a sample of this, with the four quadrants each being used to select two leaders for different team sizes and configurations. For this process, the white circles indicate the desired positions of the agents, not their actual positions. Especially at the beginning of the process, the agents can be in any arbitrary position, and so using the desired position ensures that the optimal agents will be selected by the process.

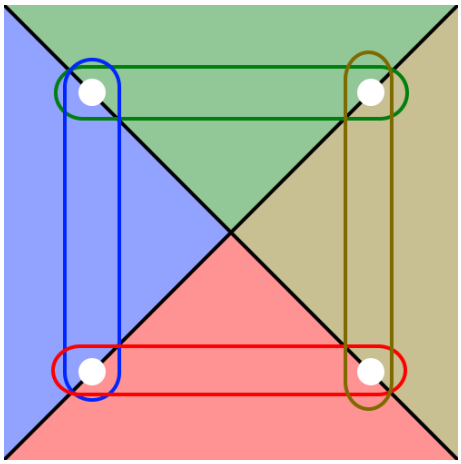

Figure 5.2: Visualization of the leader selection process.

The process can be designed to calculate new leaders depending on various circumstances. In this work it is set such that agents will assign new leaders at the initial time and when a goal is reached. Alternative methods include update based on a period of time, or using ETC.

In this case, depending on the location of the goal point, a set of leaders are selected and a network of followers can then be constructed from those leaders. Fig.5.3 shows a sample communication topology for four agents. Any adjacent pair of agents can serve as the leaders for this case, and an equivalent communication network could be produced.

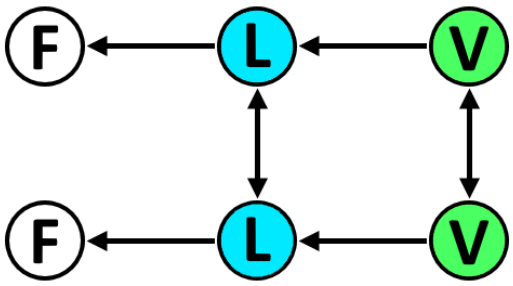

Figure 5.3: An example communicatoin topology with two virtual leaders.

A set of lines are defined within a local coordinate system where the origin is the center of the formation. As such, all of these lines will be of the form $y = mx$. A vertical line of $x = 0$ can also be deployed if the formation requires it. These lines are designed to pass through the desired positions of the agents within the formation. The local coordinate system is offset from the global position by x_c and y_c where x_c is the center of formation in the x direction and y_c is the center of formation in the y direction. There is no rotational change between the coordinate systems.

From this the current goal point (x_g, y_g) is translated into the local coordinate system by applying the following translation

$$\begin{cases} x_{g,local} = x_g - x_c, \\ y_{g,local} = y_g - y_c, \end{cases} \tag{5.1}$$

where $x_{g,local}$ and $y_{g,local}$ refer to the position of the goal point in the local coordinate system. From here, the local goal position is compared against each of the lines, and depending on the set of responses, the leaders are selected. This can be conceptualize based on Fig.5.2, where lines of $y = x$ and $y = -x$ are used. Based on the values of $x_{g,local}$ and $y_{g,local}$, $y_{g,local}$ is then compared to $y(x_{g,local})$ to determine the quadrant. The generalized mapping is presented in Table 5.1.

Table 5.1: Mapping of leader selection based on current goal position.

$y = x$	$y = -x$	Quadrant
$y_{g,local} \geq y(x_{g,local})$	$y_{g,local} \geq y(x_{g,local})$	Green
$y_{g,local} \geq y(x_{g,local})$	$y_{g,local} < y(x_{g,local})$	Blue
$y_{g,local} < y(x_{g,local})$	$y_{g,local} \geq y(x_{g,local})$	Yellow
$y_{g,local} < y(x_{g,local})$	$y_{g,local} < y(x_{g,local})$	Red

A similar mapping of quadrants can be constructed for any arbitrary formation. The process of leader selection can be run in a number of ways. It could be done at specific time intervals, or by way of event-triggered control. In the cases which will be discussed here, the leaders are selected from initialization, and will be recalculated each time a goal point is reached.

This basic process can be scaled up to an arbitrary number of agents and leaders, though depending on the desired formation it may be desirable to specify some agents as potential leaders and others as always being followers. This would depend on the exact desired setup. A number of additional formations are considered using 3 or 6 agents in Fig.5.4.

In Fig.5.4(a) there is a set of three agents, with 120 degree windows instead of the 90 degree seen with four agents. In Fig.5.4(b) and (c), two different configurations of 6 agent formations are considered. It should also be noted that while only two leader setups are implemented here, this is not a requirement. In the case of (c), the system could be deployed with a dynamic number of leaders. So a goal above or below the formation would have three leaders, while to the left and right only two leaders would be deployed. The system is capable of handling any number of leaders, the priority is simply to have them operating without any other agents in front of them as they navigate towards the goal point.

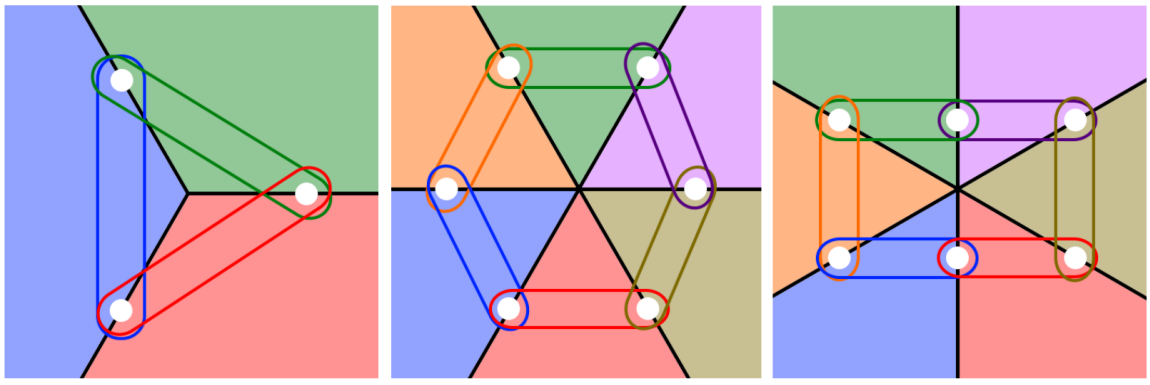

Figure 5.4: Visualization of three additional leader selection processes.

Larger teams can also be deployed. A team of 16 agents in a grid are considered two different ways in Fig.5.5. First, only the corner agents can be considered as leaders, with the remaining 12 agents all acting as followers. Here the leader selection

process would be the same as in the previous square case. Additionally, a case with all exterior agents capable of acting as leaders is presented. In this case, the four agents along the edge that the formation travels would act as leaders and be capable of collaboration. This framework allows the formation to scale without having increasingly large numbers of leaders.

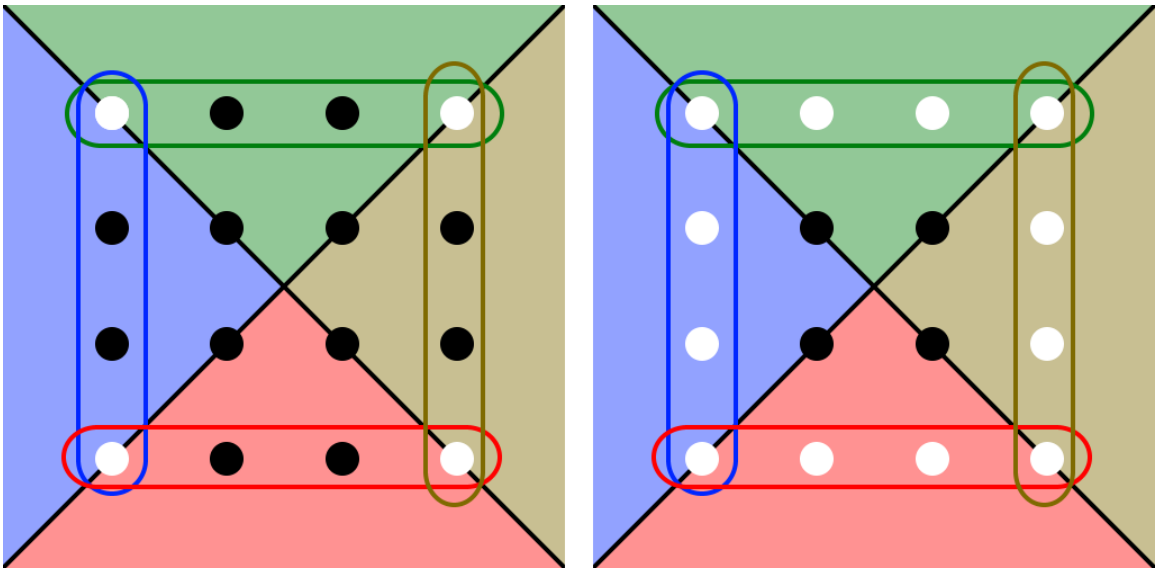

Figure 5.5: Leader selection for teams with a large number of agents.

5.1.2 Formation Shaping Control

In order to properly navigate in an unknown environment, the shape of formations can be adjusted in order to allow for the formation to fit through various passages. Sensor data is combined from multiple agents in order to produce a rough point cloud of the local environment. In the event that a wall appears to have formed, the agents will look for a region to pass through, which may be a gap in the wall, or the end of the wall. Once such a gap is found, the agents will approximate the size of the available passageway, and make adjustments to the formation in order to allow passage if such adjustments appear to be required. The overall process for control of the agents can be summarized as in Algorithm 5.1.

In a number of formulations for this type of formation control, a bearing based approach to formation shaping is implemented. This is a useful approach, as the definition of the formation does not change. Fig. represents this by showing a square

Algorithm 5.1 Determination of Formation Shaping

```
while Goal not reached do
    Calculate center of formation
    Determine quadrant
    Assign leaders
    Create topology
    Scan environment
    Detect obstacles
    if Path to goal obstructed then
        Evaluate obstacle positions
        Determine if gaps exist
        if Gap smaller than current formation then
            Begin formation shaping computations
        end if
    end if
end while
```

formation. In it, the formation is defined by the headings of the five lines. These headings will remain constant at all times, but the formation can change without its definition changing.

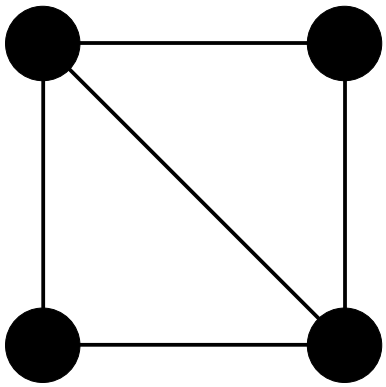

Figure 5.6: A visualization of the basics of a bearing-based formation.

However, the rigidity of this formulation can provide limitations. By requiring the bearings between agents to remain constant, the system is limited in its ability

to respond. As such, the position based approach considered provides a more robust set of options to the formation. Some of this is still possible within a bearing-based formation in which the bearings can be varied, however variation in bearings is rarely considered in literature, and can have some very pronounced limitations depending on the use case. For example, a line formation would be difficult to properly implement as all angles are equal within the formation.

In order to determine the changes that are required to the system, measurements of the environment are needed. A variety of approaches to simultaneous location and mapping (SLAM) are available, however these all provided a much more comprehensive, and by extension computationally expensive, approach than what was needed for this case. The agents are not developing a complete map, they are simply evaluating if the passage that they are travelling in, or about to travel in, is too small for the formation.

Of particular note, this is specifically designed for passages and gaps between obstacles, and so singular obstacles are ignored as not being relevant to this particular style of avoidance. In such cases only the agents which would be impacted by the obstacle will move.

Shaping

After determining the gap size in a wall, the agents change only the desired positions relative to the center of the formation, and the controller will have the necessary adjustments. As the formation control in this case is displacement based, the goal point can be modified based on the gap available. The process is defined in Algorithm 5.2.

The primary limitation of this method is the sizing of the robots. The algorithm adjusts the scale of the formation, but it does not change the fundamental shape. As such, if a square formation with four agents is attempting to manoeuvre through a gap that is smaller than two agents in width, it will not be possible for the formation to pass through the gap.

For the formation shaping, it is possible to define a desired formation distance x_f between the two leaders of the system. This exists in practice as a vector, as the agents have a specifically desired configuration, however we will only account for the

Algorithm 5.2 Determination of Formation Shaping

```
while Goal not reached do
    Scan environment
    Identify walls from sensor data
    if Obstruction has gap then
        Calculate if formation can pass
        if Gap large enough for team to pass then
            Determine formation size
            Calculate formation adjustments
            Begin formation shaping computations
        end if
    end if
end while
```

magnitude for the moment. A general function can then be defined for the formation such that

$$x_i = x_f^p. \tag{5.2}$$

Here, p defines a performance measure, and is used to adjust how the rest of the formation will adjust. x_i is the distance between agents at the ith edge of the communication matrix. For this displacement value, different values can be set for different pairs of agents as is necessary. Generally $p \leq 1$, as making it larger would bring the agents closer together as the formation grew tighter. The idea of having a p value less than 1 is that the agents do not need to get as close in the direction they are travelling, as a sudden breaking incident could lead to a collision between agents.

This is visualized below in Fig.5.7 with a p value of 0.33. Here the formation is initially (in red) a square. The formation then compresses down such that x_f is 50% of the original size. As such, with $p = 0.33$, x_i becomes 80% of the original distance behind the leaders. Keeping this separation helps to reduce stuttering in the follower agents, as the APF algorithm can cause prominent repulsive forces to be applied to their controllers, which can lead to some chaotic results.

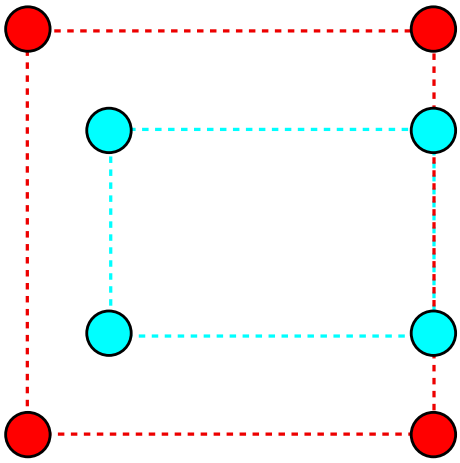

Figure 5.7: Visualization of the formation modifier with $p = 0.33$.

5.2 Simulations

Three simulations were conducted in order to validate the controller. The first of these simply passes a group of four agents through a gap. The second has the team move through a much more complex environment with multiple leader changes. Finally, a team of six agents is run through the same environment to show that the algorithm can be applied to any team size.

It is worth noting that the radius of the APF algorithm's repulsive field was lessened in this experiment. This was done due to agents getting close together during the tightening of the formation. Depending on circumstances this can certainly lead to concerns, however with the slow speeds on the agents it was deemed acceptable.

5.2.1 Case 1: Test Case

For this first case, the agents will pass through a gap in a wall. The agents are initialized in a vertical line, however any arbitrary positions can be used. Once the two leader agents (on the right side of the formation) have passed through the gap the formation expands back to its original size. It can be seen that the agent in the top left corner of the formation has a close call with the corner of the wall, but the local obstacle avoidance causes it to turn just enough to avoid the potential collision. Fig.5.8 plots these results.

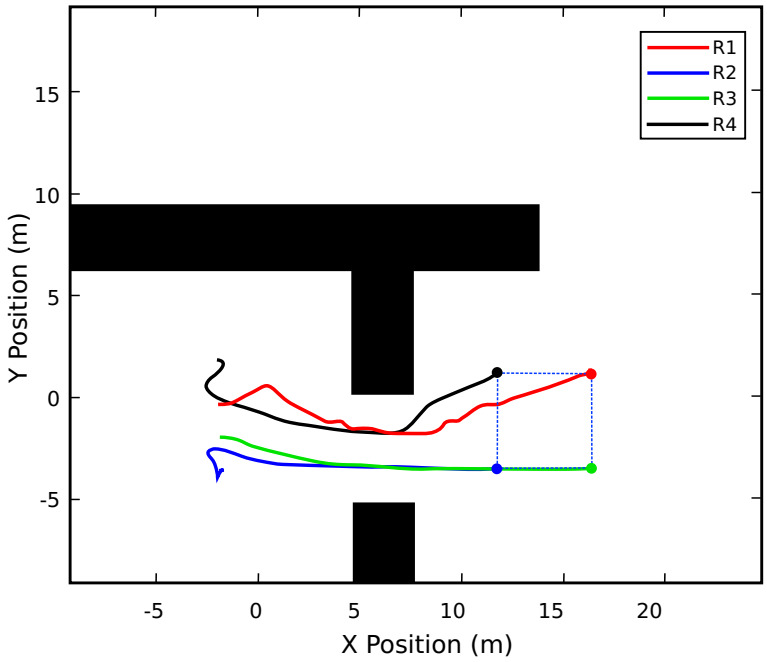

Figure 5.8: Simualtion results for the first case.

The agents are able to get into formation in a reasonable time frame, and the errors generally stay at zero once reached. There are some small fluctuations, such as in Agent 1 at $t = 15$s, as the APF causes the agent to move out of the way of the corner of the wall.

5.2.2 Case 2: Four Agents

For the next two simulations, an environment was designed with three goal points which must be reached sequentially. The aim of the environment is to present a case of moving through a gap in a wall, as well as more open motion. In addition, multiple different sets of agents in the formation assume a role as leader.

Once the center of the formation has reached the goal point, the next goal is assigned and new leaders are determined. This process is repeated until the agents reach their final goal point. For the first case with this setup, a group of 4 agents in a square formation move through the environment, as shown in Fig.5.9. This shows the agents at four different times: first while passing through the gap, and then at the point of reaching each of the three subsequent goal points. To reach the final goal point takes 80 seconds. At any point in time, the two leaders will always have an

equal position error, as their desired position is relative to each other.

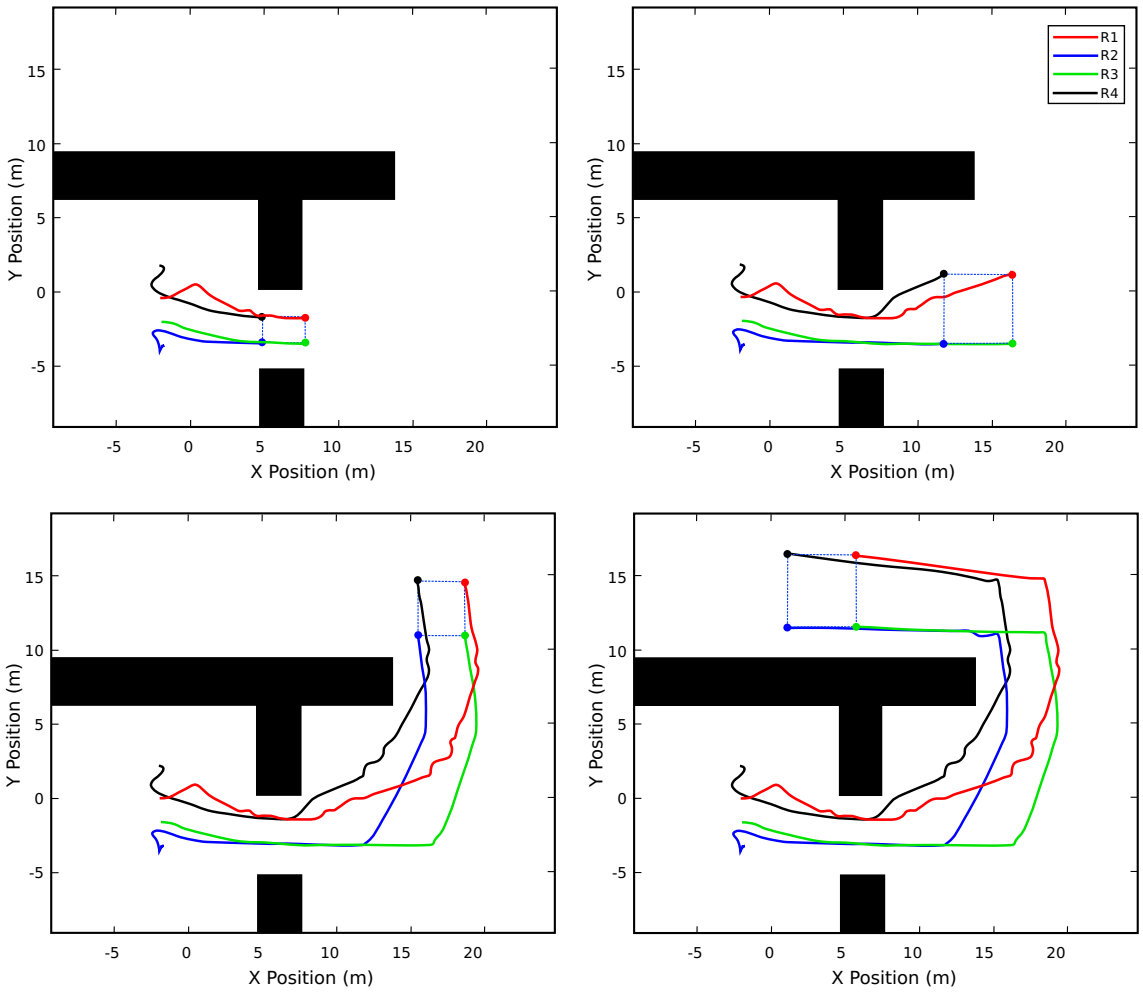

Figure 5.9: Simualtion results for the second case.

Here it is seen that the agents are able to get into formation after approximately 15 seconds (shortly after the first agents have passed through the gap in the wall) and maintain the formation until reaching the first goal point. From here, small errors become present when the agents are assigned a new goal point, but they remain relatively small and correct back to approximately 0 before the agents reach their next goal.

5.2.3 Case 3: Six Agents

Finally, a new simulation was conducted which used the same environment and goal points, but was completed with a team of six agents in a hexagonal formation. For this case, the starting positions of the agents are changed from the vertical line arrangement from the previous simulations. This was completed to demonstrate the robustness of the control scheme. Fig.5.10 shows the results of the agents moving through the environment.

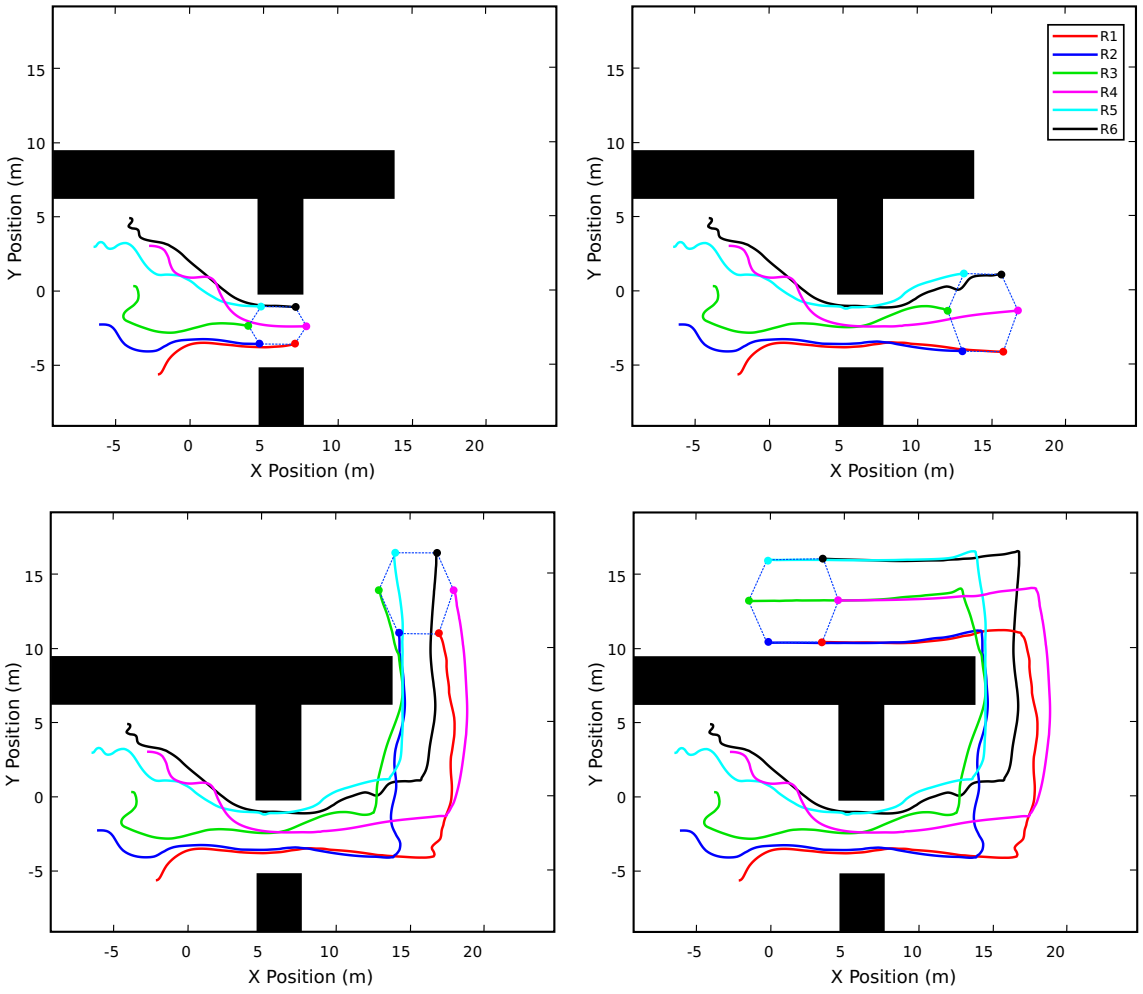

Figure 5.10: Simualtion results for the third case.

From these results we see the error values behave similarly to the four-agent case, with small perturbations when a new goal point is assigned, and the agents returning to the desired formation shortly after.

5.3 Further Implementations

5.3.1 Combination with Gaussian Process Motion Planning

This approach to formation shaping was also combined with a GPMP2 approach to path planning to develop a system capable of navigating a defined environment with approximately optimal trajectories. This uses a role-based collaboration (RBC) process based on E-CARGO, a system that uses "Environments, Classes, Agents, Roles, Groups and Obstacles"

The agent a_i can find an approximately optimal process role through a posterior function and reliability, based on Gaussian density. The process role is represented by a continuous function that is demonstrated with a Gaussian Process (GP) and a corresponding Markov model. With a small number of states n_s and their interpolation, the GP function can be efficiently computed using a gradient-based optimization algorithm. The Markov model of GP, which involves dynamic states, can be used to express GP. The role r is updated by probabilistic inference that considers direct and indirect factors. In each iteration, the MAP estimation of the function is computed, based on a set of likelihoods, to find the optimal role r.

$$r = \arg\max_{r} P(r|e), \tag{5.3}$$

where e is a set of random binary events of interest. The proportion of posterior distribution of r given e can be derived from the prior $P(r)$ and likelihood $L(e|r)$ by Bayes rule:

$$P(r|e) \propto P(r)L(e|r). \tag{5.4}$$

This formula will be represented as the product of a series of factors [88],

$$P(r|e) \propto \prod_{n_f=1}^{N_f} f_n(r_{n_f}), \tag{5.5}$$

where f_n are n_f factors on state subsets r_{n_f}. It is shown in [88] that this MAP problem can be solved efficiently by exploiting the sparsity in the factor graph. If process roles are interdependent, such as two trajectories that require a certain formation, agents must communicate and share their assigned roles and reach a Bayesian consensus to ensure that they all agree on the next actions to take.

Simulated Results

Initially, a control simulation was conducted which only used the TSMC, without the GP inference or any formation shaping, to adjust the size of the formation so that we can see how this approach improves the results later. Fig. 5.11 shows the results of this simulation. The agents are able to navigate through the environment, however the controller is constantly trying to maintain the desired formation while avoiding the obstacles with the local obstacle avoidance. As a quick note, this test was done prior to the novel APF development, and so this is using a classic APF approach. The classic APF leads to rough paths and increased risk of collisions. There are clear avenues for improvement in this control simulation that more optimized trajectories can provide. The addition of the GPMP2 and formation shaping seek to achieve in the subsequent simulations using the GP inference algorithm.

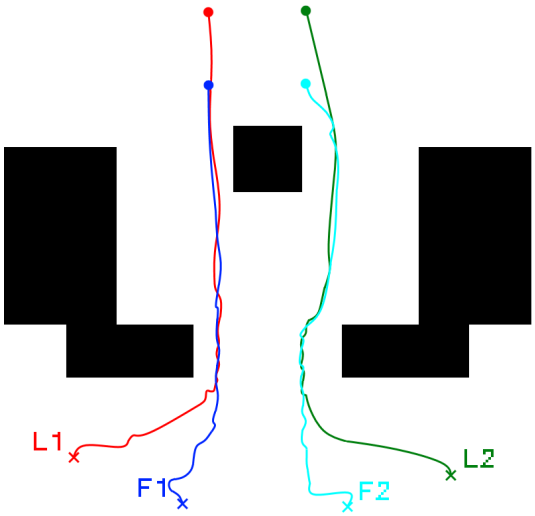

Figure 5.11: Four agents using just TSMC and obstacle avoidance without GP inference.

The GP inference algorithm was designed to create an approximately optimal trajectory for the agents in the simulation. In Fig. 5.12, an example is shown in the environment we will be testing in. The algorithm determines an approximately optimal trajectory. For the first case being studied (Case 1), the final goal points for the leaders are known in advance, and the GP inference algorithm is creating the trajectories based on this information.

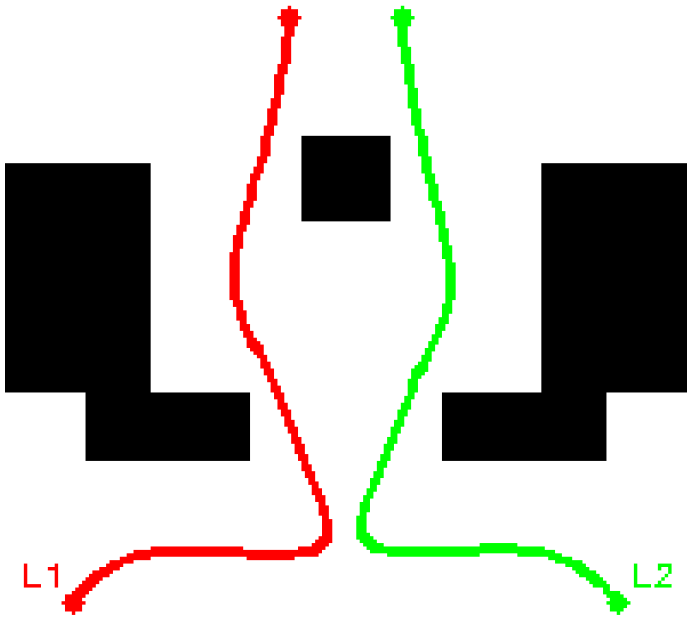

Figure 5.12: Leader trajectories without formation factor

It can be seen that the GP inference will generally avoid producing sharper turns than necessary. The motion includes large varieties in the spacing between agents in order to ensure that they will not collide with each other or with the obstacles. This is effective for getting from the initial point to the goal point, however the agents do not maintain a formation during this process. By applying an additional formation factor, the agents can maintain a much more consistent formation while navigating through the environment. This will allow for the two agents controlled here to serve as leaders for a formation, with followers being controlled using the TSMC.

For Case 1, shown in Fig. 5.13, the formation factor is utilized to develop the trajectories. The dashed line is the originally calculated trajectory from Fig. 5.12, and the solid line indicates the newly calculated trajectory when accounting for the formation factor. The GP inference algorithm provides a very smooth trajectory through the environment, with the formation varying in size depending on the current situation. The followers will then be able to use this information to determine where they should be relative to the leaders.

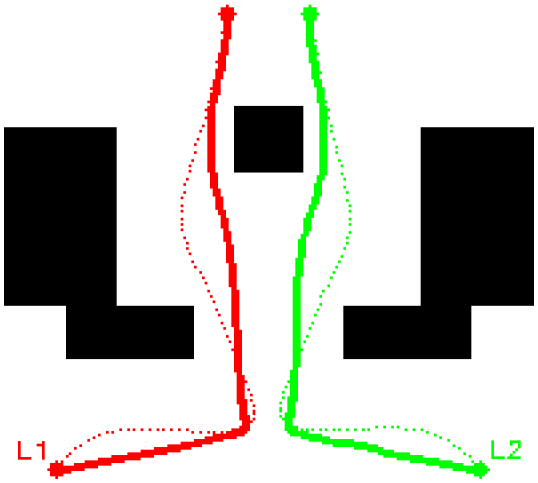

Figure 5.13: Trajectory generation for the leaders in Case 1.

Using the results collected from the GP inference simulation, a simulation in Gazebo was conducted. As in Fig.5.14, the four agents are shown to follow the desired trajectories quite well. The leaders are indicated in red (L1) and green (L2), with the followers (F1 and F2) indicated in blue and cyan, with crosses used to mark the initial positions, and circles used to mark the final positions.

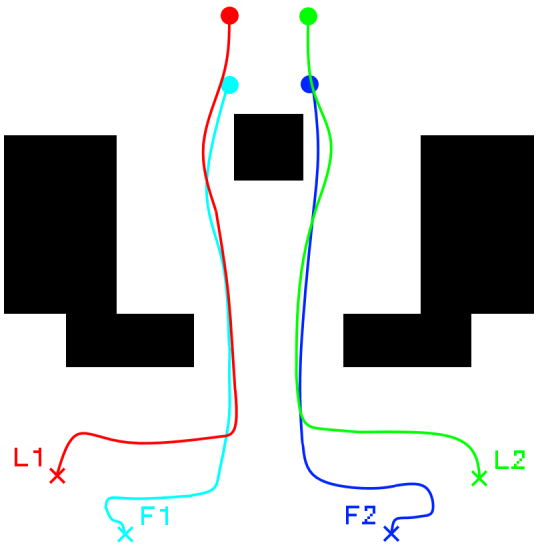

Figure 5.14: The two leader agents (red and green) along with the two follower agents (blue and cyan) in a Gazebo simulation in Case 1.

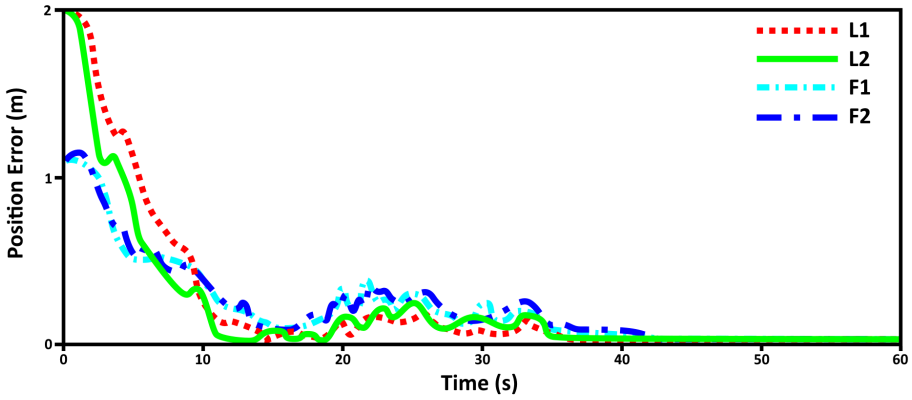

Figure 5.15: Positional errors for agents based on the desired formation.

As can be seen, the agents will create a smaller formation while passing through the gap and then more smoothly transition to a wider formation to get around the small block in their way to the final destination. It can be seen that initially the agents do not immediately track perfectly to the desired trajectory. The desired trajectories are developed from an algorithm that does not consider the dynamics of any given robot, and so the non-holonomic nature of the 2WMRs mean that their initial heading is not directly towards the desired trajectory, instead moving directly forward slightly before the angular input has been able to have a prominent effect on the agents.

The only major perturbation that remains is in the cyan agent, which needs to avoid the top right corner of the final obstacle. This is a product of it trying to follow behind its leader, but shows there is still potential improvements in the control scheme. The leaders do not perfectly track the trajectories assigned, especially in the initial conditions.

Two additional simulations (Case 2 and Case 3) were conducted which use two more environment configurations. The idea was to present two cases which are very similar to one another, but produce different results, then showing that the algorithm would be able to accomplish both of cases. Case 2, shown in Fig.5.16 and Fig.5.17.

What we see here is a fairly simple trajectory. The leaders have a mostly clear path to the final goal points, and so the trajectories follow a mostly smooth path. As L1 shifts to avoid the square obstacle in the top middle, we see that L2 also pushes to the right in order to compensate.

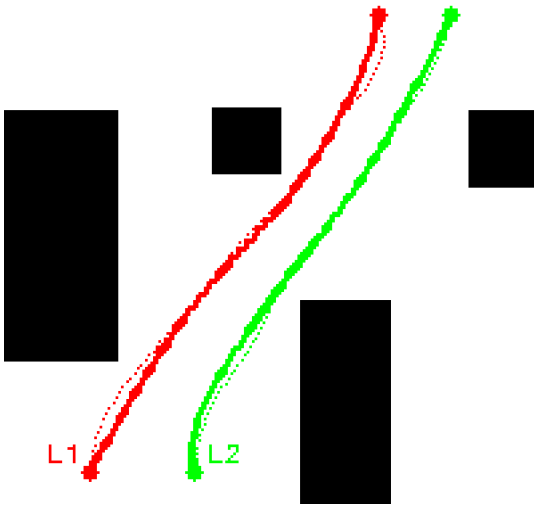

Figure 5.16: Trajectory generation for the leaders in Case 2.

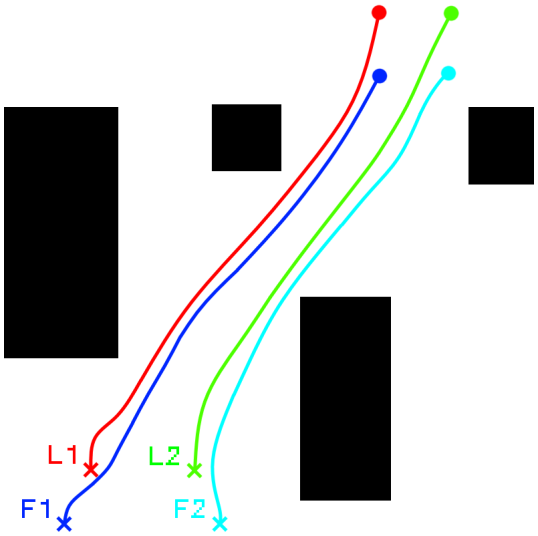

Figure 5.17: The two leader agents (red and green) along with the two follower agents (blue and cyan) in a Gazebo simulation in Case 2.

The generalities of the full simulation are as expected. L1 does not immediately get on its trajectory, but within a few seconds is on track, and more or less stays as such. Following these simple trajectories is not an issue for the controller, and the followers are able to maintain the formation quite effectively throughout.

The subsequent simulations for Case 3 (shown in Fig. 5.18 and Fig. 5.19) showcase

a very similar environment with different results. Here the obstacle in the top-center of the formation has been shifted a small degree to the right. In doing so, we see that the optimized trajectory now passes around this obstacle. In order to maintian the best possible formation case, $R2$ in Fig. 5.18 will navigate along the side of the obstacle instead of moving directly towards the goal point, a trajectory that is completely unimpeded.

Some future changes may be implemented to have the agents prioritize not having obstacles come between them if that is deemed desirable, however the current approach can be seen to do an excellent job of maintaining the broad structures of the formation, and adjusting it as necessary to work around the obstacles in the environment.

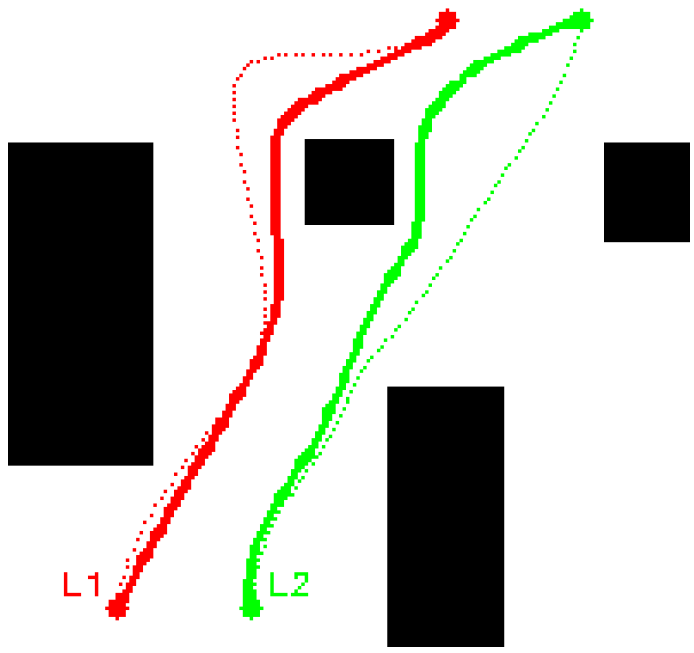

Figure 5.18: Trajectory generation for the leaders in the Case 3.

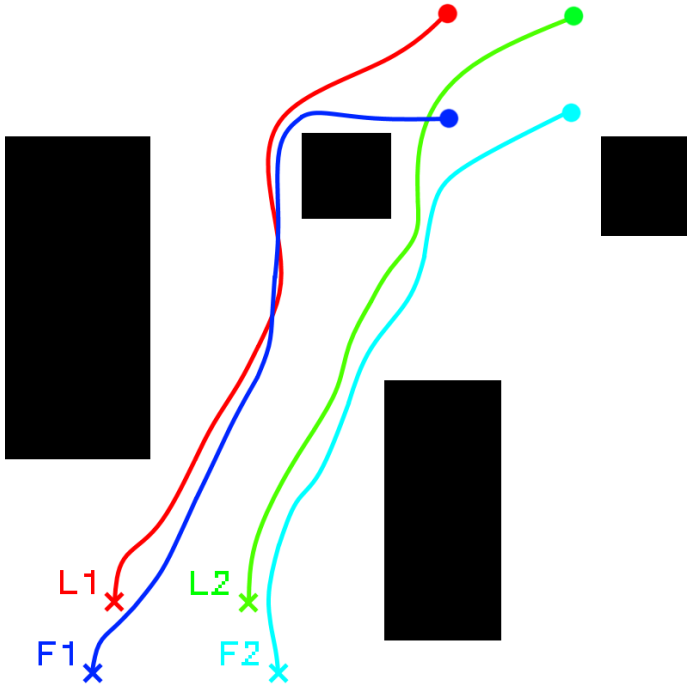

Figure 5.19: The two leader agents (red and green) along with the two follower agents (blue and cyan) in a Gazebo simulation in Case 3.

Experimental Results

An experiment was conducted using a group of four Turtlebot3 agents (3 Burger units and 1 Waffle unit). For this experiment, a set of boxes were used as obstacles for the agents to navigate. The environment for this experiment was designed to approximately replicate the Case 1 simulation. Fig. 5.20 shows the experiment, with (a) indicating the initial conditions, (b) and (c) indicating the agents in motion, and (d) showing the final positions of the agents.

As shown in the final conditions of the experiment (Fig. 5.20(d)), the agents approximately maintain their square formation, with the final positions of the Turtlebots coming to a roughly square formation. However, there are some small errors present in the final positions. This is caused by the limits of the onboard odometry. In this experiment, the agents do not have a known global coordinate system. Instead, the agents track their own positions via their IMUs, and have known relative offsets from one another in the initial conditions. In particular, small deviations in the initial

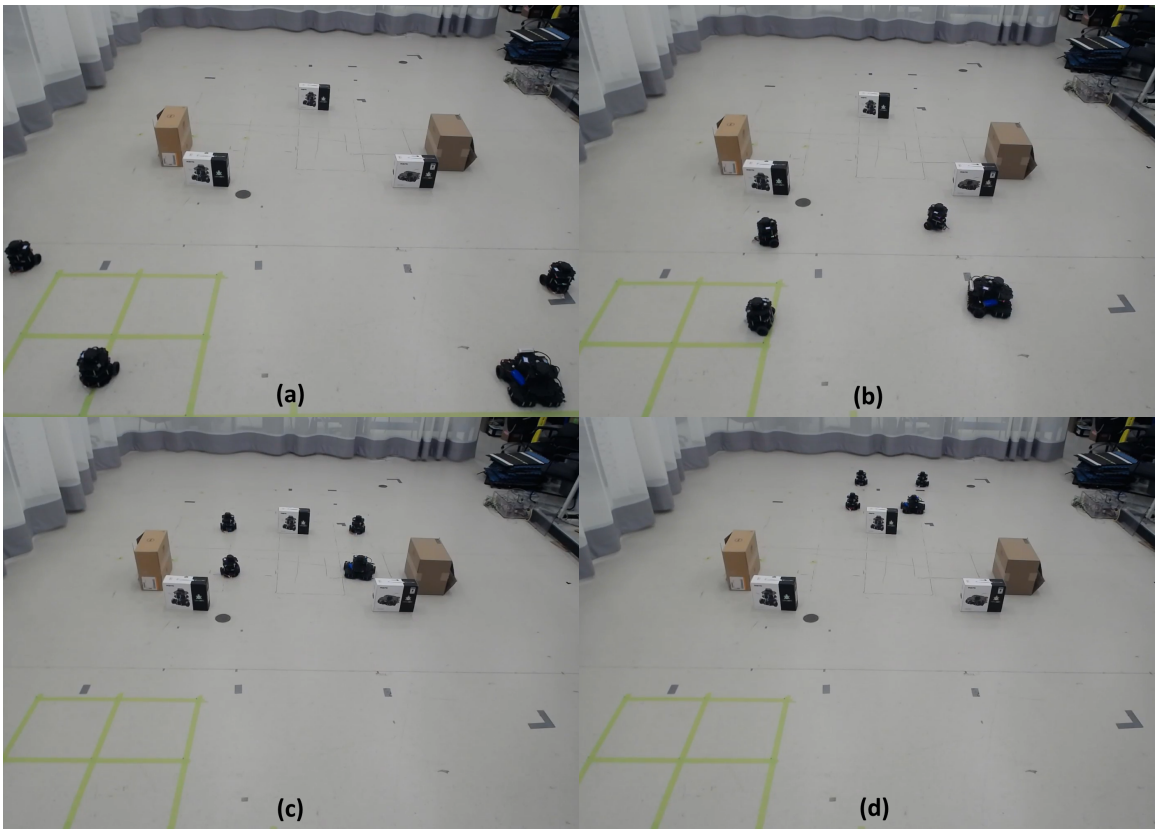

Figure 5.20: Experimental testing with four Turtlebots over time. From (a) to (d), occurring at $t = 0$ s, $t = 20$ s, $t = 40$ s, and $t = 60$ s.

orientation of the agent propogate through the experiment as a result of the non-holonomic nature of the agents. However the robots in the experimental testing still largely behave the same way as the simulations suggested. There are only minor shifts in the final position compared to what is expected. Having a global tracking system for the agents would alleviate this.

The map is provided to the GP inference algorithm as before, though there is potential room for error as the placement and measurement of the obstacles will not be as perfectly accurate as it is in simulations. It appeared while testing that the measurements were sufficiently accurate, as the behaviour of the agents was largely as expected. While the obstacles used were not identical to those used in the simulation of Case 1, they make for a serviceable stand-in, and largely replicated the results of the original simulation. Fig. 5.21 shows the results of the experiment. Video of this experiment is available at https://www.youtube.com/watch?v=03v7z-x6IUw.

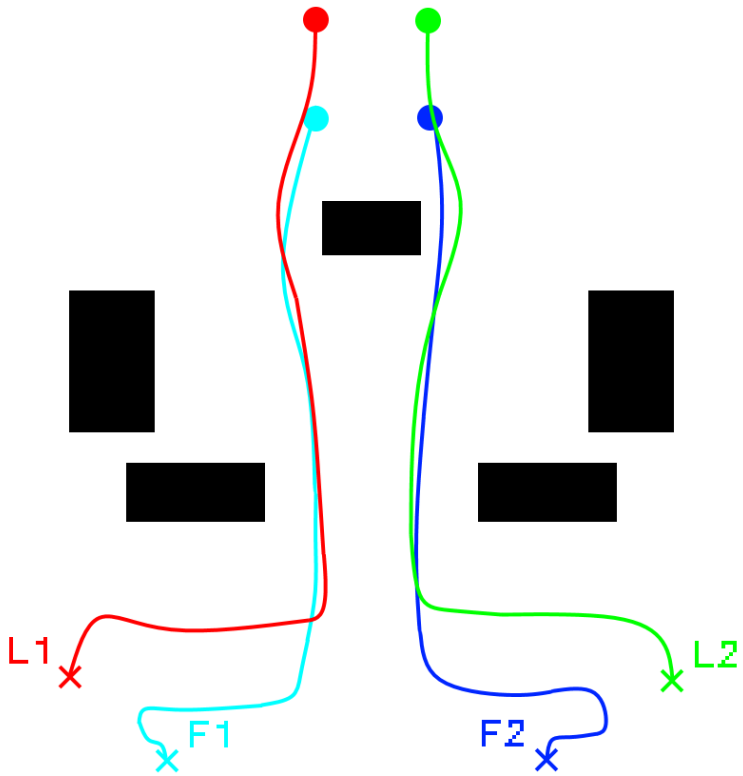

Figure 5.21: The results of the experimental testing.

5.3.2 Combination with Decentralized Tracking

The center of formation, denoted by $\boldsymbol{x}_c$, is estimated by each agent using their local information and information form neighbours. This can be represented as

$$\tilde{\boldsymbol{x}}_{c_i}(t) = \frac{\sum_{j=1}^{N} a_{ij}\tilde{\boldsymbol{x}}_{c_j}(t - T_{ij}) + \boldsymbol{\nu}_i(t)}{\sum_{j=1}^{N} a_{ij} + 1}, \tag{5.6}$$

$$\boldsymbol{\nu}_i(t) = \frac{\sum_{j=1}^{N} a_{ij}\boldsymbol{F}_j^i(t - T_{ij})\boldsymbol{x}_j(t - T_{ij}) + \boldsymbol{F}_i^i(t)\boldsymbol{x}_i(t)}{\sum_{j=1}^{N} \boldsymbol{F}_j^i(t - T_{ij})}, \tag{5.7}$$

where $\boldsymbol{F}_j^i(t)$ is the jth row of the formation weighting matrix for the ith agent. This uses a known desired formation, which improves the accuracy of the center of formation estimation. T_{ij} refers to random time-delays between agents [95]. The elements of $\boldsymbol{F}^i(t) \in \mathbb{R}^{N\times 2}$ are computed as follows:

$$F_{j,l}^i(t) = \begin{cases} (a_{ij}^* \bar{a}_i^* S_l^j(t))^{-1} & \text{if } a_{ij}^* \bar{a}_i^* S_l^j(t) \neq 0 \\ 0 & \text{if } a_{ij}^* \bar{a}_i^* S_l^j(t) = 0 \end{cases}, \tag{5.8}$$

where a^*_{ij} denotes the elements of $\mathcal{A}^* = \mathcal{A} + I_N$, $\bar{a}^*_i \in \mathbb{R}^{1\times N}$ is the ith row of the $\mathcal{A}^*$ matrix, and the kth element of $S^j_l \in \mathbb{R}^{N\times 1}$ is determined by

$$S^j_{k,l} = \begin{cases} 1 & \text{if } \delta_{k,l} = \delta_{j,l} \\ 0 & \text{if } \delta_{k,l} \neq \delta_{j,l} \end{cases}, \tag{5.9}$$

where $\boldsymbol{\delta} = \left[\boldsymbol{\delta}_1(t), \boldsymbol{\delta}_2(t), \ldots, \boldsymbol{\delta}_N(t)\right]^T$, $k \in \{1, \ldots, N\}$, $l \in \{1, 2\}$. In this method, each agent continuously maintains an accurate estimation of the center of formation by considering the specified formation and the communication connectivity among agents.

Based on the results of the decentralized calculations of the center of formation. Each agent will calculate a value defining how confident they are that the leaders selected are correct for the current case. This is done by defining how far into the selected region the goal point is by defining a confidence value C, which is calculated using a value D

$$D = \frac{2k\pi}{n} - ||\tan^{-1}(\frac{y_G - y_c}{x_G - x_c})||,$$
$$C = e^{-D},$$

where n is the number of robots in the system, k is the region number as in Fig.5.22, x_G and y_G are the coordinates of the current goal, and x_c and y_c are the center of formation as estimated by the agent. This will work generally for any number of agents as long as the formation is an equilateral polygon.

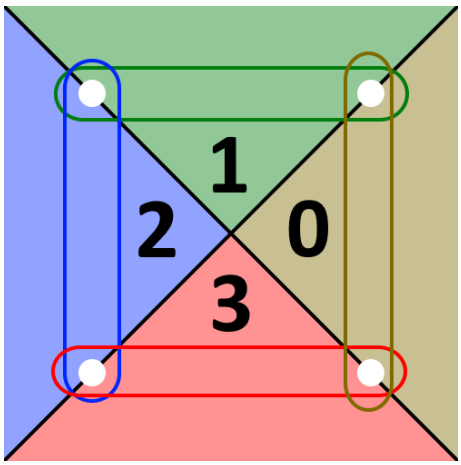

Figure 5.22: Numbering of segments for computations.

The greater the value of C, the better the estimate of the leaders is treated. From this, if Agent i receives a confidence value lower than its own, it will defer to this result, and send it out in place of its own. If the robot receives back its own signal through the network, it will ignore this signal. This process resets when a goal point is reached.

Simulations were conducted using Robot Operating System (ROS) with the Gazebo simulator. The simulation uses teams of 4 simulated Turtlebot3 robots.

In the simulation, the agents are tasked with arranging themselves in a square formation and navigate through the environment based on pre-determined goal points. The environment can be seen below in Fig. 5.23 with the goal points identified as a sequence of red stars. Each goal point is the goal for the center of the formation, and each agent has a predetermined position within the formation. The agents have a desired formation of a square with sides of 1.5 meter lengths, and the environment is a square with roughly 5 meter lengths.

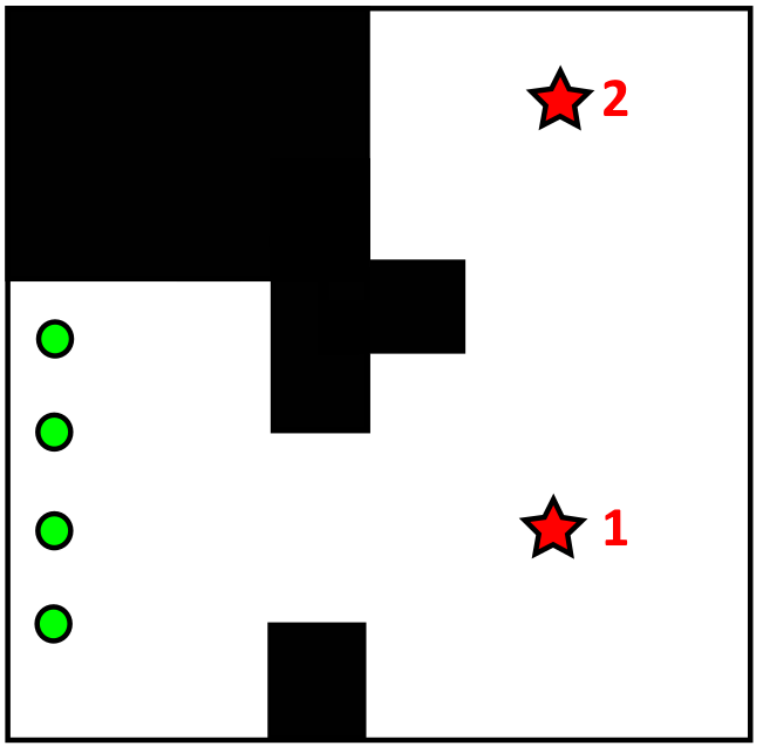

Figure 5.23: The environments used in simulations, with initial positions of agents shown as green circles, and the sequence of goal points marked by red stars.

The results of the simulation are showcased in Fig. 5.24. The initial positions

are indicated by the coloured circles with black outlines, the agents positions when reaching the each of the goal points are indicated by coloured X marks, and the estimated centers of formation at the two goal points are indicated by coloured circles without an outline.

In the simulation, the agents start arranged in a vertical line. They will initially move to their desired square formation as the leaders begin to move towards the goal point. The center of the initial configuration is placed at the origin for convenience. From here, the agents will navigate towards the gap in the wall as they move towards Goal Point 1, and compress the formation in order to pass through. After passing through the gap, the formation expands back to its original desired size. For the motion to the second goal point, there is an obstacle that will impact the agents on the left side, but this does not create a sufficiently narrow passageway such that the total formation needs to change. As such, the right side of the formation is not impacted and continues unperturbed. Once past the obstacle, the agents navigate to the goal and come to a stop.

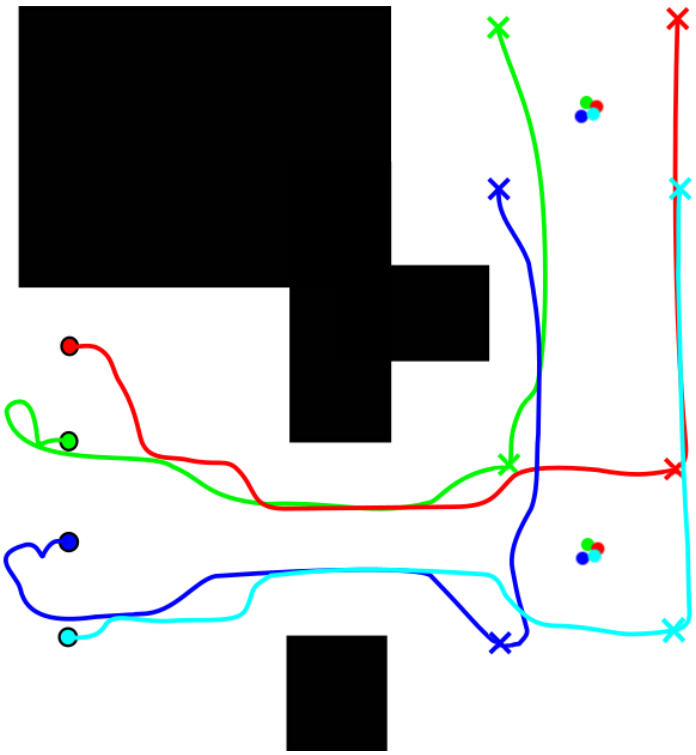

Figure 5.24: The environments used in simulations, with initial positions of agents shown as green circles, and the sequence of goal points marked by red stars.

As can be seen in the final positions, the agents do not make a perfect square. There are some minor deviations as a result of the differences in the center of formation calculation between the two leader agents. These deviations are relatively quite small, and so the formation is still kept close to the desired case.

5.4 Conclusions

In this chapter a formulation was presented for a novel approach to obstacle avoidance. The focus here is on allowing agents to shift between a discrete set of formations, while allowing for a continuous spectrum of formations in between these. This is done to minimize the amount of change in the formation to reduce the possibility of collisions between agents.

Chapter 6

Continuously Variable Formation

A common approach to time-varying formation is to have a set of discrete formations which the system can shift between as required. While this can be an effective approach, it can also lead to large shifts in the formation at times where it may not be completely necessary. As such, a novel approach to time-varying formation is considered in which the formation can operate along a continuous spectrum of formations about a discrete set. This is referred to here as, "Continuously Variable Formation," taking inspiration from the "Continuously Variable Transmission" used in many vehicles. This is a reactive approach to obstacle avoidance, as the system can almost immediately respond based on the feedback from sensors, and no new trajectory is calculated.

6.1 Objective and Assumptions

In Chapter 6 the formation was shrunk in order to pass through obstacles. This is a workable approach, however it only works up to a certain point before the agents get too close together. A more flexible approach can thus provide a great deal of value.

Assumption 6.1 *Agents are able to communicate within a defined topology without constraints on communication.*

Assumption 6.2 *All obstacles are constant in the Z-direction and can be defined in the X-Y plane.*

Assumption 6.3 *No time delays or communication constraints are placed on the communication between agents.*

The objective of the continuously variable formation is to reduce the amount that the system needs to change its formation in order to properly navigate in an unknown environment, while providing flexibility in the navigation. Within the scope

of this work, agents will pass through gaps in obstacles and change between multiple formations, with allowance for the agents to operate along a spectrum of formations between a set of discretely defined formations.

6.2 Formation Controller

For simplicity in the discussion, here we assume that we are operating with a 2D case. However it can apply equally to the 3D case. The environment is investigated with sensors, and a simple algorithm determines if there is any need for the formation to be adjusted.

Algorithm 6.1 Determination of Formation Shaping

```
while Goal not reached. do
  Scan environment
  Detect obstacles
  if Path to goal obstructed then
    Evaluate obstacle positions
    Determine if gaps exist
    if Gap smaller than current formation then
      Begin formation shaping computations
    end if
  end if
end while
```

A series of n formations are implemented. These can be set to anything, and are predetermined based on the intended formation and the potential minimum widths that might be expected for the MAS. For each of these formations, i, a minimum width M_i is determined. This is the minimum gap which the formation could pass through with appropriate spacing. This width should broadly be considered as the total width of a given formation, plus the desired clearance for the formation. The same baseline approach to shaping is deployed as in Chapter 6 to determine the gap that is being passed through.

Each formation is composed of positions $x_{i,j}$ and $y_{i,j}$ where i is the formation number and j is the agent number. $i = 1$ will represent the widest formation, and

as i increases the formation becomes narrower. Once the width of the obstruction becomes equal to M_i, then the transition towards the next formation $(i+1)$ will begin to occur. Given a set of n formations, if the passable gap has a width of less than M_n, the gap is considerable impassable, and an alternative route would be required.

Fig.6.1 designs a set of two formations for a MAS consisting of three agents. These formations are a triangle $(i = 1)$ and a line $(i = 2)$. In this case, the leading agent does not have any change in position between the two formations, though this is not an inherent requirement. The other two agents shift along the black lines shown.

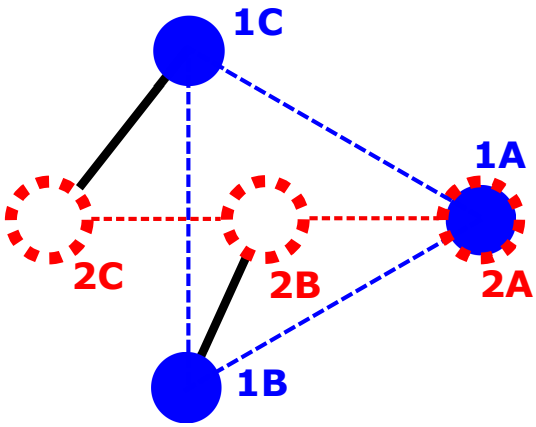

Figure 6.1: The line and triangle formations for a triangle agent case.

In environments where there is no current passageway to be traversed, or if the current passageway is larger than the largest minimum then no change to the formation will be necessary. However, if the current passageway is smaller than the largest minimum value, the formation will change in order to properly navigate the environment. By using this approach, the change in the formation can be minimized.

A new variable ζ is defined as

$$\zeta = \frac{W - M_{i+1}}{M_i - M_{i+1}}, \tag{6.1}$$

where W is the measured width of the passage. ζ then gives us a value over the continuous spectrum of formations which can be applied to the formation. We can thus conceptualize the process as in Algorithm 7.2.

The new desired positions of the agents are

$$x_i = \zeta x_{i,j} + (1 - \zeta) x_{i+1,j}, \tag{6.2}$$

$$y_i = \zeta y_{i,j} + (1 - \zeta) y_{i+1,j}. \tag{6.3}$$

Algorithm 6.2 Calculation of ζ

Calculate W from sensors.
if $W < M_n$ **then**
  Path Non-viable.
else if $W > M_1$ **then**
  No formation change required.
else
  while $W < M_i$ **do**
    if $W \geq M_i$ **then**
      $\zeta = \frac{W - M_{i+1}}{M_i - M_{i+1}}$
    end if
  end while
end if

This allows the formation to exist at any point on the line between formations i and $i + 1$. This framework uses the current environmental data from the Lidar scanner, and updates in response to new data.

6.3 Simulated Results

A number of simulations were conducted to test the functionality of the continuously variable formation algorithm. These involve formations comprised of different numbers of agents in various environments to demonstrate the effectiveness of the control scheme.

6.3.1 Homogeneous MAS

Two homogeneous simulations were conducted in order to validate the baseline functionality of the formation shaping and see how it would interact with the novel APF protocol. These tests are designed to apply the core fundamentals of the continuously variable formation to determine the validity of the approach. As such, one two-agent test and one three-agent test were conducted.

The first test was conducted with a team of two agents passing through a single obstacle. In this case, the formation would tighten together, but not form a line.

The intention here was to focus on the interaction with the novel APF. Therefore, the simulation was conducted two times, once with the novel APF and once with a classic APF. Fig.6.2 shows the novel APF with solid lines, and the classic APF with dashed lines.

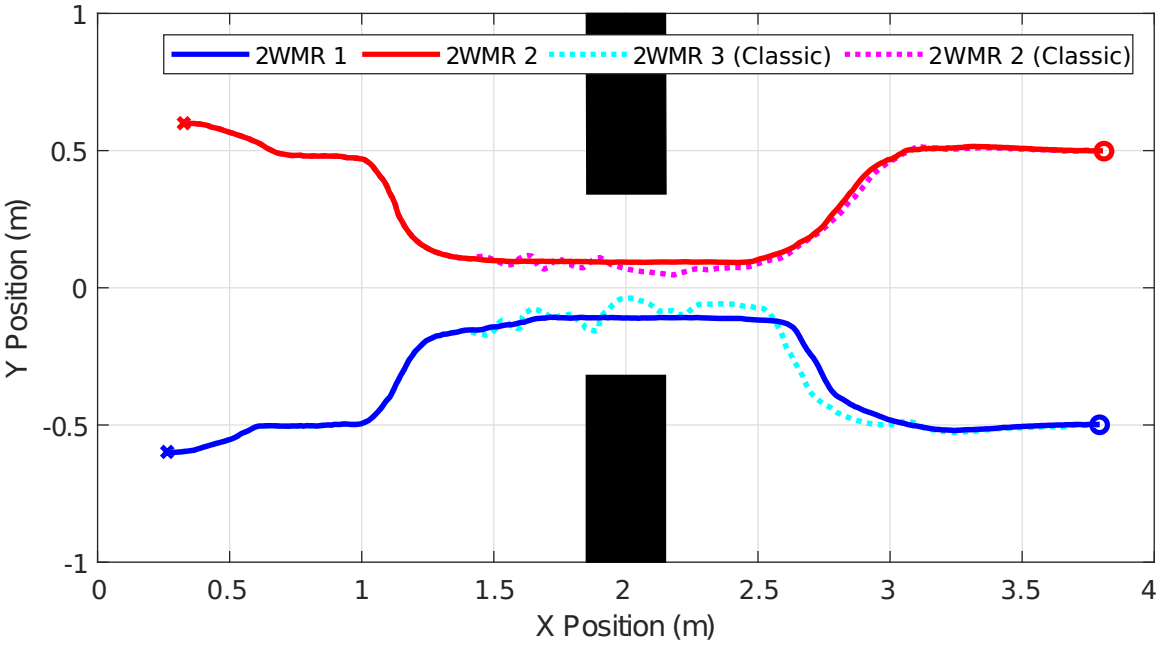

Figure 6.2: Simulation of two agents tested with the classic and novel APF algorithms.

Within this result a much steadier result is seen from the solid lines of the novel APF. As the agents are moving in parallel with each other and the walls, by the time they are passing through the gap in the obstacles, the APF is producing only small repulsive forces, and the impact on the agents is fairly minimal. There are some very minor perturbations during the simulation, but they quickly correct and are not notable in the final result.

With the classic APF, the agents fluctuating between being pushed by each other and pushed by the obstacles, leading to much poorer performance. The radius of the classic APF could be further reduced to reduce its impact on the agents in these tight corners, but further reductions start to approach the minimum measurement limits of the LiDAR sensor itself, and the

Another simulation was conducted with a team of three agents. These have a variable formation structure in the same style as Fig.6.3, converting from a triangular formation to a linear formation. This simulation has the two agents pass through two

different gaps in obstacles to demonstrate the effectiveness of the controller.

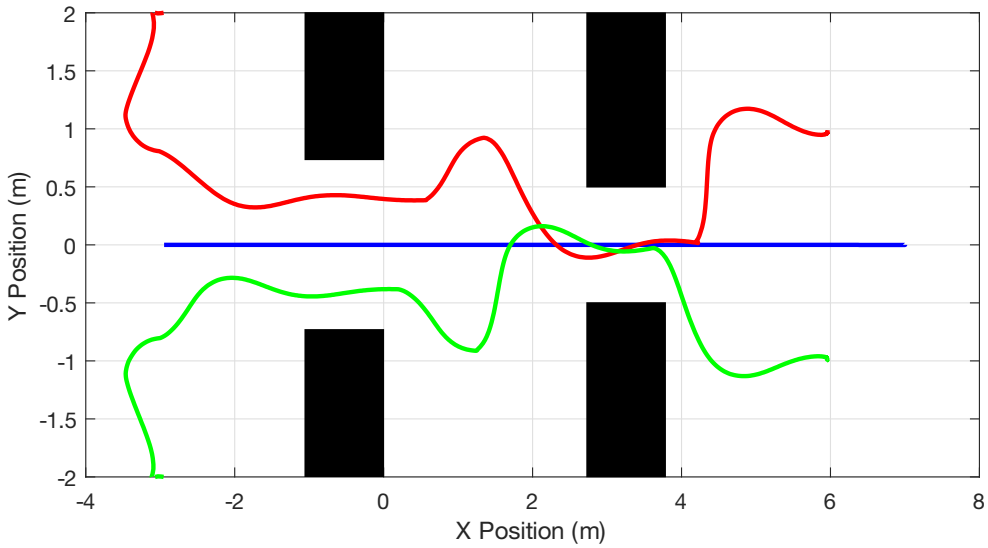

Figure 6.3: A simulated result for a three agent case passing through obstacles.

6.3.2 Heterogeneous MAS

For the following simulations, a four-agent topology was used which has a defeault of a square formation. In order to navigate the environments, this may need to be changed into a line formation. As a result, the following spectrum of formations will be deployed:

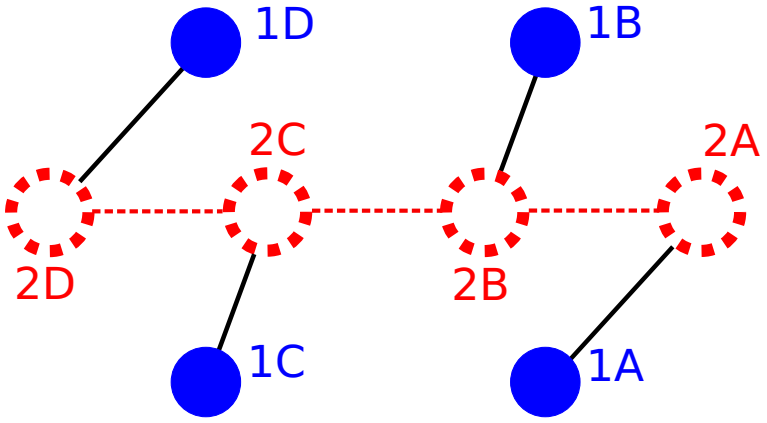

Figure 6.4: The line and square formations for a four-agent case.

With this, we will see the agents able to transition from a square formation to a line formation, as well as anything within the spectrum between these two formations.

For the heterogeneous MAS, a number of simulations were conducted to effectively demonstrate the effectiveness of the approach.

The communication topology is shown in Fig.6.5 with the leaders highlighted in cyan. Here we see that the four ground vehicles send position information to the aerial vehicle, which then tracks the center of their formation.

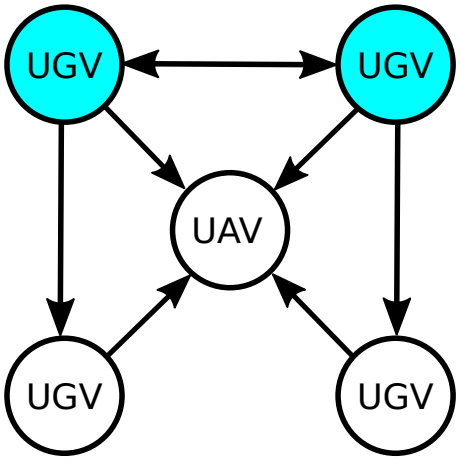

Figure 6.5: The communication topology for the simulations

In the first heterogeneous simulation there are two sets of obstacles which present narrow passages for the agents to navigate through. The four mobile robots have both a square and line formation available. The ground robots are able to travel through both sets, however with the first obstacles, there is more available space. As a result, the agents are able to navigate without fully transitioning to the line formation (here $\zeta = 0.5$), whereas in order to pass through the second set of obstacles the formation must completely transition into a line.

The quadrotor also adjusts with the rest of the formation. In Fig.6.6, the positions at various times are indicated, and the vertical position of the quadrotor is also indicated. The target vertical position of the quadrotor decreases as the formation shifts from a square formation to a line formation, showing how different agent types can be incorporated into these dynamic formations.

The total position error of the system is also shown in Fig.6.7. This indicates the sum of the errors of all five agents. We see that the system initially converges towards zero error as the agents pass through the first obstacle. After this there is an error that builds when the agents return to the normal formation after passing

the first obstacle. The agents begin to converge to zero error, but are not able to do so before another change in formation occurs. They do quickly recach the line formation and maintain minimal error passing through the second gap. Again, once they pass through and return to the regular formation, there is a spike in the error as they settle into formation, which will shrink back down as they stabilize.

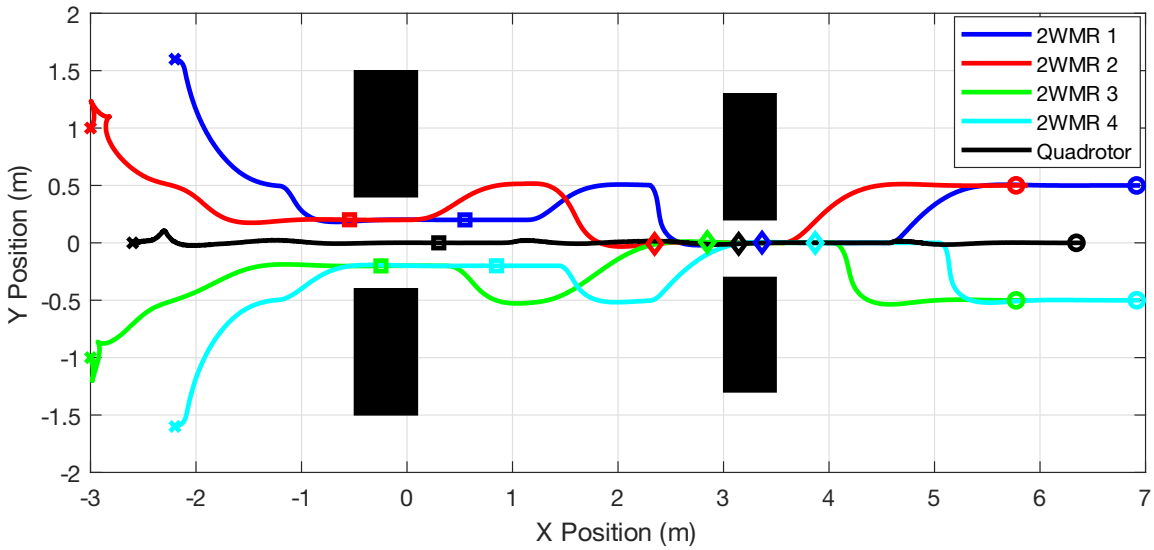

Figure 6.6: Results of the first heterogeneous simulated case.

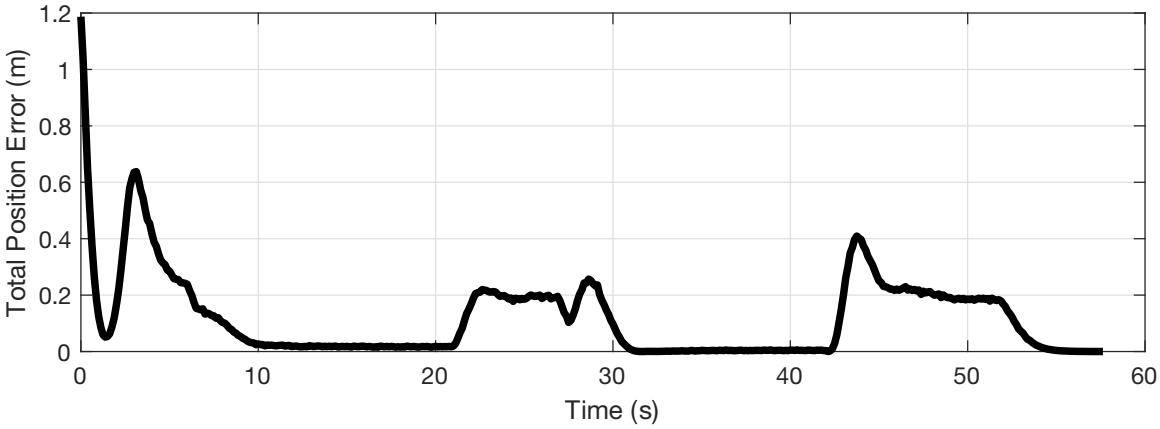

Figure 6.7: Results of the first heterogeneous simulated case.

The second heterogeneous simulation uses a circular trajectory, and the agents pass through two obstacles, and then when they loop back around on the other side, two of the agents move to avoid an obstacle. The results, plotted in 2D, are shown in Fig.6.8 In this case, it is shown that the agents will adjust their formation based

on the gaps in obstacles, as on the left side of the image. However, on the right side as the agents reach the end of the trajectory, the obstacle no longer presents an obstructed path which the formation needs to squeeze through. Only 2WMR 3 and 2WMR 4 will be impacted by this obstacle. As a result, the variable formation is not deployed. Instead, 2WMR 4 adjusts its path based on the effects of the novel APF, and 2WMR 3 adjusts course to continue tracking its leader.

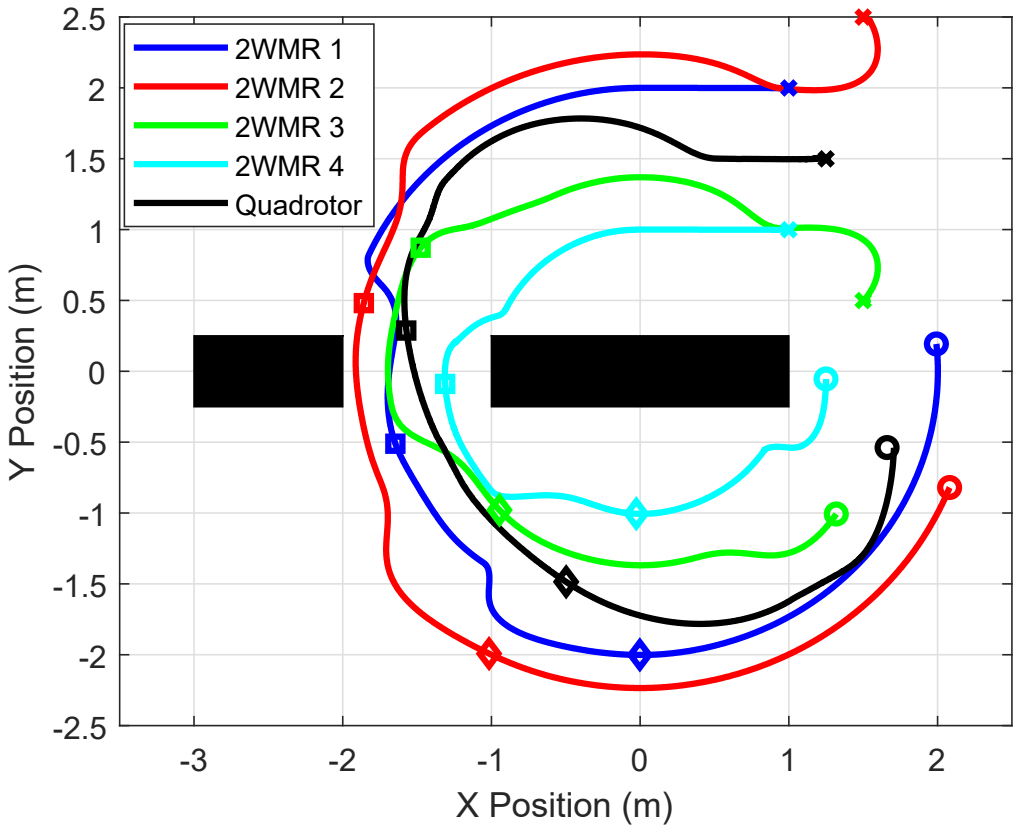

Figure 6.8: Results of the second heterogeneous simulated case.

The quadrotor here roughly tracks the center of the formation, and lowers its hovering point as the formation passes through the gap between the two obstacles. Otherwise it maintains a mostly level height of 1 meter. Fig.6.9 is presented in 3D to make it easier to see the trajectory of the quadrotor.

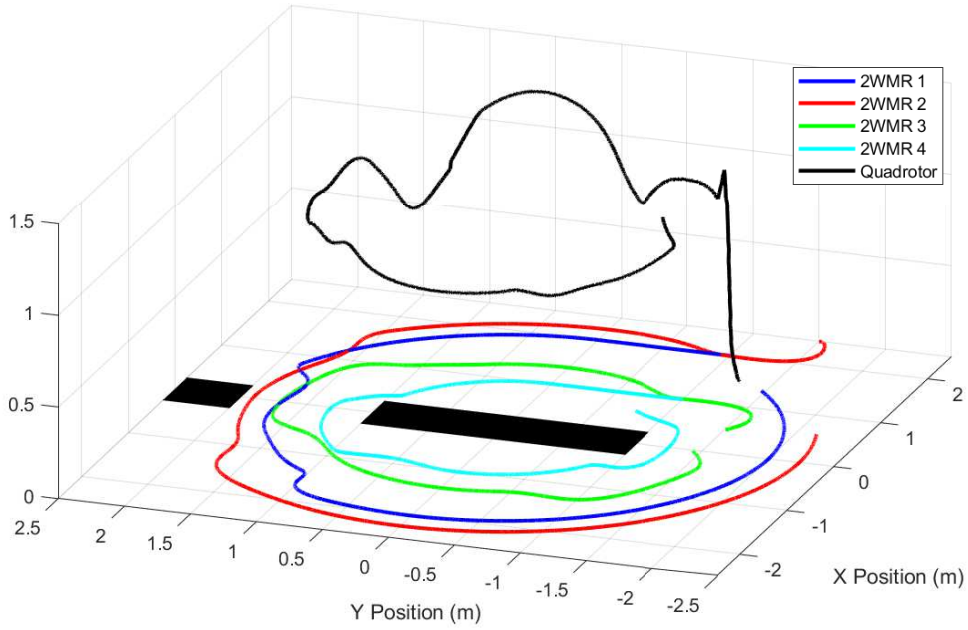

Figure 6.9: Results of the second heterogeneous simulated case.

6.4 Experimental Results

Two experimental tests were conducted to validate the results of the continuously variable formation process. One of these was with a homogeneous MAS, and the other with a heterogeneous MAS.

6.4.1 Homogeneous MAS

A test was conducted with a team of three mobile robots. This mimics the triangular formation case previously seen. Here the agents must pass through two obstacles, as well as avoid an additional obstacle slightly past these two. This demonstrates the ability of the controller to adjust the total formation, as well as to navigate around a single obstacle without needing to deploy the variable formation approach.

Fig.6.10 shows the results of the test. The leader agent (in blue) has a simple trajectory that has it move forward at a specified velocity. The two follower agents (in green and red) are moving to a triangular formation with the blue agent. However, the gap the agents pass through is 1 meters wide while the original formation is more 2

meters wide. In order to pass through, the agents transition to a formation about 75% of the way to a linear formation, and then return to the original triangular formation once the MAS has successfully passed through the obstacle. The basic behaviour demonstrated here matches very closely with what was seen in the simulated testing.

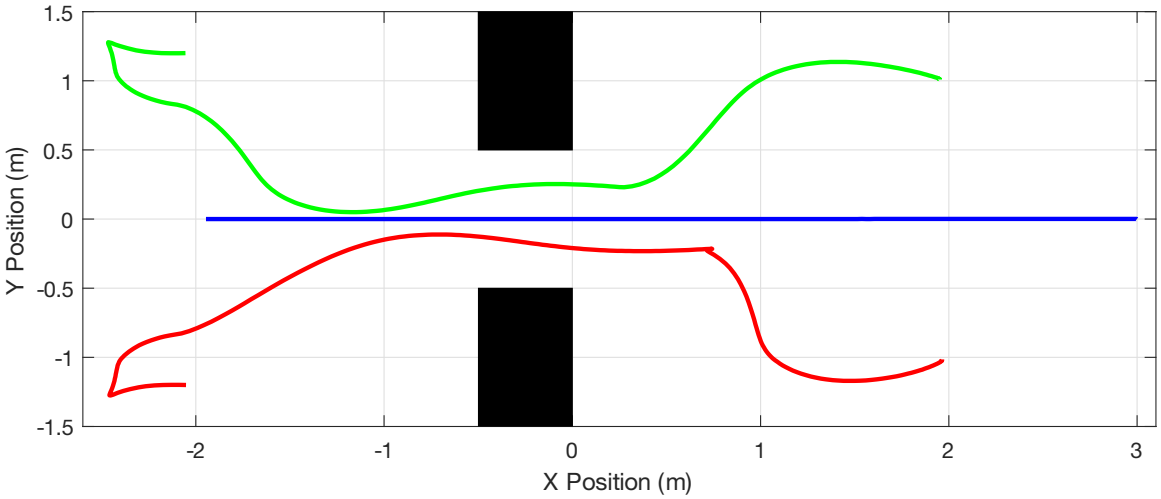

Figure 6.10: Positon data for the heterogeneous experimental test.

6.4.2 Heterogeneous MAS

An experiment using a heterogeneous team was also conducted. This case was intended to replicate the first heterogeneous simulation from the simulated testing, but on a smaller scale due to space constraints. In this experiment, a team of five agents are navigating through two different gaps of differing sizes. When passing through the gap, the formation does not form the perfect line which was seen in the simulation. This is simply the result of the formation reaching a different point on the continual spectrum of possible formations. The team of agents running the second experiment is shown in Fig.6.5. The two cyan agents represent the leaders of the system.

For these formations, we have an M_1 value of 1.5 meters and an M_2 value of 0.5 meters. With a gap of about 0.8 meters, we would expect expect to see a γ value of 0.7. Fig.6.11 shows an image of the experiment in progress with all five agents as they pass through the obstacles. Fig.6.12 shows the movement of the agents through the gap towards the goal point.

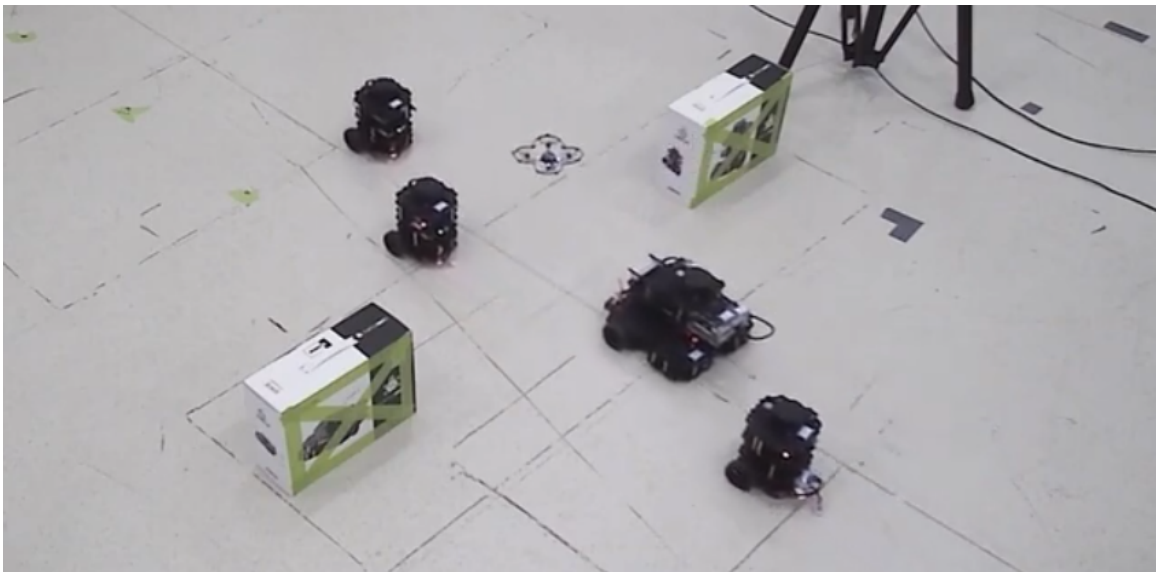

Figure 6.11: The five agent formation passing through obstacles.

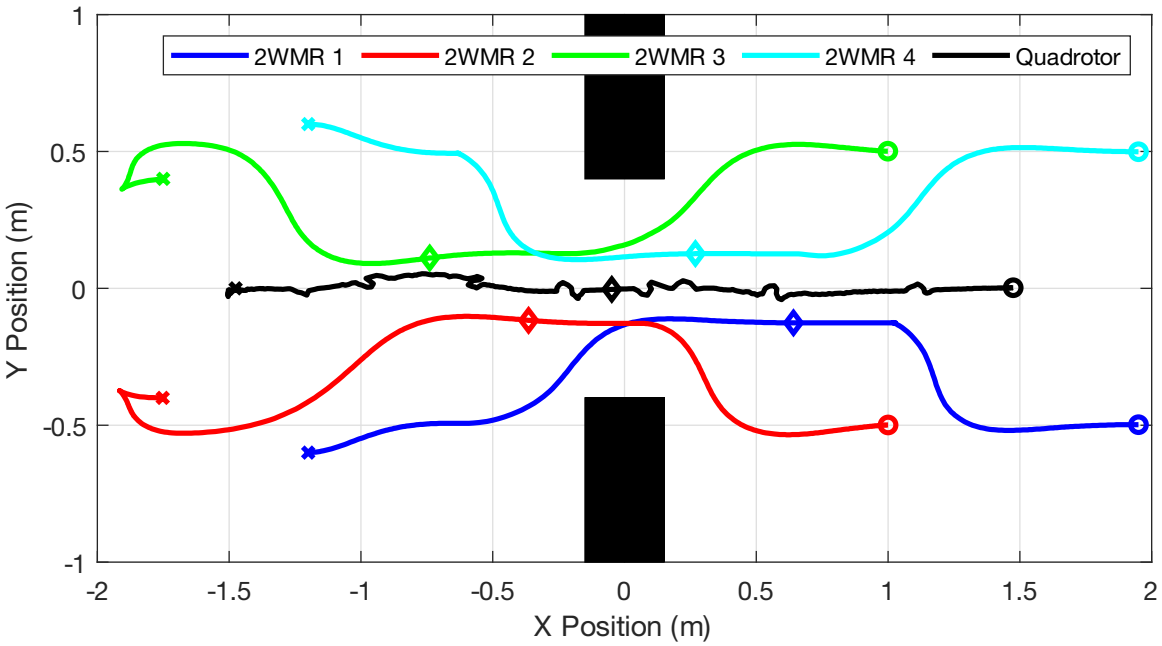

Figure 6.12: Positon data for the heterogeneous experimental test.

In this experiment, we find that a γ value of 0.64 is calculated by the agents. This is a bit smaller than the expected value, but is still a sufficient approximation that the agents are able to travel through the obstacles without any problems. The M_2 value of 0.5 meters is well above the actual width of one of the Turtlebots, and a prominent safety factor is used to ensure the agents will not approach the obstacles.

In general, the behaviour of the Turtlebots very closely matched the expectations from the simulations, though their onboard odometry was not always perfect, which would sometimes lead to minor deviations in the final formation. This is a result of the Turtlebots having an assumed initial position and heading, and subsequent measurements are taken relative to these initial conditions. If the initial heading is off by a small amount it can propogate through the entire experiment. Video of this experiment is available at https://www.youtube.com/watch?v=FI_XXSSewl0.

The Crazyflie quadrotor tended to follow the center of formation fairly well, however it did still have limitations. Due to the very lightweight nature of the quadrotor, it is not especially well suited to a slow trajectory that is prone to variations. As such, the experiment sees the quadrotor juttering somewhat while trying to track the center of the formation, and adjust its height based on the current positions in the continuous spectrum of possibilities. It is still adept at maintaining the formation, but as can be seen in Fig.6.12, it sometimes will move about 5 cm off of the desired position before quickly correcting itself.

The position error of the experiment is shown in Fig.6.13. There are notable spikes in this result that occur shortly after 5 seconds and 20 seconds as a result of the formation shifting and the followers becoming trailing behind by a small amount. There are also a number of small pertebations throughout caused by the juttering of the Crazyflie. A small steady-state error existed in the final result. This is a result of the system halting once all four Turtlebots are within 3 cm of the final goal point.

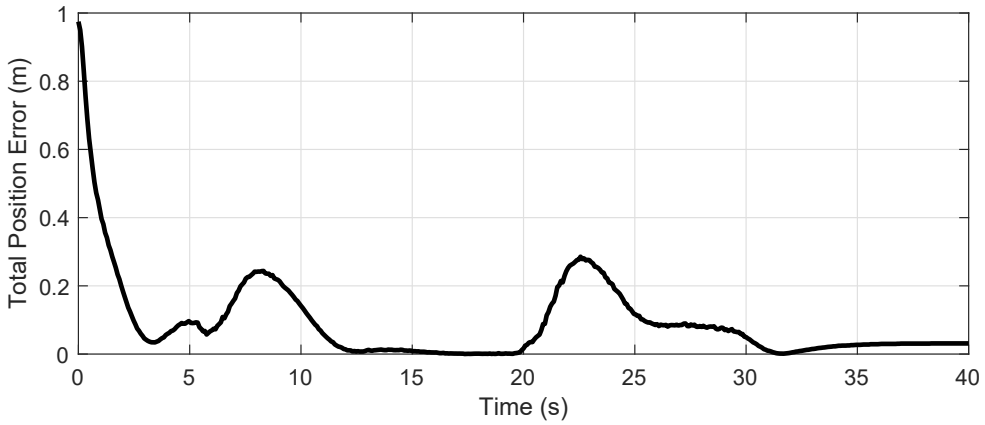

Figure 6.13: The line and square formations for a four-agent case.

6.5 Conclusion

Within this chapter, a formulation for a novel time-varying formation algorithm was developed. This utilizes a set of formations, and allows the system to transition between them in order to accomplish the current objective. This was validated in both simulated and experimental tests with both homogeneous and heterogeneous teams of agents.

Chapter 7

Conclusion

Through this thesis, developments in formation of multi-agent systems, obstacle avoidance, and time-varying formation have been presented. Four core developments have been presented in this work: SMC and TSMC based controllers for 2WMRs and quadrotors, a novel approach to APF obstacle avoidance that accounts for the heading of an agent relative to obstacles, a distance based methodology for time-varying formation with dynamic leader selection, and a continuously variable approach for time-varying formation.

7.1 Results

In Chapter 4, two pairs of controllers were demonstrated. A pair of classical sliding mode controllers and a pair of terminal sliding mode controllers were applied to teams of ground and air vehicles. It was shown that these controllers could be used to achieve consensus and formation, including time-varying formations in which the formation translates in plane, but otherwise remains constant.

In Chapter 5, a novel APF algorithm was developed. Four factors were added to the typical artificial potential field to improve the overall performance. In testing, it was found that the heading factor, which adjusts the impact of the potential field based on the heading between the agent and obstacle, was the most prominent of these. Testing demonstrated a pronounced improvement over a classic APF, including reduced distance travelled and time to reach the goal, as well as being able to navigate environments that were otherwise not possible with the classic APF.

In Chapter 6, an approach to time-varying formation was developed in which a team of agents adjusts its shape based on the environment around it. This can allow for the formation to scale itself, or make specific alterations to the shape of the formation depending on what is desirable. This base process was also implemented

in combination with a GPMP2 algorithm for centralized operation in a known environment, as well as with a center of formation estimation algorithm for decentralized operation in an unknown environment.

In Chapter 7, another approach to time-varying formation was developed in which a team of agents is able to dynamically swap between a discrete set of formations, as well as a continuous spectrum of values between these, in order to navigate through an environment. This was tested in both simulations and experiments using a heterogeneous team of agents.

7.2 Future Work

Research is always a stepping stone to more research, and there are a number of potential future avenues for this research to be elaborated on. A number of potential avenues for future work would include:

1. For the novel APF algorithm, the current control gains were optimized in testing for specific cases, but are not ideal for all cases. A reinforcement learning (or similar) approach to this could be used to develop dynamic gains which would be adjusted based on the current environment and agent scenario. This could provide more optimal control.

2. The novel APF did not use sophisticated filtering techniques when using the LiDAR sensor. This meant that the results could run into problems in experimental testing. Further work should be pursued in this regard.

3. The decentralized control scheme which was implemented with the formation shaping controller from Chapter 6 could also be implemented with the continuously variable formation approach used in Chapter 7. Additionally, the leader selection protocol could also be implemented with this approach to offer increased flexibility.

4. Obstacle avoidance in the quadrotor was not feasible due to its inability to sense obstacles. However, using a different quadrotor platform, it would be possible to apply the novel APF developed here to these agents as well, allowing for more

dynamic possibilities. Expanding it to a 3-dimensional scope would require additional development.

5. Simulated work was conducted using multiple quadrotor agents in Chapter 4, however this was not completed in experimental work. Due to the light weight of the Crazyflie, operating with multiple agents introduces new problems in the dynamics, and so it was not attempted. This is an avenue which could be pursued in future work.

6. The novel APF was designed with a differential factor to allow it to account for more dynamic obstacles, however dynamic environments were never properly tested as part of this work. An expansion of this work accounting for the behaviour of a dynamic environment is worth pursuing, as the formulation of the gains can likely be optimized quite a great deal from such testing.

Bibliography

[1] Fei Chen; Wei Ren, "On the Control of Multi-Agent Systems: A Survey", *now*, 2019.

[2] Y. Cao, W. Yu, W. Ren and G. Chen, "An Overview of Recent Progress in the Study of Distributed Multi-Agent Coordination," in *IEEE Transactions on Industrial Informatics*, vol. 9, no. 1, pp. 427-438.

[3] D. Steiner, "Industrial applications of multi-agent technology," *Proceedings International Conference on Multi Agent Systems* (Cat. No.98EX160), Paris, France, 1998, pp. 12-13.

[4] X. Liu et al., "Autonomous Multi-Agent Reconnaissance in Military Environments: A Distributed Solution," *2023 China Automation Congress (CAC)*, Chongqing, China, pp. 263-267.

[5] V. Duchenchuk and V. Boublik, "Multi-Agent Software Architecture for Distributed Virtual Reality Systems," *2020 10th International Conference on Advanced Computer Information Technologies (ACIT)*, Deggendorf, Germany, pp. 529-532.

[6] P. Twu, Y. Mostofi and M. Egerstedt, "A measure of heterogeneity in multi-agent systems," *2014 American Control Conference*, Portland, OR, USA, pp. 3972-3977.

[7] A. Dorri, S. S. Kanhere and R. Jurdak, "Multi-Agent Systems: A Survey," in *IEEE Access*, vol. 6, pp. 28573-28593, 2018.

[8] Q. Ali and S. Montenegro, "Role of graphs for multi-agent systems and generalization of Euler's Formula," *2016 IEEE 8th International Conference on Intelligent Systems (IS)*, Sofia, Bulgaria, pp. 198-204.

[9] P. Shi and B. Yan, "A Survey on Intelligent Control for Multiagent Systems," in *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 1, pp. 161-175, Jan. 2021.

[10] J. Yang, "A Consensus Control for a Multi-Agent System With Unknown Time-Varying Communication Delays," in *IEEE Access*, vol. 9, pp. 55844-55852, 2021.

[11] S. Su and Z. Lin, "Distributed Consensus Control of Multi-Agent Systems With Higher Order Agent Dynamics and Dynamically Changing Directed Interaction Topologies," in IEEE Transactions on Automatic Control, vol. 61, no. 2, pp. 515-519, Feb. 2016.

[12] H. D. Aghbolagh, E. Ebrahimkhani and A. Ghiasi, "Scaled consensus of multi-agent systems under time-varying delay," 2015 2nd International Conference on Knowledge-Based Engineering and Innovation (KBEI), Tehran, 2015, pp. 729-732.

[13] A. Pawar and Y.J. Pan, "Leader-following Consensus Control of Multi-Agent Systems with Communication Delays and Random Packet Loss", In Proceedings of the IEEE American Control Conference, June 2016, Boston, USA, pp.4464-4469.

[14] W. Ren and R.W. Beard, "Distributed consensus in multi-vehicle cooperative control," *Springer*, 2008.

[15] K. M. Lo, L. Y. Lo, P. K. Wong and K. S. Leung, "Multi-objective Multiple Quadcopter Path Planning in Urban City," *2018 3rd International Conference on Control, Robotics and Cybernetics (CRC)*, Penang, Malaysia, pp. 67-72.

[16] E. Yazid, M. Garrat and F. Santoso, "Optimal PD Tracking Control of a Quadcopter Drone Using Adaptive PSO Algorithm," *2018 International Conference on Computer, Control, Informatics and its Applications (IC3INA)*, Tangerang, Indonesia, pp. 146-151.

[17] A. S. Aghdam, M. B. Menhaj, F. Barazandeh and F. Abdollahi, "Cooperative load transport with movable load center of mass using multiple quadrotor UAVs," *2016 4th International Conference on Control, Instrumentation, and Automation (ICCIA)*, Qazvin, Iran, pp. 23-27.

[18] J. Panerati, H. Zheng, S. Zhou, J. Xu, A. Prorok and A. P. Schoellig, "Learning to Flya Gym Environment with PyBullet Physics for Reinforcement Learning of Multi-agent Quadcopter Control," *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Prague, Czech Republic, pp. 7512-7519.

[19] M. Y. Amir and V. Abbass, "Modeling of Quadrotor Helicopter Dynamics," *2008 International Conference on Smart Manufacturing Application*, Goyangi, Korea (South), pp. 100-105.

[20] X. Kan, J. Thomas, H. Teng, H. G. Tanner, V. Kumar and K. Karydis, "Analysis of Ground Effect for Small-Scale UAVs in Forward Flight," in *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3860-3867, Oct. 2019.

[21] R. C. Avram, X. Zhang and J. Muse, "Nonlinear Adaptive Fault-Tolerant Quadrotor Altitude and Attitude Tracking With Multiple Actuator Faults," in *IEEE Transactions on Control Systems Technology*, vol. 26, no. 2, pp. 701-707, March 2018.

[22] X. Peng, Z. Sun, K. Guo and Z. Geng, "Mobile Formation Coordination and Tracking Control for Multiple Nonholonomic Vehicles," in *IEEE/ASME Transactions on Mechatronics*, vol. 25, no. 3, pp. 1231-1242, June 2020.

[23] S. Thabit and A. Mohades, "Multi-Robot Path Planning Based on Multi-Objective Particle Swarm Optimization," in *IEEE Access*, vol. 7, pp. 2138-2147, 2019.

[24] Y. Wan, Y. Zhong, A. Ma and L. Zhang, "An Accurate UAV 3-D Path Planning Method for Disaster Emergency Response Based on an Improved Multiobjective Swarm Intelligence Algorithm," in IEEE Transactions on Cybernetics, vol. 53, no. 4, pp. 2658-2671, April 2023.

[25] Z. Yang, N. Ma and Y. Yao, "A Survey on Cooperative Control of Multi-Agent Systems," *2023 2nd International Conference on Artificial Intelligence, Human-Computer Interaction and Robotics (AIHCIR)*, Tianjin, China, pp. 511-515.

[26] H. Sun, Y. Liu, F. Li and X. Niu, "A survey on optimal consensus of multi-agent systems," *2017 Chinese Automation Congress (CAC)*, Jinan, China, pp. 4978-4983.

[27] J. Wang et al., "Cooperative and Competitive Multi-Agent Systems: From Optimization to Games," in IEEE/CAA Journal of Automatica Sinica, vol. 9, no. 5, pp. 763-783, May 2022.

[28] H. Zhu, "Avoiding conflicts by group role assignment," in *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 46, no. 4, pp. 535547, April 2016.

[29] Haibin Zhu, "RoleBased Collaboration," in *E-CARGO and Role-Based Collaboration: Modeling and Solving Problems in the Complex World*, IEEE, 2022, pp.69-102.

[30] Y. Yang, Y. Xiao and T. Li, "A Survey of Autonomous Underwater Vehicle Formation: Performance, Formation Control, and Communication Capability," in *IEEE Communications Surveys and Tutorials*, vol. 23, no. 2, pp. 815-841, Second quarter 2021.

[31] R. Cajo et al., "Distributed Formation Control for Multiagent Systems Using a Fractional-Order ProportionalIntegral Structure," in *IEEE Transactions on Control Systems Technology*, vol. 29, no. 6, pp. 2738-2745, Nov. 2021.

[32] M. Zhao and H. Li, "Distributed Formation Control of Quadrotors Using Model Predictive Contouring Control," IECON 2023- 49th Annual Conference of the IEEE Industrial Electronics Society, Singapore, Singapore, pp. 1-6.

[33] J. -L. Wang, Q. Wang, H. -N. Wu and T. Huang, "Finite-Time Consensus and Finite-Time H_∞ Consensus of Multi-Agent Systems Under Directed Topology," in IEEE Transactions on Network Science and Engineering, vol. 7, no. 3, pp. 1619-1632, July 2020.

[34] M. Ge and Z. -W. Liu, "Fully-distributed discontinuous consensus protocols for multi-agent systems with external disturbances," IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society, Beijing, 2017, pp. 5815-5818.

[35] H. J. Savino, C. R. P. dos Santos, F. O. Souza, L. C. A. Pimenta, M. de Oliveira and R. M. Palhares, "Conditions for Consensus of Multi-Agent Systems With Time-Delays and Uncertain Switching Topology," in IEEE Transactions on Industrial Electronics, vol. 63, no. 2, pp. 1258-1267, Feb. 2016.

[36] J. Xi, Y. Yu, G. Liu and Y. Zhong, "Guaranteed-Cost Consensus for Singular Multi-Agent Systems With Switching Topologies," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 61, no. 5, pp. 1531-1542, May 2014

[37] Z. Tu and H. Yu, "Adaptive consensus of high-order multi-agent systems with directed switching topologies," *The 27th Chinese Control and Decision Conference (2015 CCDC)*, pp. 436-441.

[38] N. Sarrafan and J. Zarei, "Bounded Observer-Based Consensus Algorithm for Robust Finite-Time Tracking Control of Multiple Nonholonomic Chained-Form Systems," in IEEE Transactions on Automatic Control, vol. 66, no. 10, pp. 4933-4938, Oct. 2021

[39] J. Lindsay et al., "Collaboration of Heterogeneous Marine Robots Toward Multidomain Sensing and Situational Awareness on Partially Submerged Targets," in *IEEE Journal of Oceanic Engineering*, vol. 47, no. 4, pp. 880-894, Oct. 2022.

[40] Y. Ma, B. Jiang, J. Wang and J. Gong, "Adaptive Fault-Tolerant Formation Control for Heterogeneous UAVs-UGVs Systems With Multiple Actuator Faults," in *IEEE Transactions on Aerospace and Electronic Systems*, vol. 59, no. 5, pp. 6705-6716, Oct. 2023.

[41] V. A. Oseledets and D. A. Yukhimets, "Method for Formation Control of AUV Group in Environment with Obstacles Based on Visual Information," *2023 International Russian Automation Conference (RusAutoCon), Sochi, Russian Federation, 2023*, pp. 576-581.

[42] L. Wang, J. Li, X. Liu and Y. Fang, "Event-Triggered Fault-tolerant Model Predictive Control of Nonlinear Multi-agent System with Time Delay and Parameter Uncertainty," *2021 40th Chinese Control Conference (CCC)*, Shanghai, China, pp. 5350-5355.

[43] B. Yan, P. Shi and C. -C. Lim, "Robust Formation Control for Nonlinear Heterogeneous Multiagent Systems Based on Adaptive Event-Triggered Strategy," in *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 4, pp. 2788-2800, Oct. 2022.

[44] Ryo Toyota, Toru Namerikawa, "Event-Triggered Formation Control of a Generalized Multi-Agent System," in 2018 57th Annual Conference of the Society of Instrument and Control Engineers of Japan, Nara, 2018, pp. 940-945.

[45] W. Zhu, W. Cao and Z. -P. Jiang, "Distributed Event-Triggered Formation Control of Multiagent Systems via Complex-Valued Laplacian," in IEEE Transactions on Cybernetics, vol. 51, no. 4, pp. 2178-2187, April 2021.

[46] S. Dai, Z. Wu, P. Zhang, M. Tan and J. Yu, "Distributed Formation Control for a Multirobotic Fish System With Model-Based Event-Triggered Communication Mechanism," in IEEE Transactions on Industrial Electronics, vol. 70, no. 11, pp. 11433-11442, Nov. 2023

[47] J. Xian, L. Shen, J. Chen and W. Feng, "Continuous Sliding Mode Control of Robotic Manipulators Based on Time-Varying Disturbance Estimation and Compensation," in *IEEE Access*, vol. 10, pp. 43473-43480, 2022.

[48] U. Javaid, H. Dong, S. Ijaz, T. Alkarkhi and M. Haque, "High-Performance Adaptive Attitude Control of Spacecraft With Sliding Mode Disturbance Observer," in *IEEE Access*, vol. 10, pp. 42004-42013, 2022.

[49] Q. Hou, S. Ding and X. Yu, "Composite Super-Twisting Sliding Mode Control Design for PMSM Speed Regulation Problem Based on a Novel Disturbance Observer," in *IEEE Transactions on Energy Conversion*, vol. 36, no. 4, pp. 2591-2599, Dec. 2021.

[50] J. Zhang, B. Zhou, S. Wei and Y. Song, "Study on sliding mode trajectory tracking control of mobile robot based on the Kalman filter," *2016 IEEE International Conference on Information and Automation*, pp. 1195-1199.

[51] R. Akbar and N. Uchiyama, "Adaptive modified super-twisting control for a quadrotor helicopter with a nonlinear sliding surface," *2017 SICE International Symposium on Control Systems (SICE ISCS)*, Okayama, Japan, pp. 1-6.

[52] C. Yi-mei, L. Yong-chao and L. Kang-li, "Sliding mode Control for Quadrotor Based on Super-twisting Algorithm Disturbance Observation and Compensation," *2019 Chinese Control And Decision Conference (CCDC)*, Nanchang, China, pp. 3123-3128.

[53] Y. Zhang, M. Su, Z. Cheng, L. Liu and B. Wang, "Adaptive Super Twisting Sliding Mode Control for Flying-Wing UAV," *2021 33rd Chinese Control and Decision Conference (CCDC)*, Kunming, China, pp. 236-241.

[54] D. Zhang, Y. Tang, W. Zhang and X. Wu, "Hierarchical Design for Position-Based Formation Control of Rotorcraft-Like Aerial Vehicles," in *IEEE Transactions on Control of Network Systems*, vol. 7, no. 4, pp. 1789-1800, Dec. 2020.

[55] D. Van Vu, M. H. Trinh and H. -S. Ahn, "Distance-based Formation Tracking with Unknown Bounded Reference Velocity," *2020 20th International Conference on Control, Automation and Systems (ICCAS)*, Busan, Korea (South), pp. 524-529.

[56] Q. V. Tran and H. -S. Ahn, "Flocking control and bearing-based formation control," *2018 18th International Conference on Control, Automation and Systems (ICCAS)*, PyeongChang, Korea (South), 2018, pp. 124-129.

[57] X. Dong, B. Yu, Z. Shi and Y. Zhong, "Time-Varying Formation Control for Unmanned Aerial Vehicles: Theories and Applications," in *IEEE Transactions on Control Systems Technology*, vol. 23, no. 1, pp. 340-348, Jan. 2015.

[58] J. Yu, X. Dong, Q. Li and Z. Ren, "Robust H Guaranteed Cost Time-Varying Formation Tracking for High-Order Multiagent Systems With Time-Varying Delays," in *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 50, no. 4, pp. 1465-1475, April 2020.

[59] J. Du, J. Li and F. L. Lewis, "Distributed 3-D Time-Varying Formation Control of Underactuated AUVs With Communication Delays Based on Data-Driven State Predictor," in *IEEE Transactions on Industrial Informatics*, vol. 19, no. 5, pp. 6963-6971, May 2023.

[60] Z. Wang, Z. Tang, Y. Lu, N. Du and J. Zhou, "Dynamic Event-triggered Time-varying Formation Control for Discrete-time Multi-agent Systems," *2022 China Automation Congress (CAC)*, Xiamen, China, pp. 6469-6474.

[61] Z. Li, J. Qian and Z. Duan, "Distributed Multi-layer Time-varying Output Formation Tracking Control for Heterogeneous Linear Multiagent Systems," *2021 33rd Chinese Control and Decision Conference (CCDC)*, Kunming, China, pp. 5566-5571.

[62] X. Zhang, X. Zhan, J. Wu, T. Han and H. Yan, "Distributed Adaptive Time-Varying Formation Tracking for Input Saturated Multi-Agent Systems With Multiple Leaders," in *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 71, no. 4, pp. 2084-2088, April 2024.

[63] S. Zhao, "Affine Formation Maneuver Control of Multiagent Systems," in *IEEE Transactions on Automatic Control*, vol. 63, no. 12, pp. 4140-4155, Dec. 2018.

[64] Z. Huang, R. Bauer and Y. -J. Pan, "Affine Formation Control of Multiple Quadcopters," *IECON 2022 – 48th Annual Conference of the IEEE Industrial Electronics Society*, Brussels, Belgium, 2022, pp. 1-5.

[65] Z. Luo, P. Zhang, X. Ding, Z. Tang, C. Wang and J. Wang, "Adaptive Affine Formation Maneuver Control of Second-Order Multi-Agent Systems with Disturbances," *2020 16th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, Shenzhen, China, pp. 1071-1076.

[66] W. Li, H. Zhang, Y. Mu and Y. Wang, "Bipartite Time-Varying Output Formation Tracking for Multi-Agent Systems With Multiple Heterogeneous Leaders Under Signed Digraph," in *IEEE Transactions on Industrial Informatics*, 2023.

[67] X. Zhao and L. Zhang, "Adaptive Bipartite Time-Varying Output Formation Tracking of Heterogeneous Multi-Agent Systems, in *IEEE Control Systems Letters*, vol. 8, pp. 394-399, 2024

[68] X. Li, M. J. Er and J. Wang, "Time-Varying Formation Control of Nonholonomic Multi-Agent Systems," *2019 2nd International Conference on Intelligent Autonomous Systems (ICoIAS)*, Singapore, pp. 118-123.

[69] D. Van Vu and M. H. Trinh, "Decentralized sliding-mode control laws for the bearing-based formation tracking problem," *2021 International Conference on Control, Automation and Information Sciences (ICCAIS)*, Xi'an, China, pp. 67-72.

[70] G. Jing and L. Wang, "Multiagent Flocking With Angle-Based Formation Shape Control," in *IEEE Transactions on Automatic Control*, vol. 65, no. 2, pp. 817-823, Feb. 2020.

[71] H. Su, C. Chen, Z. Yang, S. Zhu and X. Guan, "Bearing-Based Formation Tracking Control With Time-Varying Velocity Estimation," in *IEEE Transactions on Cybernetics*, vol. 53, no. 6, pp. 3961-3973, June 2023.

[72] J. N. Yasin, S. A. S. Mohamed, M. -H. Haghbayan, J. Heikkonen, H. Tenhunen and J. Plosila, "Unmanned Aerial Vehicles (UAVs): Collision Avoidance Systems and Approaches," in IEEE Access, vol. 8, pp. 105139-105155, 2020.

[73] D. Alves de Lima and A. Corra Victorino, "A Hybrid Controller for Vision-Based Navigation of Autonomous Vehicles in Urban Environments," in *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 8, pp. 2310-2323, Aug. 2016.

[74] Ye Bin-qiang, Zhao Ming-fu and Wang Yi, "Research of path planning method for mobile robot based on artificial potential field," *2011 International Conference on Multimedia Technology*, pp. 3192-3195.

[75] H. -I. Lin and M. -F. Hsieh, "Robotic Arm Path Planning Based on Three-Dimensional Artificial Potential Field," 2018 18th International Conference on Control, Automation and Systems (ICCAS), PyeongChang, Korea (South), 2018, pp. 740-745.

[76] Y. Hargas, A. Mokrane, A. Hentout, O. Hachour and B. Bouzouia, "Mobile manipulator path planning based on artificial potential field: Application on RobuTER/ULM," *2015 4th International Conference on Electrical Engineering (ICEE)*, Boumerdes, Algeria.

[77] S. Wasiela, N. Kasshyap, Y.J. Pan and J. Awe, "Realization of Consensus with Collision and Obstacle Avoidance in an Unknown Environment for Multiple Robots," *Proceedings of the IEEE International Conference on Robotics and Biomimetics (ROBIO)*, December 2019, Dali, China, 740-745.

[78] Y. Yan, J. Li, H. Dong, C. Gao and Y. Fang, "An Improved Artificial Potential Field Method for Formation Control and Obstacle Avoidance of the Multi-Agents Systems," *2023 China Automation Congress (CAC)*, Chongqing, China, pp. 2877-2882.

[79] Z. Pan, C. Zhang, Y. Xia, H. Xiong and X. Shao, "An Improved Artificial Potential Field Method for Path Planning and Formation Control of the Multi-UAV Systems," in *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 69, no. 3, pp. 1129-1133, March 2022.

[80] W. Pang, D. Zhu and C. Sun, "Multi-AUV Formation Reconfiguration Obstacle Avoidance Algorithm Based on Affine Transformation and Improved Artificial Potential Field Under Ocean Currents Disturbance," in *IEEE Transactions on Automation Science and Engineering*, vol. 21, no. 2, pp. 1469-1487, April 2024.

[81] W. Di, L. Caihong, G. Na, S. Yong, G. Tengteng and L. Guoming, "Local Path Planning of Mobile Robot Based on Artificial Potential Field," *2020 39th Chinese Control Conference (CCC)*, Shenyang, China, pp. 3677-3682.

[82] Y. Huang et al., "A Motion Planning and Tracking Framework for Autonomous Vehicles Based on Artificial Potential Field Elaborated Resistance Network Approach," in *IEEE Transactions on Industrial Electronics*, vol. 67, no. 2, pp. 1376-1386, Feb. 2020.

[83] X. Lin, Z. -Q. Wang and X. -Y. Chen, "Path Planning with Improved Artificial Potential Field Method Based on Decision Tree," *2020 27th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS)*, St. Petersburg, Russia, pp. 1-5.

[84] Y. Guo, X. Liu, W. Zhang, X. Liu and Y. Yang, "Obstacle Avoidance Planning for Quadrotor UAV Based on Improved Adaptive Artificial Potential Field," *2019 Chinese Automation Congress (CAC)*, Hangzhou, China, pp. 2598-2603.

[85] H. Shen and P. Li, "Unmanned Aerial Vehicle (UAV) Path Planning Based on Improved Pre-planning Artificial Potential Field Method," 2020 Chinese Control And Decision Conference (CCDC), Hefei, China, 2020, pp. 2727-2732.

[86] C. Ju, Q. Luo and X. Yan, "Path Planning Using Artificial Potential Field Method And A-star Fusion Algorithm," *2020 Global Reliability and Prognostics and Health Management (PHM-Shanghai)*, Shanghai, China, pp. 1-7.

[87] C. Liu, L. Zhai and X. Zhang, "Research on local real-time obstacle avoidance path planning of unmanned vehicle based on improved artificial potential field method," *2022 6th CAA International Conference on Vehicular Control and Intelligence (CVCI)*, Nanjing, China, pp. 1-6.

[88] M. Mukadam, X. Yan and B. Boots, "Gaussian Process Motion planning," *2016 IEEE International Conference on Robotics and Automation (ICRA)*, Stockholm, Sweden, pp. 9-15.

[89] S. Guo, B. Liu, S. Zhang, J. Guo and C. Wang, "Continuous-time Gaussian Process Trajectory Generation for Multi-robot Formation via Probabilistic Inference," *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Prague, Czech Republic, pp. 9247-9253.

[90] B. Akbari and H. Zhu, "Fault-Resilience Role Engine for an Autonomous Cooperative Multi-Robot System using E-CARGO," *2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Prague, Czech Republic, pp. 730-735.

[91] J. Meng, Y. Liu, R. Bucknall, W. Guo and Z. Ji, "Anisotropic GPMP2: A Fast Continuous-Time Gaussian Processes Based Motion Planner for Unmanned Surface Vehicles in Environments With Ocean Currents," in *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 4, pp. 3914-3931.

[92] W. Wang and B. Hu, "Multi-Manipulator Motion Planning based on Gaussian Process with Probabilistic Inference," *2022 WRC Symposium on Advanced Robotics and Automation (WRC SARA)*, Beijing, China, pp. 221-226.

[93] Z. Huang, R. Bauer and Y. -J. Pan, "Closed-Loop Identification and Real-Time Control of a Micro Quadcopter," in *IEEE Transactions on Industrial Electronics*, vol. 69, no. 3, pp. 2855-2863, March 2022.

[94] W. Benaziza, N. Slimane and A. Mallem, "Mobile robot trajectory tracking using terminal sliding mode control," *2017 6th International Conference on Systems and Control (ICSC)*, Batna, Algeria, pp. 538-542.

[95] L. Wan and Y. -J. Pan, "Bilateral Teleoperation of a Multi-Robot Formation with Time-Varying Delays using Adaptive Impedance Control," *2022 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, Sapporo, Japan, pp. 1739-1746.

Appendix A

Published Works

Journal Papers

1. **R. Adderson**, Y.J. Pan, "Continuously Varying Formation for Heterogeneous Multi-Agent Systems with Novel Potential Field Avoidance", *IEEE Transactions on Industrial Electronics*, doi: 10.1109/TIE.2024.3429620.

2. **R. Adderson**, B. Akbari, Y.J. Pan and H.B. Zhu, "Multi-Leader and Role-Based Time-Varying Formation Using GP Inference and Sliding Mode Control", *IEEE/ASME Transactions on Mechatronics*, doi: 10.1109/TMECH.2024.3385716.

3. B. Akbari, Z. Wang, H. Zhu, L. Wan, **R. Adderson** and Y. -J. Pan, "Role Engine Implementation for a Continuous and Collaborative Multirobot System," in *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 54, no. 2, pp. 866-877, Feb. 2024.

Conference Papers

1. **R. Adderson**, L. Wan and Y. -J. Pan, "Decentralized Time-Varying Formation with Dynamic Leader Selection," *IECON 2023- 49th Annual Conference of the IEEE Industrial Electronics Society*, Singapore, Singapore, 2023, pp. 1-6.

2. **R. Adderson** and Y. -J. Pan, "Formation Shaping Control for Multi-Agent Systems with Obstacle Avoidance and Dynamic Leader Selection," *2022 IEEE 31st International Symposium on Industrial Electronics (ISIE)*, Anchorage, AK, USA, 2022, pp. 1082-1087.

3. **R. Adderson** and Y. -J. Pan, "Terminal Sliding Mode Control for the Formation of a Team of Quadrotors and Mobile Robots," *IECON 2021 47th Annual*

Conference of the IEEE Industrial Electronics Society, Toronto, ON, Canada, 2021, pp. 1-6.

4. **R. Adderson**, Y. -J. Pan and H. Shen, "Application of Sliding Mode Control for the Formation of Heterogeneous Multi-Agent Systems," *2021 IEEE Conference on Control Technology and Applications (CCTA)*, San Diego, CA, USA, 2021, pp. 777-782

5. T. Hubbard and **R. Adderson**, "2D Digital Image Correlation for Sub nm MEMS Measurements," *Progress in Canadian Mechanical Engineering.* Volume 3, Jun. 2020.