Swarm Coordination Scheme for Fixed-Wing UAVs

by

John Kelly Mills

A thesis submitted to the Graduate School of
Florida Institute of Technology
in partial fulfillment of the requirements
for the degree of

Master of Science
in
Mechanical Engineering

Melbourne, Florida
March, 2015

We, the undersigned committee, hereby approve the attached thesis, "Swarm Coordination Scheme for Fixed-Wing UAVs" by John Kelly Mills.

Tiauw Go, Ph.D.
Major Advisor
Associate Professor, Aerospace Engineering

Matthew Jensen, Ph.D.
Committee Member
Assistant Professor, Mechanical Engineering

James Brenner, Ph.D.
Committee Member
Assistant Professor, Chemical Engineering

Hamid Hafazi, Ph.D.
Academic Unit Head
Department Head, Mechanical and Aerospace Engineering

Abstract

Title: Swarm Coordination Scheme for Fixed-Wing UAVs

Author: John Kelly Mills

Advisor: Tiauw Go, Ph. D.

The objective of this thesis is to develop a swarm model and control laws that satisfy the requirements of flocking, target tracking, and speed regulation while respecting the performance limits of a fixed-wing aircraft. The control architecture is decentralized and involves two levels: a swarm coordination level and a flight control and stabilization level. The swarm coordination level is the focus of this thesis. The agents are modelled in both two and three dimensions as a point mass with attitude coordinates described by a simplification of the direction cosine kinematic equations. The swarm controller is a weighted sum of gradient controllers based on a Lyapunov function related to the relative states of the system. Stability analysis shows that the proposed design is locally stable at a minimum. Simulations suggest that the controller can tolerate more extreme initial conditions with strategic placement of informed agents. The dynamic network topology is based on a fixed sensing radius and can cause an undesirable squeezing effect on the swarm as well as discontinuities in the control inputs. Three methods of artificial control saturation are proposed that will suppress the controls according to the UAV's performance limits. Results indicate that the method, here called pre-saturation, can actually improve performance by redistributing the control effort over time.

Keywords: swarm coordination, fixed-wing, UAV, nonholonomic, decentralized, artificial potential function, Lyapunov function, gradient controller, saturation

Table of Contents

List of Figures

List of Tables

Acknowledgements

I would like to thank my adviser, Dr. Tiauw Go, for his guidance and support for the past ten months as I completed the research presented in this document. Halfway through my graduate degree, I decided that swarm robotics and advanced control theory was a primary interest of mine, but I did not know how to approach such a substantial research field. Through Dr. Go's guidance, I was able to stay focused on my research goals, complete my research, and further my education in engineering.

I would like to thank Dr. Matthew Jensen for supporting me financially for the first year of my graduate degree and for serving on my defense committee. I would also like to thank Dr. James Brenner for serving on my committee and for being an inspiring teacher both inside and outside of the classroom.

I would like to thank my coworkers and friends for their support and patience while I committed a great deal of my time and attention to this thesis, especially in the last few months. I am grateful for the amazing coworkers I have come to know in my two years at the Cape and the small, yet extraordinary group of friends I have made over the years. I want to especially thank my good friend, Casey, for combing through this document with a fine-tooth comb to catch all the little errors and typos.

Lastly, I would like to thank my parents and grandparents for their continuous love and support throughout my arduous pursuit of higher education in engineering.

Dedication

To my father and my mother,

And to mommom and granddad,

For giving me the tools needed to build my own success and see the world beyond myself.

For providing the guidance I need when decisions seem impossible.

For the love and support that gives me strength and hope.

Chapter 1
Introduction

Background

Multi-Agent Systems

Research in multi-agent systems (MAS) as it applies to mobile robots has seen much attention in the last three decades. In general, MAS refers to any system in which multiple dynamic and *intelligent* agents share a common workspace and must meet objectives either cooperatively or independently. The field of MAS covers a broad range including both robotic agent and software agent systems and problems ranging from simple distributed agreement to more complex tasks like cooperative search or gradient climbing. The language of MAS is well defined in the fields of computer science (CS) and artificial intelligence (AI) and will be used in the following paragraphs to explain the various components of MAS; however, in the later chapters the focus will be on multi-robot systems and related control theory while research in AI will receive little attention.

The environment is generally unknown by the agents except for the local region it can sense. A set of agents, called the informed agents, will know the objective state which can be any set of environment states that the designer wants to find or track. Examples of environment states are relative speed and position of an adversarial agent(s) to an informed agent(s) or spatially varying properties of the environment such as air or ocean temperature. This means that the *performance measure* [1] of the system is directly related to all agents' ability to respond to the environment; when all informed agents achieve the objective state then the system's performance measure is maximized and the mission is successful. There could, of course, be other *internal* performance measures

related to each individual agent that quantify how well the agent performs with its limited resources and motion capability; for example, the performance measure of a single unmanned aerial vehicle (UAV) could be related to its fuel and power consumption and relative position to other agents. In control theory, these performance measures are represented in the form of cost functions that are either used for optimal control or the development of an explicit control law.

An agent is defined above as being both dynamic and intelligent. The dynamic part is intuitive, but the definition of intelligent in this case is not trivial. Here, the definition of intelligent agents will be further clarified by borrowing some terminology and definitions from the fields of CS and AI [1]. An agent is simply defined as anything that can see its environment through sensors and act upon those readings via actuators. This definition is very basic and does not give any guidance in the design of an agent that will be successful in its mission; this is why AI practitioners also introduce the concept of rational agents. A rational agent performs actions that are *expected* to maximize its performance measure based on sensor readings it takes from the environment, the history of sensor readings, and any built-in knowledge that it possesses. Emphasis is placed on "expected" because an agent that knows the actual actions that will maximize its performance measure in the future would not be considered rational because that means it has perfect knowledge of future events and the environment and, therefore, does not need to use "rational thought" to determine its actions. This type of agent would be considered omniscient and is impossible in reality. This concept is explained in more detail with anecdotes in reference [1].

An agent can be further characterized based on how it uses the information it collects or knows. An agent can be *a simple reflex type,* which means it only reacts based on its current sensor readings, while a *model-based* reflex agent will react based on both the current sensor readings and a history of sensor readings that allow it to infer other properties of the environment. A model-based reflex agent uses an observer from control theory to estimate properties of the environment that it cannot normally see with its sensors; an example is an agent that can only measure its own speed and its relative position to a nearby agent or obstacle but uses the history of that information to estimate its relative velocity to the nearby agent or obstacle. An agent can also be *goal-based* and

utility-based. A goal-based agent picks actions based on the goal that it must achieve and a utility-based agent will pick the sequence of actions that achieves the goal in the most efficient way. In essence, a utility-based agent is designed to optimize a cost function that is designed to guide the agent to its goal, but also do it in a way that minimizes its expected cost. In this work, all agents will be considered simple and goal-based.

Agents can also be classified based on their role in the system. The most important type is the informed agent (also known as the leader agent). Informed agents know the information necessary to complete the mission. The set of informed agents can be a single agent, a group of agents, or all agents. If an agent is not an informed agent, then it is a follower agent; these types of agents are designed to follow the lead of the informed agents. Follower agents do not need to directly follow an informed agent; rather, they can follow any set of follower agents that are being led by an informed agent. Choosing a suitable ratio of informed agents to follower agents has been the topic of several research efforts [2]. Another type of agent that was introduced above is an adversarial agent; as one might suspect, this kind of agent acts against the system. An adversarial agent cannot be directly controlled by the designer and may act to minimize the system's performance measure. Examples of adversarial agents are hostile aircrafts that a group of UAVs are tasked with capturing or neutralizing, static or dynamic obstacles in the environment, and autonomous cars competing for a parking spot.

A system of agents that are the same type is called homogenous (i.e. a formation of identical aircrafts). Conversely, a system where differences exist between some or all agents is called heterogeneous (i.e. a system of cooperating aircrafts and ground vehicles). Agents can differ in their physical configuration, functional capability, mathematical model, control law, sensing mechanism, communication method, etc. Essentially a difference between agents is defined as anything that causes a significant change in the way agents can interact with each other or the environment or a change in the way they are controlled. A heterogeneous system is more difficult to design than a homogenous system because not all agents can run the same algorithm; the amount of extra design effort depends on the degree of heterogeneity.

Swarm Coordination and Control

The class of problems in MAS research considered in this thesis is swarm coordination and control. The goal of swarm problems is to achieve emergent behaviors such as flocking, social foraging, formation flight, and distributed agreement. An emergent behavior occurs when a collection of agents with simple behaviors interactively generate a complex group behavior. In this context, simple is interpreted relative to the group behavior. A prime example is the shape and movement of a flock of birds; each bird can control simple properties like its speed and orientation but due to the complex quantity and nature of interactions between birds, group properties such as flock shape, size, speed, direction, and density will vary in a highly complex manner. See the work from Gazi and Passino for a more thorough review of the different emergent behaviors seen in nature that are of engineering interest [3].

The definition of emergent behaviors used in engineering systems primarily comes from biological research of animal species that exhibit swarm behavior such as herds of land animals, schools of fish, flocks of birds, and swarms of bees. Each of the emergent behaviors listed above have characteristics that uniquely define them. Some of them are based on only local interactions of the agents while others are generated based on the group's response to the environment. All of the emergent behaviors of interest will be explained in more explicit detail in the next chapter.

In the literature on swarm coordination and control, researchers have investigated many different ways to model and control a multi-robot system to solve the various problems previously mentioned. Early research efforts started with linear kinematic and dynamic models, which provided very useful results including the well-known solution to the distributed agreement problem [4]. To solve the more complex problems in swarm robotics, some techniques from nonlinear control systems were adopted such as artificial potential functions (APF) [5], [6], navigation functions [7], [8], sliding mode control [9], [10], backstepping [11], [12], control Lyapunov functions [13], and optimal control [14].

Applications

Swarms of robots can be useful in many real world problems that occur in a large environment or involve a large task that is impractical for one agent to perform. According to a recent review of swarm robotics (SR) [15], there are several issues that need to be overcome before robot swarms will be seen in actual use; moreover, no researchers have proposed a solution that successfully unifies techniques from all fields of study (i.e. robotics, control theory, AI, CS, etc.) and solves all issues identified in the review. Here is the list of issues to be addressed simultaneously, reproduced from [15]:

1. Devising a hybrid distributed scheme that overcomes the drawbacks of pure centralized and decentralized systems;
2. Devising a self-organizing SR system that can be supervised on an abstract level;
3. Ensuring that devised mechanisms are scalable and robust;
4. Generating and maintaining energy-efficient connected formations that support the task at hand;
5. Ensuring that any robot can facilitate mapping and localization mechanisms;
6. Incorporating path planning and obstacle avoidance mechanisms that do not depend on a particular specialized leader;
7. Devising mechanisms that enable SR systems to manipulate and transport objects efficiently in the application area at hand; and
8. Taking measures to prevent the SR system from shutting down prematurely as a result of depleted energy sources.

Most physical test systems for swarm robotics exist only in a lab and are controlled using motion capture system [13], [16], [17]. This works well for testing and a small class of real world scenarios, but most problems do not facilitate the use of motion capture systems as a control mechanism. One swarm system that actually operates in a real environment was recently reported by the U.S. Navy; the system is called CARACaS (Control Architecture for Robotic Agent Command and Sensing) and was developed on the Office of Naval Research (ONR) to command a group of unmanned surface vehicles (USVs) to autonomously identify, target, and enclose hostile vessels while they escort and defend a high value asset [18]. The system was demonstrated for two weeks in

August 2014 at James River, Virginia and involved 13 autonomous boats participated in the escort and threat neutralization task. The hardware and software of CARACaS was adapted from NASA's Jet Propulsion Lab (JPL) Mars Rover program, which has been tested by the Navy for the past eight years on USVs and unmanned underwater vehicles (UUVs) [19]. This control system incorporates many aspects of AI that are necessary to create rational agents with the ability to make complex decisions based on the information they receive from the environment. This system does exhibit aspects of both centralized and decentralized control schemes since each agent determines its own path, but sensor information is shared throughout the swarm and a human operator has the ability to monitor and control certain aspects of the swarm.

A very common application considered for swarms is the deployment of mobile sensor networks. The idea is that each swarm member will be armed with one or more sensors that will measure properties of the environment so that the entire network of sensors can characterize a large area of the environment. Each agent can either record the data or use it as feedback to guide its trajectory through the environment. The advantage of such a system is that many inexpensive robots can be used to cover a larger area and provide a more accurate and detailed map of environment than would be obtained from a single robot. Examples of such systems would be unmanned underwater vehicles (UUVs) that follow temperature gradients in the ocean.

The primary application of swarms in the defense industry is surveillance and reconnaissance missions [20]. An autonomous swarm of small UAVs can cover a much greater area with fewer operators than a single large UAV. Although the swarm requires many more UAVs, they can be designed to be smaller and less complex and with the inclusion of an effective autonomous control system each UAV can generate its own path and require fewer personnel at the ground station to operate it. This is essentially the same concept as a mobile sensor network. Other possible applications are coordinated defense in which a swarm of autonomous agents defends a high value target [21], and coordinated tracking in which a swarm tracks and eventually neutralizes a target.

Objectives

The objective of this thesis is to develop a swarm model and control laws that achieve swarm coordination objectives and that respects the performance limits of a fixed-wing aircraft. The control architecture considered here is decentralized and involves two levels; level one handles the swarm coordination tasks, while level two handles the flight control and aircraft stabilization tasks. This scheme allows the aircrafts to be modelled as simpler entities at the swarm control level while the flight control level takes into account the more specific aspects of the aircraft's form and function.

The *swarm controller* is tasked with taking information from the environment and generating a trajectory that each agent can follow to achieve swarm behavior. An objective of the swarm controller design is to use a simple model that still respects the motion constraints of the aircraft. In addition, the control law should be designed in such a way that it respects the performance limits of the aircraft such as maximum thrust, minimum turning radius, and maximum rate of climb. The *flight controller* is tasked with following the trajectory generated by the swarm controller and maintaining stable flight. In this work, only the swarm controller is considered.

Contributions

The contributions of this work are as follows:

1. Derived a gradient controller based on control law presented in [22] that satisfies objectives of flocking, target tracking, and speed regulation.
2. Proved that the proposed controller is locally stable according to the Lyapunov theorem for local stability.
3. Defined evaluation criteria that can be used to measure how successful a swarm system is at meeting the objectives of flocking.
4. Presented simulations that test the proposed controller when subjected to extreme initial conditions, limited communication, and large number of agents.
5. Proposed three methods to suppress control effort using artificial saturation.

Outline

This thesis is organized in the following manner: Chapter 2 presents a review of the literature related to swarm coordination and control problems; Chapter 3 discusses the equations of motions and associated theory used to model dynamic agents in this thesis; Chapter 4 addresses the design of a stabilizing controller for swarm coordination and control based on Lyapunov stability theory; Chapter 5 presents the results from testing the control laws developed in Chapter 4 against evaluation criterion designed to quantify the objectives of swarm coordination; Finally, Chapter 6 closes the discussion with conclusions and future work.

Chapter 2
Literature Review

Objectives in Swarm Coordination

In this thesis, the objectives of swarm coordination are divided into two categories: internal and external. Internal objectives refer to those objectives which depend on inter-agent states. If the multi-robot system is viewed as it is roaming free in an ideal and open environment, then the internal objectives describes the way in which the designer wants the agents to interact and ultimately how the system behaves as a collective. In other words, the internal objectives are criterion to achieve an emergent behavior. The external objectives refer to objectives which depend on the relative states between the environment and the informed agents. External objectives describe the way in which the designer wants the multi-robot system to behave while still satisfying the internal objectives if it is placed in a real and closed environment. In other words, the external objectives are the mission objectives for a particular application as well as any objective which must be satisfied to maintain the system's emergent behavior. The specific objectives will be discussed in the following sections.

Internal Objectives

Swarm coordination has been a problem of interest in the research field of multi-agent systems since the groundbreaking work of Craig Reynolds in 1987 [23] in which he created the first computer simulation of flocking birds. Based on research of biological swarms, Reynolds proposed three rules that are a minimum to achieve flocking behavior. These rules are:

1. Separation: Agents avoid collisions with their neighbors.
2. Velocity Matching: The velocities of all agents converge to the same value.
3. Cohesion: Agents attempt to stay close to their neighbors.

In other words, flocking occurs when all agents travel in the same direction as a cohesive group while maintaining an acceptable separation distance. One may notice that these rules do not give any guidance as to what direction the flock will travel in; Reynolds' rules only govern the relative motions of all agents. Therefore, a control law designed according to the rules will yield a swarm that achieves flocking behavior, but has no capability to navigate through the environment. That means Reynolds' rules can achieve only internal objectives. Methods to achieve external objectives will be discussed later.

Flocking describes the emergent behavior, but initially a set of agents may not be in a configuration that satisfies the flocking requirements. To reach such a configuration, the swarm must aggregate, or gather together as a cohesive group. The group does not need to be organized or shaped in a particular way; this is defined as a formation and will be discussed later. Once a set of agents has aggregated, they can participate in flocking and achieve some external objective.

Formation control has three main tasks associated with it: formation acquisition, maintenance, and reconfiguration [3]. Acquisition is like a specific case of aggregation in which the relative position of agents must be exactly achieved, usually to generate a geometric formation that is ideal for the application. Formation maintenance is the task of keeping the geometric formation during the mission; this is analogous to flocking. Formation reconfiguration occurs when the formation must change its size and shape in order to avoid an obstacle in the environment or complete part of the mission.

A popular topic in swarm coordination and control is the concept of consensus or distributed agreement. Distributed agreement is simply when some or all of the states or outputs of all agents converge to the same value. This is a generalization of Reynolds' velocity matching rule for any state. Over the years, researchers have studied many aspects of the consensus problem including consensus over a time-varying network

topology [24], consensus without velocity measurement [25], output agreement [26] and controllability [27] in high-dimensional systems, consensus of multiple rigid spacecraft using a leader-follower framework [28], consensus in the presence of time delays and intermittent communication [29], [30], and distributed observer-based consensus protocols using a leader-follower framework [31].

Most work on consensus protocols use linear integrator models or nonlinear unicycle models with integrators that make it possible to use a linear control law to achieve consensus. Some research has gone into investigating the use of coupled oscillators as consensus protocols [32]. In fields such as biology and chemistry, this model is a natural choice based on system dynamics and is often referred to as the Kuramoto model [33]. In swarm coordination and control, the coupled oscillator function is typically used in an alternate form when a vector based dynamic model is used [22]. For a more detailed discussion of consensus protocols, refer to [4].

External Objectives

The direction that the swarm travels can be controlled based on some prior knowledge or measurement of the environment. Two common types of external objectives are social foraging and swarm tracking [3]. Social foraging occurs when agents cooperatively search the environment for favorable regions and avoid dangerous regions according to the performance measures that the designer chooses. Social foraging is crucial in the design of a mobile sensor network. Swarm tracking is similar to social foraging and flocking, but the task is to track and follow a specific target. Swarm tracking is a type of pursuer-evader problem in which the pursuer is designed to follow and capture the evader [34]. The problem can also be reversed such that the swarm is designed to evade some perceivable threat in the environment. This could be considered a dynamic version of the obstacle avoidance problem. Typically obstacles are static entities in the environment, but when an obstacle is dynamic and adversarial, it becomes an evasion problem.

Formation reconfiguration, or switching, involves changing the geometric formation during the mission due to a change in the environment or mission requirements. An example of a change is a swarm of UAVs encountering a narrow passage that is too small

for the formation to pass through. This is an aspect of overall swarm system control in which each agent has one or more parameters that can be manipulated to change the overall swarm size and shape. These parameters can be manipulated by a global broadcast from a centralized unit or an observer programmed into each agent's software that estimates the swarm parameters [35]. Typically these parameters are related to the swarm size, shape, or centroid.

Review of Dynamic Agent Models

Dynamic agents taking part in swarm control tasks can be modelled in various ways depending on the design requirements. The type of model is strongly dependent on the type of robot that the agent represents and whether or not the swarm control task is separated from the robot control task. When the two control tasks are handled separately, a simpler agent model can be implemented in the swarm control task while the robot control task uses a model that is exactly tailored to its specific dynamics.

In general, models can be classified as being kinematic or dynamic; linear or nonlinear; continuous time or discrete time; and fully actuated or nonholonomic. In a kinematic model, the velocity of the agent's states is controlled. In a dynamic model, the acceleration of the agent's states is controlled. In a nonlinear model, the relationship between the agent's states and their first derivative is represented by a nonlinear function. Most real systems exhibit nonlinearities, but sometimes they can be ignored if the performance range is small enough. When analyzing a system or designing a controller mathematically, the models are typically considered in continuous time; but when it comes time for actual implementation, a discrete time model is used.

An important classification for a dynamic model is whether it is fully actuated or nonholonomic. Fully actuated models allow every state to be independently controlled. Nonholonomic models have constraints that limit the degrees of freedom of the system, yet the entire state space is still reachable [36]. In other words, fully actuated systems have the same number of degrees of freedom as generalized coordinates, and nonholonomic systems have fewer degrees of freedom than generalized coordinates.

The most popular models in swarm coordination literature are the single integrator [5], [27], [34], [37], [38], double integrator [2], [6], [9], [12], [14], [25], [27], [29], [30], [35], [39]–[45], and unicycle model [16], [20], [46]–[57]. The single integrator represents agents as kinematic particles and the double integrator represents an agent as a point mass. Both of these models are linear and fully actuated. The single integrator model is given by:

$$\dot{\vec{p}}(t) = \vec{v}(t)$$

where $\vec{p}(t)$ is the agent's position vector and $\vec{v}(t)$ is the agent's velocity vector as well as the control input. The double integrator model is found by adding another integrator:

$$\dot{\vec{p}}(t) = \vec{v}(t)$$
$$\dot{\vec{v}}(t) = \vec{a}(t)$$

where $\vec{a}(t)$ is the acceleration vector and the control input. Both of these models can be used in two- or three-dimensions by simply changing the vector dimensions.

The unicycle model is a nonholonomic model and can only be used in two-dimensional space. It can be used in both a kinematic and dynamic form. The dynamic form of the unicycle model is given by:

$$\dot{x}(t) = v(t)\cos\theta(t)$$
$$\dot{y}(t) = v(t)sin\theta(t)$$
$$\dot{v}(t) = a(t)$$
$$\dot{\theta}(t) = \omega(t)$$

where $[x(t), y(t)]$ is the agent's position vector, $v(t)$ is the agent's speed, $\theta(t)$ is the agent's heading angle, $a(t)$ is the agent's forward acceleration, and $\omega(t)$ is the agent's angular velocity. The control inputs for this model are $a(t)$ and $\omega(t)$. The unicycle model can also be assumed to have constant or unit speed, and in the case of UAV swarm applications, it can be assumed that a lower level controller regulates altitude.

Literature surrounding the topic of three-dimensional swarm control is quite scarce. The transition from a two- to three-dimensional framework brings several choices of attitude coordinates. The Frenet-Serret and Bishop frames have been used for swarming and flocking control by a few researchers in the field [22], [58]–[60]. The Euler angle model has seen attention in UAV applications [7], [61]. The quaternion attitude model has been used for control of multiple spacecraft [28].

Review of Swarm Stability and Control

Designing a controller to generate swarm behavior is a challenging task. The controller must not only satisfy a set of rules such as the ones proposed by Reynolds, but must also guide the agents based on external stimuli such as targets and obstacles. Satisfying all of these conditions simultaneously requires a complex, often nonlinear, controller. The tools available to control system designers for nonlinear systems are scarce compared to linear systems. Typically, nonlinear control systems are designed and verified as stable using Lyapunov stability theory [62]. Other approaches exist such as nonlinear model predictive control (NMPC) [63] and various robust control designs [64], but most methods fall back on the design of a Lyapunov function for the purposes of design or stability analysis.

Designing a swarm controller comes down to designing functions that model how an agent interacts with other agents and the environment. Part of this interaction is the consensus protocols discussed earlier, but another component is the spacing dynamics that control relative positions. Spacing dynamics is usually modeled using artificial potential functions (APF) with near field repulsion and far field attraction behavior for agent-to-agent interactions [5]. More complex forms of potential functions have been proposed for swarm coordination and control that provide some interesting possibilities. Harmonic functions, which are commonly used in fluid mechanics to model streamlines, have been discussed as candidates for obstacle avoidance functions in swarm control [54], [65]. A class of Dipolar Navigation Functions have been used to control agents to move towards targets and away from obstacles and each other while respecting the aircraft's performance limits [7], [8], [66]. A novel concept of morphing potential

functions has been proposed in [67], which uses a common APF candidate that has been modified to “morph” the shape of the potential based on angle of approach, relative velocity, and kinematic aircraft constraints. The purpose of this scheme is to give the aircraft more time to respond by shifting the centroid of the potential field ahead of the obstacle and the shape of the potential near the centroid.

In recent years, various different solutions have been proposed to solve different problems in swarm coordination. Some approaches attempt to achieve swarm behavior with limited sensing capability. Rastgofter and Jayasuriya [38] describe an approach that uses an initial geometric layout of the agents in two-dimensions such that the leader agents make up a convex boundary around the follower agents. The agents determine their paths based on their *perception* of where the neighboring agents are instead of exact positions. Thus, peer-to-peer communication is not necessary. Maintaining connectivity in multi-agent systems is a notable concern since some conditions may force an agent to stray too far away from the swarm. Mao *et al*. [44] proposes a solution in which each agent maintains an estimate of the system’s algebraic connectivity and uses it for control such that flocking is achieved while maintaining global connectivity. Unlike other approaches, this method allows flexibility in that a communication link can be broken as long as global connectivity is not compromised.

Some of the focus in swarm coordination research has gone to control strategies that can support mobile sensor networks. Dhananjay and Kristiansen [57] developed a guidance strategy to guide multiple cooperative agents to track circular paths using gradient search methods. Ögren *et al*. [40] used virtual bodies and artificial potentials to perform adaptive gradient climbing missions for mobile sensor networks. Zhang *et al*. [68] proposed a method to estimate the environmental field along the averaged trajectories of the mobile sensors using distributed measurements and a Kalman filter.

More advanced nonlinear control techniques have also been implemented. A popular nonlinear control method based on Lyapunov theory is sliding mode control and has been implemented by several researchers in swarm coordination. It has been used by Gazi *et al*. [55] to solve the target tracking problem and Han *et al*. [10] to follow a reference

trajectory while avoiding collisions with obstacles and pop-up threats. Stastny *et al*. [67] propose a very intricate approach using "morphing" potential functions, online predictive and prioritized waypoint planning, and NMPC to achieve real-time collision and obstacle avoidance behavior in fixed-wing UAV systems with high speed and inertia. The UAVs must operate in a congested environment such as a city with tall buildings. This approach is very novel and is supported by excellent simulation results. Wang and Xin [14] use a unified optimal control approach in which all the objectives of flocking behavior and navigation are represented by a nonquadratic cost function, and the optimal control law is derived analytically by a novel inverse optimal control strategy. Li *et al*. [56] implement a multi-level control algorithm which uses an inner-loop $\mathcal{L}_1$ adaptive control law to follow a trajectory generated by an outer-loop guidance law. Chiew *et al*. [13] presented a control approach using the robust control Lyapunov function design described in [64].

Justh and Krishnaprasad [22] proposed a simple control law for nonholonomic agents described by the Frenet-Serret equations to achieve stable formation and flocking behavior in UAVs. Morgan and Schwartz [59] presented a modification to this work in which the control law can be changed in real-time to break the swarm into independent subswarm clusters. Justh and Krishnaprasad later extended their work to three-dimensions using the Bishop frame equations [58]. These equations come from differential geometry of curves where they are used to describe the orientation and position of a unit speed particle on a curve using a set of orthonormal unit vectors. In two dimensions, the model is essentially the same as the unicycle model. An advantage of framing the model in this way is that the control laws developed in two-dimensions can be easily ported to three-dimensions since the two models have the same form. The scheme presented by Justh and Krishnaprasad is intuitive and produces the desired flocking behavior, but its synthesis appears to be guided only by intuition, and the justification behind the choice of Lyapunov function is not clear. In this thesis, a variation of this scheme will be discussed with special attention to the control design process and stability analysis.

Chapter 3
Modelling of Dynamic Agents

Agent Attitude Coordinates

In this thesis, a dynamic agent is considered to be a fixed-wing UAV with control inputs of forward thrust and pitch, yaw, and roll rates. First, the choice of attitude model will be discussed and then in a later section the forward thrust model will be added. The attitude of an aircraft is modelled by a set of nonlinear kinematic differential equations. These equations are a fundamental construct in rigid body kinematics which involves the construction of a direction cosine matrix (DCM) that embodies the transformation from an earth-fixed coordinate system to a body-fixed coordinate system located on the aircraft. There are several ways to construct the DCM but in this thesis the fundamental nine (9) parameter representation is considered. Other representations use fewer parameters, but by using all nine (9), the attitude model yields a convenient form for control design of swarm agents that will be discussed in the next chapter.

The DCM is a 3x3 matrix that describes the relationship between two sets of orthonormal base vectors; each row represents the three (3) projections of one of the resultant base vectors onto the original base vectors. The differential equations that describe the evolution of the DCM are given as:

$$[\dot{C}] = -[\tilde{\omega}][C] \tag{3.1}$$

$$[\tilde{\omega}] = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix}, \qquad [C] = \begin{bmatrix} \cos\alpha_{11} & \cos\alpha_{12} & \cos\alpha_{13} \\ \cos\alpha_{21} & \cos\alpha_{22} & \cos\alpha_{23} \\ \cos\alpha_{31} & \cos\alpha_{32} & \cos\alpha_{33} \end{bmatrix}$$

where $[C]$ is the DCM expressed in polar notation and $[\widetilde{\omega}]$ is the skew-symmetric angular velocity matrix (refer to a book on analytical mechanics such as [69] for detailed derivation). The DCM can also be expressed in Cartesian notation as:

$$[C] = \begin{bmatrix} \hat{T}^T \\ \hat{N}^T \\ \hat{B}^T \end{bmatrix} = \begin{bmatrix} \mathrm{T_x} & T_y & T_z \\ N_x & N_y & N_z \\ B_x & B_y & B_z \end{bmatrix}$$

where $\hat{T}$ is the tangent vector, $\hat{N}$ is the normal vector, and $\hat{B}$ is the binormal vector. The set $(\hat{T}, \hat{N}, \hat{B})$ forms an orthonormal set of base vectors that describe the body-fixed coordinate system of the rigid body of interest (in this case, a dynamic agent). In this work, the Cartesian notation will be used.

The fundamental DCM model has several advantages and some disadvantages. Since the polar and Cartesian representations of the DCM completely describe the rigid body's base vectors, it does not exhibit any geometric singularities that are present in other representations of the DCM; this advantage comes at the price of a highly redundant set of equations. The minimum number of parameters needed to describe the attitude of a rigid body is three (3), but the polar and Cartesian DCM has nine (9); that means there are six (6) redundant equations that need to be solved [69]. Besides eliminating the geometric singularities, the benefit of this formulation is that it contains only quadratic nonlinearity at a very low level.

Another disadvantage is that numerical solutions to the fundamental DCM model are susceptible to computation errors which may cause the base vectors to eventually fail to be orthogonal [70]. This is primarily due to the fact that every entry in the DCM has a value between [-1, 1], which means that the acceptable error tolerance is very low. There are several ways to compensate for numerical errors seen in the fundamental DCM model [70]. For example, to maintain orthogonality of the base vectors, they can be normalized after some time steps to ensure that they are unit vectors going into the next time step.

Other formulations for the DCM such as Euler angles and quaternions have fewer parameters than the fundamental formulation and therefore yield fewer differential

equations to solve. The Euler angle formulation uses three (3) parameters to describe attitude, namely the set (ϕ, θ, ψ), which are the roll, pitch, and yaw angles, respectively. The DCM is constructed by multiplying three (3) independent rotation matrices in a particular order. The drawback of this formulation is that it has trigonometric nonlinearity and geometric singularities.

The quaternion formulation uses four (4) parameters and is a popular choice for spacecraft attitude control since it does not exhibit any geometric singularities. The quaternion parameters, also called Euler parameters, describe the instantaneous rotation angle and axis. The kinematic differential equations governing the quaternion attitude description exhibit quadratic nonlinearity and no geometric singularities just like the fundamental DCM model but has only four (4) equations instead of nine (9). While quaternions are an attractive choice for rigid body attitude coordinates, designing a controller for swarm control applications using quaternions is not very intuitive. The fundamental DCM model can be written in a compact vector form that facilitates the design of simple control laws for swarming which will be shown in the next chapter.

It was stated earlier that the DCM represents the set of orthonormal base vectors of the body-fixed coordinate system. The angular velocity matrix represents the body-fixed angular velocities which in aircraft terms are typically written as (p, q, r) being the roll, pitch, and yaw rates, respectively. Now the fundamental DCM model can be rewritten in terms of the body-fixed base vectors and angular velocities in vector-matrix form as:

$$\begin{bmatrix} \dot{\hat{T}} \\ \dot{\hat{N}} \\ \dot{\hat{B}} \end{bmatrix} = \begin{bmatrix} 0 & r & -q \\ -r & 0 & p \\ q & -p & 0 \end{bmatrix} \begin{bmatrix} \hat{T} \\ \hat{N} \\ \hat{B} \end{bmatrix} \quad \textbf{(3.2)}$$

or written out as:

$$\dot{\hat{T}} = r(t)\hat{N} - q(t)\hat{B}$$
$$\dot{\hat{N}} = -r(t)\hat{T} + p(t)\hat{B}$$
$$\dot{\hat{B}} = q(t)\hat{T} - p(t)\hat{N}$$

in which each equation represents a set of three differential equations, one for each component of the base vectors. The transpose on each vector is left out for simplicity. System (3.2) represents the complete attitude description of an aircraft in which all three (3) attitude rates are in effect. In the next section this representation will be related to some concepts from differential calculus and geometry which use simplifications of this model to describe the shape and trajectory of a space curve.

Differential Geometry of Curves

Two simplifications of the fundamental DCM model can be seen in differential calculus, specifically in the geometry of space curves. All functions are parameterized in terms of s, which is the curve arc length. One of these models is the Frenet-Serret frame, which is described as [71]:

$$\begin{aligned} \frac{d\hat{T}}{ds} &= \kappa(s)\hat{N} \\ \frac{d\hat{N}}{ds} &= -\kappa(s)\hat{T} + \tau(s)\hat{B} \\ \frac{d\hat{B}}{ds} &= -\tau(s)\hat{N} \end{aligned} \tag{3.3}$$

where $\kappa(s)$ is called curvature and $\tau(s)$ is called torsion of the curve. The Frenet-Serret model can be seen as the fundamental DCM model where the pitch curvature is zero. The other model is the Fermi-Walker frame, also known as the Bishop frame, and is given by [72]:

$$\begin{aligned} \frac{d\hat{T}}{ds} &= k_1(s)\hat{N} + k_2(s)\hat{B} \\ \frac{d\hat{N}}{ds} &= -k_1(s)\hat{T} \\ \frac{d\hat{B}}{ds} &= -k_2(s)\hat{T} \end{aligned} \tag{3.4}$$

where $k_1(s)$ and $k_2(s)$ are *natural curvatures* of the curve. The Bishop model can be seen as the fundamental DCM model with the roll curvature equal to zero and the direction of the pitch curvature is reversed. In general, the normal and binormal vectors in the Bishop frame are given the names M_1 and M_2, but this notation is not used to maintain consistency with the previous definitions.

The previous sets of differential equations can be parameterized in terms of time by introducing the speed of the frame moving along the curve, which is defined as:

$$u(t) = \frac{ds}{dt} \tag{3.5}$$

By multiplying both sides of the equations (3.3) and (3.4) by equation (3.5), the result is

$$\left(\frac{ds}{dt}\right)\frac{d\hat{T}}{ds} = u(t)\kappa(s)\hat{N}$$
$$\left(\frac{ds}{dt}\right)\frac{d\hat{N}}{ds} = -u(t)\kappa(s)\hat{T} + u(t)\tau(s)\hat{B}$$
$$\left(\frac{ds}{dt}\right)\frac{d\hat{B}}{ds} = -u(t)\tau(s)\hat{N}$$

$$\left(\frac{ds}{dt}\right)\frac{d\hat{T}}{ds} = u(t)k_1(s)\hat{N} + u(t)k_2(s)\hat{B}$$
$$\left(\frac{ds}{dt}\right)\frac{d\hat{N}}{ds} = -u(t)k_1(s)\hat{T}$$
$$\left(\frac{ds}{dt}\right)\frac{d\hat{B}}{ds} = -u(t)k_2(s)\hat{T}$$

On the left-hand side of each equation, ds cancels to yield a derivative in terms of time. Also by noting that arc length is a function of time, it is obvious that the curvatures are also functions of time. In addition, it can be seen through dimensional analysis that the product of speed and curvature yields units of rad/s, which means that the product is an angular rate.

Finally, after comparing with system (3.2), the following relationships can be made:

$$p(t) = u(t)\tau(t)$$
$$q(t) = u(t)k_2(t)$$
$$r(t) = u(t)\kappa(t) = u(t)k_1(t)$$

With all of these facts, the Frenet-Serret frame can be rewritten as:

$$\begin{aligned} \dot{\hat{T}} &= r(t)\hat{N} \\ \dot{\hat{N}} &= -r(t)\hat{T} + p(t)\hat{B} \\ \dot{\hat{B}} &= -p(t)\hat{N} \end{aligned} \tag{3.6}$$

and the Bishop frame can be rewritten as:

$$\begin{aligned} \dot{\hat{T}} &= r(t)\hat{N} + q(t)\hat{B} \\ \dot{\hat{N}} &= -r(t)\hat{T} \\ \dot{\hat{B}} &= -q(t)\hat{T} \end{aligned} \tag{3.7}$$

Agent Motion Dynamics

To complete the model of a dynamic agent, a differential equation of the agent's absolute position must be included to describe how it moves in space. A fixed-wing aircraft uses the thrust force generated by its engines to drive it. This means that the control input would be forward acceleration. Here, an agent is modelled as a point mass with its velocity vector aligned with its body-fixed tangent vector. The resulting model is:

$$\dot{\vec{R}} = u(t)\hat{T}$$
$$\dot{u} = a(\vec{x}, t)$$

where $\vec{R}(t)$ is the absolute position vector of the body, $u(t)$ is the body's speed, $\hat{T}$ is the body's tangent unit vector, $a(\vec{x}, t)$ is forward acceleration, and $\vec{x}(t)$ is the state vector.

Reduction to Two-Dimensions

Up to this point, the model developed has been in a three-dimensional framework. This same model can be reduced to two-dimensional space to yield the well-known unicycle model in vector form by noting that the pitch and roll rates are zero:

$$\begin{bmatrix} \dot{\vec{R}} \\ \dot{\hat{T}} \\ \dot{\hat{N}} \end{bmatrix} = \begin{bmatrix} 0 & u(t) & 0 \\ 0 & 0 & \omega(\vec{x},t) \\ 0 & -\omega(\vec{x},t) & 0 \end{bmatrix} \begin{bmatrix} \vec{R} \\ \hat{T} \\ \hat{N} \end{bmatrix}$$

$$\dot{u} = a(\vec{x},t)$$

where $\omega(\vec{x},t)$ is the steering control and $a(\vec{x},t)$ is the forward acceleration control. The unicycle model in the traditional polar form is given as:

$$\dot{R}_x = u(t)\cos\theta(t)$$
$$\dot{R}_y = u(t)\sin\theta(t)$$
$$\dot{u} = a(\vec{x},t)$$
$$\dot{\theta} = \omega(\vec{x},t)$$

Summary of the Dynamic Agent Model

In this thesis, two models are defined based on the concepts previously discussed. Each model has unique characteristics that provide useful insight into the swarm control of a few types of vehicles. The models will be summarized in the following paragraphs and the reasoning for such a model choice will be provided. Note that the constant N is the number of agents in the system.

Model 1: Two-Dimensional Kinematic

The first model is two-dimensional and uses yaw rate and forward acceleration as control inputs. Attitude is described using the tangent and normal unit vectors as discussed in the previous sections. This model was considered first because it is simple and gives insight into how the control laws described in the next chapter cause the agents to behave.

Initially the agents are assumed to travel at a constant cruise speed, and then the acceleration control is added in to see if agents who are initially far away will eventually achieve a desirable separation distance. A simple model such as this is a great starting place because it involves only one attitude control instead of the two or three that would be required in the three-dimensional case. This model provides a framework to intuitively design a set of control laws to manipulate each agent's velocity vector in order to achieve the desired objectives of swarm coordination. This simple two-dimensional model is given as:

$$\begin{aligned} \dot{\vec{R}}_i &= u_i \hat{T}_i \\ \dot{\hat{T}}_i &= r_i(\vec{x}, t)\hat{N}_i \\ \dot{\hat{N}}_i &= -r_i(\vec{x}, t)\hat{T}_i \\ \dot{u}_i &= a_i(\vec{x}, t) \end{aligned} \tag{3.8}$$

where

$$\vec{x}(t) = \left(\vec{R}, \vec{T}, \vec{N}, \vec{u}\right)^T$$

$$\begin{aligned} \vec{R} &= \left(\vec{R}_1, \vec{R}_2, \dots, \vec{R}_N\right) \\ \vec{T} &= \left(\hat{T}_1, \hat{T}_2, \dots, \hat{T}_N\right) \\ \vec{N} &= \left(\hat{N}_1, \hat{N}_2, \dots, \hat{N}_N\right) \\ \vec{u} &= (u_1, u_2, \dots, u_N) \end{aligned}$$

Model 2: Three-Dimensional Kinematic

The second model extends model 1 to the three-dimensional domain by using the DCM attitude description introduced earlier in the form of the Bishop frame equations written as a functions of time. The control inputs of this model are yaw rate, pitch rate, and forward acceleration. This version of the DCM attitude model is chosen so that only yaw and pitch rates are considered, which yields an intuitive way to simply control a fixed-wing UAV in three-dimensional space. It is assumed that this model will be used in a

trajectory generator level, and actual attitude stabilization and control will take place at a lower control level. By controlling yaw and pitch rates, a controls designer can set saturation limits of the controller in terms of minimum turning radius and maximum flight path angle or angle of attack, respectively. In addition, the control laws developed for the two-dimensional case can be easily adapted to the three-dimensional framework, and the pitch control will have a very similar form to the yaw control. The three-dimensional kinematic model is given as:

$$\begin{aligned}
\dot{\vec{R}}_i &= u_i \hat{T}_i \\
\dot{\hat{T}}_i &= r_i(\vec{x}, t)\hat{N}_i + q_i(\vec{x}, t)\hat{B}_i \\
\dot{\hat{N}}_i &= -r_i(\vec{x}, t)\hat{T}_i \\
\dot{\hat{B}}_i &= -q_i(\vec{x}, t)\hat{T}_i \\
\dot{u}_i &= a_i(\vec{x}, t)
\end{aligned} \tag{3.9}$$

where

$$\vec{x}(t) = \left(\vec{R}, \vec{T}, \vec{N}, \vec{B}, \vec{u}\right)^T$$

$$\begin{aligned}
\vec{R} &= \left(\vec{R}_1, \vec{R}_2, \dots, \vec{R}_N\right) \\
\vec{T} &= \left(\hat{T}_1, \hat{T}_2, \dots, \hat{T}_N\right) \\
\vec{N} &= \left(\hat{N}_1, \hat{N}_2, \dots, \hat{N}_N\right) \\
\vec{B} &= \left(\hat{B}_1, \hat{B}_2, \dots, \hat{B}_N\right) \\
\vec{u} &= (u_1, u_2, \dots, u_N)
\end{aligned}$$

Page Intentionally Left Blank

Chapter 4
Swarm Stability and Control

Preliminaries

Algebraic graph theory can be used to describe the communication topology of a multi-agent system. Communication in this sense is any transmission or reception of information about the states of the system. This can be explicit two-way communication between agents in which the information is transferred through a protocol such as WiFi or Bluetooth, or it can be achieved through local sensing in which the states of neighboring agents are measured or estimated through sensors and software manipulation. In both cases, there is a limited range of interaction based on the effective range of the communication device or sensing instruments.

Graph theory allows for this information to be expressed in a mathematical form that can aid in stability analysis, as well as, simplify the computational effort in a control law by minimizing the number of agents considered. Graph theory also allows designers to define an explicit communication topology to yield configurations such as leader-follower or can allow information about the entire system or the environment to be shared among agents. In other words, some agents will act as relays so information can be diffused through the group.

The primary matrices in graph theory are the degree matrix, adjacency matrix, and the Laplacian matrix. Every agent represents a vertex of the graph and a communication link between agents is represented by edges. The matrices describe the sets of edges connected to each agent. The degree matrix (D) is a diagonal NxN matrix where each entry is the number of agents that share an edge with a particular agent (i.e. the number of neighbors the agent has). The adjacency matrix (A) is a symmetric NxN matrix where an

element $a_{ij} = 1$ means agent i and agent j share an edge (i.e. the two agents are neighbors). The Laplacian matrix (L) is defined as the difference between the degree matrix and the adjacency matrix, $L = D - A$. This matrix indicates whether or not the graph is connected (i.e. each agent has at least one edge); if the graph is connected, then the smallest eigenvalue is zero. The Laplacian matrix is used extensively in swarm control to solve consensus problems because it mathematically represents the neighbors of each agent. By using this matrix in a swarm control law, the computational effort of all agents is reduced since they only need to consider a small set of other agents instead of all of them. This is especially useful when solving swarm problems with a large number of agents. For a more detailed discussion of algebraic graph theory, refer to [73].

Swarm Control Design

The design of a control scheme for swarm coordination is a daunting task as the objectives and requirements of such a design are often numerous and conflicting. It is common to employ a decentralized scheme when designing a swarm controller [15]. In a decentralized or distributed scheme, every agent is designed to compute its own trajectory using only locally sensed and communicated information. This is an effective method because it distributes the computational effort among all agents and increases robustness by eliminating reliance on a single communication link to a centralized unit.

However, in a practical swarm system, a hybrid centralized/decentralized scheme will be necessary so that the group can receive information necessary for mission success, such as, changes in the mission objective, or the presence of new intelligence about the environment. In addition, a centralized link is needed to provide the operator supervisory control over the swarm. In this thesis, the control scheme is primarily decentralized, but there are some agents, called *informed agents*, who know about the group's objectives; in this case, the objectives are waypoints that compose the swarm's flight path.

The control design for a decentralized system is still challenging because each agent must consider a set of complex and often conflicting objectives. An agent must keep track of the positions, headings, and speeds of its neighboring agents and use this information to

generate control inputs that act to satisfy Reynolds' rules for flocking. In addition, each agent must be able to sense various kinds of targets in the environment and generate control inputs that drive it towards the targets. The generated commands must also satisfy the performance limitations of the aircraft such as thrust capability, maximum yaw and pitch rates, and maximum speed. Synthesizing a control law that satisfies all of these conflicting requirements at the same time is the goal of swarm coordination. This can be done by either implementing intelligent switching logic that chooses the best function to satisfy the most critical requirements at a given time, or by designing a well-balanced control law, which is a superposition of functions that achieve the individual requirements. In both scenarios, a set of functions must be chosen and designed to efficiently satisfy the requirements and respect the aircraft's performance limitations. Here, the latter method is chosen, but it is recognized that a practical system will require some intelligent logic to be successful in various types of missions and environments.

Lyapunov Function Design

A controller for swarm coordination is often designed by considering a Lyapunov function that represents the energy state of the system in terms of relative coordinates. For example, instead of considering the kinetic energy of each individual agent, a Lyapunov function for a swarm of agents will include a relative kinetic energy term that is a positive definite function of the relative speeds between agents. Such a function would look like:

$$K(\vec{u}) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j<i}\left(u_i - u_j\right)^2 \qquad \textbf{(4.1)}$$

or in vector-matrix form:

$$K(\vec{u}) = \frac{1}{2}\vec{u}^T L_c \vec{u} \qquad \textbf{(4.2)}$$

where $L_c = NI_{N\times N} - 1_{N\times N}$ is the complete graph Laplacian.

Similarly, the potential energy of the swarm will contain functions of the relative positions and attitude of all agents as well as relative positions with respect to targets in the environment. These functions are broken into two groups to reinforce the fact that swarm control must achieve not only internal objectives related to emergent behaviors, but also external objectives that include some interaction with the environment. The first group represents the swarm's internal potential energy and is given by:

$$U_I(\vec{T}, \vec{R}) = J_H(\vec{T}) + J_F(\vec{R}) \tag{4.3}$$

where $J_H(\cdot)$ is the energy related to differences in heading between agents and $J_F(\cdot)$ is the energy related to relative distances between agents. Equations (4.1) and (4.3) completely describe how well the swarm satisfies Reynolds' rules for flocking. The combination of $K(\cdot)$ and $J_H(\cdot)$ describes the energy required to satisfy Reynolds' first rule (velocity alignment) and $J_F(\cdot)$ describes the energy required to satisfy Reynolds' second and third rules (collision avoidance and cohesion). When these functions reach their global minima, then flocking is achieved according to Reynolds' rules.

The internal kinetic and potential energy functions are positive definite functions with minima at the equilibrium points described by:

$$u_i - u_j = 0 \ \ \forall i, j \neq i \tag{4.4}$$

$$\hat{T}_i \cdot \hat{T}_j = 1 \ \ \forall i, j \neq i \tag{4.5}$$

$$\left\| \vec{R}_{ij} \right\| = r_{sep} \ \ \forall i, j \neq i \tag{4.6}$$

where r_{sep} is the desired separation distance between agents. Equation (4.6) can be relaxed to include a narrow range of distances that will yield a natural flocking behavior.

The following relationships are defined to simplify notation:

$$\vec{R}_{ij} = \vec{R}_i - \vec{R}_j, \qquad \vec{R}_{iT} = \vec{R}_i - \vec{R}_T$$

$$\hat{R}_{ij} = \frac{\vec{R}_{ij}}{\|\vec{R}_{ij}\|}, \qquad \hat{R}_{iT} = \frac{\vec{R}_{iT}}{\|\vec{R}_{iT}\|}$$

where $\vec{R}_T$ is the location of the target or waypoint and $\vec{R}_{ij}$ and $\vec{R}_{iT}$ are the relative distance vectors between agent i and agent j and the target, respectively. The unit vectors will be used later in this chapter.

The first term in equation (4.3) can be represented by a function such as:

$$J_H(\vec{T}) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j<i}(\hat{T}_i - \hat{T}_j)\cdot(\hat{T}_i - \hat{T}_j) \tag{4.7}$$

or in vector-matrix form:

$$J_H(\vec{T}) = \frac{1}{2}\vec{T}^T(L_c \otimes I_{3\times 3})\vec{T} \tag{4.8}$$

By noting that $\hat{T}_i$ is a unit vector and applying the distributive property of dot products, equation (4.7) can be simplified to yield:

$$J_H(\vec{T}) = \frac{N(N-1)}{2} - \sum_{i=1}^{N}\sum_{j<i}\hat{T}_i \cdot \hat{T}_j \tag{4.9}$$

which can easily be shown to have a global minimum at zero when equation (4.5) is satisfied.

The second term in equation (4.3) is given the form:

$$J_F(\vec{R}) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i} C_0 + J_{ij}(\|\vec{R}_{ij}\|) \tag{4.10}$$

where $\mathcal{N}_i$ is the set of agents within the sensing range of agent i, $J_{ij}(\cdot)$ is an artificial potential function that defines the interaction energy between agents, and C_0 is a constant

that is chosen for the specific choice of J_{ij}, so that their sum has a global minimum at zero. The design of artificial potential functions will be discussed later in this chapter.

The second group of potential energies is the swarm's external potential energy that describes the swarm's interactions with the environment. The design of this energy function dictates the swarm's mission objectives and requirements. This function can include potential energy related to target tracking, rendezvous tasks, obstacle avoidance, etc. In general, it contains attractive potentials that drive informed agents toward the mission objectives and repulsive potentials that drive them away from obstacles and other objects in the environment that attempt to minimize mission success. In this thesis, only an attractive potential related to a fixed target or waypoint is considered. The total external potential energy including all informed agents is given by:

$$U_E(\vec{R}, \vec{R}_T) = J_T(\vec{R}, \vec{R}_T) = \sum_{i \in \mathcal{I}} J_{iT}(\|\vec{R}_{iT}\|) \qquad \textbf{(4.11)}$$

where $\mathcal{I}$ is the set of informed agents and $J_{iT}(\cdot)$ is the artificial potential function describing the interaction energy between an informed agent and the mission target. The inclusion of this term is crucial to successfully produce flocking behavior [41]. Without it, the system will only achieve flocking behavior for a small set of initial conditions. By including a term that provides navigational guidance to some informed agents, the follower agents will tend to adopt the same guidance through equation (4.10). This means that a larger set of initial conditions can be supported.

The complete Lyapunov function is the sum of equations (4.1), (4.3), and (4.11):

$$V(\vec{x}) = K(\vec{u}) + J_H(\vec{T}) + J_F(\vec{R}) + J_T(\vec{R}, \vec{R}_T) \qquad \textbf{(4.12)}$$

Now that the Lyapunov function has been defined, a stabilizing swarm controller can be devised which acts to minimize the system's energy state. The form of the controller is simply a superposition of the negative gradients of each energy function in equation (4.12). Each term represents a particular goal of the system. The set of controls needed to achieve each goal varies. The gradient of equation (4.1) is added to the acceleration

control input and the gradient of equation (4.7) is added to the yaw and pitch control input because they deal with consensus of those particular motions. Equations (4.10) and (4.11) are added to all three control inputs because all are required to change an agent's position. Gradients with respect of $\vec{R}$ and $\vec{T}$ are broken into tangent, normal, and binormal components for the acceleration, yaw, and pitch controls, respectively. The complete control laws are given by:

$$a_i = -k_{c1}\frac{\partial K}{\partial u_i} - k_{p1}\nabla_{\vec{R}_i}J_F \cdot \hat{T}_i - k_{o1}\nabla_{\vec{R}_i}J_T \cdot \hat{T}_i - k_d(u_i - u_{crs}) \quad \textbf{(4.13)}$$

$$r_i = -k_{c2}\nabla_{\hat{T}_i}J_H \cdot \hat{N}_i - k_{p2}\nabla_{\vec{R}_i}J_F \cdot \hat{N}_i - k_{o2}\nabla_{\vec{R}_i}J_T \cdot \hat{N}_i \quad \textbf{(4.14)}$$

$$q_i = -k_{c3}\nabla_{\hat{T}_i}J_H \cdot \hat{B}_i - k_{p3}\nabla_{\vec{R}_i}J_F \cdot \hat{B}_i - k_{o3}\nabla_{\vec{R}_i}J_T \cdot \hat{B}_i \quad \textbf{(4.15)}$$

where the sets of k_c, k_p, k_o, and k_d are control gains. The first terms in equations (4.13) – (4.15) are responsible for distributed agreement, or consensus of a particular state. The second terms are based on artificial potential functions that are responsible for collision avoidance and cohesion of the swarm. The third terms are based on artificial potential functions responsible for target tracking and are only applied to informed agents. An additional term is included in the acceleration control law to make sure all agents settle back to u_{crs}, the desired cruise speed. In the following sections, these terms will be described in detail.

Distributed Agreement

The consensus problem has been studied extensively in the field of swarm control and is strongly based around algebraic graph theory [4]. A consensus control law uses the Laplacian matrix to mathematically represent the relative states between all neighboring agents. This result can be used as a control law by negating the function. This is the same as taking the negative gradient of the relative kinetic energy described in equation (4.1). One difference is that the consensus law uses the Laplacian matrix, so each agent only considers its immediate neighbors. This function in vector-matrix and summation form is given by:

$$\dot{\vec{u}}_C = -k_{c1}\nabla_{\vec{u}}\left(\frac{1}{2}\vec{u}^T L\vec{u}\right) = -k_{c1}L\vec{u} \tag{4.16}$$

$$\dot{u}_C^i = -k_{c1}\sum_{j\in\mathcal{N}_i}(u_i - u_j) \tag{4.17}$$

where L is the NxN Laplacian matrix. It has been proven that this control law exhibits global asymptotic stability as long as the Laplacian matrix stays connected [4]. If an agent loses all communication links, then there is no guarantee that it will rejoin the group unless it is given higher level decision logic. If the graph is connected, then the smallest eigenvalue of the Laplacian matrix is zero; meaning that all of the other eigenvalues are positive and real. Since the system is linear, applying the negative sign ensures that the system is stable.

The previous control law is perfect for the acceleration control law, but the yaw and pitch control laws require a nonlinear controller since the agent model uses vectors to track attitude. This control law takes the form of a dot product between attitude vectors between each agent. Just like the acceleration consensus law, the yaw and pitch consensus laws end up being the negative gradient of the internal potential energy given by equation (4.9); the negative gradient of this equation with respect to the i^{th} tangent vector is found to be:

$$-\frac{\partial}{\partial \hat{T}_i}\left(\frac{N(N-1)}{2}-\sum_{i=1}^{N}\sum_{j<i}\hat{T}_i\cdot\hat{T}_j\right)=\sum_{j\neq i}\hat{T}_j$$

Because the gradient is a vector, it must be multiplied by the normal and binormal vectors using the dot product in order to get a scalar function. In addition, the consensus laws are written in terms of only neighboring agents instead of all agents. The resulting yaw and pitch consensus laws are given as:

$$r_C^i = k_{c2}\sum_{j\in\mathcal{N}_i}\hat{T}_j\cdot\hat{N}_i \tag{4.18}$$

$$q_C^i = k_{c3}\sum_{j\in\mathcal{N}_i}\hat{T}_j\cdot\hat{B}_i \tag{4.19}$$

This form of consensus control is often referred to as coupled oscillators [32]. In this form, it is not obvious why such a name is given, but if the alternate dot product form is used, then it becomes a summation of sinusoidal functions and is given by:

$$r_C^i = -k_{c2}\sum_{j\in\mathcal{N}_i}\sin(\theta_i-\theta_j) \tag{4.20}$$

where θ represents the attitude angle related to yaw motion. If equation (4.20) is linearized, then it is obvious that the consensus law will take the same linear form as equation (4.17). This means that, near the equilibrium point, the system has the same stable behavior. It will be shown later that the range of initial conditions is limited due to the nonlinearity, but the range tends to grow with the neighborhood size of each agent.

Artificial Potential Functions

The typical form for an APF used to model agent interactions is given by:

$$J_{ij}(\|\vec{R}_{ij}\|) = J_a(\|\vec{R}_{ij}\|) - J_r(\|\vec{R}_{ij}\|) \qquad \textbf{(4.21)}$$

where $J_a(\cdot)$ is the attractive potential function and $J_r(\cdot)$ is the repulsive potential function. Equation (4.21) is designed to have a global minimum when the desired separation distance is achieved (i.e. equation (4.6) is satisfied), a positive increasing derivative when $\|\vec{R}_{ij}\| < r_{sep}$, and a negative decreasing derivative when $\|\vec{R}_{ij}\| > r_{sep}$. An example of such functions can be seen in Figure 1.

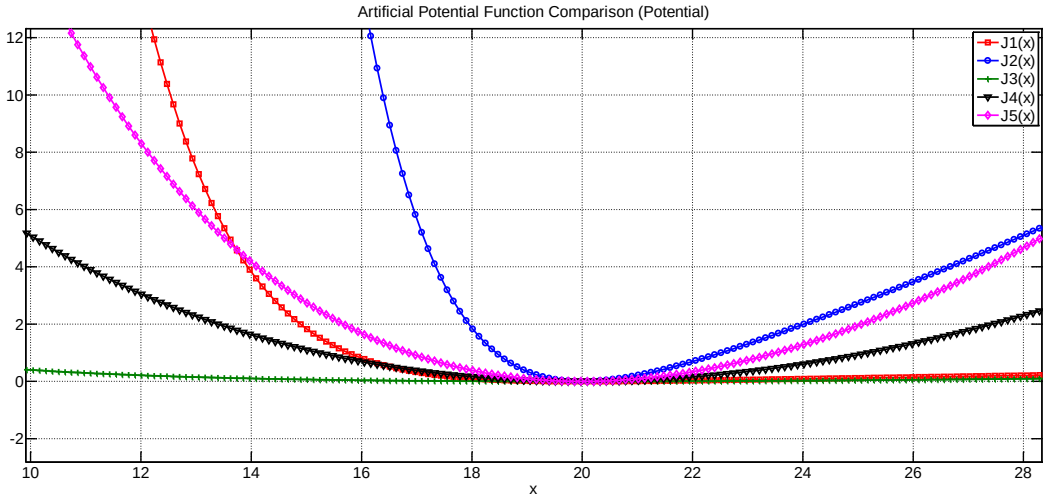

Figure 1 Examples of five artificial potential functions

It can easily be shown that the gradient of a potential function of the form in equation (4.21) is given by:

$$\nabla_{\vec{R}_i} J_{ij}(\|\vec{R}_{ij}\|) = g(\|\vec{R}_{ij}\|)\hat{R}_{ij} = [g_a(\|\vec{R}_{ij}\|) - g_r(\|\vec{R}_{ij}\|)]\hat{R}_{ij} \qquad \textbf{(4.22)}$$

where $g_a(\cdot)$ and $g_r(\cdot)$ are the derivatives of the attraction and repulsion potentials with respect to distance between agents. Examples of these functions can be seen in Figure 2.

An APF can be designed in the following two ways: the derivatives $g_a(\cdot)$ and $g_r(\cdot)$ can be designed and then integrated to obtain the potential functions $J_a(\cdot)$ and $J_r(\cdot)$ or the potential functions can be designed and then differentiated. In this work, the former method is chosen because it is more intuitive to design a function that can be used for control. Designing the derivatives allows for direct control of the function shape around the equilibrium point. First, the types of attraction and repulsion functions are chosen, and then a set of shaping constants is chosen. The coefficient of the attraction function is chosen first, and then the coefficient of the repulsion function is found by solving $g(x_d) = 0$ where x_d is the desired separation distance.

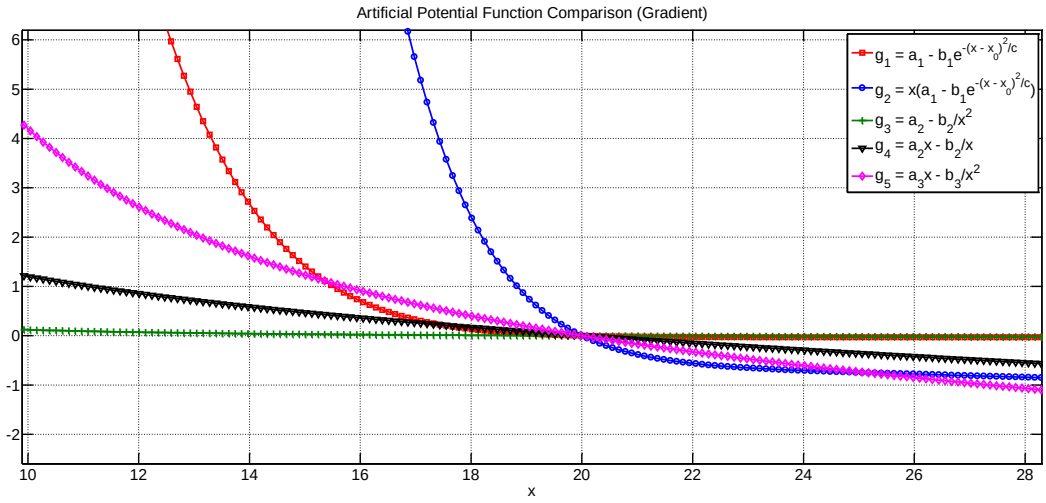

Figure 2 Derivatives of five artificial potential functions as used in control. All functions have been multiplied by a constant, k = -0.01. Design constants are: $a_1 = 3$, $b_1 = 9883$, $c = 40$, $x_0 = 2$, $a_2 = 4$, $b_2 = 1600$, $a_3 = 6$, $b_3 = 48{,}000$, and $x_d = 20$.

The functions in Figure 2 have two types of attraction functions. The first is defined as $g_a = a$ and yields a constant attraction potential. The second is defined as $g_a = ax$ and yields a linearly increasing attraction potential. Higher ordered terms can be used to help ensure cohesion of the swarm even if some agents drift away from the group. The repulsion functions have three forms in Figure 2, but there are two main kinds of repulsion functions: bounded and unbounded. A function of the form $g_r = be^{-x^2}$ is bounded, while a function of the form $g_r = b/x^n$ where n is any positive integer is unbounded. For a more detailed discussion of potential functions used for stable swarm

control, refer to [5]. In this work, the following function is chosen because it provides a faster response near the equilibrium point. Using the definition $\|\vec{R}_{ij}\| \equiv r_{ij}$, the result is:

$$g(r_{ij}) = ar_{ij} - br_{ij}e^{-(r_{ij}-r_0)^2/c} \quad \textbf{(4.23)}$$

where r_0 is the wingspan of the UAV. The agent-to-target potential is defined similar to the agent-to-agent potential except only an attraction potential is used. The gradient of the agent-to target potential is given by:

$$\nabla_{\vec{R}_i} J_{iT}(\|\vec{R}_{iT}\|) = h(\|\vec{R}_{iT}\|)\hat{R}_{iT} \quad \textbf{(4.24)}$$

The gradient function, $h(\cdot)$, is simply defined as a linear attraction function. Using the definition $\|\vec{R}_{iT}\| \equiv r_{iT}$, the result is:

$$h(r_{iT}) = r_{iT} \quad \textbf{(4.25)}$$

Stability Analysis

In order to prove local stability of the proposed swarm control system, it is sufficient to show that the time derivative of the Lyapunov function is always negative semi-definite in a region around the equilibrium point [62]. Note that the following equations make use of vectrix notation, and it should be understood that when two vectors are multiplied that the operation is a dot product. The time derivative is given by:

$$\dot{V}(\vec{x}) = \frac{\partial V}{\partial \vec{x}}\dot{\vec{x}} = \left(\frac{\partial K}{\partial \vec{u}}\right)^T \dot{\vec{u}} + \left(\frac{\partial J_H}{\partial \vec{T}}\right)^T \dot{\vec{T}} + \left(\frac{\partial J_F}{\partial t}\right) + \left(\frac{\partial J_T}{\partial \vec{R}}\right)^T \dot{\vec{R}}$$

$$\dot{V}(\vec{x}) = \vec{u}^T L_c \dot{\vec{u}} + \vec{T}^T (L_c \otimes I_{3x3})\dot{\vec{T}} + \sum_{i=1}^{N} \sum_{\substack{j \in \mathcal{N}_i \\ j<1}} g(r_{ij}) \left(\hat{R}_{ij} \cdot \dot{\hat{R}}_{ij}\right) + \sum_{i \in \mathcal{I}} h(r_{iT}) \left(\hat{R}_{iT} \cdot \dot{\hat{R}}_i\right) \quad \textbf{(4.26)}$$

Now the time derivative will be looked at term-by-term to see if the criterion for local stability is met. This analysis will only consider the two-dimensional case, but the yaw and pitch controls have a common form, so the extension to three-dimensions is similar. First, the kinetic energy term will be considered. The stack vector of all acceleration controls is given by:

$$\begin{aligned} \dot{\vec{u}} = -k_{c1} L \vec{u} &- k_{p1} col_{\mathcal{G}} \left(\sum_{k \in \mathcal{N}_i} g(r_{ik}) \left(\hat{R}_{ik} \cdot \hat{T}_i\right) \right) \\ &- k_{o1} col_{\mathcal{I}} \left(h(r_{iT}) \left(\hat{R}_{iT} \cdot \hat{T}_i\right) \right) \\ &- k_d \left(\vec{u} - u_{crs}(1_{Nx1})\right) \end{aligned} \quad \textbf{(4.27)}$$

where $col_{\mathcal{G}}(g_i)$ represents an Nx1 column vector of the argument g_i for all i and $col_{\mathcal{I}}(h_i)$ represents an Nx1 column vector where $h_i = h(r_{iT})\left(\hat{R}_{iT} \cdot \hat{T}_i\right)$ for all $i \in \mathcal{I}$ and $h_i = 0$ otherwise.

After substituting equation (4.27) into the first term in equation (4.26), the result is:

$$\dot{V}_K = \vec{u}L_c\left(-k_{c1}L\vec{u} - k_{p1}col_{\mathcal{G}}\left(\sum_{k\in\mathcal{N}_i} g(r_{ik})(\hat{R}_{ik}\cdot\hat{T}_i)\right) - k_{o1}col_{\mathcal{I}}\left(h(r_{iT})(\hat{R}_{iT}\cdot\hat{T}_i)\right) - k_d\vec{u} + k_d u_{crs}(1_{N\text{x}1})\right)$$

$$\dot{V}_K = -k_{c1}\vec{u}L_cL\vec{u} - k_d\vec{u}L_c\vec{u} + k_d u_{crs}\vec{u}L_c(1_{N\text{x}1}) - k_{p1}\vec{u}L_c col_{\mathcal{G}}\left(\sum_{k\in\mathcal{N}_i} g(r_{ik})(\hat{R}_{ik}\cdot\hat{T}_i)\right) - k_{o1}\vec{u}L_c col_{\mathcal{I}}\left(h(r_{iT})(\hat{R}_{iT}\cdot\hat{T}_i)\right)$$

By noting that the sum of any row or column from the Laplacian matrix yields zero, the following simplifications can be made:

$$L_cL = (NI_{N\text{x}N} - 1_{N\text{x}N})L = NL, \qquad L_c(1_{N\text{x}1}) = \vec{0}$$

Now the time derivative can be written as:

$$\dot{V}_K = -k_{c1}N\vec{u}L\vec{u} - k_d\vec{u}L_c\vec{u}$$

$$\dot{V}_P' = -k_{p1}\vec{u}L_c col_{\mathcal{G}}\left(\sum_{k\in\mathcal{N}_i} g(r_{ik})(\hat{R}_{ik}\cdot\hat{T}_i)\right)$$

$$\dot{V}_T' = -k_{o1}\vec{u}L_c col_{\mathcal{I}}\left(h(r_{iT})(\hat{R}_{iT}\cdot\hat{T}_i)\right)$$

The time derivative is broken up because later the potential function terms can be grouped together. The prime notation is used here to identify a subcomponent of the term it is attached to (i.e. $\dot{V}_P = \dot{V}_P' + \dot{V}_P''$). First, the agent-to-agent potential will be considered. By writing this term in summation form and regrouping, the result is:

$$\dot{V}_P' = -k_{p1}\sum_{i=1}^{N}\sum_{k\neq i}(u_i - u_k)\sum_{j\in\mathcal{N}_i} g(r_{ij})(\hat{R}_{ij}\cdot\hat{T}_i)$$

$$\dot{V}_P' = -k_{p1}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i}\left(\sum_{k\neq i}u_i\, g(r_{ij})(\hat{R}_{ij}\cdot\hat{T}_i)\right) - \left(\sum_{k\neq i}u_k\, g(r_{ij})(\hat{R}_{ij}\cdot\hat{T}_i)\right)$$

$$\dot{V}_P' = -k_{p1}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i}\left((N-1)g(r_{ij})(\hat{R}_{ij}\cdot u_i\hat{T}_i)\right) - \left(\sum_{k\neq i}u_k\, g(r_{ij})(\hat{R}_{ij}\cdot\hat{T}_i)\right)$$

Using the fact that $\dot{\hat{R}}_i = u_i\hat{T}_i$ and $\hat{R}_{ij} = -\hat{R}_{ji}$, this equation can be rewritten as:

$$\dot{V}_P' = -k_{p1}\sum_{i=1}^{N}\sum_{\substack{j\in\mathcal{N}_i\\ j<i}}\left((N-1)g(r_{ij})\left(\hat{R}_{ij}\cdot\dot{\hat{R}}_{ij}\right)\right) - \left(\sum_{k\neq i}u_k\, g(r_{ij})\left(\hat{R}_{ij}\cdot(\hat{T}_i-\hat{T}_j)\right)\right)$$

Now it can be seen that $\dot{V}_P'$ and the third term in equation (4.26) have common terms that can be combined. By appending $\dot{V}_P'$ to include this other term the result is:

$$\dot{V}_P' = -\left(k_{p1}(N-1)-1\right)\sum_{i=1}^{N}\sum_{\substack{j\in\mathcal{N}_i\\ j<i}} g(r_{ij})\left(\hat{R}_{ij}\cdot\dot{\hat{R}}_{ij}\right) + k_{p1}\sum_{i=1}^{N}\sum_{\substack{j\in\mathcal{N}_i\\ j<i}} g(r_{ij})\left(\hat{R}_{ij}\cdot(\hat{T}_i-\hat{T}_j)\right)\sum_{k\neq i}u_k$$

Now the agent-to-target potential will be considered. By writing it in summation form and regrouping, the result is:

$$\dot{V}_T' = -k_{o1}\sum_{i\in\mathcal{I}}\sum_{j\neq i}(u_i-u_j)h(r_{iT})(\hat{R}_{iT}\cdot\hat{T}_i)$$

$$\dot{V}_T' = -k_{o1}(N-1)\sum_{i\in\mathcal{I}}h(r_{iT})\left(\hat{R}_{iT}\cdot\dot{\hat{R}}_i\right) + k_{o1}\sum_{i\in\mathcal{I}}\sum_{j\neq i}h(r_{iT})(\hat{R}_{iT}\cdot u_j\hat{T}_i)$$

This equation can be appended with the last term of equation (4.26) to yield:

$$\dot{V}_T' = -(k_{o1}(N-1)-1)\sum_{i\in\mathcal{I}}h(r_{iT})\left(\hat{R}_{iT}\cdot\dot{\hat{R}}_i\right) + k_{o1}\sum_{i\in\mathcal{I}}\sum_{j\neq i}h(r_{iT})(\hat{R}_{iT}\cdot u_j\hat{T}_i)$$

Now the internal potential energy term will be considered. By making use of the properties of the dot product and Laplacian matrix as well as vectrix notation, the time derivative can be written as:

$$\dot{V}_H = \vec{T}^T L_c col_{\mathcal{G}}(r_i \hat{N}_i) = \sum_{i=1}^{N} \sum_{j \neq i} (\hat{T}_i - \hat{T}_j)(r_i \hat{N}_i) = -\sum_{i=1}^{N} \sum_{j \neq i} r_i \hat{T}_j \cdot \hat{N}_i$$

By substituting the yaw control law given as:

$$r_i = \sum_{j \in \mathcal{N}_i} \left(k_{c2} \hat{T}_j \cdot \hat{N}_i - k_{p2} g(r_{ij}) (\hat{R}_{ij} \cdot \hat{N}_i) \right) - k_{o2} h(r_{iT}) (\hat{R}_{iT} \cdot \hat{N}_i) \quad \textbf{(4.28)}$$

the time derivative becomes:

$$\dot{V}_H = -\sum_{i=1}^{N} \sum_{j \neq i} (\hat{T}_j \cdot \hat{N}_i) \left(\sum_{k \in \mathcal{N}_i} k_{c2} \hat{T}_k \cdot \hat{N}_i - k_{p2} g(r_{ik}) (\hat{R}_{ik} \cdot \hat{N}_i) \right) + k_{o2} \sum_{i \in \mathcal{I}} \sum_{j \neq i} (\hat{T}_j \cdot \hat{N}_i) h(r_{iT}) (\hat{R}_{iT} \cdot \hat{N}_i)$$

$$\dot{V}_H = -\sum_{i=1}^{N} \sum_{j \neq i} \sum_{k \in \mathcal{N}_i} k_{c2} \hat{T}_j \cdot \hat{T}_k - k_{p2} g(r_{ik}) (\hat{R}_{ik} \cdot \hat{T}_j) + k_{o2} \sum_{i \in \mathcal{I}} \sum_{j \neq i} h(r_{iT}) (\hat{R}_{iT} \cdot \hat{T}_j)$$

To analyze the triple summation in this equation, it can be broken into two parts: terms where $k = j$ and terms where $k \neq j$. For the first case, the result is:

$$(\dot{V}_H)_{k=j} = -\sum_{i=1}^{N} \sum_{j \neq i} \sum_{\substack{k \in \mathcal{N}_i \\ k=j}} k_{c2} \hat{T}_j \cdot \hat{T}_k - k_{p2} g(r_{ik}) (\hat{R}_{ik} \cdot \hat{T}_j)$$

$$(\dot{V}_H)_{k=j} = -\sum_{i=1}^{N} \sum_{j \in \mathcal{N}_i} k_{c2} \hat{T}_j \cdot \hat{T}_j - k_{p2} g(r_{ij}) (\hat{R}_{ij} \cdot \hat{T}_j)$$

$$(\dot{V}_H)_{k=j} = -\sum_{i=1}^{N} \left(k_{c2} L_{ii} - \sum_{j \in \mathcal{N}_i} k_{p2} g(r_{ij}) (\hat{R}_{ij} \cdot \hat{T}_j) \right)$$

$$\left(\dot{V}_H\right)_{k=j} = -C_{\mathcal{N}} + \sum_{i=1}^{N} \sum_{j \in \mathcal{N}_i} k_{p2} g(r_{ij}) \left(\hat{R}_{ij} \cdot \hat{T}_j\right)$$

where $C_{\mathcal{N}} = k_{c2} tr(L)$ is a constant that depends on the number of neighbors linked to each agent. When the swarm has all-to-all communication, then $C_{\mathcal{N}} = k_{c2} N(N-1)$ which represents the maximum possible value for a given gain. Now for the second case when $k \neq j$, the result is:

$$\left(\dot{V}_H\right)_{k \neq j} = -\sum_{i=1}^{N} \sum_{j \neq i} \sum_{\substack{k \in \mathcal{N}_i \\ k \neq j}} \left(k_{c2} \hat{T}_j \cdot \hat{T}_k - k_{p2} g(r_{ik}) \left(\hat{R}_{ik} \cdot \hat{T}_j\right) \right)$$

By noting that $\hat{R}_{ik} = -\hat{R}_{ki}$ it can be shown that the summation of the second term is zero. Using this fact yields:

$$\left(\dot{V}_H\right)_{k \neq j} = -k_{c2} \sum_{i=1}^{N} \sum_{j \neq i} \sum_{\substack{k \in \mathcal{N}_i \\ k \neq j}} \hat{T}_j \cdot \hat{T}_k$$

This summation can be expanded for a simple case to reveal a pattern where all terms repeat with a multiplicity related to the Laplacian matrix (see Appendix A). Using this knowledge, the summation can be rewritten as:

$$\left(\dot{V}_H\right)_{k \neq j} = -k_{c2} \sum_{i=1}^{N} \sum_{j<i} \left(L_{ii} + L_{jj} + 2L_{ij}\right) \hat{T}_j \cdot \hat{T}_i$$

The coefficient applied to each term is simply the number of edges connected to the pair of nodes on the graph excluding the edge joining them, if it exists. Now $\dot{V}_H$ can be restructured so that the heading, agent-to-agent, and agent-to-target potentials are separated. Using the fact that $\hat{R}_{ij} = -\hat{R}_{ji}$, the result is:

$$\dot{V}_H = -C_{\mathcal{N}} - k_{c2} \sum_{i=1}^{N} \sum_{j<i} \left(L_{ii} + L_{jj} + 2L_{ij}\right) \hat{T}_j \cdot \hat{T}_i$$

$$\dot{V}_P'' = -k_{p2} \sum_{i=1}^{N} \sum_{\substack{j \in \mathcal{N}_i \\ j<i}} g(r_{ij}) \left(\hat{R}_{ij} \cdot (\hat{T}_i - \hat{T}_j) \right)$$

$$\dot{V}_T'' = k_{o2} \sum_{i \in \mathcal{J}} \sum_{j \neq i} h(r_{iT}) (\hat{R}_{iT} \cdot \hat{T}_j)$$

Now all of the time derivative terms previously derived will be regrouped and discussed individually. The Lyapunov time derivative has been restructures such that is it given by:

$$\dot{V}(\vec{x}) = \dot{V}_K + \dot{V}_H + \dot{V}_P + \dot{V}_T$$

where $\dot{V}_K$ represents the terms that involve relative speed, $\dot{V}_H$ represents the terms that involve relative heading, $\dot{V}_P$ represents the terms that involve agent-to-agent potential energy, and $\dot{V}_T$ represents the terms that involve agent-to-target potential energy.

The first term of the Lyapunov derivative is given by:

$$\dot{V}_K = -k_{c1} N \vec{u} L \vec{u} - k_d \vec{u} L_c \vec{u}$$

Since the Laplacian is a symmetric matrix, the first two terms are quadratic functions of relative speeds. The first term will be negative semi-definite as long as the graph is connected, and the second term is always negative semi-definite. Therefore, the consensus and artificial damping components of the acceleration control law will always tend to minimize the energy state of the system described by equation (4.12) as long as the Laplacian of the graph describing the system's communication topology is connected. The only requirement on the control gains is that they are positive.

The second term of the Lyapunov derivative is given by:

$$\dot{V}_H = -C_{\mathcal{N}} - k_{c2} \sum_{i=1}^{N} \sum_{j<i} (L_{ii} + L_{jj} + 2L_{ij}) \hat{T}_j \cdot \hat{T}_i$$

By recognizing that the dot product $\hat{T}_j \cdot \hat{T}_i = \cos(\theta_i - \theta_j)$ and the coefficient $(L_{ii} + L_{jj} + 2L_{ij}) \geq 0$, it can be said that the second term of $\dot{V}_H$ is negative definite in the

region surrounding the equilibrium condition. By itself, this term is negative only when the relative heading angle satisfies the inequality $-pi/2 < \theta_i - \theta_j < pi/2$, but the constant $C_{\mathcal{N}}$ will contribute to making the complete term negative. Therefore, the range of relative heading angles that can be tolerated is actually larger and depends strongly on the level of connectivity that the system possesses according to the graph Laplacian. This is an important result because it says that the system can be locally stable even for some $|\theta_i - \theta_j| > pi/2$. The only requirement on the control gain is that it is positive.

The third term of the Lyapunov derivative is given by:

$$\dot{V}_P = \dot{V}_P' + \dot{V}_P'' = \left(1 - k_{p1}(N-1)\right) \sum_{i=1}^{N} \sum_{\substack{j \in \mathcal{N}_i \\ j<i}} g(r_{ij}) \left(\hat{R}_{ij} \cdot \dot{\hat{R}}_{ij}\right)$$
$$+ \sum_{i=1}^{N} \sum_{\substack{j \in \mathcal{N}_i \\ j<i}} \left(k_{p1} \sum_{k \neq i} u_k - k_{p2}\right) g(r_{ij}) \left(\hat{R}_{ij} \cdot \left(\hat{T}_i - \hat{T}_j\right)\right)$$

First, it should be noted that if velocity consensus is achieved, then both terms of $\dot{V}_P$ are zero, regardless of the relative distance between agents. Next, it should be noted that the relative locations of the agents is irrelevant (i.e. whether agent i is in front of or behind agent j) because $\hat{R}_{ij} \cdot \dot{\hat{R}}_{ij} = \hat{R}_{ji} \cdot \dot{\hat{R}}_{ji}$.

Consider a case when two agents have a relative position of $r_{ij} > r_{sep}$ and agent i is behind agent j; therefore, $g(r_{ij}) > 0$, because it is in the attraction region. This attraction will control agent i to attain a higher speed than agent j and control both agents to steer towards the other's general direction. In this scenario, the angle between $\hat{R}_{ij}$ and $\dot{\hat{R}}_{ij}$ will tend to be greater than $pi/2$ which means that $\hat{R}_{ij} \cdot \dot{\hat{R}}_{ij} < 0$.

Now, consider a case when two agents have a relative position of $r_{ij} < r_{sep}$ and agent i is behind agent j; therefore, $g(r_{ij}) < 0$, because it is in the repulsion region. This repulsion control will cause agent j to attain a higher speed than agent i and cause both agents to

steer away from the other's general direction. In this scenario, the angle between $\hat{R}_{ij}$ and $\dot{\hat{R}}_{ij}$ will tend to be less than $pi/2$, which means that $\hat{R}_{ij} \cdot \dot{\hat{R}}_{ij} > 0$.

Figure 3 shows the vector relationships discussed in the previous cases. The desired separation distance is called r_0 in this figure. Note that these relationships may not exist at every instance in time, but the control law tends to create these conditions and thus tends to guide the agents towards a minimum energy state.

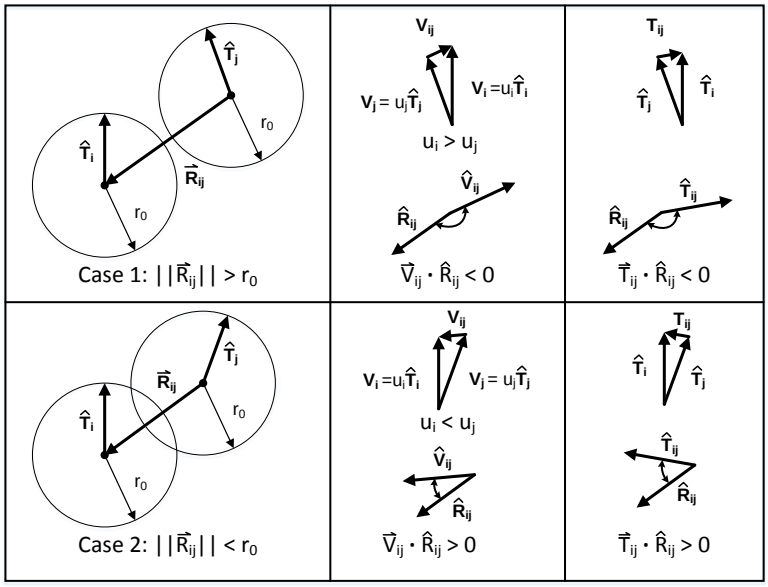

Figure 3 Relationships between relative position and velocity vectors.

Based on these two cases, the first term of $\dot{V}_P$ will tend to be negative semi-definite in the region around equilibrium when the following inequality is satisfied:

$$0 < k_{p1} < \frac{1}{N-1}$$

The second term of $\dot{V}_P$ has a very similar form to the first term and therefore exhibits similar behavior. Since the quantity $k_{p1} \sum_{k \neq i} u_k \gg k_{p2}$, this term tends to be negative semi-definite in the region around equilibrium. Unlike the first term, there is no guidance in the design of the control gain, k_{p2}. In the model, yaw rate is part of the differential equations that govern the change in the unit vectors, which means the overall value of

yaw rate is expected to be fairly small. In addition, it has been recommended by Sepulchre and Paley [32] that the part of the control law that deals with spacing dynamics (i.e. the APFs) should be multiplied by a small constant so that a timescale difference between the spacing dynamics and velocity consensus dynamics is created. Therefore, it is recommended that the yaw control gain for the agent-to-agent potential should satisfy:

$$0 < k_{p2} \ll k_{c2}$$

The fourth term of the Lyapunov derivative is given by:

$$\dot{V}_T = \dot{V}_T' + \dot{V}_T'' = \left(1 - k_{o1}(N-1)\right)\sum_{i\in\mathcal{I}} h(r_{iT})\left(\hat{R}_{iT}\cdot\dot{\hat{R}}_i\right) + k_{o1}\sum_{i\in\mathcal{I}}\sum_{j\neq i} h(r_{iT})\left(\hat{R}_{iT}\cdot u_j\hat{T}_i\right) + k_{o2}\sum_{i\in\mathcal{I}}\sum_{j\neq i} h(r_{iT})\left(\hat{R}_{iT}\cdot\hat{T}_j\right)$$

If it is assumed that agent i is traveling in the general direction of the target, then $h(r_{iT}) > 0$ and $\left(\hat{R}_{iT}\cdot\dot{\hat{R}}_i\right) < 0$. Therefore, for the first term to be negative semi-definite, the control gain must be designed such that it satisfies the inequality:

$$0 < k_{o1} < \frac{1}{N-1}$$

The second two terms of $\dot{V}_T$ are interesting because they involve states of all the other agents in the swarm. The second term of $\dot{V}_T$ is always negative semi-definite, as long as, the conditions discussed in the previous paragraph are satisfied. The third term is only negative semi-definite when all other agents in the swarm are also heading towards the target. Since all agents are controlled to achieve and maintain velocity consensus, all agents will eventually head towards the target as long as the informed agents head towards the target. Similarly to the agent-to-agent potential terms, there is no guidance for the design of the control gain, k_{o2}, except that it must be positive. Since it is applied to the yaw rate control, it should be kept relatively small, but at the same time, it needs to be large enough to overcome the consensus and spacing controls. A good choice of this

control gain is dependent on the choices of the other control gains as well as the APFs $g(\cdot)$ and $h(\cdot)$ that are chosen.

It has been shown that every term of the Lyapunov derivative tends to be negative semi-definite, which means that local stability is guaranteed. Velocity consensus is always achieved as long as the graph is connected. Speed consensus can be achieved for any speed differential. Heading consensus can be achieved for a dynamic set of heading differentials depending on the degree of connectivity according to the graph Laplacian. Heading consensus can be guaranteed if $\left|\theta_i - \theta_j\right| < pi/2\,, \forall i, j \neq i$, but larger heading differences can be tolerated if there is a high degree of connectivity. Separation and cohesion are guaranteed in all regions of the state space in which the graph is connected. It can also be seen that the time derivative terms that deal with agent-to-agent potentials can be zero when the desired separation distance is achieved, when consensus is achieved, or when all terms of the summation cancel. Therefore, the desired separation distance may not be achieved if (1) the agent-to-agent potential cannot overcome the velocity consensus terms, (2) if the agents reach a balanced spatial formation, or (3) a combination of the two. It will be seen in the next chapter that the desired separation distance is typically not achieved because of a complex balancing in the control law terms, as well as, the emergence of balanced spatial formations. Finally, interception of the target is achieved as long as the informed agents can steer and accelerate towards it.

Control Gain Design

In this section, the design methodology for the control gains is discussed. The stability analysis in the previous section provided some insight into the proper design of the gains; however, there are a few additional considerations that must be addressed. For instance, as the number of agents in the swarm increase, so do the control law components that describe agent-to-agent interactions. Both the consensus and agent-to-agent potential function laws depend on the number of agents in the neighborhood of a particular agent; as the size of this neighborhood grows and shrinks, so does the magnitude of the control law. When an agent enters or leaves another agent's neighborhood, both agents will experience an instantaneous change in control input. In this thesis, a gain function is proposed that will help soften this change and attempt to keep the magnitude range of the control law fixed with respect to changes in the swarm size. This function is given by:

$$k_*^i = \frac{k_*}{N-1}\left(1 - \frac{L_{ii}}{N+1}\right) \qquad \textbf{(4.29)}$$

where the subscript * is substituted for one of the control gain subscripts associated with either agent-to-agent potentials or consensus functions (i.e. $p1, c1, p2$, etc.) and L_{ii} is the number of agents in agent i's neighborhood. The first part of equation (4.29) is chosen to scale the gain with the swarm size, and the second part is chosen to scale the gain with the neighborhood size. Since $\max(L_{ii}) = N - 1$, the gain function is always positive; therefore, the stability analysis previously discussed is still valid.

The design of the individual gains is guided by the results of the Lyapunov stability analysis; although, fine-tuning is done to achieve a certain level of performance. As a rule of thumb, the consensus gains are much larger than the other gains for reasons discussed in the previous section. To determine the changes that should be made from one simulation to another, it is necessary to analyze all of the data available from the simulation. The primary data sets to look at are the relative distances, speeds, and headings over time; the acceleration, yaw, and pitch control inputs; the speed of all agents; and the individual functions that compose the control laws for some agents. Looking at the relative states will tell the designer whether or not the consensus terms

need more weighting. Looking at the acceleration, yaw, pitch, and speed histories will tell the designer which control laws need to be scaled down in order to stay within physical limits imposed by the aircraft model. The individual control functions provide very useful insight into the control law design. The functions used in this thesis are the consensus law, agent-to-agent potential, the agent-to-target potential, and the artificial damping. Looking at graphs of these functions together will show which functions tend to agree and which ones tend to fight for dominance. From these graphs, the designer can decide which component needs to have more weight in order to achieve its goal faster or more effectively. The design of these control law gains is not simple, but a methodology can be devised that helps guide the designer down the right path.

Control Saturation and Suppression

In swarm control design where optimization is not applied, control laws are typically created by the superposition of multiple, likely conflicting, functions of the states. This conflict arises due to the fact that each function is designed to drive the agent to a specific goal (i.e. velocity matching, collision avoidance, cohesion, target tracking, rendezvous, etc.). These functions cannot be designed perfectly for every application and will likely give high-valued control inputs if saturation is not applied, especially if potential functions are used. Saturation is often necessary because the property that is being controlled, and sometimes its derivative, is limited to a certain range. For example, in the case of an airplane's control surfaces, the elevator, rudder, and ailerons all have mechanical limits that dictate how much they can be deflected. This saturation will often come naturally, but in design and simulation, the saturation must be taken into account so that the resulting controller can respond appropriately in the field.

In order to account for the performance limitations of an aircraft, it is natural to look at control input saturation because it directly applies limits to the acceleration, yaw, and pitch controls. When considering saturation as applied to swarm control, there are three ways in which it can be applied. The proposed swarm control laws are a superposition of conflicting functions that are intended to eventually balance near an equilibrium state. Based on this form of control law, saturation can be applied *before* the functions are

added, *after* the functions are added, or *as* the functions are added. These concepts will be discussed and depicted graphically with block diagrams in the following paragraphs.

First, the concept of post-saturation will be discussed and is depicted in Figure 4 as it applies to the acceleration control. Post-saturation is what most people think about when they hear the term "saturation". In this method, a saturation limit is applied to the final control value that is computed. This directly simulates the performance limitations of a vehicle by placing a finite bound on the maximum and minimum values of acceleration, yaw rate, and pitch rate.

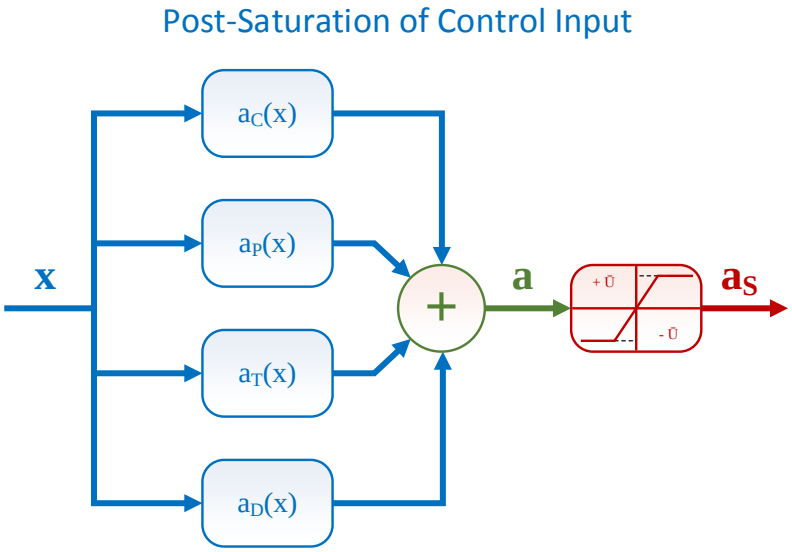

Figure 4 Block diagram of post-saturation scheme for acceleration control

The next concept is pre-saturation and is depicted in Figure 5 as applied to acceleration control. In this scheme the saturation limits are applied before all the functions are added together. A percentage of the overall saturation limit is given to each function such that the sum of percentages equals unity. This means the overall saturation limit is observed, but each function is only allowed to use a certain amount of the available "energy". In the next chapter, simulations will be presented that show that this scheme actually acts as a kind of control "suppression". The result is similar to placing a high weight on control effort in an optimal control scheme. This means that some performance is sacrificed in order to minimize control effort.

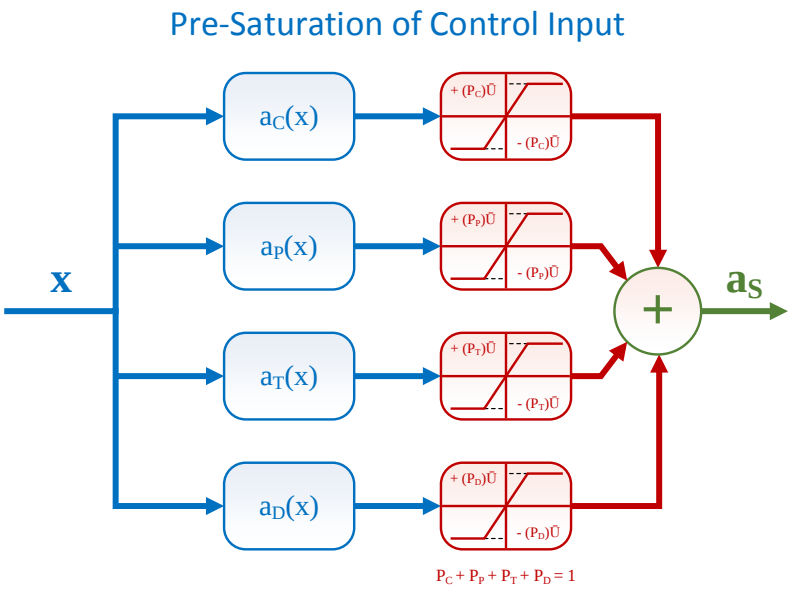

Figure 5 Block diagram of pre-saturation scheme for acceleration control

The percentages can be fixed for all time, or they can be redistributed dynamically. To accomplish this, a priority must be applied to each function, and then saturation is applied to the functions in order of priority. If a high priority function does not use all of its allocated "energy", then this excess can be redistributed to all of the lower priority functions. Then the process can be repeated for all functions. At the next time step, the percentages are reset to the original value, and the process is repeated. Obviously, there are many ways that this can be implemented, especially when a lot of functions are used. The priority assignment is based on the functions used and the system objectives. In this work, the priority order is as follows: consensus, target, APF, and damping. Consensus is given the highest priority because, when it is achieved, collision avoidance among agents and collective motion towards the target are guaranteed.

The final scheme is considered progressive saturation and is depicted in Figure 6 as applied to acceleration control. In this scheme, the functions are added together incrementally, and in-between each sum, the overall saturation limit is applied. This is almost identical to post-saturation except for a case when the total sum of the function values is less than the saturation limit, but an individual function value or an intermediate

sum of function values is greater than the saturation limit. This yields a smaller control law value than would be seen in post-saturation. So this scheme is the same as post-saturation except it has the capability for control suppression under certain scenarios. The degree of suppression is dependent on what order the functions are added together. If two functions that tend to agree are added together first, then a higher level of suppression will be seen when compared to two functions that tend to disagree. In addition, if the first function tends to see high control values then more suppression will be observed. The order shown in Figure 6 is used in simulations because the APF function will tend to yield higher control values, but the consensus function tends to fight the APF function such that a moderate level of suppression is expected.

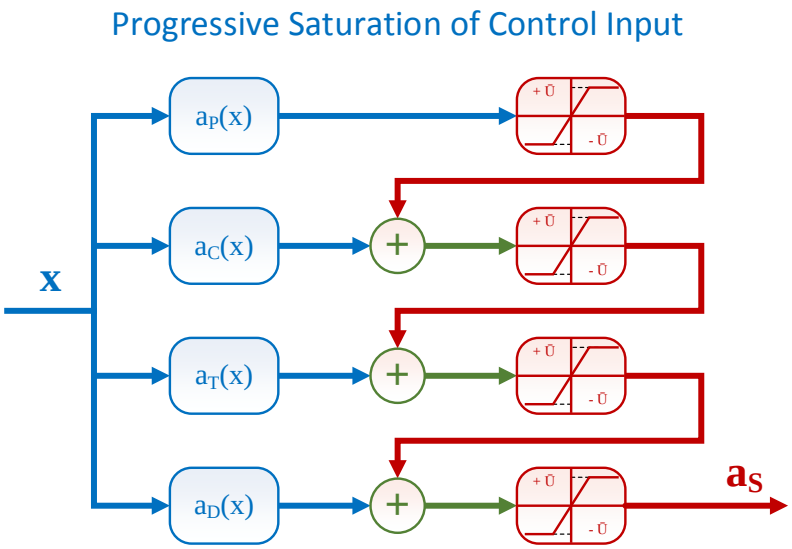

Figure 6 Block diagram of progressive saturation scheme for acceleration control

Page Intentionally Left Blank

Chapter 5
Simulations

Simulation Architecture

The swarm simulation was built using MATLAB and numerically solves the equations of motion using a custom-made function based on the Runge-Kutta 4th order method (RK4). During preliminary testing, the two-dimensional equations of motion for two and three agents were solved using MATLAB's built-in differential equation solvers for both stiff and non-stiff equations and there was no noticeable difference between the results. The code prompts the user for the number of agents, potential function type, saturation type, control law type, and simulation time. A fixed time step of $dt = 1\ ms$ is used.

Agent Description

In this work, agents are considered to be meter-scale fixed-wing UAVs with a large sensing range. Table 1 shows all of the parameters used to describe an agent.

Table 1 Agent parameters

Simulation Parameter	Symbol	Value
Wingspan [m]	r_0	2.0
Sensing Range [m]	r_{sen}	150.0
Desired Separation Distance [m]	r_{sep}	20.0
Cruise Speed [m/s]	u_{crs}	10.0
Max Speed [m/s]	u_{max}	25.0
Max Acceleration [m/s^2]	$\bar{U}_a$	5.0
Max Yaw Rate [rad/s]	$\bar{U}_r$	2.0
Max Pitch Rate [rad/s]	$\bar{U}_q$	2.0

Evaluation Criteria

The simulations are evaluated based on criteria that indicate how well the system achieves flocking behavior. The following criteria must be satisfied:

1.	Collision Avoidance:	$r_{ij}(t) > r_0$	$\forall t > 0, i, j \neq i$
2.	Cohesion:	$\lim_{t\to\infty} r_{ij}(t) < r_{ij}(0)$	$\forall i, j \neq i$
3.	Speed Consensus:	$\lim_{t\to t_w} \lvert u_i(t) - u_j(t) \rvert < \delta_u$	$\forall i, j \neq i$
4.	Heading Consensus:	$\lim_{t\to t_w} \lvert \hat{T}_j(t) \cdot \hat{N}_i(t) \rvert < \delta_h$	$\forall i, j \neq i$
5.	Pitch Consensus:	$\lim_{t\to t_w} \lvert \hat{T}_j(t) \cdot \hat{B}_i(t) \rvert < \delta_q$	$\forall i, j \neq i$
6.	Separation Distance Error:	$\lim_{t\to\infty} \lvert \min r_{ij}(t) - r_{sep} \rvert < \delta_r$	$\forall i, j \neq i$
7.	Maximum Speed:	$u_i(t) < u_{max}$	$\forall t > 0, i$
8.	Maximum Acceleration:	$\lvert a_i(t) \rvert < \overline{U}_a$	$\forall t > 0, i$
9.	Maximum Yaw Rate:	$\lvert r_i(t) \rvert < \overline{U}_r$	$\forall t > 0, i$
10.	Maximum Pitch Rate:	$\lvert q_i(t) \rvert < \overline{U}_q$	$\forall t > 0, i$

where t_w is the time at which a new waypoint is assigned. In other words, criteria 3 through 5 must be satisfied as the system approaches a waypoint. Criterion 6 states that the set of relative positions must satisfy the inequality, $r_{sep} - \delta_r < \min r_{ij}(t) < r_{sep} + \delta_r$, for all agent pairs as time goes to infinity. Table 2 shows the tolerances for criteria 3 through 6. The tolerances are chosen to be 10% of the cruise speed, maximum value of the sine of relative heading angle, and desired separation distance.

Table 2 Tolerances for evaluation criteria

Evaluation Parameter	Symbol	Value
Speed Consensus Tolerance [m/s]	δ_u	1.0
Heading Consensus Tolerance	δ_h	0.1
Pitch Consensus Tolerance	δ_q	0.1
Separation Distance Error [m]	δ_r	2.0

All of these criteria involve many terms so the data will be represented using the minimum, maximum, and average of each quantity at each time step. This information will be considered the *statistics* of the relative states of all agents over time.

Initial Conditions

The choice of initial conditions plays a critical role in the stability of a swarm coordination algorithm. The initial conditions include the number of agents, the locations and orientations of all agents, and the choice of informed agents. One of the major limiting factors in the success of a swarm coordination scheme is the effective sensing range of an agent. As the number of agents grows, the sensing range becomes more significant. The sensing range and initial conditions dictates the initial connectivity of the system, and if this is too low, then the swarm is at greater risk of becoming fragmented. Fragmentation occurs when at least one agent loses all connectivity with the rest of the swarm (i.e. $L_{ii} = 0$). Fragmentation can be avoided by intelligent assignment of informed agents and initial positions. Figure 7 shows an example of swarm fragmentation due to an insufficient sensing range. Two agents become disconnected from the swarm, and two sub-clusters are formed that are weakly connected.

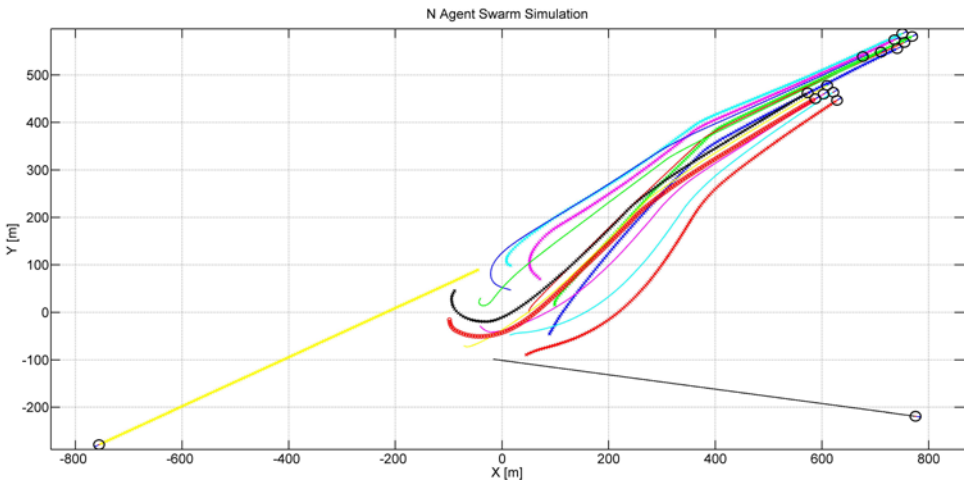

Figure 7 Example of swarm fragmentation

It was shown in the previous chapter that the time derivative of the Lyapunov function is guaranteed to be negative semi-definite when $|\theta_i - \theta_j| < \pi/2\,, \forall i, j \neq i$. With this choice of initial orientations, the risk of fragmentation is very low. Unfortunately, this set is not the most practical for use with fixed-wing UAVs because they typically cannot hover in place, unless it is of the vertical takeoff and landing (VTOL) type. A fixed-wing UAV that does not have VTOL capability is required to maintain a minimum airspeed in flight. In addition, a typical takeoff location will likely have a small limit on the number of aircrafts that can simultaneously takeoff. Therefore, it is likely that some UAVs will have to wait in the air while the full swarm is mobilized.

There are many ways that this problem can be approached and many factors that must be considered. In this thesis, it is assumed that agents will takeoff and enter a circular trajectory centered around or near the takeoff location. All agents will eventually reach an initial configuration in which they are traveling on concentric circles and each circle holds a certain number of agents. Once this configuration is reached, the swarm coordination algorithm is activated. The algorithm that controls the agents to reach this configuration is outside the scope of this thesis.

Mission Objective

In this thesis, a set of waypoints are defined that the informed agents must navigate to. This serves as a simple way to guide the swarm through the environment and to reduce the risk of collisions and fragmentation due to the initial conditions described previously. The simulation algorithm is designed to change waypoints when the informed agent closest to the current waypoint is within 20 meters of the waypoint. The waypoints used in the following two- and three-dimensional simulations are:

$$W = \begin{bmatrix} 400 & 400 \\ 800 & 600 \end{bmatrix} (2D, All\ runs), \qquad W = \begin{bmatrix} 500 & 500 & 1400 \\ 1000 & 750 & 1540 \end{bmatrix} (3D, Run\ 1),$$

$$W = \begin{bmatrix} 500 & 500 & 1400 \\ 1000 & 750 & 1120 \end{bmatrix} (3D, Run\ 2)$$

Phase 1: Two-Dimensional Swarm Simulations

Simulation Setup

The first set of simulations use the two-dimensional agent model described by model (3.8). Table 3 shows all of the simulation parameters used in the simulations described in this section. The first simulation demonstrates the control method described in Chapter 4 as applied to six (6) agents. The agents are considered to be small meter-scale fixed-wing UAVs with a large sensing range. As a result, the first simulation will demonstrate a case in which there is all-to-all communication between agents, or in other words, the graph Laplacian is always the complete Laplacian (i.e. $L = L_c$). The second simulation is a case when the number of agents is increased. The third and fourth simulations show the effect of sensing radius (i.e. connectivity) on the swarm performance.

Table 3 Simulation parameters for two-dimensional simulations

Simulation Parameter	Symbol	Run 1	Run 2	Run 3
Number of Agents	N	6	15	
Acceleration Consensus Gain	$k_{c,1}$	5.0	6.0	
Acceleration APF Gain	$k_{j,1}$	0.100	0.100	
Acceleration Target Gain	$k_{o,1}$	0.008	0.080	
Acceleration Damping Gain	k_d	10.0	5.0	
Yaw Consensus Gain	$k_{c,2}$	6.0	7.0	
Yaw APF Gain	$k_{j,2}$	0.006	0.006	
Yaw Target Gain	$k_{o,2}$	0.003	0.040	
APF Attraction Coefficient	a	2.0	2.0	
APF Repulsion Coefficient	b	1303.9	20,958	
APF Shaping Constant	c	50	35	
Sensing Range [m]	r_{sen}	150	150	80

Simulation 1: Small Swarm with All-to-All Communication

In the first simulation, the swarm consists of six (6) agents, two (2) of which are informed agents. The agents start on two concentric circles with three (3) agents on each, as shown in Figure 8. The circles representing each agent have a diameter equal to the agent's wingspan. Using the sensing range indicated in Table 3 with these initial conditions yields a swarm that has all-to-all communication.

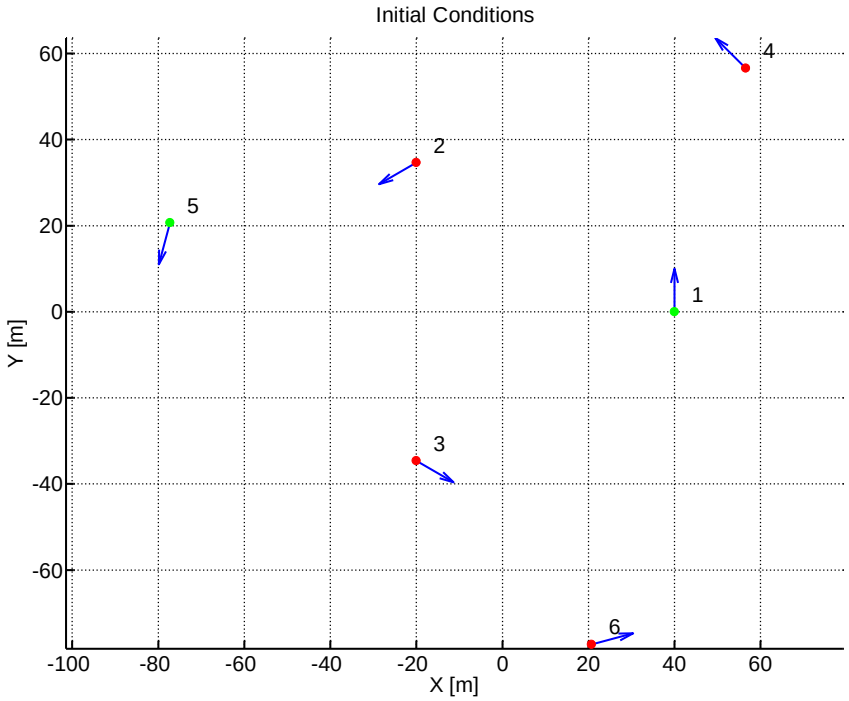

Figure 8 Initial conditions of 6 agent simulation. Informed agents (1 and 5) are designated by green circles and follower agents are designated by red circles.

Figure 9 shows the trajectory of the six (6) agents after 80 seconds. Figure 10 shows the final positions of all six (6) agents. At first glance, it looks like cohesion, collision avoidance, and velocity matching are all achieved in this simulation. There is obviously some error in separation distance by the end, but a closer look at the data is required to determine if the evaluation criteria is satisfied.

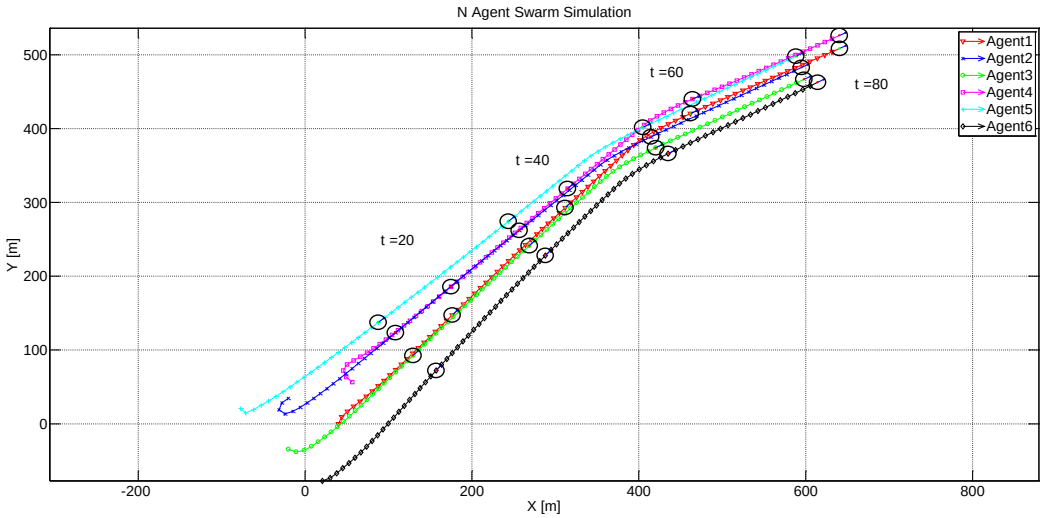

Figure 9 Trajectory of 6 agents over 80 seconds.

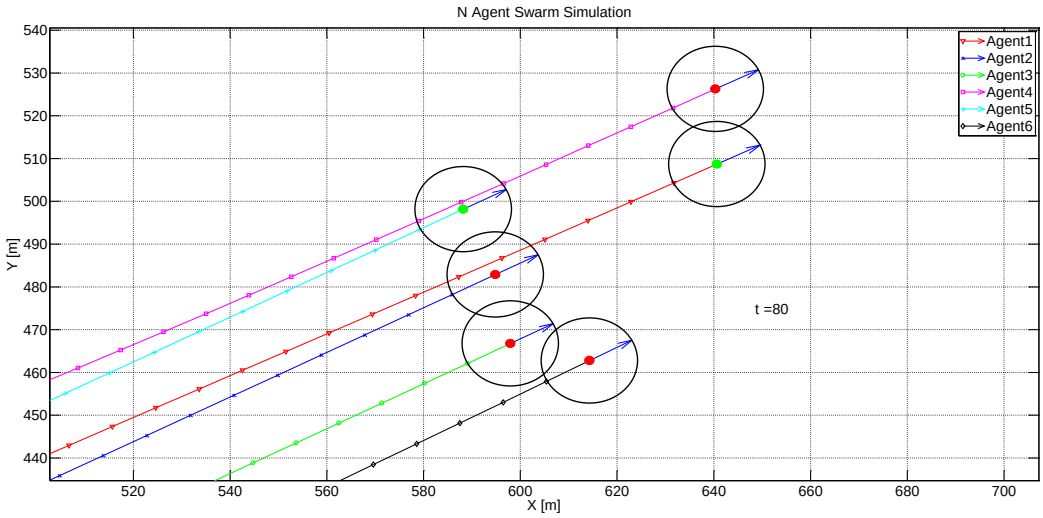

Figure 10 Final positions of 6 agents.

Figure 11 shows the relative position statistics, and Figure 12 shows a detailed view of the minimum relative position curve. From these figures, it is obvious that criterion 1 (collision avoidance) and criterion 2 (cohesion) are satisfied. According to Figure 12, criterion 6 (separation distance error) is not satisfied, but the deviation from the tolerance band is not very high and the minimum curve tends to increase near the end. The reason for this behavior will be discussed later.

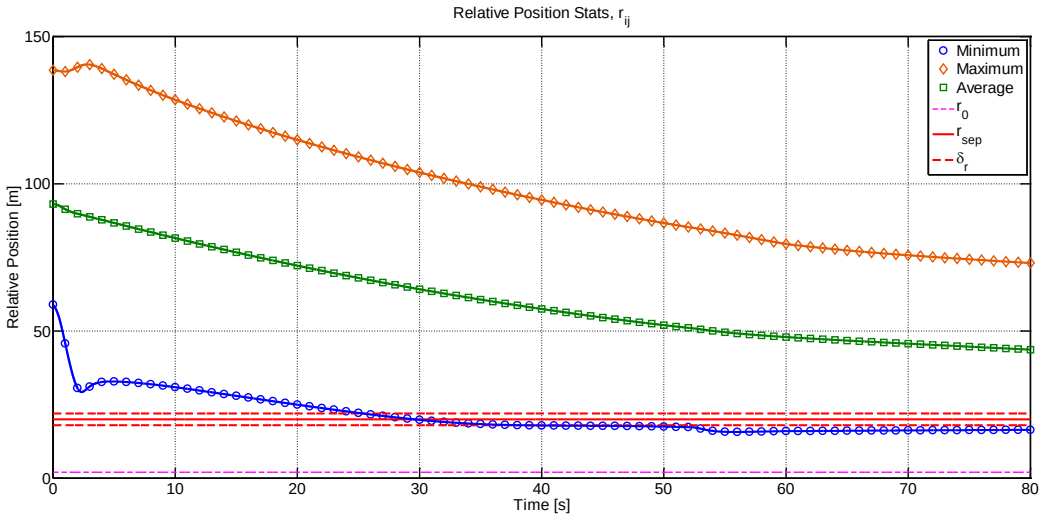

Figure 11 Relative position statistics for 6 agent simulation.

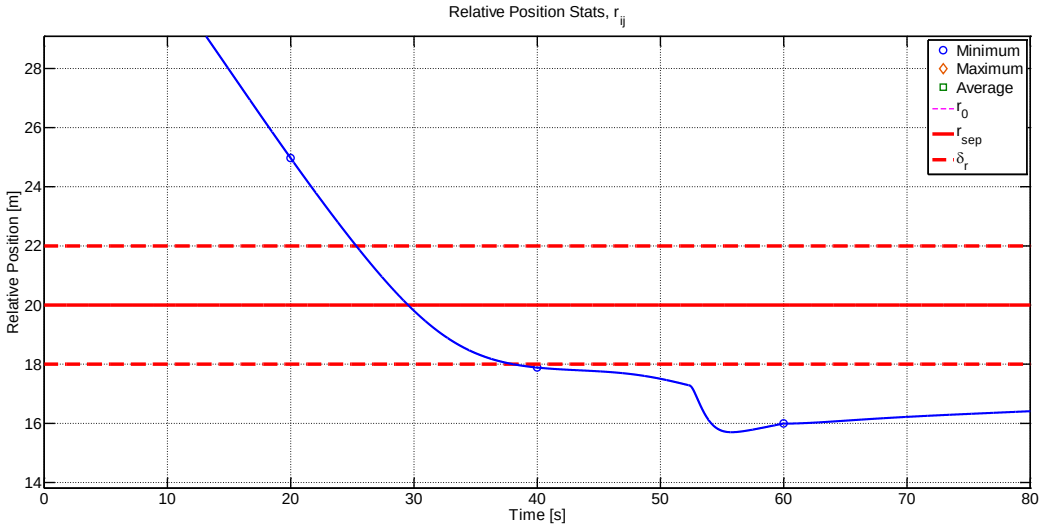

Figure 12 Detailed view of minimum relative position history.

Figure 13 shows the relative speed statistics, and Figure 14 shows the relative heading statistics of the swarm over time. To determine if criterion 3 (speed consensus) and criterion 4 (heading consensus) are satisfied, the maximum and average curves must be observed as the swarm approaches the waypoint, which is crossed at $t \sim 52\ s$. Once both of these curves fall below the tolerance line, it can be said that consensus is achieved. According to both figures, velocity consensus is achieved just before and after the waypoint is crossed.

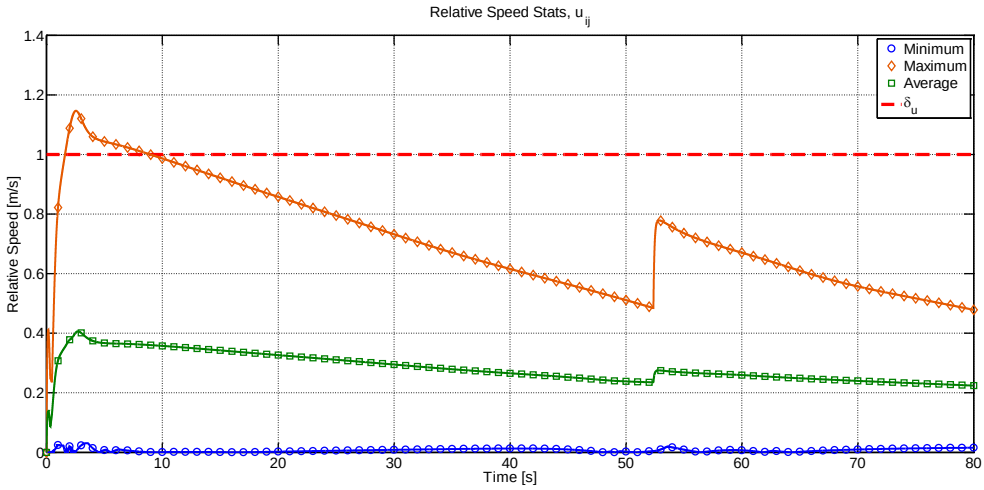

Figure 13 Relative speed statistics for 6 agent simulation.

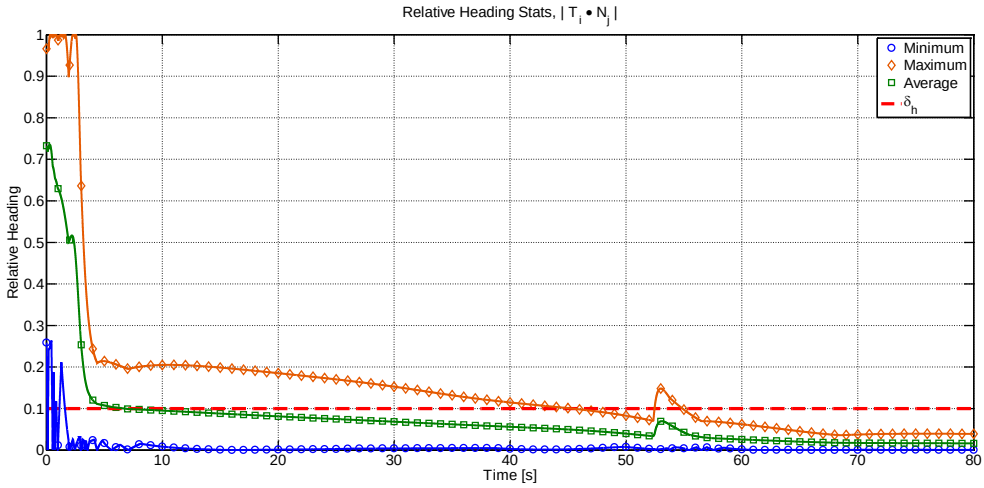

Figure 14 Relative heading statistics for 6 agent simulation.

According to the results thus far, it appears that this simulation yielded a stable swarm over the entire time period, but to verify stability the Lyapunov function should be computed over time and then differentiated. Figure 15 shows the Lyapunov function time history, and Figure 16 shows the derivative of this data. The derivative starts slightly positive, but remains negative for the rest of time. The positive values are likely attributed to the fact that the agents had to overcome the large heading differences in the initial conditions. According to these figures, the swarm was stable over all time.

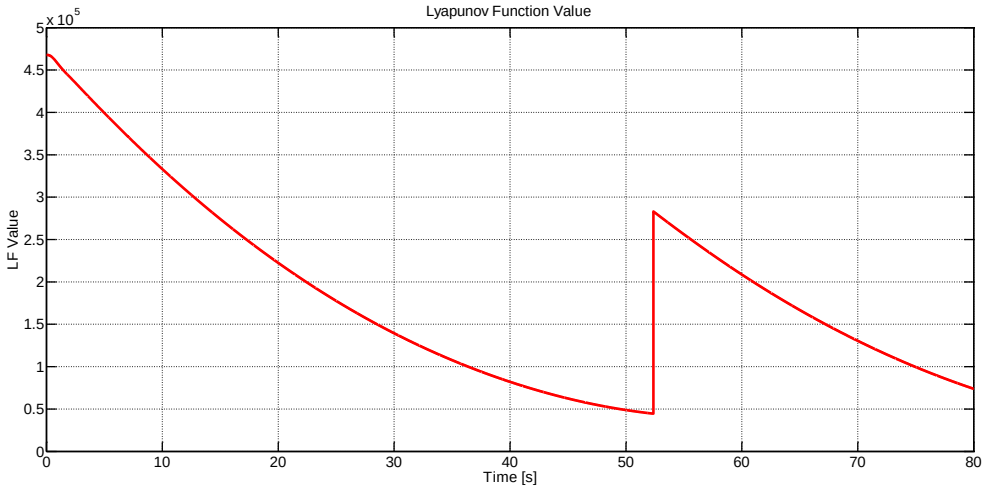

Figure 15 Lyapunov function time history of 6 agent simulation.

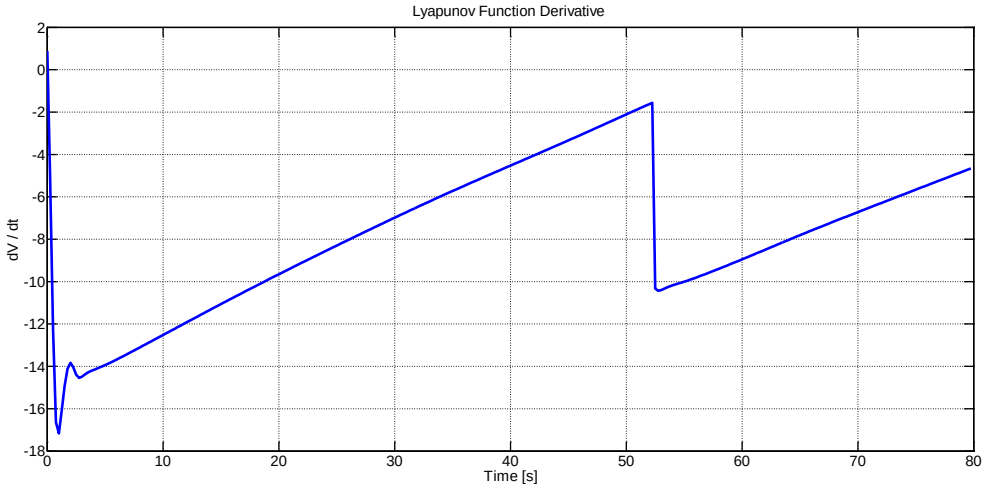

Figure 16 Lyapunov function time derivative. Points are plotted every ¼ second.

Now the remaining criterion will be checked. Figure 17 shows the speed of all six (6) agents over time. From this figure, it can be seen that criterion 7 (maximum speed) is satisfied, and full speed consensus is nearly achieved by the end of the simulation. In addition, all agents stay very close to the desired cruise speed. Figure 18 shows the acceleration control input for all six (6) agents. It can be seen that criterion 8 (maximum acceleration) is satisfied and that all agents slowly accelerate/decelerate back to the cruise speed after the initial response. Figure 19 shows the yaw control input to all six (6) agents. It can be seen that criterion 9 (maximum yaw rate) is not exactly satisfied, but with fine-tuning of the control gains, the deviation can be eliminated.

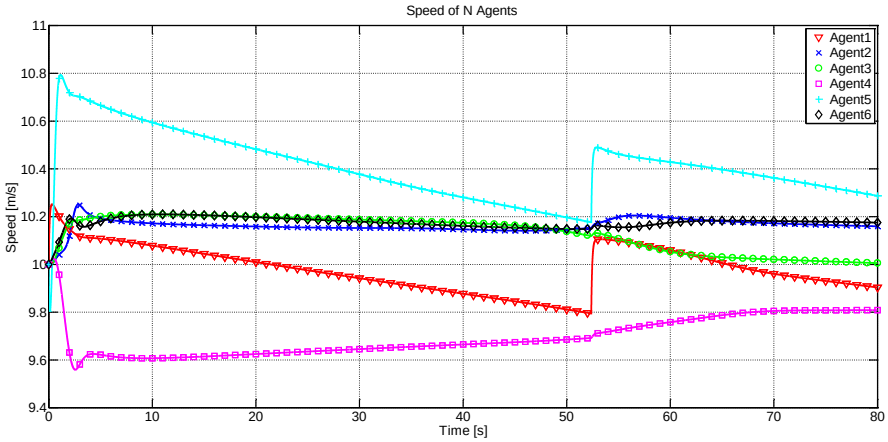

Figure 17 Speed of 6 agents

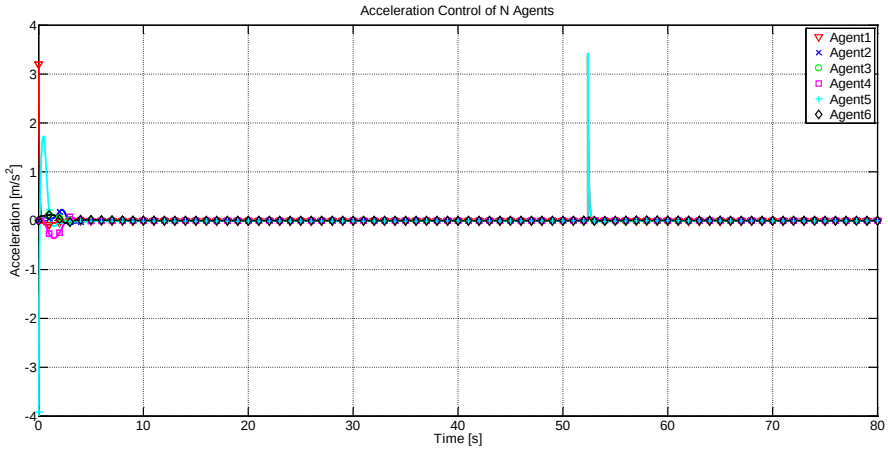

Figure 18 Acceleration control of 6 agents

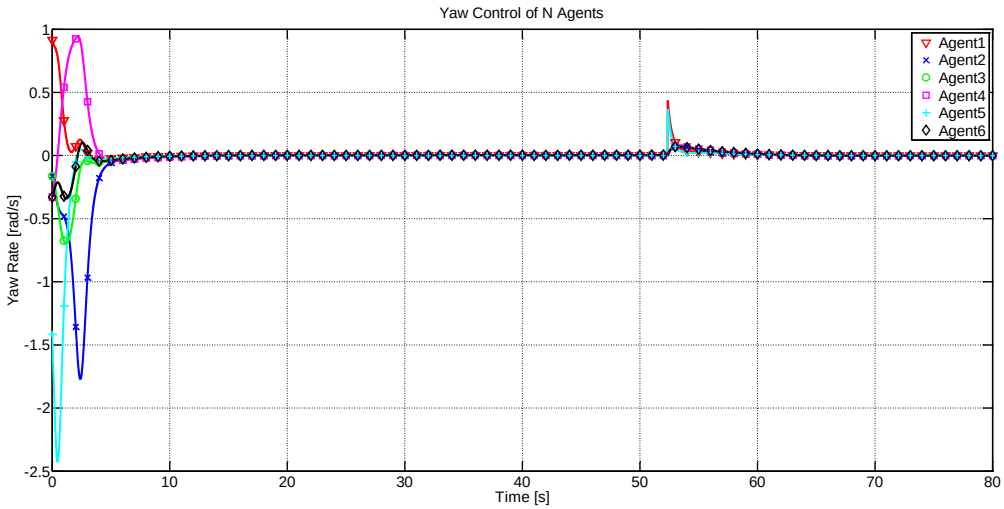

Figure 19 Yaw control for 6 agents

The previous figures both indicate that, at the start of the simulation and after the first waypoint is crossed, the controls exhibit a fast behavior followed by a nearly constant control about zero. This means that the critical swarm objectives are achieved within a very short time period after a change in the mission details. If all of the figures presented in this section are considered together, then it can be seen that heading consensus and target/waypoint alignment are achieved first. Speed consensus, aggregation, and cohesion are achieved over a much longer time period. In other words, the spacing dynamics occur slower than the orientation dynamics, which is exactly the behavior that is desired. The difference between the timescales of these two dynamics can be controlled by modifying the control gains.

In the previous figures and discussion, there were a few behaviors that were noted that deserve some explanation. Specifically, the fact that the cruise speed and separation distance are not achieved by all agents over time. The reason for these behaviors can be seen by looking at the individual components of the control laws.

Figure 20 and Figure 21 show the components of the acceleration control law for agent 1 and 2, respectively. The curve labeled *damping* is responsible for bringing the agent back to its cruise speed. The other components do not reached their equilibrium conditions by the end of the simulation. Therefore, the swarm is still trying to reach its minimal energy state, but its trajectory is slowed down due to the conflicting functions that compose the control law. The swarm may even reach a trajectory in which all the components cancel out and cause the swarm to settle at a local minimum instead of the global minimum described by the equilibrium conditions.

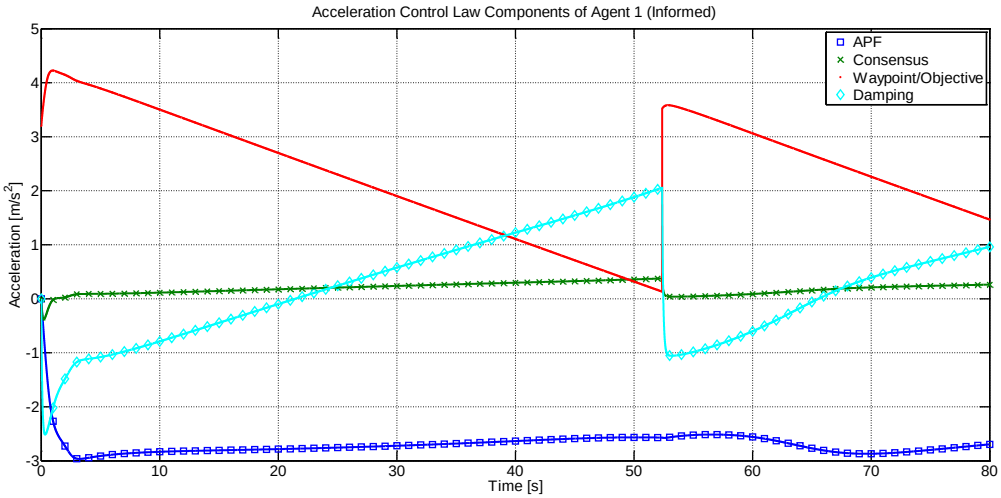

Figure 20 Components of acceleration control law for agent 1.

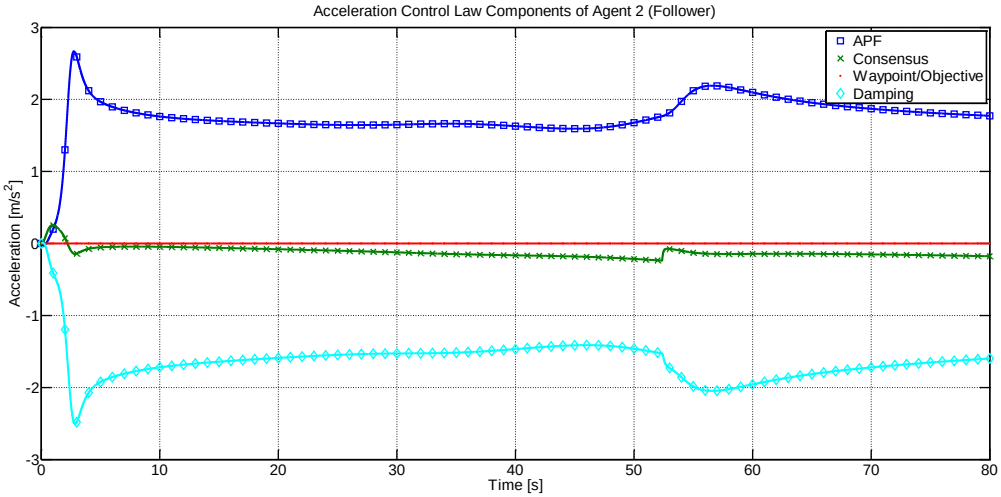

Figure 21 Components of acceleration control law for agent 2.

Figure 22 and Figure 23 show the components of the yaw control law for agent 1 and 2, respectively. From these figures, it can be seen that the local minimum phenomenon described in the previous paragraph is exactly what causes the separation distances to settle outside of the tolerance band. This can be seen clearly in Figure 23 as time approaches the first waypoint crossing and the end of the simulation. This is a common problem in swarm control problems where a fixed separation distance is desired.

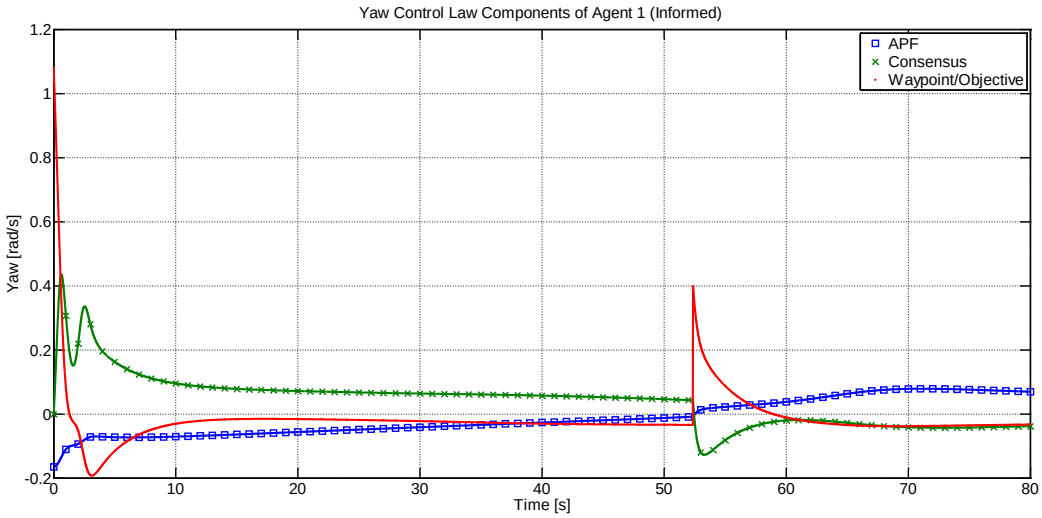

Figure 22 Components of yaw control law for agent 1 (informed).

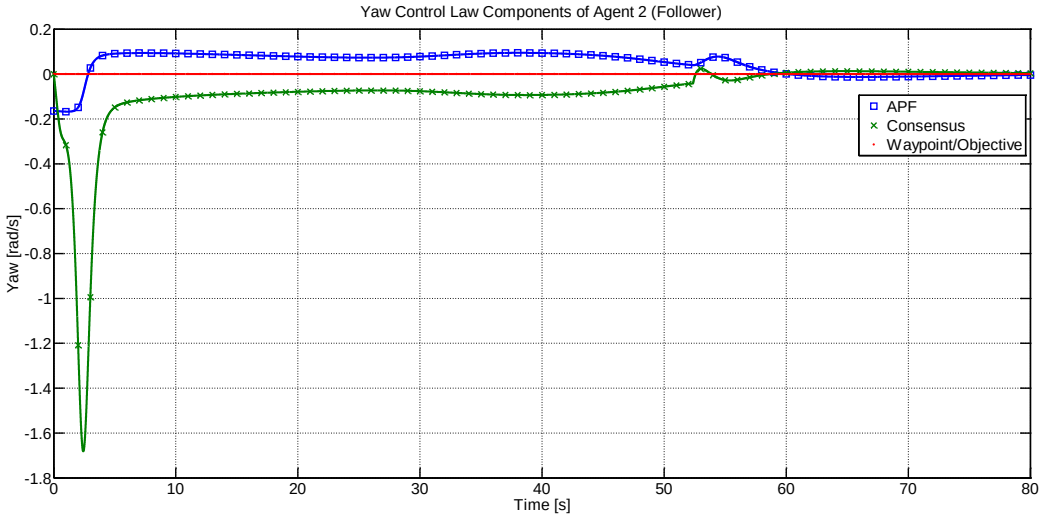

Figure 23 Components of yaw control law for agent 2 (follower).

Simulations 2-3: Large Swarm with Limited Communication

The next two simulations present cases with a much larger swarm and limited communication. The swarm has 15 agents with initial conditions shown in Figure 24. Five (5) informed agents are selected in strategic locations to prevent fragmentation of the swarm. The initial configuration consists of two concentric circles with five (5) agents on the inner circle and ten (10) agents on the outer circle. Two (2) sensing radius values are tested to see how the level of connectivity affects the swarm performance.

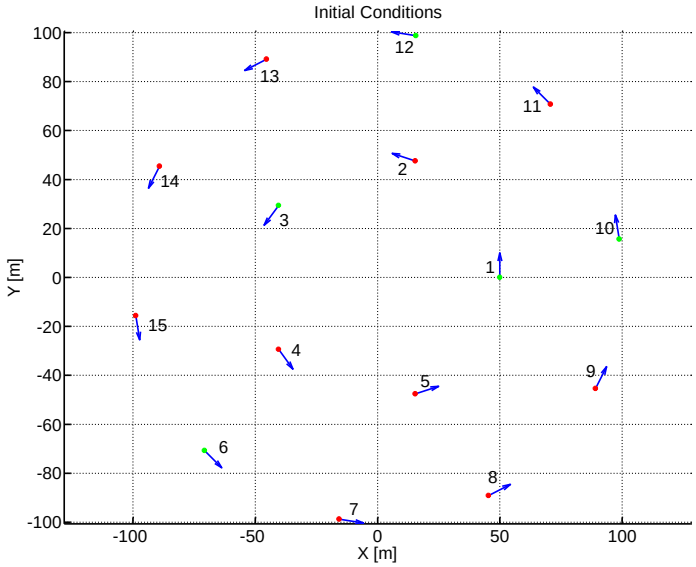

Figure 24 Initial conditions of 15 agent simulation. Informed agents (1, 3, 6, 10, and 12) are designated by green circles and follower agents are designated by red circles.

The connectivity of a swarm is typically measured by looking at the second smallest eigenvalue of the Laplacian matrix. The maximum value of this eigenvalue is the number of agents in the swarm and represents the case of full connectivity.

Table 4 Swarm Connectivity

Simulation #	Initial	Final
2	6.76	15.00
3	0.88	8.62

Table 4 shows the measure of swarm connectivity for both simulation cases at the start and end of the simulation. For the first case, the swarm ends with full connectivity while in the second case, the swarm starts with fairly low connectivity and ends with partial connectivity. The differences between these simulations will be discussed next.

Table 5 shows the results of the simulation evaluations based on the criteria defined in the beginning of this chapter. Each simulation is given a "pass" or "fail" to indicate whether or not it met the criterion. For criteria 3 and 4, the simulation is evaluated at the two waypoints (WP). Since the second waypoint is not reached in both simulations, the second waypoint is considered the end of the simulation. Simulation 3 failed to meet the speed consensus criterion, but if the simulation was run with a larger simulation time then it would eventually pass the test. It will be seen later in this section that the second swarm is still in the process of achieving a minimal energy state at the end of the simulation. The separation error criterion is not satisfied for the same reasons discussed in the previous section with the smaller swarm. All of the graphs used to deduce these results can be found in Appendix B.

Table 5 Evaluation criteria results for simulations 2 and 3.

Criterion	Nomenclature	Simulation 2	Simulation 3
1	Collision Avoidance	PASS	PASS
2	Cohesion	PASS	PASS
3	Speed Consensus	FAIL at WP 1 PASS at WP 2	FAIL at WP 1 FAIL at WP 2
4	Heading Consensus	FAIL at WP 1 PASS at WP 2	FAIL at WP 1 PASS at WP 2
6	Separation Error	FAIL	FAIL
7	Max Speed	PASS	PASS
8	Max Acceleration	PASS	PASS
9	Max Yaw Rate	FAIL*	PASS

*Failed by a very small deviation from the limit.

Figure 25 shows the trajectory of a swarm with a sensing radius of 150 m and Figure 26 shows the trajectory of a swarm with a sensing radius of 80 m. It is obvious from these figures that the swarm with the smaller sensing radius takes more time to completely aggregate than the other swarm.

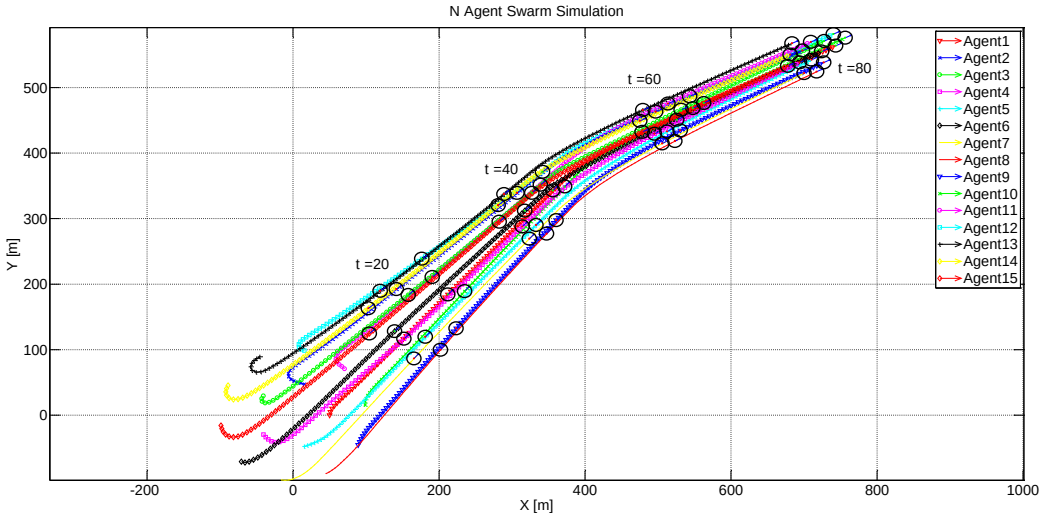

Figure 25 Trajectory of 15 agents with a sensing radius of 150 m.

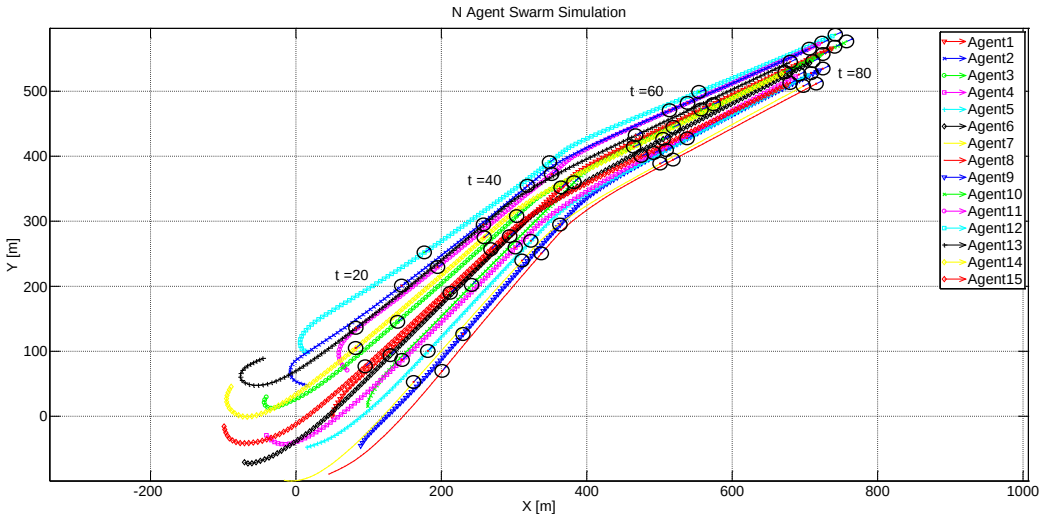

Figure 26 Trajectory of 15 agents with a sensing radius of 80 m.

Figure 27 and Figure 28 show the final positions of the swarm in both simulations cases. It is more obvious from these figures that the swarm in the second simulation has not completely aggregated. In addition, it can be observed that in both simulations the informed agents form a sub-cluster within the swarm. This is a result of the strong potential function that attracts the informed agents to the waypoint.

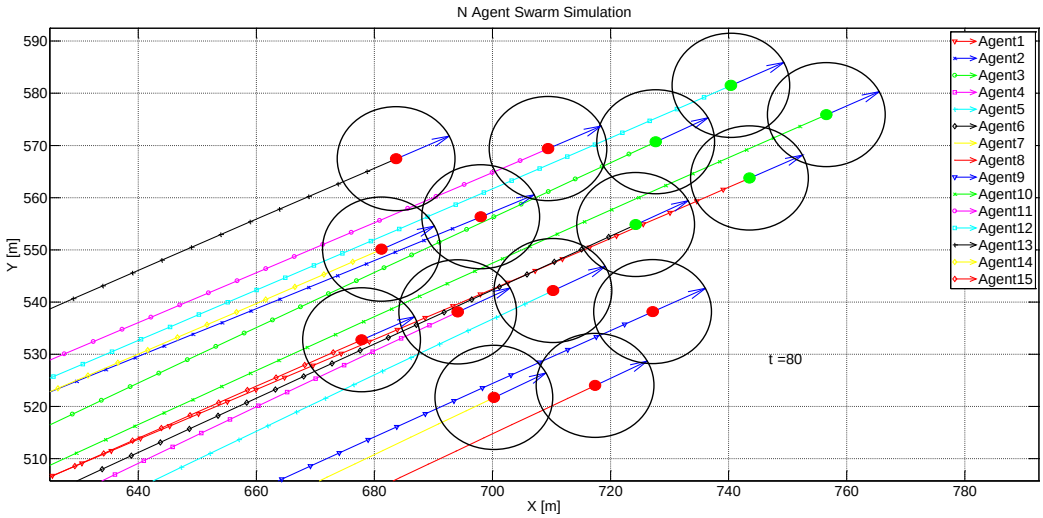

Figure 27 Final positions of 15 agents with a sensing radius of 150 m.

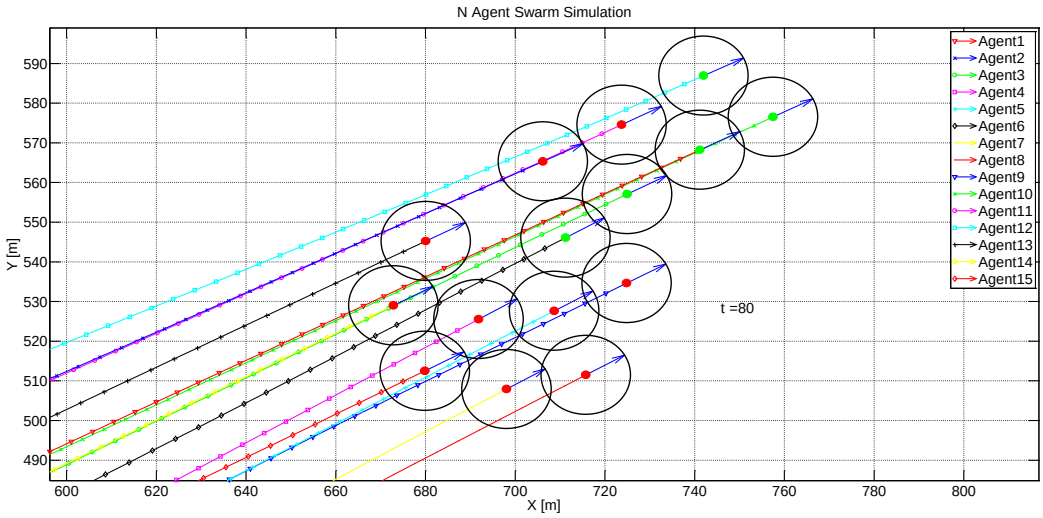

Figure 28 Final positions of 15 agents with a sensing radius of 80 m.

Figure 29 and Figure 30 show the Lyapunov function history for the two simulation cases. The simulation with low connectivity has a smaller magnitude overall compared with the simulation with high connectivity. In addition, the change in the Lyapunov function value over the entire time period is smaller in the low connectivity case than the high connectivity case. This supports the previous claim that the second swarm is slower to aggregate than the first. Another observation is that both curves exhibit highly discontinuous behavior at many points along the timeline. This is a result of the constantly changing neighborhood size of every agent.

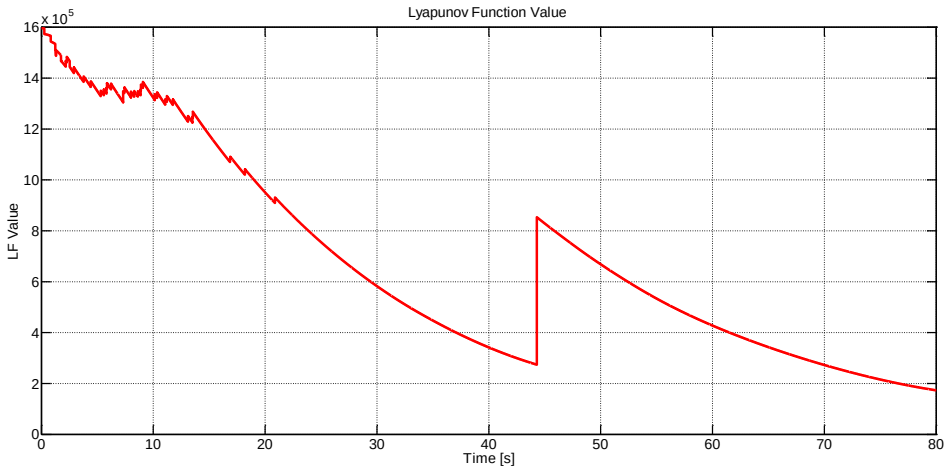

Figure 29 Lyapunov function history for swarm with sensing radius of 150 m.

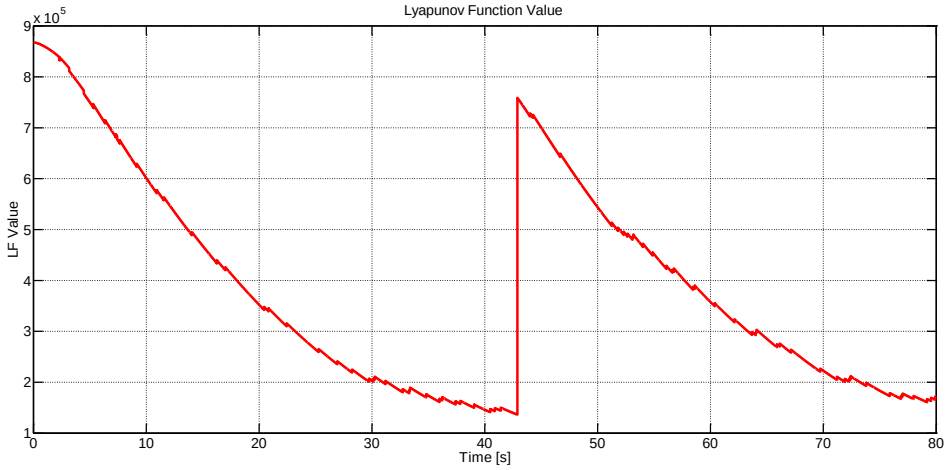

Figure 30 Lyapunov function history of swarm with sensing radius of 80 m.

Figure 31 and Figure 32 shows the time derivative of the Lyapunov functions previously presented. Both curves are negative for the entire simulation time, which indicates that both swarms are stable according to Lyapunov function theory. Both curves have been plotted with points at every ¼ second to remove the sharp spikes caused by the varying neighborhood sizes. This behavior does not affect the stability of the system, but it does cause noticeable discontinuities in the control inputs.

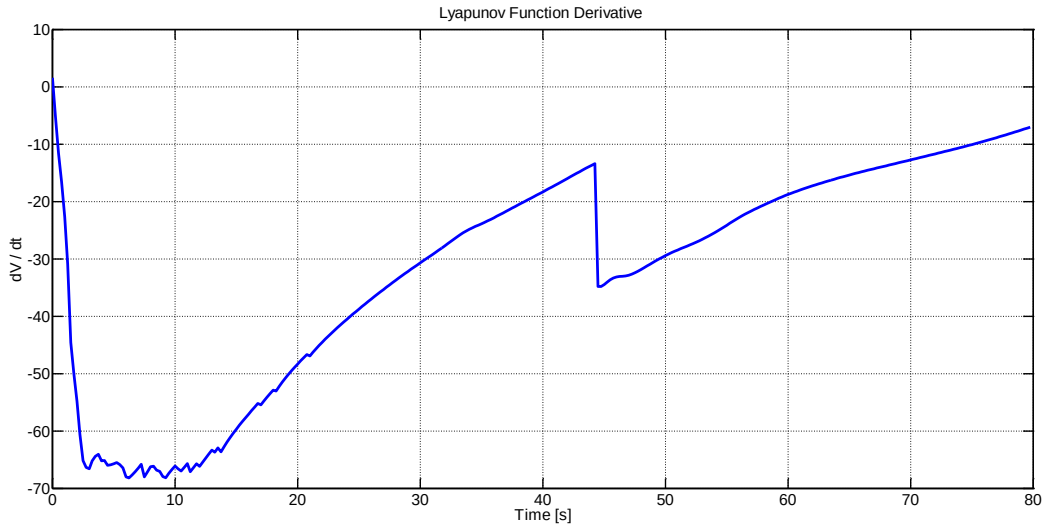

Figure 31 Lyapunov function time derivative for simulation 2.

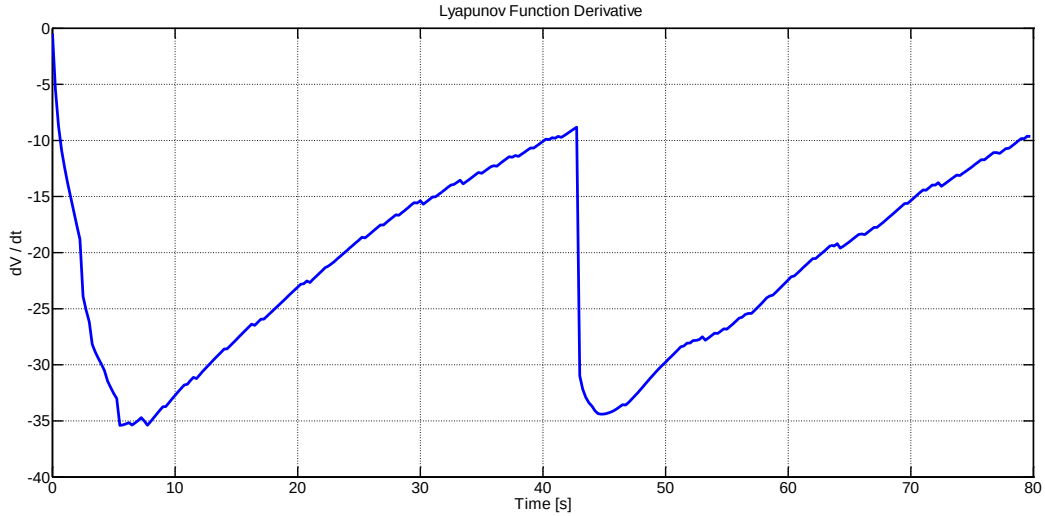

Figure 32 Lyapunov function time derivative for simulation 3.

Figure 33 and Figure 34 show the acceleration control inputs of select follower agents, and Figure 35 and Figure 36 show the yaw control inputs of select follower agents from the two simulation cases, respectively. The change in neighborhood sizes has a more pronounced affect in the acceleration control inputs compared with the yaw inputs. In addition, the simulation case with a lower sensing radius exhibits significantly more discontinuities than the former case. The gain function introduced in the previous chapter obviously does not completely alleviate the discontinuous behavior.

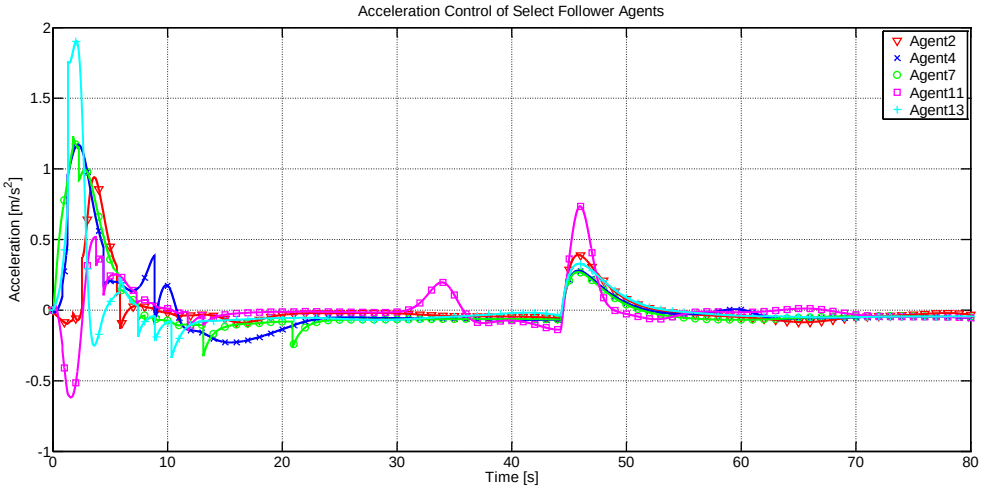

Figure 33 Acceleration control of select follower agents from simulation 2.

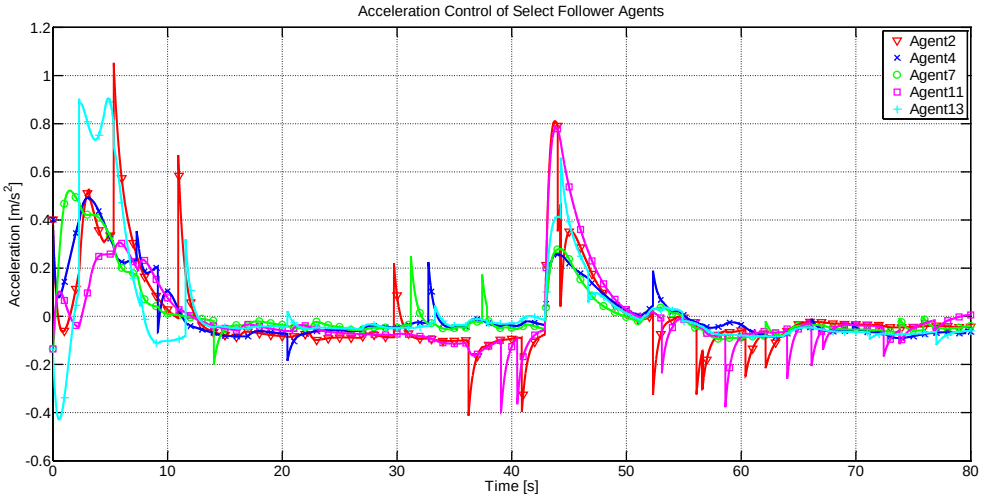

Figure 34 Acceleration control of select follower agents from simulation 3.

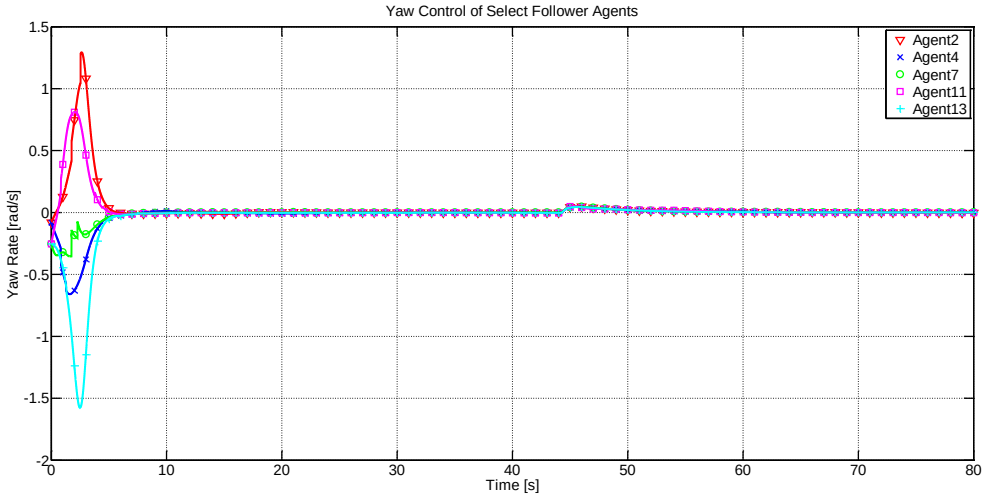

Figure 35 Yaw control of select follower agents from simulation 2.

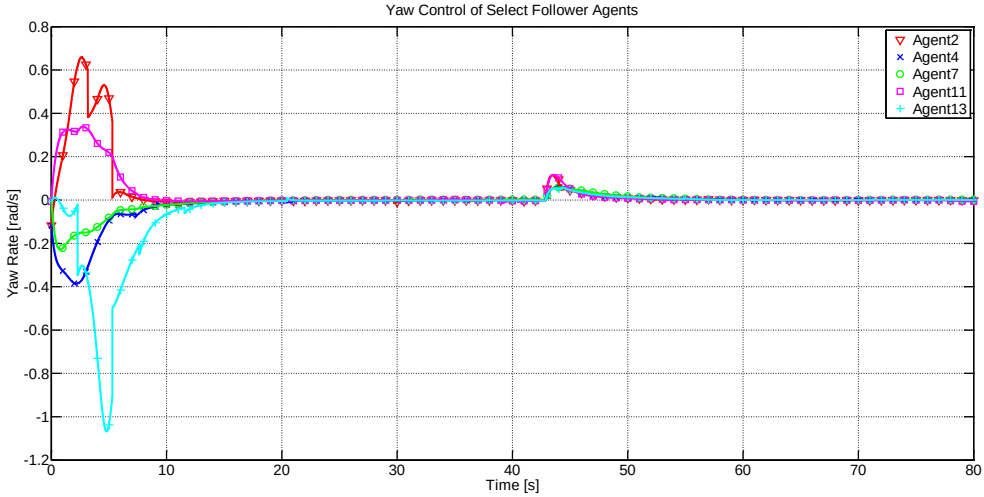

Figure 36 Yaw control of select agents from simulation 3.

Additional design modifications need to be implemented to reduce the level of discontinuity in the control inputs. There are two ways in which this can be accomplished: (1) change how the network topology is synthesized, and (2) change the control design. If a way to minimize the number of links in the network topology and suppress the dynamic behavior of the network could be found, then the effects on the control law will be minimized.

Another possibility is modifying the existing control framework such that it can detect a change in the neighborhood size and perform an interpolation between the new control law value and an *assumed extension* of the previous control law value until the error between the two values falls below a defined tolerance. The *assumed extension* can be some function such as a line with a slope in the direction of the discontinuity (i.e. if the control input sees an instant decrease then the slope would be negative) such that the extension of the old control input tends to intersect with the actual control input after the discontinuity. Such a function would look like:

$$u_{ext}(t) = u_{act}(t_d - \Delta t) + k\ sgn(\Delta u)(t - t_d)$$
$$\Delta u = u_{act}(t_d) - u_{act}(t_d - \Delta t)$$

where Δu is the magnitude of the discontinuity, t_d is the time at which the discontinuity occurs, Δt is the time step, u_{act} is the actual control law value, u_{ext} is the assumed extension of the control law before the discontinuity, k is an arbitrary slope.

Weighting can be added to the interpolation as well to better smooth the control input. Without weighting, the continuity would still exist but the change would be half the magnitude. The functions could be weighted such that the extension gets full weight at first and then the weight is linearly shifted to the actual control input value over time. Such a scheme would look like:

$$u_{new}(t) = \frac{1}{2}\left[\big(1 - w(t)\big)u_{ext}(t) + w(t)u_{act}(t)\right]$$
$$w(t) = \frac{t - t_d}{\lambda \Delta t + t_d}$$

where u_{new} is the final value used for control, w is a weighting function, and λ is a scalar that represents the predicted number of time steps it takes for u_{ext} and u_{act} to intersect after the discontinuity occurs. This scheme would only be applied when a discontinuity occurs and then deactivated as soon as $|u_{ext} - u_{act}| < \delta_d$ for an arbitrary tolerance δ_d.

Phase 2: Three-Dimensional Swarm Simulations

Simulation Setup

The first set of simulations use the three-dimensional agent model described by model (3.9). Table 6 shows all of the simulation parameters used in the simulations described in this section. The first simulation demonstrates the control method described in Chapter 4 as applied to five (5) agents. The agents are considered to be small meter-scale fixed-wing UAVs with a large sensing range ($R_{sen} = 150\ m$). As a result, the first simulation will demonstrate a case in which there is all-to-all communication between agents. The second simulation is a case when the number of agents is very large. Not all data from the simulations is presented in this section; additional plots can be found in Appendix B.

Table 6 Simulation parameters for three-dimensional simulations

Simulation Parameter	Symbol	Run 1	Run 2
Number of Agents	N	5	45
Acceleration Consensus Gain	$k_{c,1}$	5.000	6.000
Acceleration APF Gain	$k_{j,1}$	0.500	0.080
Acceleration Target Gain	$k_{o,1}$	0.006	0.006
Acceleration Damping Gain	k_d	6.000	12.000
Pitch Consensus Gain	$k_{c,3}$	4.000	6.000
Pitch APF Gain	$k_{j,3}$	0.003	0.010
Pitch Target Gain	$k_{o,3}$	0.001	0.001
Yaw Consensus Gain	$k_{c,4}$	6.000	6.000
Yaw APF Gain	$k_{j,4}$	0.005	0.010
Yaw Target Gain	$k_{o,4}$	0.001	0.001
APF Attraction Coefficient	a	2.000	2.000
APF Repulsion Coefficient	b	1304	6589
APF Shaping Constant	c	50	40

Simulation 1: Small Swarm with All-to-All Communication

In the first simulation, the swarm consists of five (5) agents, two (2) of which are informed agents. The initial conditions place each agent at even angular increments around the origin. Each agent is at a radius from the origin within the range 30-50 meters and an altitude within the range 1180-1216 meters. The choices for exact values are based on an assignment algorithm using a for loop and arbitrary equations. Figure 37 shows the initial conditions from the isometric, top, and side views. Since the number of agents is small and the initial conditions are tight, all agents are able to communicate with each other. Therefore, this simulation presents a case with full connectivity for all time.

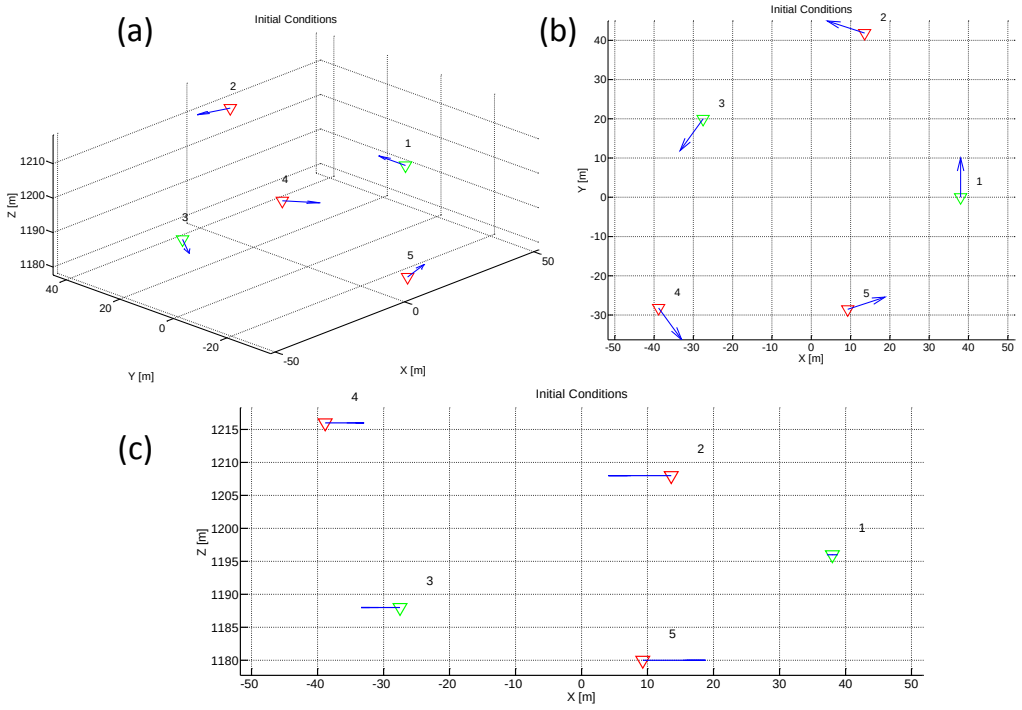

Figure 37 Initial Conditions for three-dimensional simulation 1. Informed agents are shown as green triangles (agents 1 and 3) and follower agents are red triangles. (a) three-dimesional view, (b) XY-plane view, (c) XZ-plane view.

Figure 38 shows the trajectories of the five (5) agents in 3D space. It is important to note that all agents quickly achieve heading consensus and remain cohesive for the entire simulation time. There appears to be no major change to the swarm shape over time, and the agents never reach the same altitude. This objective was not considered in the controller, but an altitude consensus law can be designed to augment the pitch controller. Next, the simulation data will be analyzed to see if the evaluation criterion was met.

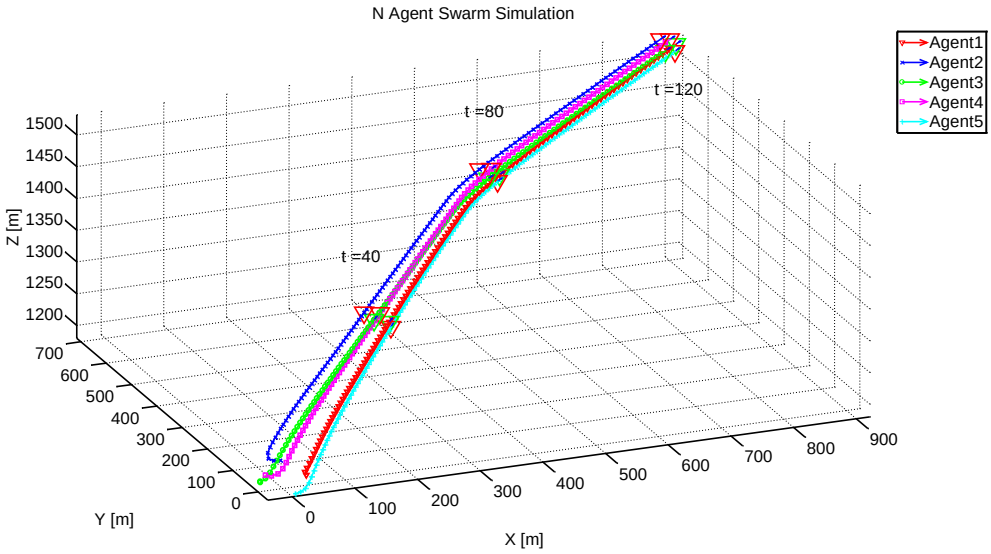

Figure 38 Trajectory of 5 agent three-dimensional simulation.

Figure 39 shows the relative position statistics for this simulation. According to this figure, it can be said that criterion 1, 2, and 6 are satisfied. First, it can be confirmed that the maximum relative position is less than the sensing radius for all time. Next, it can be seen that no collisions occur and that both the minimum and average relative positions fall within the separation distance tolerance band by the end of the simulation. The latter observation was not seen in any of the two-dimensional simulations. The reason for this behavior is likely the additional degree of freedom given by the altitude coordinate and the small size of the swarm. With so few agents in the swarm, the agents have very few attraction/repulsion forces acting on them. Therefore, the demand on each agent is more relaxed than in the two-dimensional case.

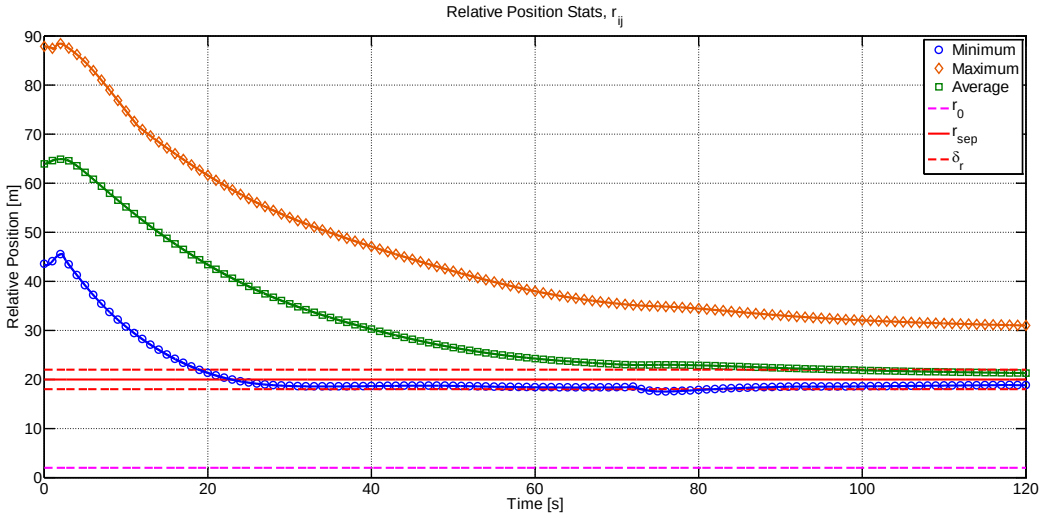

Figure 39 Relative position statistics for 5 agent three-dimensional simulation.

Figure 40 and Figure 41 show the relative heading (yaw) and pitch statistics, respectively. As suspected above, heading and pitch consensus are achieved rather quickly in this simulation. Even after the first waypoint is crossed, relative heading and pitch quantities remain within the tolerance band. Therefore, criterion 4 and 5 are satisfied.

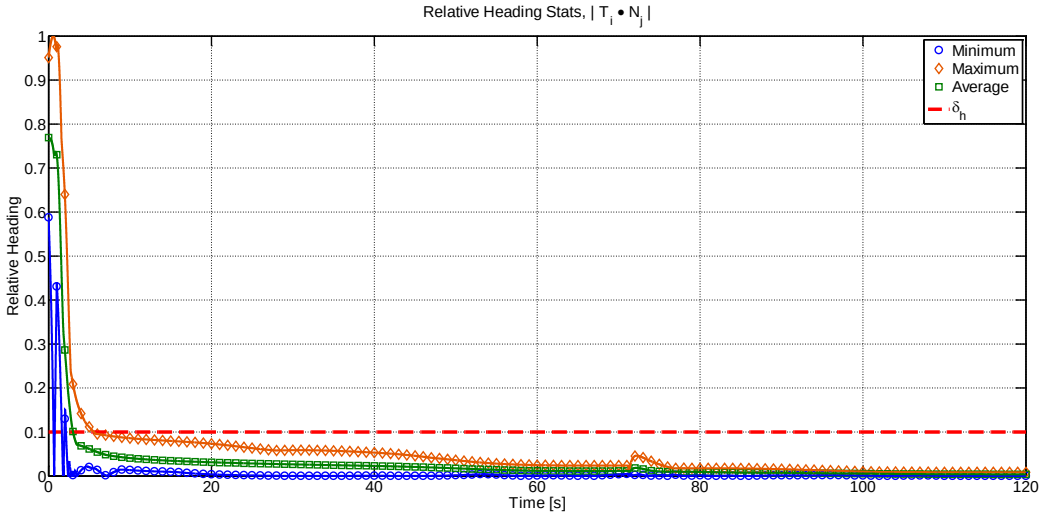

Figure 40 Relative heading statistics for 5 agent three-dimensional simulation.

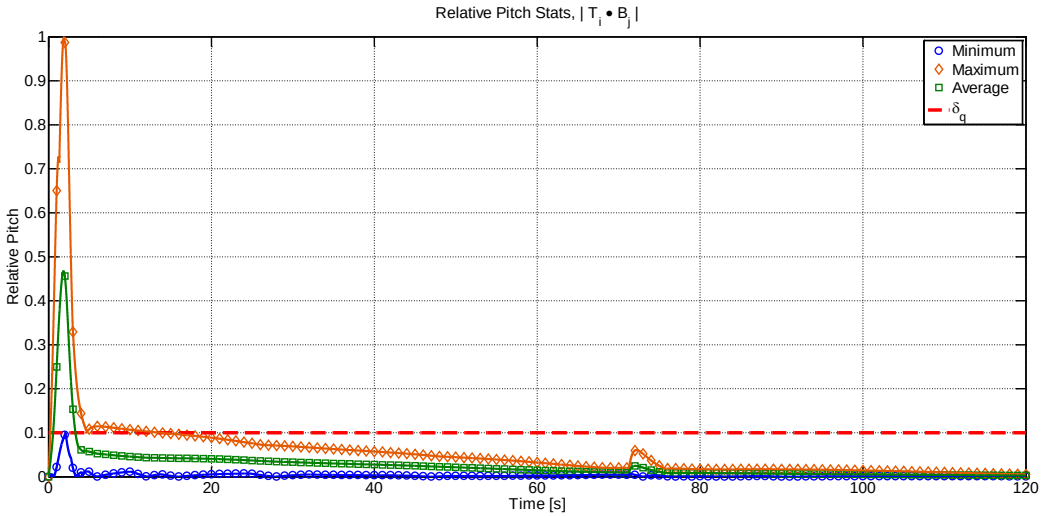

Figure 41 Relative pitch statistics for 5 agent three-dimensional simulation.

Figure 42 shows the relative speed statistics. Speed consensus is achieved about 10-15 seconds after heading and pitch consensus are achieved. According to this figure, criterion 3 is satisfied.

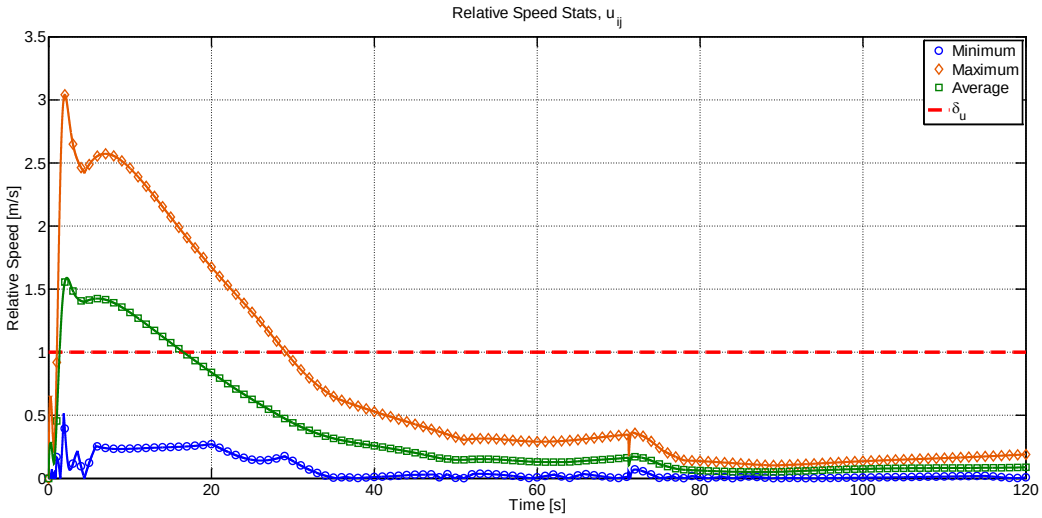

Figure 42 Relative speed statistics for 5 agent three-dimensional simulation.

Figure 43 shows the speed history of all agents. The maximum speed is not exceeded; therefore, criterion 7 is satisfied.

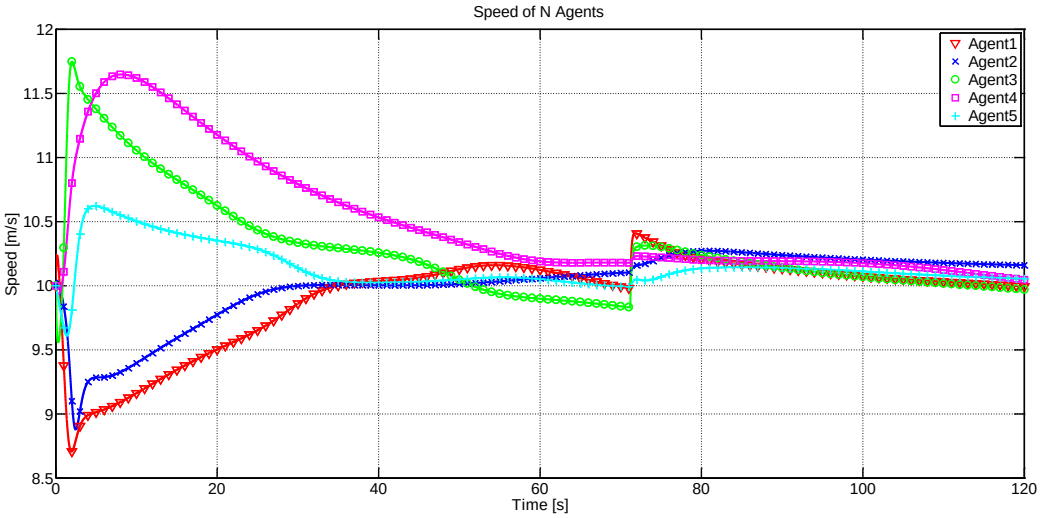

Figure 43 Speed of 5 agents in three-dimensional simulation.

Figure 44 shows the full time history of acceleration control inputs for all agents. The maximum acceleration is not exceeded for all time; therefore, criterion 8 is satisfied. Figure 45 shows the first 10 seconds of the acceleration control input. The inputs all die out very quickly. An interesting observation is that agent 1 starts by accelerating and agent 3 starts by decelerating. Agent 1 accelerates because it is already almost aligned with the waypoint bearing vector, while agent 3 is heading almost directly opposite to the waypoint. Therefore, agent 3 slows down so it can make the turn. This is an example that the control laws, despite being complex and conflicting, exhibit desirable behaviors.

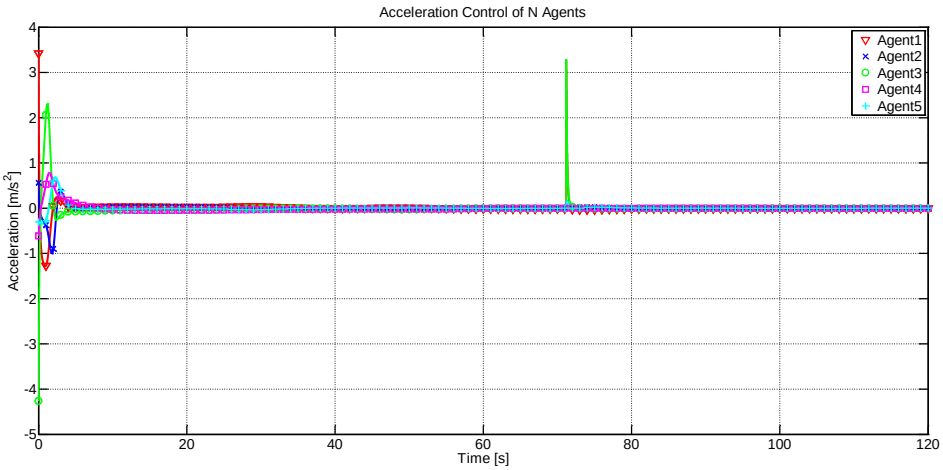

Figure 44 Acceleration control of 5 agents in three-dimensional simulation.

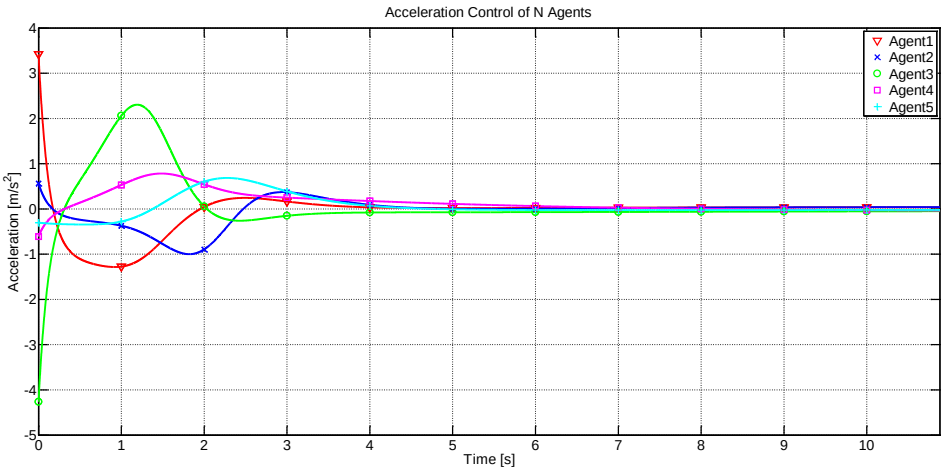

Figure 45 First 10 seconds of acceleration control input.

Figure 46 shows the full time history of pitch control inputs for all agents. The maximum pitch rate is not exceeded; therefore, criterion 10 is satisfied. Figure 47 shows the first 30 seconds of the pitch control inputs.

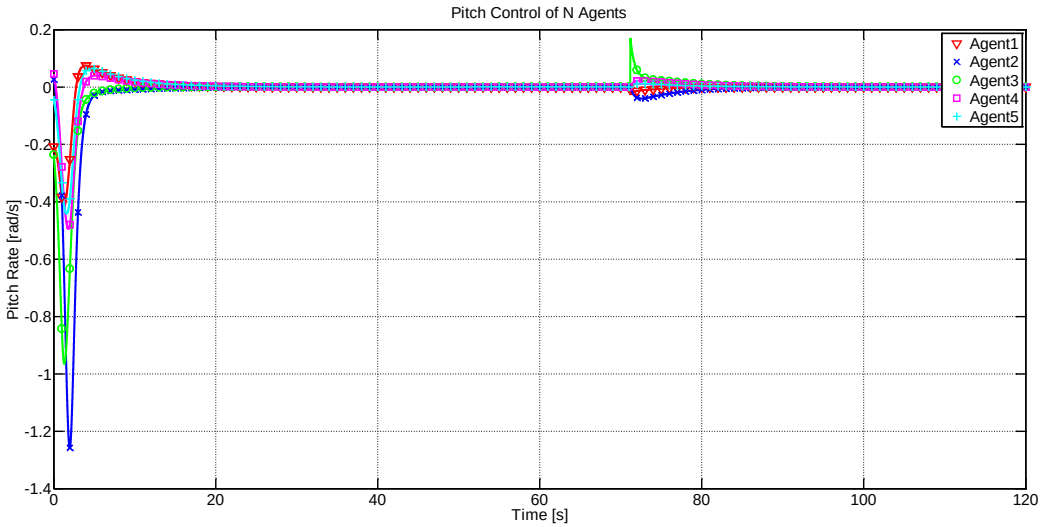

Figure 46 Pitch control of 5 agents in three-dimensional simulation.

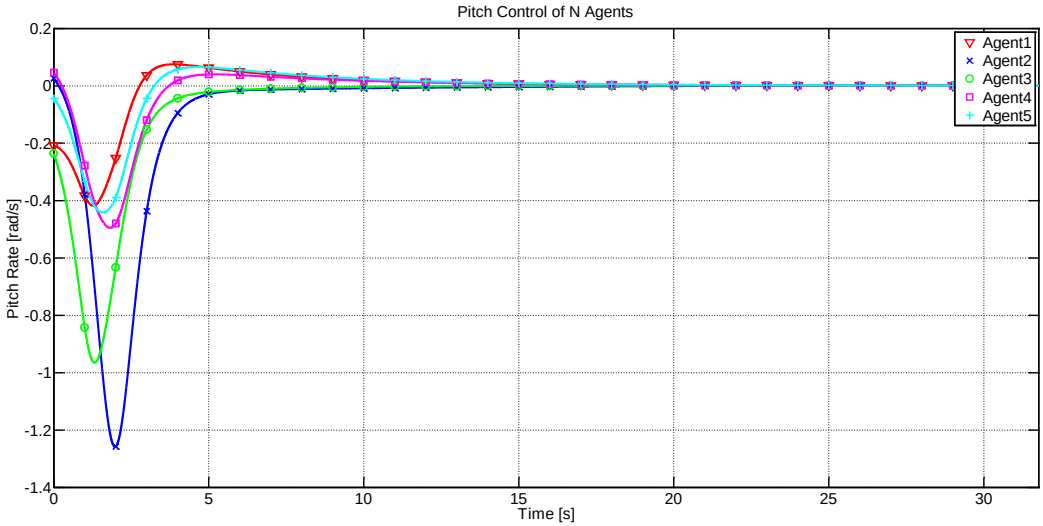

Figure 47 First 30 seconds of pitch control input.

Figure 48 shows the full time history of yaw control inputs for all agents. The maximum yaw rate is not exceeded; therefore, criterion 9 is satisfied. Figure 49 shows the first 30 seconds of the yaw control inputs.

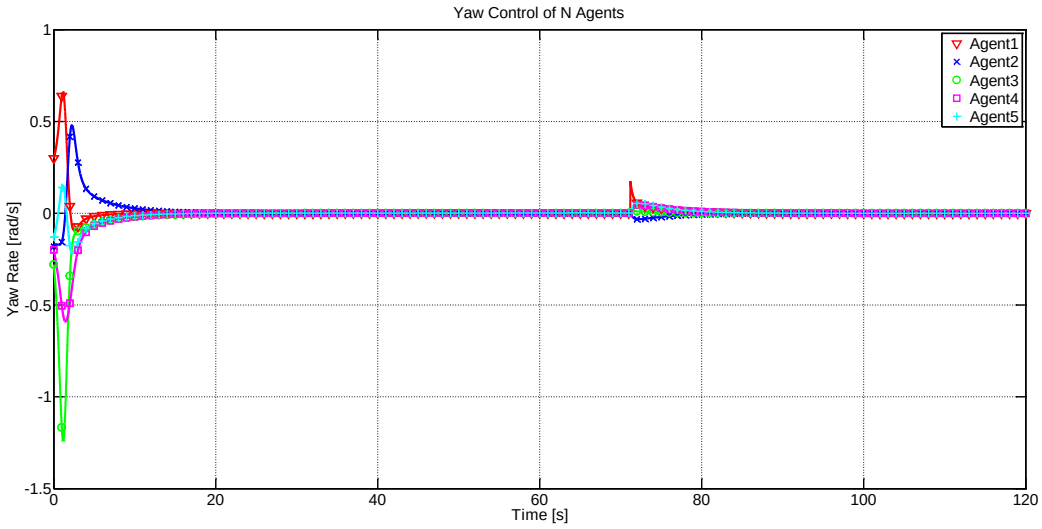

Figure 48 Yaw control of 5 agents in three-dimensional simulation.

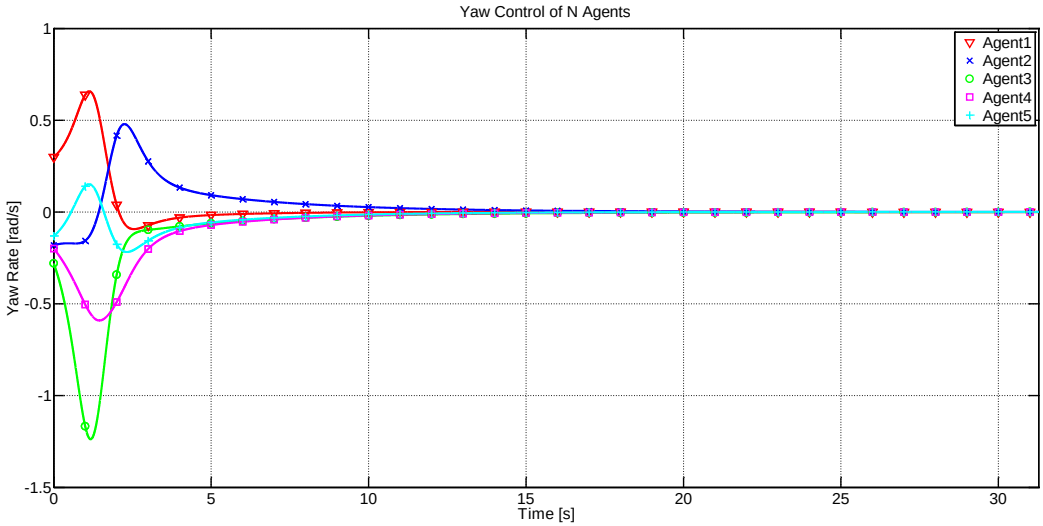

Figure 49 First 30 seconds of yaw control input.

Figure 50 shows the Lyapunov function time history, and Figure 51 shows its time derivative. The time derivative remains negative for the entire simulation time except for a small interval within the first second. This is likely due to the agents reacting to the extreme initial conditions. Otherwise, these figures suggest stable swarm behavior.

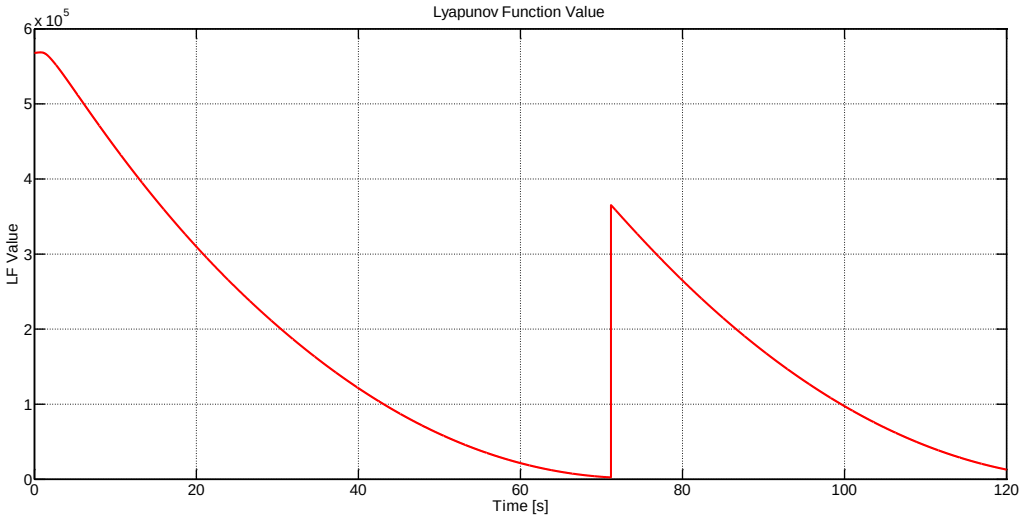

Figure 50 Lyapunov function time history.

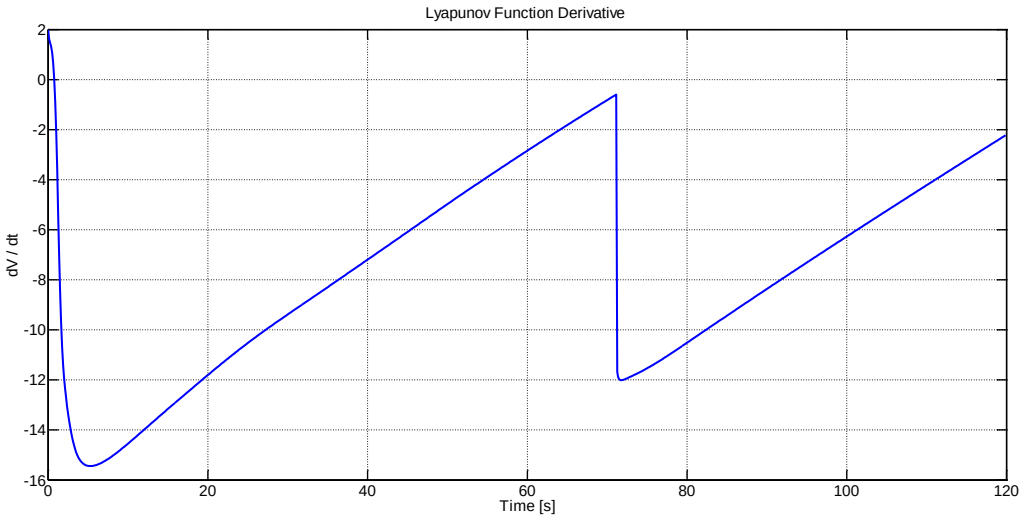

Figure 51 Lyapunov function time derivative. Points are plotted every ¼ second.

Simulation 2: Large Swarm with Limited Communication

The second three-dimensional simulation represents a case with a much larger swarm. Due to the large size of the swarm, the time step was reduced to $dt = 2\ ms$ to reduce the amount of time required to run the simulation. The swarm has initial conditions given in Figure 52. A slightly different assignment algorithm is used compared with the previous simulation. This design has three (3) sets of fifteen (15) agents, each at a different average altitude. At each altitude there is one ring of five (5) agents and one ring of ten (10) agents. The angular spacing is equal for each ring and the average starting radii from the origin are 60 m for the five (5) agents and 150 m for the ten (10) agents. The average altitudes are 1200 m, 1320 m, and 1440 m.

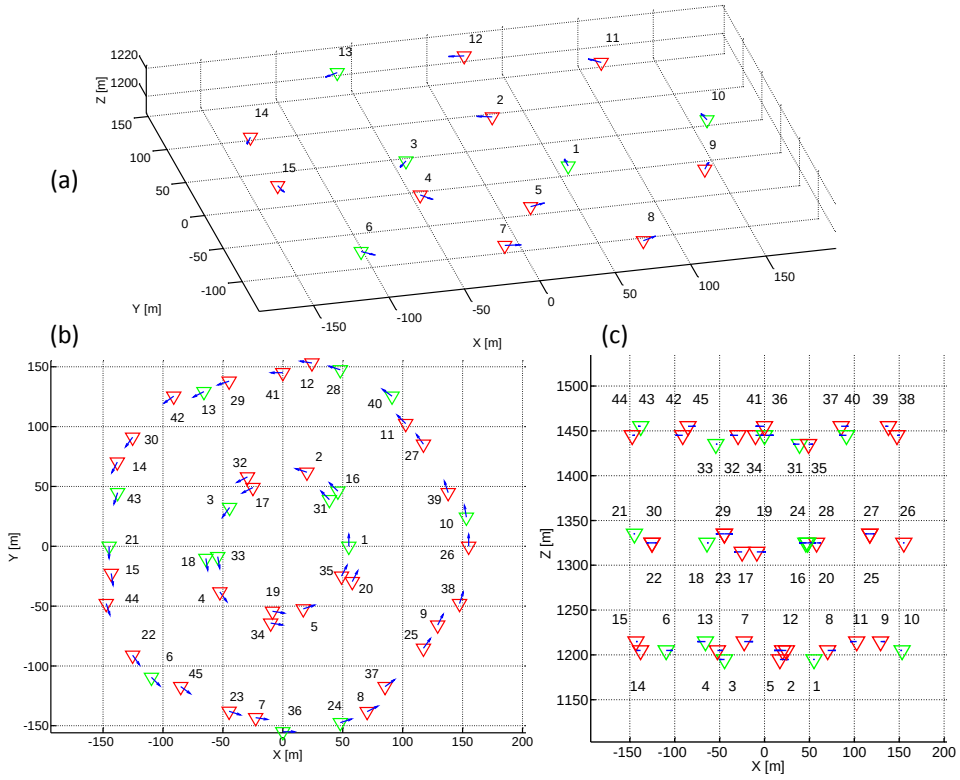

Figure 52 Initial conditions for simulation 2. Informed agents are shown in green (1, 3, 6, 10, 13, 16, 18, 21, 24, 28, 31, 33, 36, 40, and 43), and follower agents are red. (a) Three-dimensional view of 1/3 of the agents; the full swarm is 3 variations of this formation at different altitudes. (b) XY-plane view and (c) XZ-plane view of swarm.

The large size of the swarm and the initial conditions described above create a condition where all agents cannot sense every other agent. This will still be the case when the swarm has aggregated. The algebraic connectivity described earlier in this chapter can be used to quantify the connectivity of the swarm at every time step. Figure 53 shows the connectivity of the swarm in this simulation. The connectivity starts very low, but continues to increase over time. Even at the end of the simulation, the swarm is not fully connected. This simulation case is an excellent test of the proposed controller's ability to handle a varying network topology.

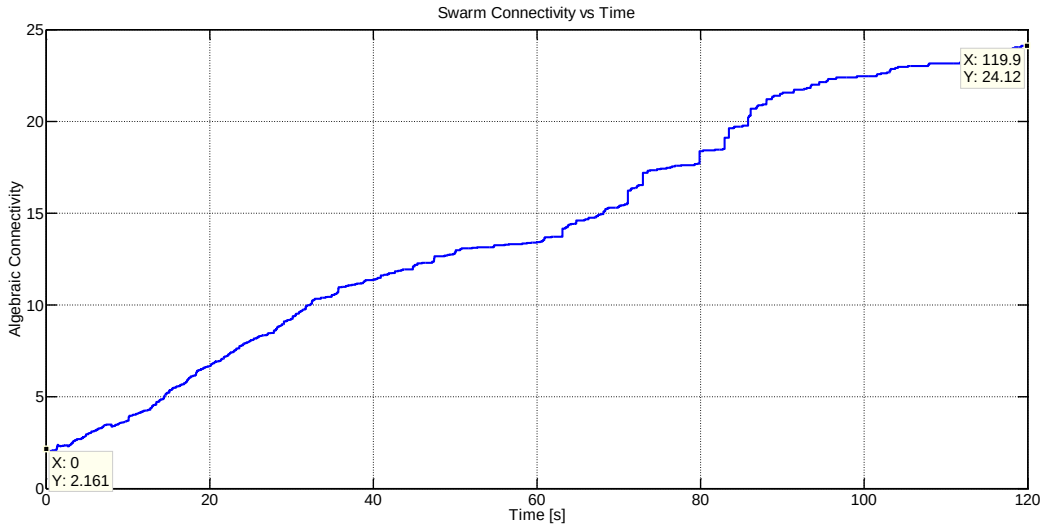

Figure 53 Swarm algebraic connectivity history.

Figure 54 shows the trajectory of all 45 agents. Figure 55 and Figure 56 show the XY- and XZ-plane views of the trajectory, respectively. These figures show that the agents successfully reached the first waypoint and were approaching the second waypoint. The swarm became cohesive and eventually reached an elongated box shape. Without any kind of shape controls, the swarm will create an arbitrary shape that represents a local minimum of the energy (Lyapunov) function. The final configuration appears to be very tight and dense. A look at the relative positions statistics will give a better picture of the relative spacing behavior.

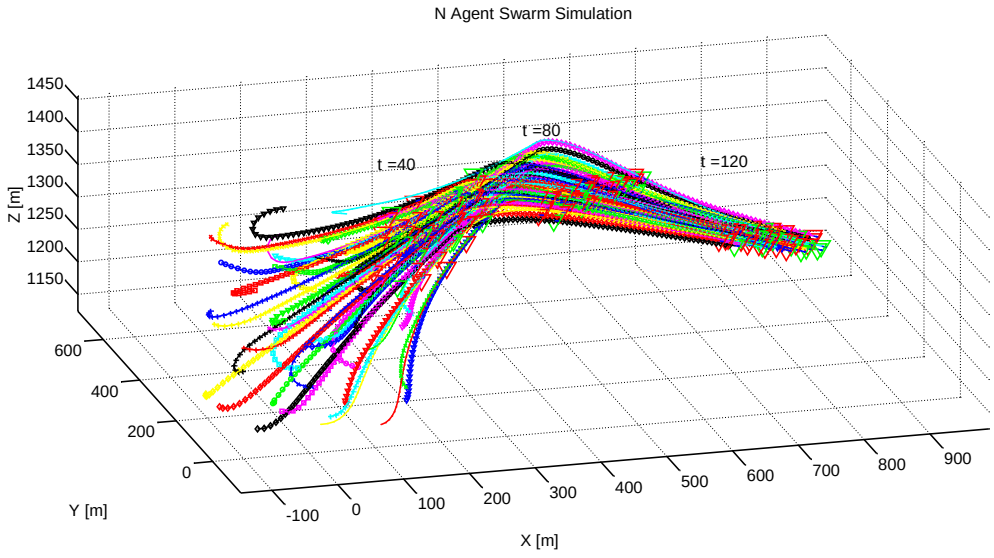

Figure 54 Trajectory of 45 agents.

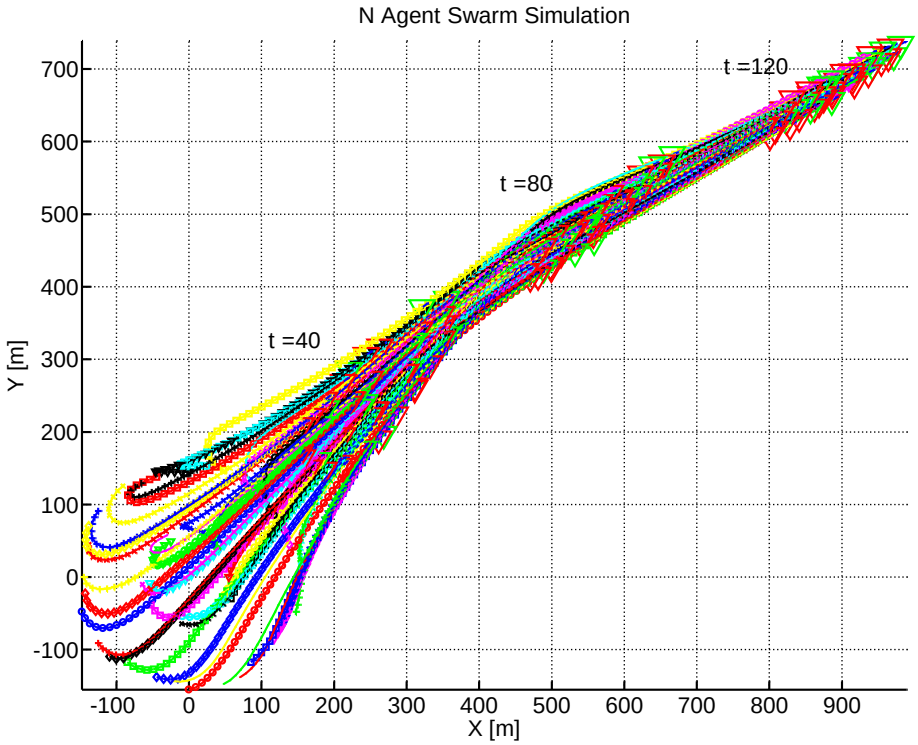

Figure 55 XY-plane view of trajectory.

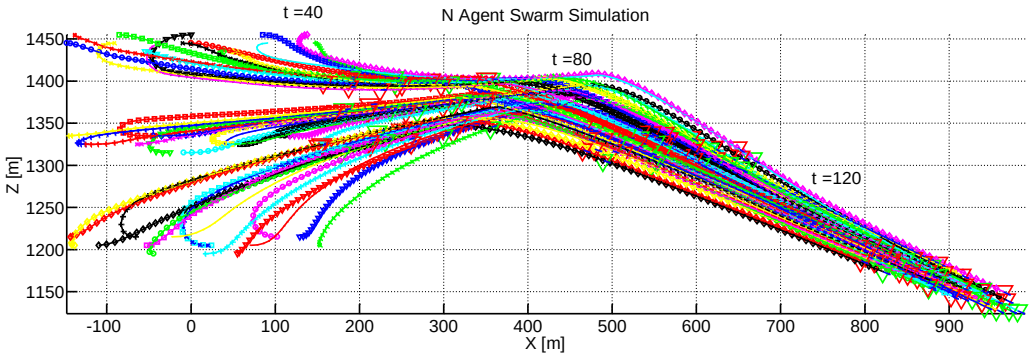

Figure 56 XZ-plane view of trajectory.

Figure 57 shows the relative position statistics for this simulation, and Figure 58 shows a close up view of the minimum relative position curve. Criterion 1 and 2 are both satisfied according to the first figure. Criterion 6 is not satisfied, and this simulation represents the largest separation distance error compared to all previous simulations in this chapter. There are likely two factors that are contributing to this phenomenon. First, the informed agents are dispersed throughout the swarm and have a strong attraction to the waypoint that will tend to make them diffuse through to the head of the swarm. This may cause the informed agents to move through other agent's repulsion region. Second, the sensing radius of the agents is so large that agents on the outside of the swarm will see repulsion from the interior agents they are too close to, but also attraction to agents who are even closer to the swarm center. If the attraction forces start to outweigh the repulsion forces, then the swarm will experience a sort of *squeezing* effect. This may explain the elongated box shape seen in the previous figures.

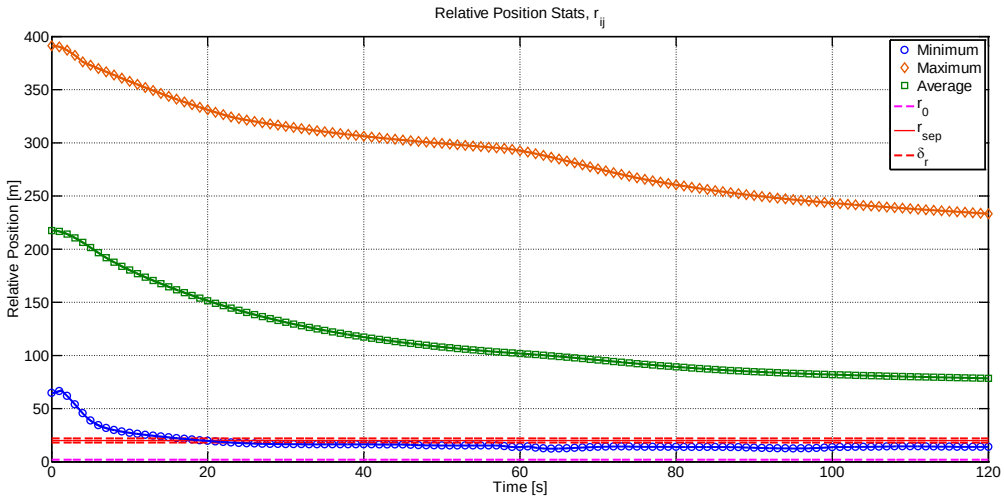

Figure 57 Relative position statistics for 45 agent swarm.

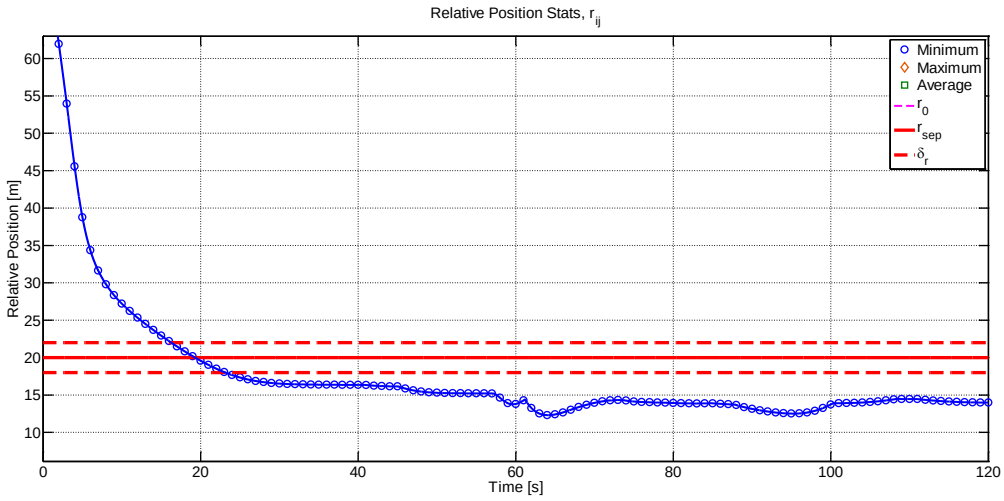

Figure 58 Minimum relative position for 45 agent swarm.

The issue noted in the previous paragraph can be addressed several ways. First, the addition of shape controls can reduce the number of possible formations that the swarm can achieve to a more specific subset. Also, the repulsion potential can be redesigned such that the separation distance requirement is relaxed. In essence, this modification is adding a deadband to the control function. Another possible solution is a redesign to the network topology synthesis such that the number of neighboring agents is minimized. This would help to eliminate the *squeezing* effect noted above.

Figure 59 shows the Lyapunov function time history, and Figure 60 shows its time derivative. The time derivative remains negative for all of time, but there are many spikes that have been removed for visual clarity. The spikes are a result of the changing network topology. This causes the initial increasing trend in the Lyapunov function. The large quantity of spikes in the time derivative can't be seen because it was plotted every 1/3 second. The increase is merely a result of the aggregation that caused more terms to be added to the Lyapunov function over time. The overall trend is a decreasing energy state, which suggests stable swarm behavior.

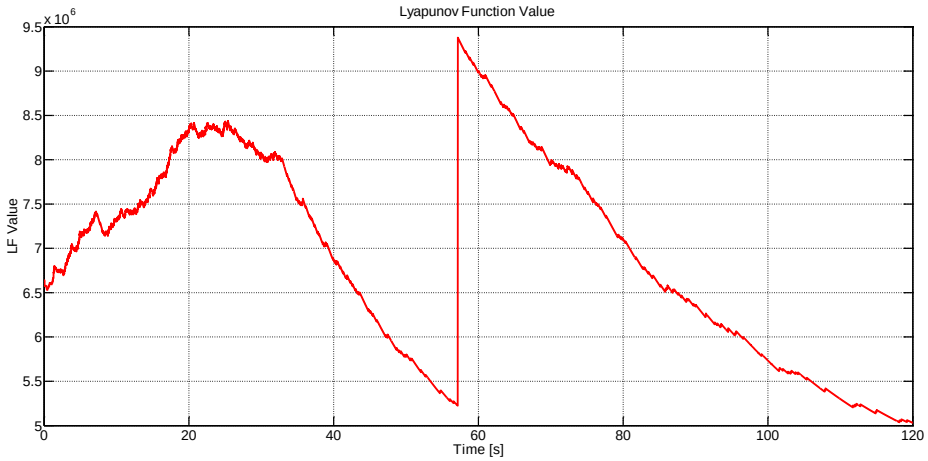

Figure 59 Lyapunov function time history for 45 agent swarm.

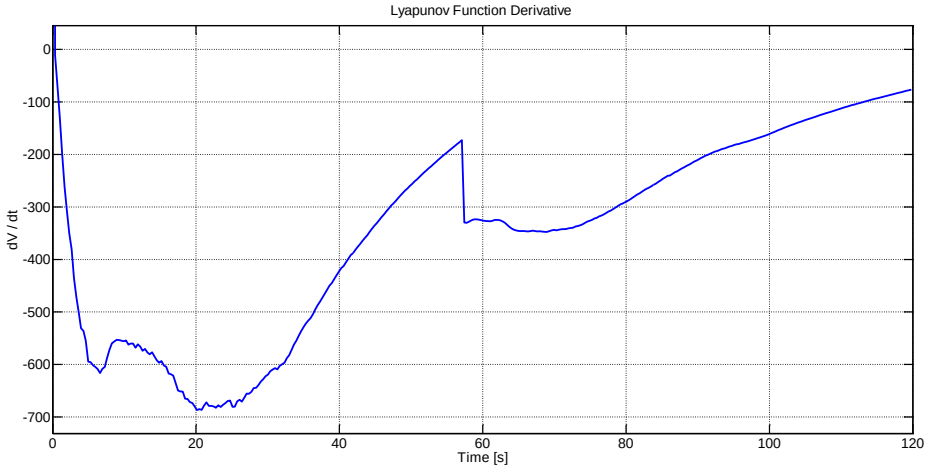

Figure 60 Lyapunov function time derivative.

Figure 61 shows the first 10 seconds of the acceleration control inputs to select follower agents. The inputs contain many discontinuities just like in the two-dimensional simulations. Some interesting observations can be made about the interacting control law components and their relation to the discontinuities. Figure 62 shows the components of the acceleration control input of agent 2. It can be seen that the primary contributor to the discontinuous behavior is the APF component. In addition, it can be seen that the damping component quickly tries to match the APF component, but because it is a reactive function, the discontinuity appears and dies off very quickly. Therefore, the damping component can be used to control the decay from the discontinuity, but it will never be able to eliminate it. The discontinuities are very small and fast, like noise in the control input. Depending on the hardware being used, this behavior may be tolerable. Regardless, a way to clean up the control signals should be devised.

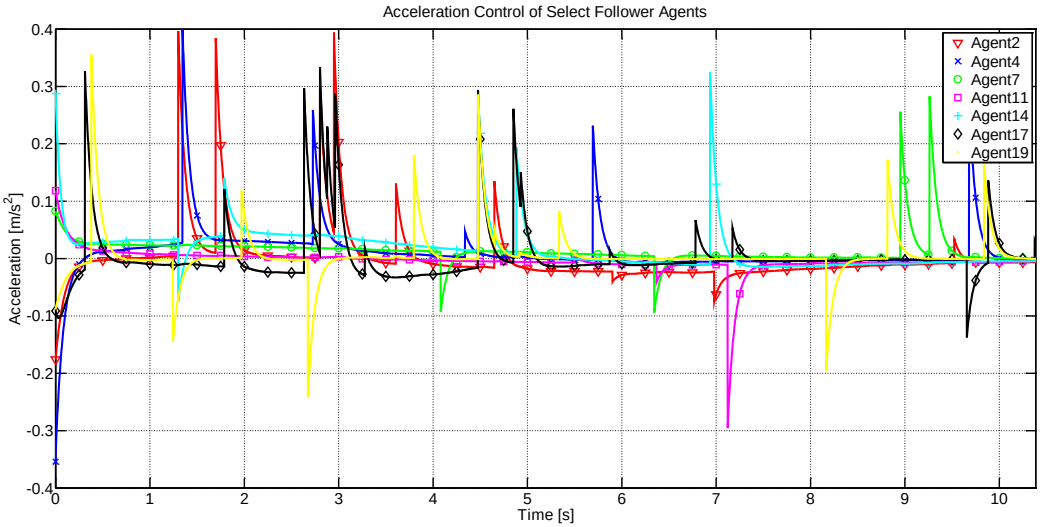

Figure 61 Acceleration control input for select follower agents.

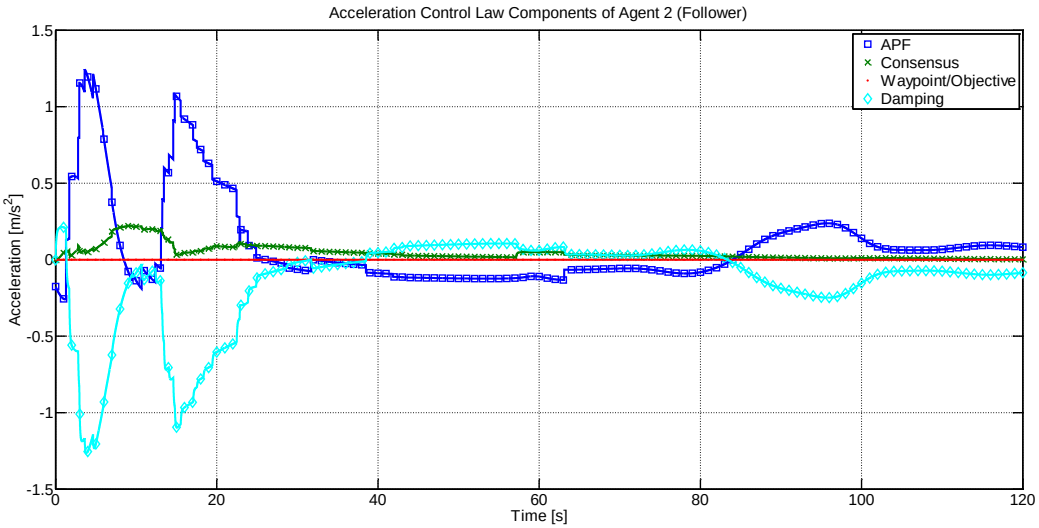

Figure 62 Components of acceleration control law of agent 2.

Figure 63 shows the first 20 seconds of the pitch control inputs for select follower agents. This control also exhibits some discontinuities, but they are not as pronounced as seen in the acceleration control input. This can be explained by looking at the individual control law components.

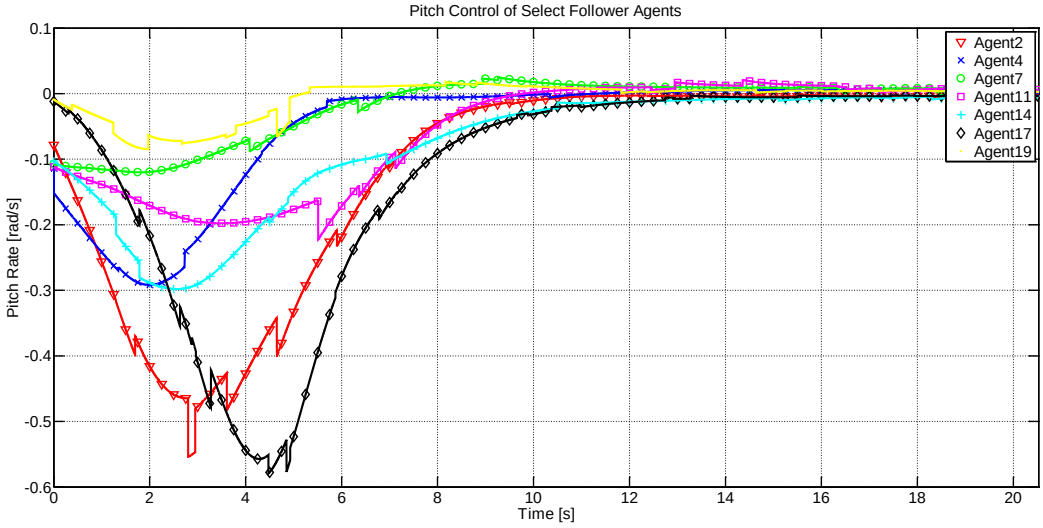

Figure 63 Pitch control inputs for select follower agents.

Figure 64 shows the components of the pitch control law of agent 2. Both the consensus and APF components are dependent on the network topology and both components tend to conflict with each other. As a result, when a change in the network topology occurs, both components change at the same time, and they will tend to change in the opposite direction. As long as the control gains are chosen such that the functions are well balanced, the discontinuities will tend to cancel out. This can be clearly seen in the figure, especially in the 10-20 second interval.

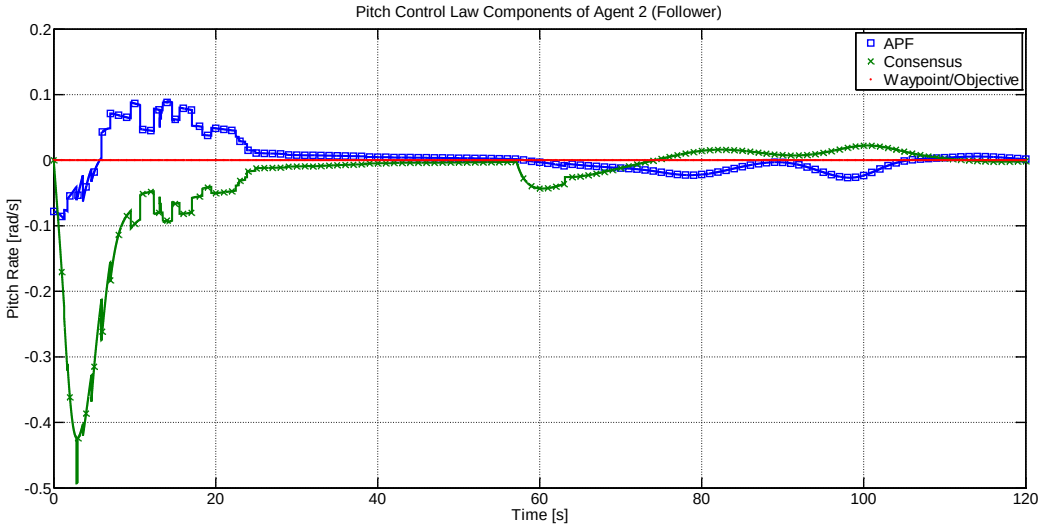

Figure 64 Components of pitch control law of agent 2.

Phase 3: Control Saturation Simulations

Simulation Setup

The final phase of simulations presented in this thesis demonstrates the effect of the saturation schemes presented at the end of Chapter 4. The same simulations parameters and initial conditions used in the first two-dimensional simulation are used here so that comparisons can be made. The maximum acceleration and yaw limits have been redefined so that saturation is induced. Table 7 shows all of these parameters. Not all of the simulation data is presented in the following sections. Interested readers should refer to Appendix B for the complete set of figures.

Table 7 Simulation parameters for control saturation simulations

Simulation Parameter	Symbol	Run 1	Run 2	Run 3
Saturation Scheme	NA	Post	Pre	Progressive
Number of Agents	N	6		
Acceleration Consensus Gain	$k_{c,1}$	5.0		
Acceleration APF Gain	$k_{j,1}$	0.100		
Acceleration Target Gain	$k_{o,1}$	0.008		
Acceleration Damping Gain	k_d	10.0		
Yaw Consensus Gain	$k_{c,2}$	6.0		
Yaw APF Gain	$k_{j,2}$	0.006		
Yaw Target Gain	$k_{o,2}$	0.003		
APF Attraction Coefficient	a	2.0		
APF Repulsion Coefficient	b	1303.9		
APF Shaping Constant	c	50		
Accel. Saturation Limit [m/s^2]	$\overline{U}_a$	2.5		
Yaw Saturation Limit [rad/s]	$\overline{U}_r$	1.5		

Simulation 1: Effects of Post-Saturation

Figure 65 shows the trajectories from a simulation with no saturation compared to a simulation with post-saturation. The change in performance between the two simulations is minimal. Both swarms do not completely aggregate and form two groups of agents. Both groups contain an informed agent and are within sensing range of each other. This means the flocking behavior can be maintained.

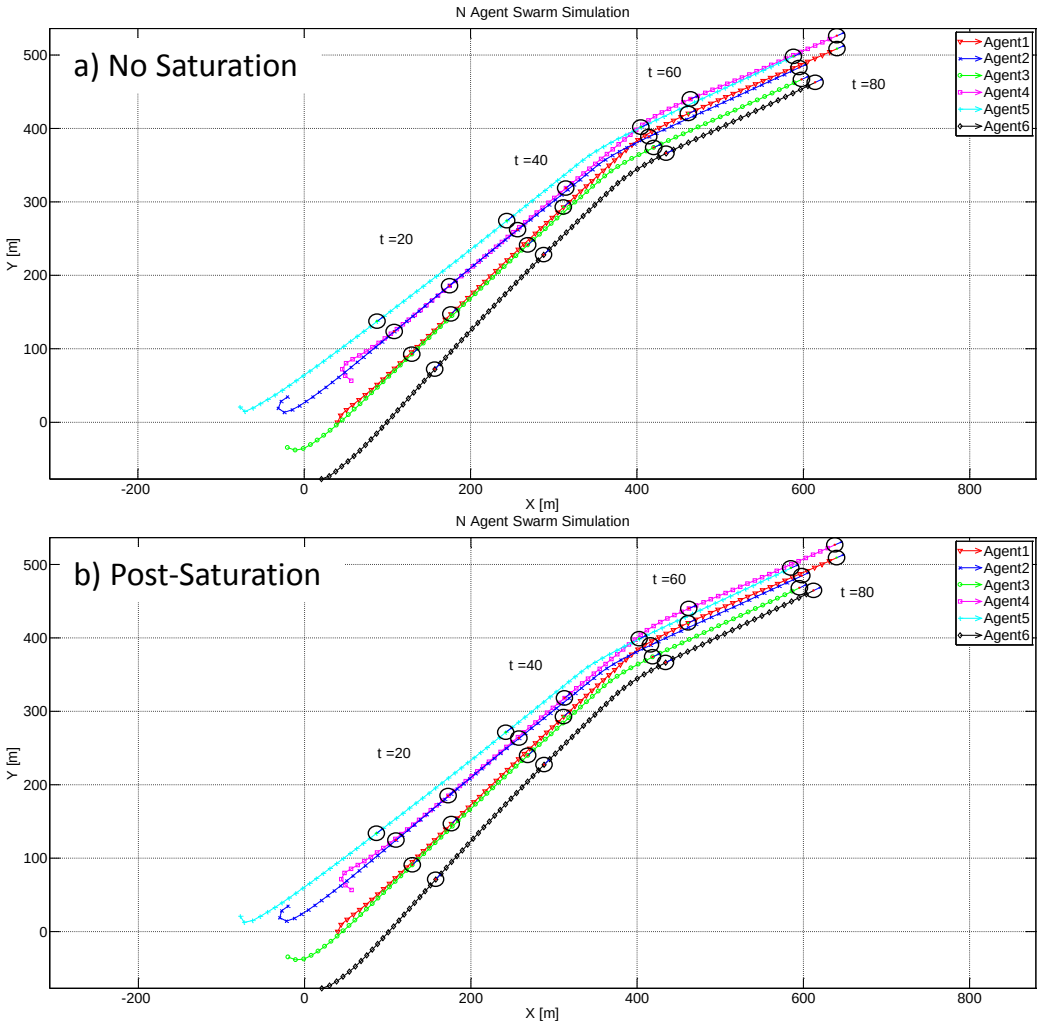

Figure 65 Comparison of swarm trajectories (post-saturation).

Figure 66 shows the first 10 seconds of the acceleration control inputs of all agents for the case without saturation and with post-saturation. The overall behavior of the two sets of controls is similar, but it can be seen that the control effort is reduced in the post-saturation case. The informed agents (1 and 5) are the only agents to experience the saturation cutoff. The follower agents see miniscule changes in control effort.

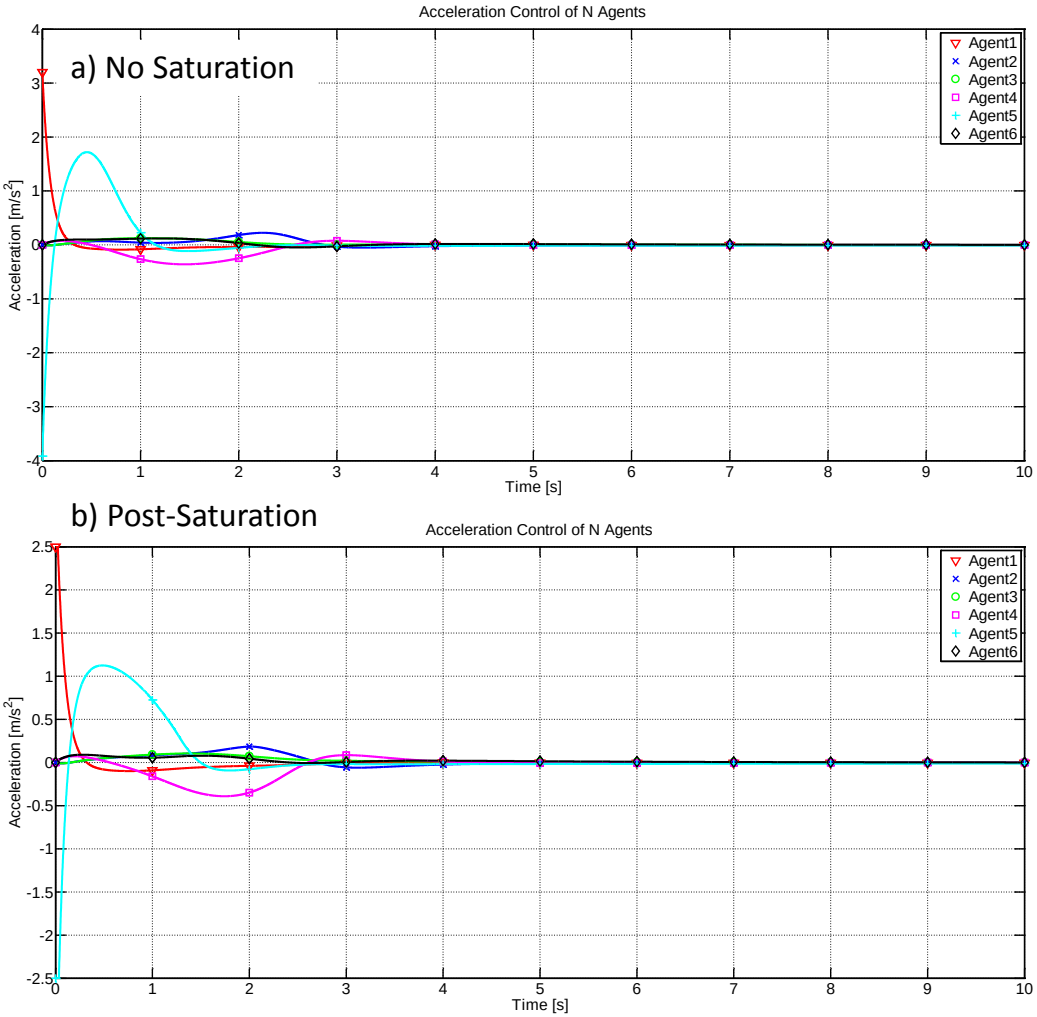

Figure 66 Comparison of first 10 sec of acceleration control (post-saturation).

Simulation 2: Effects of Pre-Saturation

Figure 67 compares the trajectories of a swarm without saturation and a swarm with pre-saturation. Unlike the post-saturation case, the change in performance using the pre-saturation scheme is significant. The swarm that uses pre-saturation actually aggregates faster than the swarm without saturation. In addition, the second swarm travels slightly farther than the first swarm in the same timeframe. This indicates an improvement in performance using the pre-saturation scheme.

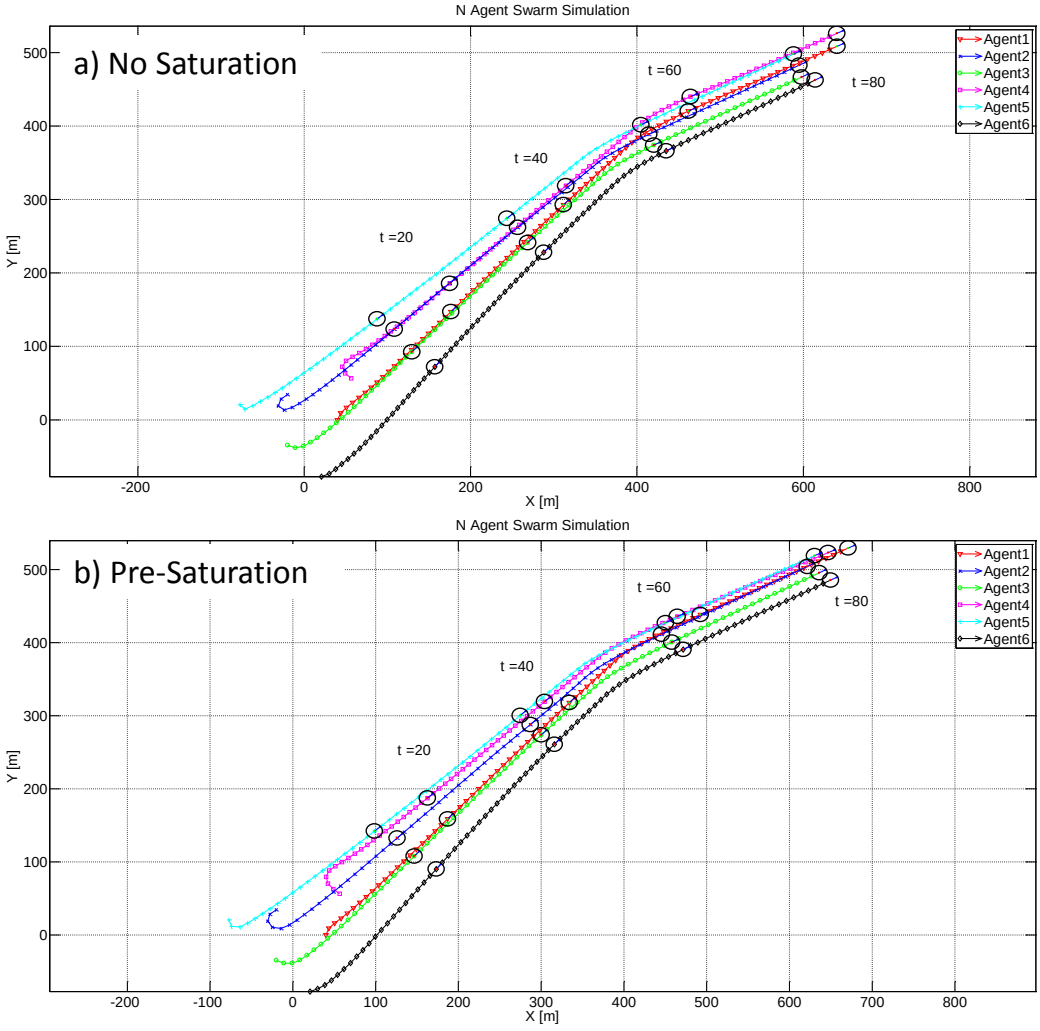

Figure 67 Comparison of swarm trajectories (pre-saturation).

Figure 68 compares the first 10 seconds of the acceleration control inputs of all agents from the two swarms. Two interesting observations can be made by looking at the pre-saturation control inputs. First, none of the controls ever approach or exceed the saturation limit. Second, suppression of the control magnitude comes with an increase of control effort over time. This phenomenon is not localized to the start of the simulation, but actually occurs throughout, as seen in Figure 69.

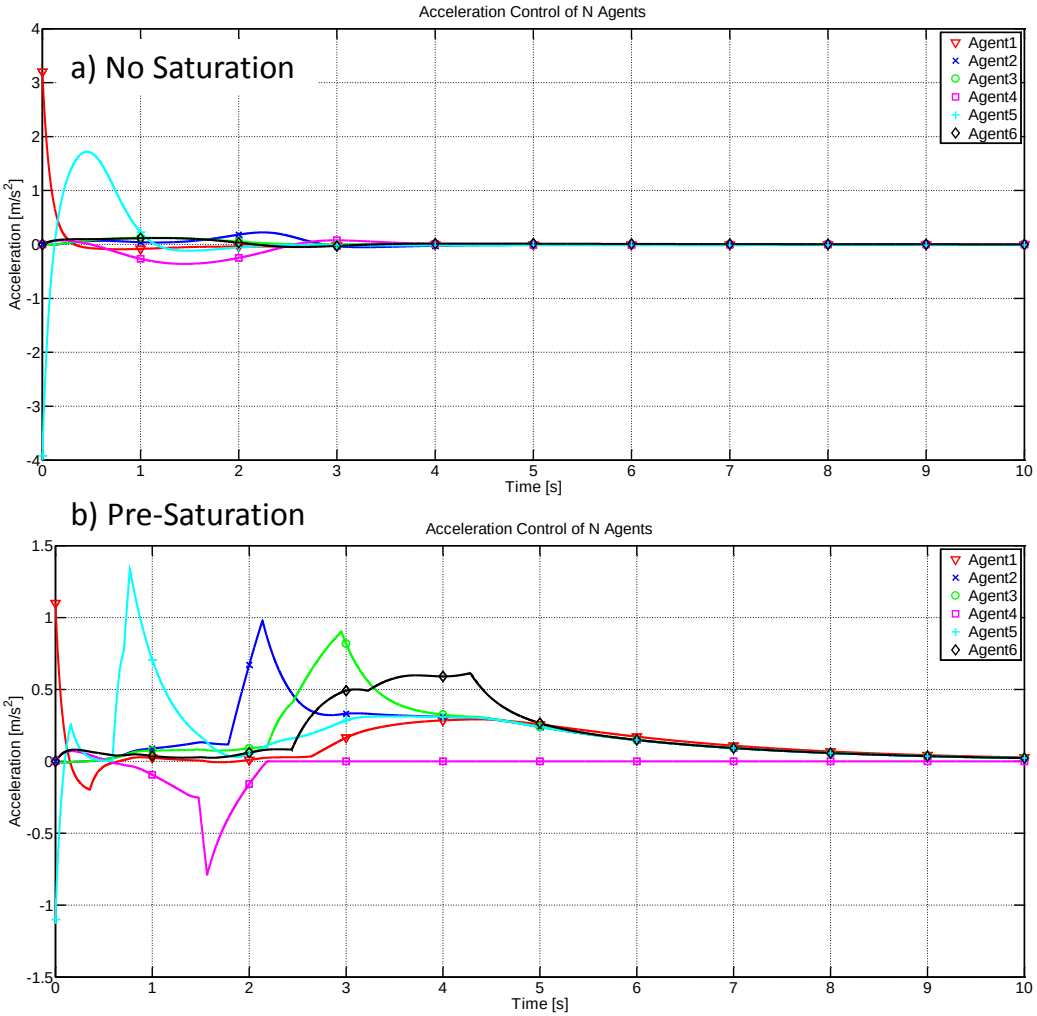

Figure 68 Comparison of first 10 sec of acceleration control (pre-saturation).

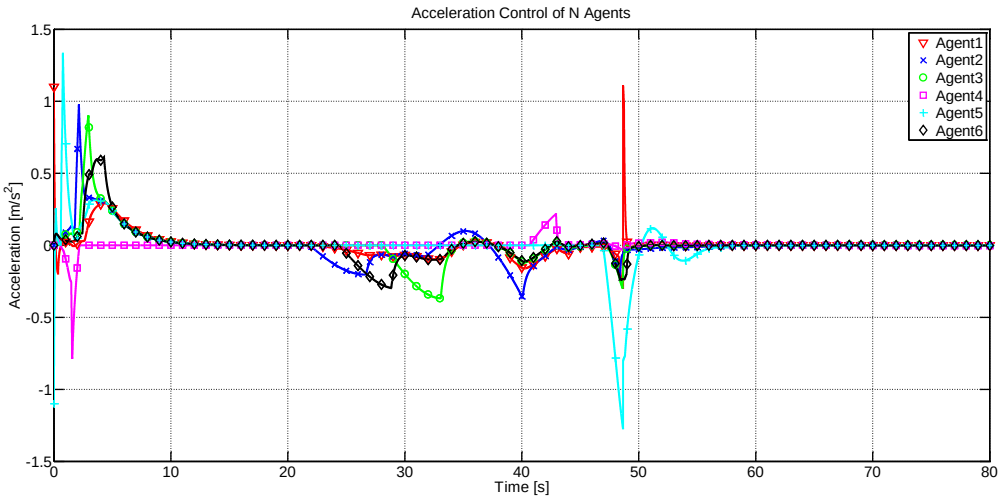

Figure 69 Acceleration control for 6 agents using the pre-saturation scheme.

The behavior of the acceleration controls using pre-saturation is quite interesting and suggests that control effort is not necessarily reduced, but is actually spread out more evenly over time compared with a case without saturation. The yaw control inputs do not experience this to the same degree. This is likely because the acceleration control has artificial damping and affects second-order dynamics as opposed to first-order dynamics. Figure 70 shows the yaw controls for the swarm with pre-saturation.

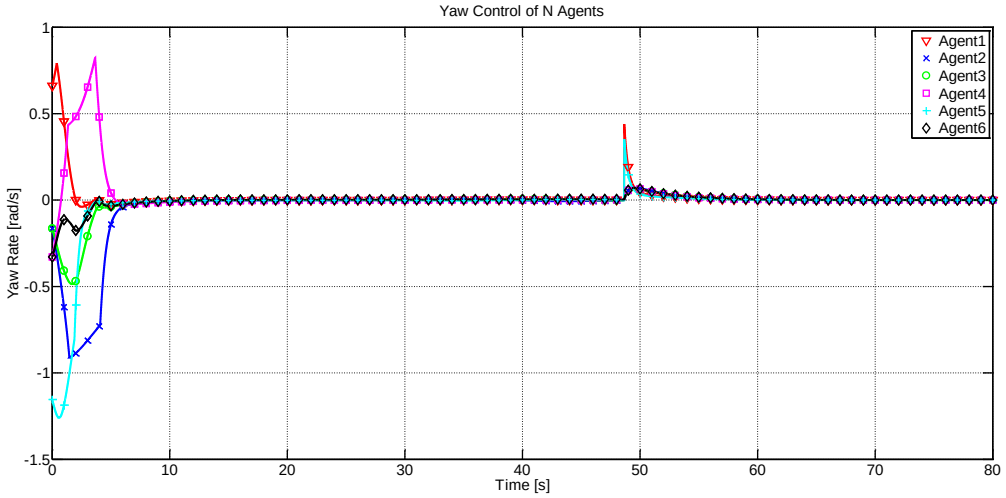

Figure 70 Yaw control for 6 agents using the pre-saturation scheme.

Figure 71 shows the acceleration control law components for agent 1 (informed). This figure helps explain the phenomena described above. There is obviously a high level of suppression in each component. All components are attempting to minimize their respective energy quantity, but they all have a limited amount of control effort they can use at any given moment. The limit creates an interesting case in the informed agent where the objective term is always active and steady. This creates a condition where the other terms are attempting to fight the objective term and end up exhibiting similar behavior. The small variations in each function are mostly attributed to the dynamic reallocation algorithm described in the previous chapter.

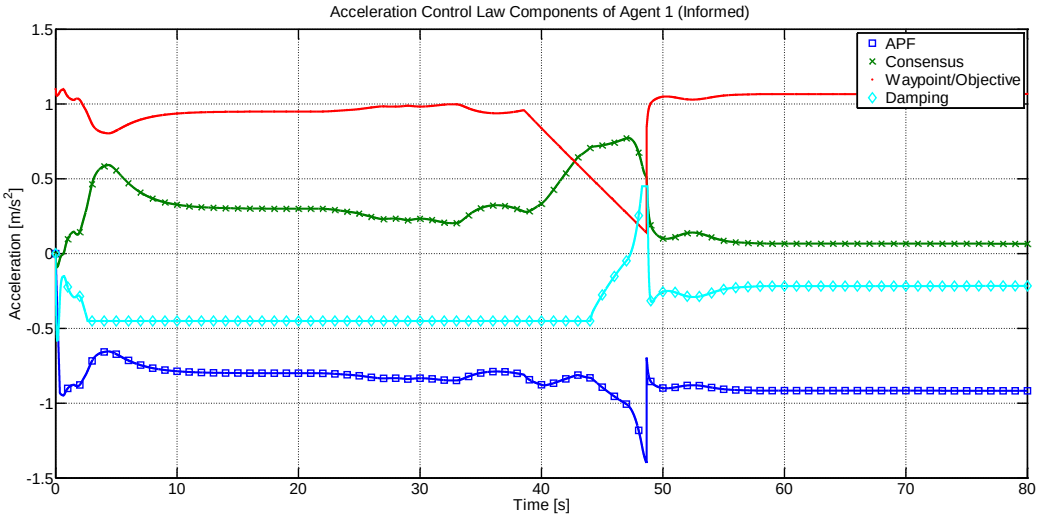

Figure 71 Components of acceleration control input of agent 1 (informed).

Figure 72 shows the acceleration control law components for agent 2 (follower). This figure exhibits a more interesting behavior. Without the objective term, the other functions are allowed to show more dynamic behavior. The APF and damping terms both get some of the control effort that is reserved for the objective term. Since the consensus term is processed first in this scheme, its percentage of control effort is fixed for all time. There are a few instances when the consensus and damping terms are steady and yield minimal control effort; there are other instances when they are more dynamic which creates a high level of control effort. By the end of the simulation, the consensus term is nearly zero, and the APF and damping terms are canceling each other out. This means that speed consensus is achieved, and the APF term is attempting to repel from some nearby agents. However, the artificial damping and consensus terms are negating the repulsion forces.

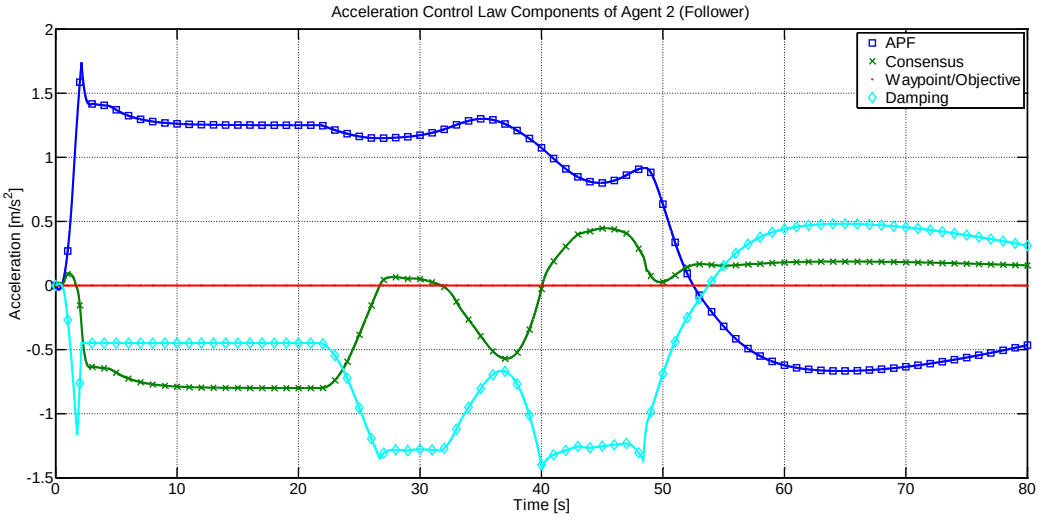

Figure 72 Components of acceleration control input of agent 2 (follower).

Simulation 3: Effects of Progressive Saturation

Figure 73 compares the trajectories of a swarm without saturation and one using the progressive saturation scheme. The change in performance is minimal between the two simulations. The comparison is similar to the post-saturation case.

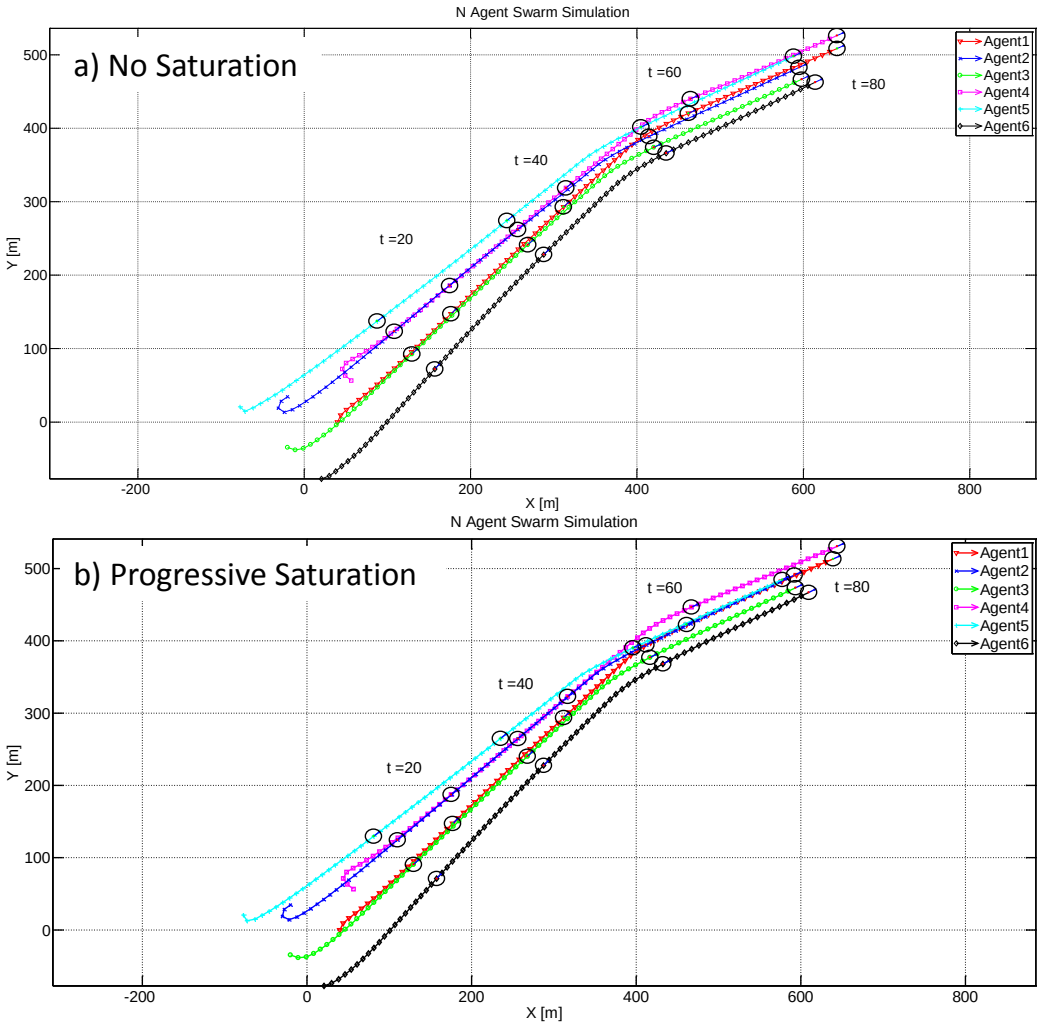

Figure 73 Comparison of swarm trajectories (progressive saturation).

Figure 74 compares the first 10 seconds of the acceleration control inputs of all agents from the two swarms. The control effort is suppressed, but the change is quite different than was seen for the post-saturation case. This means that the concept of progressively saturating the control input does produce a different result compared to post-saturation.

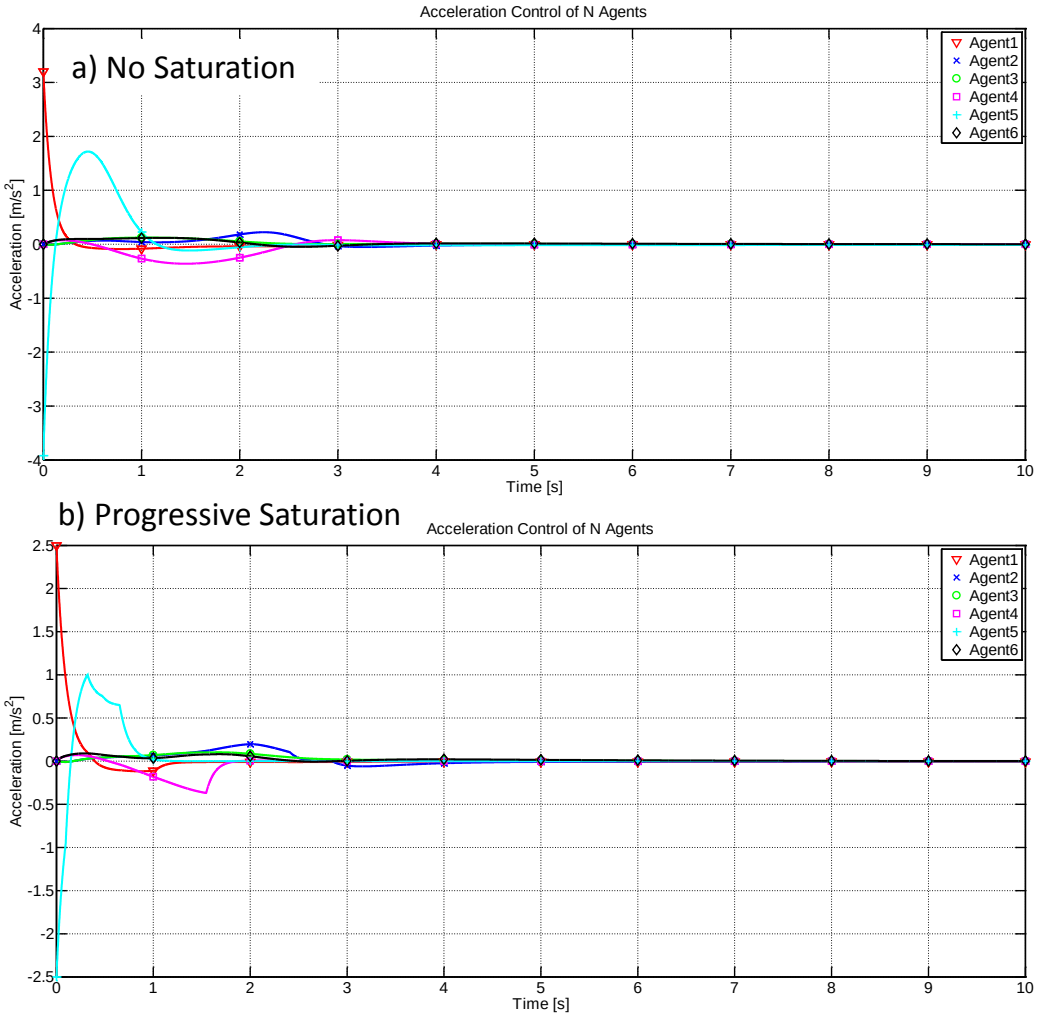

Figure 74 Comparison of first 10 sec of acceleration control (progressive saturation).

Page Intentionally Left Blank

Chapter 6
Conclusions and Future Work

Discussion and Conclusions

In this thesis, the modelling and control of swarm systems comprised of fixed-wing UAVs was discussed. A two degree of freedom control scheme was assumed in which the swarm control is handled by a high level controller, and the aircraft control task is handled by a low level flight controller. The flight controller must follow the trajectory given by the swarm controller. In order to maximize effectiveness, it is desirable to use a model that accurately describes the type of vehicle being used. Fixed-wing aircrafts have very specific motion constraints that require an attitude model to accurately capture them. In addition, the swarm controller should respect the aircrafts performance limitations.

In Chapter 3, the direction cosine matrix (DCM) attitude model was considered and simplified to yield two models that can be used for swarm control. The first model is a two-dimensional kinematic model with acceleration and yaw rate control. The second model is a three-dimensional kinematic model with acceleration, pitch rate, and yaw rate controls. These models are well suited for describing fixed-wing UAVs because of their nonholonomic constraints. In Chapter 4, a control scheme inspired by the work of Justh and Krishnaprasad [22], [58] was discussed. A variation of the control laws that they proposed was derived in this document using Lyapunov stability theory and a gradient controller design. Moreover, the control law was augmented to include artificial damping, objective/target potential functions, and dependence on the dynamic network topology. Lyapunov stability analysis was used to prove that the controller is locally stable with appropriate choice of control gains. Only the two-dimensional case was proven, but the three-dimensional case has a similar form to the two-dimensional case, so

the analysis is essentially the same. In addition, three forms of artificial control saturation were presented that can be used to impose aircraft performance limits on the controller.

In Chapter 5, several simulations were presented that tested the two models described in Chapter 3. The simulations are broken into two phases; the first phase presents the two-dimensional simulations and the second phase presents the three-dimensional simulations. Evaluation criteria were defined to quantify the level of success of each simulation. The initial conditions were designed to a represent an initial configuration of fixed-wing UAVs that is plausible and structured. The initial configuration proposed here places all agents on concentric circles centered at the origin with their tangent vectors directed tangent to the circles. In the three-dimensional simulation, all agents are at different altitudes. The objective given to all informed agents is to sequentially rendezvous at a set of waypoints.

The phase 1 simulations presented three (3) test cases of the two-dimensional system. The first test case demonstrated a small six (6) agent swarm with full connectivity. The purpose of this test case is to demonstrate and prove the effectiveness of the proposed control scheme using a simple system. This is an ideal size of an agent's neighborhood in a much larger swarm, because it is the maximum number of agents that can surround an agent if the desired separation distance is the same for all agents. The results of this simulation met all but the separation distance error criterion. This can be explained by the conflicting nature of the control law components and the design of the APF. It was shown that agent-to-agent potential and the consensus term in the yaw control law tend to fight each other. Once heading consensus is achieved, the consensus term fights any other term that tries to steer the agent. In addition, the APF is designed to have a single separation distance that makes it zero. If a deadband is added to the potential function, then the error may be eliminated.

The second two simulations in phase 1 demonstrated a larger fifteen (15) agent swarm with dynamic connectivity. The second simulation uses the same parameters as the first simulation, but the sensing radius is smaller. Due to the size of the swarm and the

chosen sensing radii, not all agents can sense each other. These simulations tested the controller's ability to operate with a dynamic network topology. Most criteria were satisfied in both simulations, but separation distance error was not satisfied. This is due to the same reasons given for the small swarm simulation. Another issue noticed in these simulations was the discontinuous behavior of the control inputs due to the dynamic network topology. The effect was more significant in the acceleration control input. As one would expect, the number of discontinuities increases as the sensing radius decreases. To fix this issue, the control scheme or the way the network topology is synthesized need to be changed. Some possible modifications will be noted in the following section.

The phase 2 simulations presented two (2) test cases of the three-dimensional system. The initial conditions in the three-dimensional case are more complex than the two-dimensional case due to the additional altitude coordinate. The starting altitude and radii relative to the origin are assigned to create a more imperfect formation compared with the two-dimensional case. This is done to make the initial conditions more realistic. The first test case demonstrates a small five (5) agent swarm with full connectivity. The purpose of this test case is similar to the first phase 1 simulation. An interesting result from this simulation is that all criteria, including the separation distance error criterion, was satisfied. This is believed to be attributed to the additional degree of freedom given by the altitude coordinate as well as the small size of the swarm.

The second test case in phase 2 demonstrates a large forty-five (45) agent swarm with dynamic connectivity. This test case really pushed the limits of the control scheme because it must overcome extreme initial conditions, a large number of agents, and a highly dynamic network topology. A few results from this simulation are worth noting. First, the shape of the swarm eventually reaches an elongated box-like shape. Also, the separation distance error is the highest seen in any simulation presented here. Based on these results, it is believed that the large sensing radius compared to separation distance is causing a *squeezing* effect in which the outer-most agents are being attracted to some of the agents on the interior and cannot be matched by the repulsion forces. This problem may be alleviated by adding swarm shape controls and changing the way the network topology is synthesized. The other issue noticed in the simulation was the discontinuities

in the acceleration control input. Due to the large number of agents, the number of discontinuities is much greater than was seen in the two-dimensional simulations. Upon looking at the individual control law components, it was noticed that the discontinuities are caused by the APF component, but are quickly suppressed by the artificial damping term. In this simulation, the damping gain was set very high to see how it would affect the discontinuities. The damping term is a reactive term since it is dependent on speed, so it will never be able to eliminate the discontinuities. A change to the consensus gain may actually help to suppress the discontinuities even more since it also depends on the dynamic network topology. Regardless, changes to the control gains alone will not eliminate all discontinuities. A change to the control scheme will be the most effective option.

In the third phase of simulations, a few kinds of artificial control saturation were tested. The first method, here called post-saturation, is the traditional form of saturation in engineering fields. It applies a saturation limit to the final value of the control law at each time step. The simulation showed minimal change to both performance and control effort. The amount of saturation in this simulation was minimal, but it does have an effect on the trajectory of the swarm. It is expected that a lower saturation limit or higher control gains will produce a case that yields more saturation and a more pronounced effect on overall performance.

The second form of artificial saturation presents the most interesting case. This scheme is called pre-saturation and work by applying the saturation limit to each part of the control law before it is added together. Each term is given a percentage of the overall saturation limit. In this thesis, the percentage is dynamically reallocated on a priority basis. The results of the simulation using this scheme indicate that the control magnitude actually sees an overall reduction while the amount of control effort is distributed over time. All other simulations show a large amount of control effort in a very short time period after the start of the simulation and after the waypoints is passed. The swarm with the pre-saturation scheme exhibits more control activity over time. The control inputs are subjected to a kind of suppression, but the effect on control effort appears to be more of

redistribution rather than a reduction. In addition, the suppression actually caused an improvement in performance over an identical case with no saturation.

The final form of artificial saturation is called progressive saturation. It involves applying the saturation limit as the control law terms are added together. For most cases this is the same as post-saturation, but under certain circumstances it will create additional suppression of the controls. The simulations produced very similar results to the post-saturation case, but there is a noticeable difference in the acceleration controls that was induced by the progressive saturation method.

Based on all of the results presented in Chapter 5, it can be said that the proposed control scheme successfully minimizes the energy state of the swarm system defined by the system Lyapunov function, though it may not do so in the most efficient and optimal manner. Several issues were noted that arise when the number of agents is high and the network topology becomes dynamic. These issues include discontinuities in the control inputs and errors in the separation distances. Besides these small performance issues, the scheme is effective and provides a good basis for design and research. The pre-saturation method has been demonstrated to suppress the control inputs while improving some aspects of performance and presents as an interesting topic for future research. In the next section, additional ideas will be presented that could be the focus of future research.

Future Work

Control Scheme Modifications

In this thesis, the control scheme presented is designed to achieve flocking behavior and perform target tracking, but there is very little control over the size and shape of the swarm. Each agent tries to achieve a certain separation distance with its neighboring agents, but it was demonstrated that the separation distance is not always achieved, and this type of design provides little control over the swarm shape. A modification that can be made to the control scheme in a future work is using a set of shape variables to better control the size and shape of the swarm. An interesting idea was presented in [74] where shape variables are represented by a set of inertial moments and are called *swarm*

formation statistics. Their control is given by a nonlinear gradient controller with an estimator for the swarm formation statistics. A scheme such as this can be used to augment the APF design presented here so that the swarm shape can be controlled.

The controls used in the three-dimensional case are the same used in the two-dimensional case. The pitch control law mirrors the yaw control law in every aspect. This is an effective design, but it may be desired in some applications to have more control on the attitude coordinates and altitude. For instance, it might be desirable to penalize changes in the pitch angle itself so that the agent's rate of climb or descent is not too high. In addition, by controlling just pitch and yaw, the roll angle is not restricted at all in the current scheme. If this controller is to be used with a low level flight controller, then this is acceptable because the flight controller will accomplish roll stabilization, but if it were to be used as a single level controller then roll stabilization will have to be considered. Currently, there is no control of an agent's altitude, but some applications may require all agents to eventually reach a configuration where they are all at the same altitude. An altitude consensus law, similar to the speed consensus law, can be devised that can drive all agents to achieve the same altitude.

Another modifications to the existing control scheme that can be considered is the morphing potential function idea presented in [67]. This is a novel idea that can be implemented, and perhaps augmented, to make the repulsion potential more capable of reacting to different scenarios in practice. A variation of morphing potentials can be used for both agent-to-agent potentials and agent-to-obstacle potentials. Obstacles were not considered in this work for simplicity, but they are an important element of swarm coordination problems. Future work with the current scheme should include both static and dynamic obstacles and should investigate the use of morphing potential functions.

The control scheme used in this thesis is basically a superposition of gradient controllers based on a Lyapunov function that describes the energy state of a swarm in terms of relative coordinates. Other schemes exist that can use the same Lyapunov function in a different way and are worth investigating. One such scheme is sliding mode control where a sliding surface is designed to track a particular state or output. By going through

the design process, a set of conditions can be derived that help guide the design of a control gain for a switching controller based on the sliding surface. The resulting controller is robust to modelling uncertainties, but can still satisfy the system objectives as long as the sliding surface can be reached in finite time. For a more detailed discussion of sliding mode control, refer to [62].

Another scheme that is worth considering is nonlinear model predictive control (MPC). In nonlinear MPC, an optimal control problem is solved repetitively with a receding horizon. This method can be slow and very computationally taxing, but it has several benefits that are useful for swarm control problems. Since it is an optimization technique, limits on the control inputs and states can be imposed, which is a very attractive feature for controlling a vehicle with specific performance l imitations. In addition, the predictive nature of the controller can be used to reduce the risk of issues such as swarm fragmentation. Because it is typically a slow control method, nonlinear MPC would be most effective as a high level swarm controller.

Network Topology Synthesis

The way in which the network topology is synthesized should be researched in a future work. This is the primary cause of the discontinuities in the control inputs. In addition, the control gains still require some fine tuning, despite the gain function that adjusts the final gain based on swarm and neighborhood size. A scheme should be devised that keeps the number of neighboring agents considered for control at a minimum. One possible solution is to use a geometric construct such as Voronoi diagrams to describe the network topology. Voronoi diagrams were successfully used by Chiew [13] along with proximity-based sensing. This is an excellent tool for two-dimensional systems, but it is very difficult to extend it to three-dimensional systems.

A simpler solution would be to programmatically limit the number of neighboring agents considered. This could be done by reducing the sensing radius in the software when a certain number of agents are in sensing range. The benefit of this idea is that as the swarm aggregates, the effective sensing radius will shrink until it is just large enough to sense only the agents in the immediate vicinity. This will act to minimize the complexity

of the controller over time. There will also need to be software included to reverse the process so that any changes to the swarm shape due to mission updates or obstacles do not force the swarm to become fragmented. This idea has not been tested, but it sounds promising and should be considered in future work.

Another idea that was briefly tested is to keep a constant sensing radius and program each agent to only consider the nearest six (6) agents. After an agent determines the relative positions of the sensible agents, it sorts them from smallest to largest and picks the six (6) agents with the smallest relative positions. This idea seems like it would be plausible, but after initial testing, it was found that it generates Laplacian matrices with imaginary eigenvalues. Typically, the Laplacian is a symmetric, positive semi-definite matrix, but in this case the Laplacian is non-symmetric. This matrix does not have the same properties as the symmetric Laplacian; therefore, some of the claims made in the stability analysis break down. The resulting network topology actually represents a hybrid network that contains both unidirectional and bidirectional edges. The reason this hybrid network is created is that some agents will be part of another's minimal set, but the opposite may not be true. This idea has not been tested in an actual swarm simulation, and this thesis does not focus strongly on graph theory. An interesting research topic might be swarm control using a hybrid graph such as the one created by the idea presented above. A deeper study into graph theory would be required.

Some UAVs will not have the sensing capability for full omnidirectional sensing of its surroundings. Rather, a UAV may only have forward sensing capability or a set of blind spots where sensing is not possible. If the agent's cannot communicate their states over a communications network, then the network topology must be synthesized so that it considers an agent's line-of-sight restriction. Additional control laws may need to be added to steer an agent outside of another agent's blind spots.

If a communication network does exist, then another possible modification is to implement a hierarchy to the swarm. A hierarchy can be used to simplify the communication effort of all agents so that mission specific information can be transferred effectively to all necessary agents. For example, a squad hierarchy could be imposed

where one agent is designated a squad leader, another agent is designated a communications relay, and the remaining agents are followers. The entire swarm would consist of multiple squads where squad leaders can communicate with each other and the ground station while the communication relays pass information from the leader to all follower agents in the squad. In addition, the communication relays can pass information from the follower agents back to the squad leaders, and if necessary, the ground station. Such a configuration may be useful in mobile sensor networks where a large amount of data must be transferred from the agents back to a centralized unit for storage or analysis. If a certain amount of communication delay is tolerable and the data can be time synchronized, then this method can be a plausible way to streamline data collection and synthesis.

Simulation Modifications and Experiments

Several modifications can be made to the simulation software to increase its fidelity and efficiency. Obstacle and barrier avoidance capabilities should be added to more accurately model the environment. The option to add or remove agents in the middle of a simulation should be added to test the controller's ability to adapt. To improve the fidelity of the control scheme, a flight controller and UAV model should be added to simulate how the complete control system performs. Also, noise and time delays can be added to push the limits of the control system even further. The computational efficiency of the simulation software may need to be improved, especially if more capabilities are added. In addition, a more capable computer may need to be used to run simulations. As point of reference, the forty-five (45) agent simulation used a simulation time of 120 seconds and a time step of 2 milliseconds and it required roughly five (5) hours to run. The computer used was a Dell Inspiron 7520 with 8 GB of RAM, an Intel i7-3632QM processor, and Windows 7 Professional 64-bit. Nearly all the RAM was being used by the simulation software, and all eight (8) cores of the processor were working. Either a more powerful computer is needed, or the MATLAB code needs to be modified to more efficiently use resources.

The three-dimensional simulation software could be improved by adding the capability for more detailed visualization and animation. A tool such as the Simulink 3D Animation toolbox can be used to visualize the swarm system in a virtual environment. Sets of virtual cameras could be included to pick up various views of the swarm.

Finally, all of the ideas considered that pass the simulation tests should be tested on an actual experimental platform. The initial platform will likely consist of many small, inexpensive quadrotors and a motion capture system. Eventually, it is desired to have an outdoor test platform using small scale fixed-wing UAVs in a controlled environment. To achieve this goal, a great deal of research must go into state-of-the-art UAV technology, especially sensing and communication technology.

References

[1] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd ed. Upper Saddle River, NJ: Pearson Education Inc., 2010.

[2] A. R. Paygani and M. Haeri, "On the Number of Informed Agents and Their Initial Positions in a Free Flocking," *J. Dyn. Syst. Control*, vol. 135, no. 9, pp. 1–7, 2013.

[3] V. Gazi and K. M. Passino, *Swarm Stability and Optimization*, 1st ed. Springer-Verlag Berlin Heidelberg, 2011.

[4] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and Cooperation in Networked Multi-Agent Systems," *Proc. IEEE*, vol. 95, no. 1, pp. 215–233, 2007.

[5] V. Gazi and K. M. Passino, "A Class of Attraction/Repulsion Functions for Stable Swarm Aggregations," *Int. J. Control*, vol. 77, no. 18, pp. 1567–1579, 2004.

[6] N. E. Leonard and E. Fiorelli, "Virtual Leaders, Artificial Potentials and Coordinated Control of Groups," in *Proceedings of the 40th IEEE Conference on Decision and Control*, 2001, pp. 2968–2973.

[7] G. Roussos, D. V. Dimarogonas, and K. J. Kyriakopoulos, "3D Navigation and Collision Avoidance for Nonholonomic Aircraft-like Vehicles," *Int. J. Adapt. Control Signal Process.*, vol. 24, no. 8, pp. 900–920, 2010.

[8] H. G. Tanner and A. Boddu, "Multiagent Navigation Functions Revisited," *IEEE Trans. Robot.*, vol. 28, no. 6, pp. 1346–1359, 2012.

[9] J. Yao, R. Ordóñez, and V. Gazi, “Swarm Tracking Using Artificial Potentials and Sliding Mode Control,” *J. Dyn. Syst. Meas. Control*, vol. 129, no. September, p. 749, 2007.

[10] K. Han, J. Lee, and Y. Kim, “Unmanned Aerial Vehicle Swarm Control using Potential Functions and Sliding Mode Control,” *Proc. Inst. Mech. Eng. Part G J. Aerosp. Eng.*, vol. 222, no. 6, pp. 721–730, 2008.

[11] B. Ren, S. S. Ge, T. H. Lee, and M. Krstic, “Region Tracking Control for Multi-Agent Systems with High-Order Dynamics,” in *American Control Conference (ACC)*, 2013, pp. 1266–1271.

[12] X. Cai and M. De Queiroz, “Rigidity-Based Stabilization of Multi-Agent Formations,” *J. Dyn. Syst. Meas. Control*, vol. 136, no. 1, pp. 1–7, 2014.

[13] S. H. Chiew, W. Zhao, and T. H. Go, “Swarming Coordination with Robust Control Lyapunov Function Approach,” *J. Intell. Robot. Syst.*, vol. 72, 2013.

[14] J. Wang and M. Xin, “Flocking of Multi-Agent Systems Using a Unified Optimal Control Approach,” *J. Dyn. Syst. Meas. Control*, vol. 135, no. 6, pp. 1–11, 2013.

[15] J. C. Barca and Y. A. Sekercioglu, “Swarm Robotics Reviewed,” *Robotica*, vol. 31, no. 3, pp. 345–359, 2012.

[16] J. González-Sierra, E. Aranda-Bricaire, and E. G. Hernandez-Martinez, “Formation Tracking with Orientation Convergence for Groups of Unicycles,” *Int. J. Adv. Robot. Syst.*, vol. 10, no. 180, pp. 1–8, 2013.

[17] D. Mellinger, M. Shomin, N. Michael, and V. Kumar, “Cooperative Grasping and Transport using Multiple Quadrotors,” in *Distributed Autonomous Robotic Systems*, vol. 83, Springer Berlin Heidelberg, 2013, pp. 545–558.

[18] D. Smalley, “The Future Is Now: Navy’s Autonomous Swarmboats Can Overwhelm Adversaries,” 2014. [Online]. Available: http://www.onr.navy.mil/Media-Center/Press-Releases/2014/autonomous-swarm-boat-unmanned-caracas.aspx. [Accessed: 06-Oct-2014].

[19] T. Huntsberger, H. Aghazarian, T. Estlin, and D. Gaines, “Control Architecture for Robotic Agent Command and Sensing,” *NASA Tech Briefs*, 2008. [Online]. Available: http://www.techbriefs.com/legal-footer-127/3251-npo-43635. [Accessed: 07-Oct-2014].

[20] A. Matlock, R. Holsapple, C. Schumacher, J. Hansen, and A. Girard, “Cooperative Defensive Surveillance using Unmanned Aerial Vehicles,” in *American Control Conference (ACC)*, 2009, pp. 2612–2617.

[21] E. Raboin, P. Švec, D. S. Nau, and S. K. Gupta, “Model-Predictive Asset Guarding by Team of Autonomous Surface Vehicles in Environment with Civilian Boats,” *Auton. Robots*, vol. 38, no. 3, pp. 261–282, 2014.

[22] E. W. Justh and P. S. Krishnaprasad, “A Simple Control Law for UAV Formation Flying,” College Park, MD, 2002.

[23] C. W. Reynolds, “Flocks, Herds and Schools: A Distributed Behavioral Model,” *ACM SIGGRAPH Comput. Graph.*, vol. 21, pp. 25–34, 1987.

[24] A. Jadbabaie, J. Lin, and A. S. Morse, “Coordination of Groups of Mobile Autonomous Agents using Nearest Neighbor Rules,” *IEEE Trans. Automat. Contr.*, vol. 48, no. 6, pp. 988–1001, 2003.

[25] Y. Gao, L. Wang, and Y. Jia, “Consensus of Multiple Second-Order Agents without Velocity Measurements,” in *American Control Conference (ACC)*, 2009, pp. 4464–4469.

[26] G. Xie, L. Wang, and Y. Jia, “Output Agreement in High-Dimensional Multi-Agent Systems,” in *American Control Conference (ACC)*, 2009, pp. 2243–2248.

[27] F. Jiang, L. Wang, G. Xie, Z. Ji, and Y. Jia, “On the Controllability of Multiple Dynamic Agents with Fixed Topology,” in *American Control Conference (ACC)*, 2009, pp. 5665–5670.

[28] H. Cai and J. Huang, “The Leader-Following Attitude Control of Multiple Rigid Spacecraft Systems,” in *American Control Conference (ACC)*, 2013, pp. 824–829.

[29] G. Wen, Z. Duan, W. Yu, and G. Chen, “Consensus of Second-Order Multi-Agent Systems with Delayed Nonlinear Dynamics and Intermittent Communications,” *Int. J. Control*, vol. 86, no. 2, pp. 322–331, 2013.

[30] J. Hu and Y. S. Lin, “Consensus Control for Multi-Agent Systems with Double-Integrator Dynamics and Time Delays,” *IET Control Theory Appl.*, vol. 4, no. 1, pp. 109–118, 2010.

[31] Z. Li, Z. Duan, and L. Huang, “Leader-follower Consensus of Multi-Agent Systems,” in *American Control Conference (ACC)*, 2009, pp. 3256–3261.

[32] R. Sepulchre and D. Paley, “Collective Motion and Oscillator Synchronization,” in *Cooperative control*, vol. 309, V. Kumar, N. Leonard, and A. S. Morse, Eds. Springer Berlin Heidelberg, 2005, pp. 189–205.

[33] Y. Kuramoto, *Chemical Oscillations, Waves, and Turbulence*, 1st ed., vol. 19. Springer Berlin Heidelberg, 1984.

[34] D. A. Sierra, P. McCullough, N. Olgac, and E. Adams, “Swarm Coordination Under Conflict and Use of Enhanced Lyapunov Control,” *J. Dyn. Syst. Meas. Control*, vol. 133, no. 3, pp. 1–8, 2011.

[35] S. Coogan, M. Arcak, and M. Egerstedt, “Scaling the Size of a Multiagent Formation via Distributed Feedback,” in *IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, 2011, pp. 994–999.

[36] A. De Luca and G. Oriolo, *Modelling and Control of Nonholonomic Mechanical Systems*, 1st ed., vol. 360. Vienna: Springer Vienna, 1995.

[37] Y. L. Chuang, Y. R. Huang, M. R. D'Orsogna, and A. L. Bertozzi, "Multi-Vehicle Flocking: Scalability of Cooperative Control Algorithms using Pairwise Potentials," in *IEEE International Conference on Robotics and Automation*, 2007, pp. 2292–2299.

[38] H. Rastgoftar and S. Jayasuriya, "An Alignment Strategy for Evolution of Multi-Agent Systems," *J. Dyn. Syst. Meas. Control*, vol. 137, no. 2, pp. 1–10, 2015.

[39] D. E. Chang, S. C. Shadden, J. E. Marsden, and R. Olfati-Saber, "Collision Avoidance for Multiple Agent Systems," in *Proceedings of the 42nd IEEE International Conference on Decision and Control*, 2003, vol. 1, pp. 539–543.

[40] P. Ogren, E. Fiorelli, and N. E. Leonard, "Cooperative Control of Mobile Sensor Networks: Adaptive Gradient Climbing in a Distributed Environment," *IEEE Trans. Autom. Control*, vol. 49, no. 8, pp. 1292–1302, 2004.

[41] R. Olfati-Saber, "Flocking for Multi-Agent Dynamic Systems: Algorithms and Theory," *IEEE Trans. Automat. Contr.*, vol. 51, no. 3, pp. 401–420, 2006.

[42] M. Niccolini, L. Pollini, and M. Innocenti, "Decentralized Control of Swarms with Collision Avoidance Implications," in *Proceedings of the 17th World Congress*, 2008, vol. 17, pp. 16002–16007.

[43] D. J. Bennet and C. R. McInnes, "Verifiable Control of a Swarm of Unmanned Aerial Vehicles," *Proc. Inst. Mech. Eng. Part G J. Aerosp. Eng.*, vol. 223, no. 7, pp. 939–954, 2009.

[44] Y. Mao, L. Dou, H. Fang, and J. Chen, "Distributed Flocking of Second-Order Multi-Agent Systems with Global Connectivity Maintenance," in *American Control Conference (ACC)*, 2013, pp. 976–981.

[45] J. Maidens and M. Y. Li, "Global Lyapunov Functions and a Hierarchical Control Scheme for Networks of Robotic Agents," in *American Control Conference (ACC)*, 2013, pp. 4050–4055.

[46] H. G. Tanner, A. Jadbabaie, and G. J. Pappas, “Coordination of Multiple Autonomous Vehicles,” in *IEEE Mediterranean Conference on Control and Automation*, 2003, pp. 1–6.

[47] H. G. Tanner, A. Jadbabaie, and G. J. Pappas, “Flocking in Teams of Nonholonomic Agents,” in *Cooperative Control*, Springer Berlin Heidelberg, 2005, pp. 229 – 239.

[48] D. J. Klein and K. a. Morgansen, “Controlled Collective Motion for Trajectory Tracking,” in *American Control Conference (ACC)*, 2006, pp. 5269–5275.

[49] V. Gazi, B. Fidan, Y. S. Hanay, and M. I. Köksal, “Aggregation, Foraging, and Formation Control of Swarms with Non-Holonomic Agents using Potential Functions and Sliding Mode Techniques,” *Turkish J. Electr. Eng. Comput. Sci.*, vol. 15, no. 2, pp. 149–168, 2007.

[50] B. Lei, W. Li, and F. Zhang, “Flocking Algorithm for Multi-Robots Formation Control with a Target Steering Agent,” in *IEEE International Conference on Systems, Man and Cybernetics*, 2008, pp. 3536–3541.

[51] Q. Li and Z. P. Jiang, “Flocking of Decentralized Multi-Agent Systems with Application to Nonholonomic Multi-Robots,” in *Proceedings of the 17th World Congress*, 2008, vol. 17, pp. 9344–9349.

[52] V. Gazi, B. Fidan, and R. Ordóñez, “Tracking a Maneuvering Target with a Swarm of Non-holonomic Agents Using Artificial Potentials and Sliding Mode Control,” *Control Autom. 2008 16th Mediterr. Conf.*, pp. 1174–1179, 2008.

[53] R. Falconi, S. Gowal, J. Pugh, and A. Martinoli, “Graph-Based Distributed Control for Non-Holonomic Vehicles Engaged in a Reconfiguration Task using Local Positioning Information,” in *2009 Second International Conference on Robot Communication and Coordination (ROBOCOMM)*, 2009, pp. 1–6.

[54] R. Soukieh, I. Shames, and B. Fidan, “Obstacle Avoidance of Robotic Formations Based on Fluid Mechanical Modeling,” in *Proceedings of the European Control Conference*, 2009, pp. 3263–3268.

[55] V. Gazi, B. Fidan, R. Ordóñez, and M. İlter Köksal, “A Target Tracking Approach for Nonholonomic Agents Based on Artificial Potentials and Sliding Mode Control,” *J. Dyn. Syst. Meas. Control*, vol. 134, no. 6, pp. 1–13, 2012.

[56] Z. Li, J. Vanness, and D. E. Caballero, “Distributed Target Tracking and Collision Avoidance using Multiple Nonholonomic Robots with Uncertain Dynamics,” in *AIAA Guidance, Navigation, and Control Conference*, 2011, pp. 1–15.

[57] N. Dhananjay and R. Kristiansen, “Guidance Strategy for Gradient Search by Multiple UAVs,” in *AIAA Guidance, Navigation, and Control Conference*, 2012, pp. 1–11.

[58] E. W. Justh and P. S. Krishnaprasad, “Natural Frames and Interacting Particles in Three Dimensions,” in *Proceedings of the 44th IEEE Conference on Decision and Control, and the European Control Conference, CDC-ECC ’05*, 2005, pp. 2841–2846.

[59] D. S. Morgan and I. B. Schwartz, “Dynamic Coordinated Control Laws in Multiple Agent Models,” *Phys. Lett. A*, vol. 340, no. 1–4, pp. 121–131, 2005.

[60] E. W. Justh and P. S. Krishnaprasad, “Equilibria and Steering Laws for Planar Formations,” *Syst. Control Lett.*, vol. 52, no. 1, pp. 25–38, 2004.

[61] A. Sarlette, R. Sepulchre, and N. E. Leonard, “Autonomous Rigid Body Attitude Synchronization,” in *Proceedings of the 46th IEEE Conference on Decision and Control*, 2007, vol. 45, pp. 2566–2571.

[62] J. E. Slotine and W. Li, *Applied Nonlinear Control*. Englewood Cliffs, NJ: Prentice Hall, 1991.

[63] L. Grüne and J. Pannek, “Nonlinear Model Predictive Control,” in *Nonlinear Model Predictive Control*, 2011, pp. 67–85.

[64] R. A. Freeman and P. V. Kokotovic, *Robust Nonlinear Control Design*, 1st ed. Birkhäuser Basel, 1996.

[65] S. Goldenstein, M. Karavelas, D. Metaxas, L. Guibas, and A. Goswami, “Scalable Dynamical Systems for Multi-Agent Steering and Simulation,” in *IEEE International Conference on Robotics and Automation*, 2001, pp. 3973–3980.

[66] G. Roussos and K. J. Kyriakopoulos, “Decentralised Navigation and Collision Avoidance for Aircraft in 3D Space,” *IEEE Trans. Control Syst. Technol.*, vol. 20, no. 6, pp. 1622–1629, 2010.

[67] T. J. Stastny, G. A. Garcia, and S. S. Keshmiri, “Collision and Obstacle Avoidance in Unmanned Aerial Systems Using Morphing Potential Field Navigation and Nonlinear Model Predictive Control,” *J. Dyn. Syst. Meas. Control*, vol. 137, no. 1, pp. 1–10, 2015.

[68] F. Zhang, E. Fiorelli, and N. E. Leonard, “Exploring Scalar Fields using Multiple Sensor Platforms: Tracking Level Curves,” in *Proceedings of the 46th IEEE Conference on Decision and Control*, 2007, pp. 3579–3584.

[69] H. Schaub and J. L. Junkins, *Analytical Mechanics of Space Systems*, 2nd ed. Reston, VA: AIAA Inc., 2009.

[70] J. W. Jordan, “Direction Cosine Computational Error,” Cambridg, Mass., 1969.

[71] W. Boothby, *An Introduction to Differentiable Manifolds and Riemannian Geometry*, 2nd ed. Orlando, FL: Academic Press Inc., 1986.

[72] R. L. Bishop, “There is More Than One Way to Frame a Curve,” *Am. Math. Mon.*, vol. 82, no. 3, pp. 246–251, 1975.

[73] C. Godsil and G. Royle, *Algebraic Graph Theory*, vol. 207. Springer New York, 2001.

[74] R. a. Freeman, P. Y. P. Yang, and K. M. Lynch, “Distributed Estimation and Control of Swarm Formation Statistics,” in *American Control Conference (ACC)*, 2006.

Page Intentionally Left Blank

Appendix A
Support Material for Stability Analysis

In this section, the background work performed in the stability analysis is presented. First, the time derivative of the relative heading energy will be discussed. This derivative is given as:

$$\dot{V}_H = \left(\frac{\partial J_H}{\partial \vec{T}}\right)^T \dot{\vec{T}}$$

It was shown in the section entitled "Stability Analysis" in Chapter 5 that the result of this equation is a triple summation. First, the summation was evaluated when the third summation index is equal to the second index. The simplification in this case was straightforward. For the case when the two indices are not equal, the simplification is not as easy. The resulting triple summation is given by:

$$\left(\dot{V}_H\right)_{k\neq j} = -\sum_{i=1}^{N}\sum_{j\neq i}\sum_{\substack{k\in\mathcal{N}_i \\ k\neq j}}\left(k_{c2}\hat{T}_j\cdot\hat{T}_k - k_{p2}g(r_{ik})\left(\hat{R}_{ik}\cdot\hat{T}_j\right)\right)$$

Using the fact that $\hat{R}_{ik} = -\hat{R}_{ki}$, the second part of the triple summation yields

$$g(r_{ik})\left(\hat{R}_{ik}\cdot\hat{T}_j\right) + g(r_{ik})\left(-\hat{R}_{ik}\cdot\hat{T}_j\right) = 0, \qquad \forall i,k\in\mathcal{N}_i \; and \; \forall j\neq i,k$$

The goal here is to show through a simple example that the first part of the triple summation can be simplified to yield:

$$\left(\dot{V}_H\right)_{k\neq j} = -k_{c2}\sum_{i=1}^{N}\sum_{j<i}\left(L_{ii}+L_{jj}+2L_{ij}\right)\hat{T}_j\cdot\hat{T}_i$$

A simple five (5) agent swarm with a proximity graph given by the following Laplacian matrix will be considered in this example:

$$L = \begin{bmatrix} 2 & 0 & -1 & -1 & 0 \\ 0 & 3 & -1 & -1 & -1 \\ -1 & -1 & 2 & 0 & 0 \\ -1 & -1 & 0 & 2 & 0 \\ 0 & -1 & 0 & 0 & 1 \end{bmatrix}$$

If the first part of the triple summation is expanded for this case, the result is:

$i = 1$

$$\sum_{j \neq 1} \sum_{\substack{k \in (3,4) \\ k \neq j}} \hat{T}_j \cdot \hat{T}_k = \left(\hat{T}_2 \cdot \hat{T}_3 + \hat{T}_2 \cdot \hat{T}_4\right) + \left(\hat{T}_3 \cdot \hat{T}_4\right) + \left(\hat{T}_4 \cdot \hat{T}_3\right) + \left(\hat{T}_5 \cdot \hat{T}_3 + \hat{T}_5 \cdot \hat{T}_4\right)$$

$i = 2$

$$\sum_{j \neq 2} \sum_{\substack{k \in (3,4,5) \\ k \neq j}} \hat{T}_j \cdot \hat{T}_k = \left(\hat{T}_1 \cdot \hat{T}_3 + \hat{T}_1 \cdot \hat{T}_4 + \hat{T}_1 \cdot \hat{T}_5\right) + \left(\hat{T}_3 \cdot \hat{T}_4 + \hat{T}_3 \cdot \hat{T}_5\right) + \left(\hat{T}_4 \cdot \hat{T}_3 + \hat{T}_4 \cdot \hat{T}_5\right) + \left(\hat{T}_5 \cdot \hat{T}_3 + \hat{T}_5 \cdot \hat{T}_4\right)$$

$i = 3$

$$\sum_{j \neq 3} \sum_{\substack{k \in (1,2) \\ k \neq j}} \hat{T}_j \cdot \hat{T}_k = \left(\hat{T}_1 \cdot \hat{T}_2\right) + \left(\hat{T}_2 \cdot \hat{T}_1\right) + \left(\hat{T}_4 \cdot \hat{T}_1 + \hat{T}_4 \cdot \hat{T}_2\right) + \left(\hat{T}_5 \cdot \hat{T}_1 + \hat{T}_5 \cdot \hat{T}_2\right)$$

$i = 4$

$$\sum_{j \neq 4} \sum_{\substack{k \in (1,2) \\ k \neq j}} \hat{T}_j \cdot \hat{T}_k = \left(\hat{T}_1 \cdot \hat{T}_2\right) + \left(\hat{T}_2 \cdot \hat{T}_1\right) + \left(\hat{T}_3 \cdot \hat{T}_1 + \hat{T}_3 \cdot \hat{T}_2\right) + \left(\hat{T}_5 \cdot \hat{T}_1 + \hat{T}_5 \cdot \hat{T}_2\right)$$

$i = 5$

$$\sum_{j \neq 5} \sum_{\substack{k=2 \\ k \neq j}} \hat{T}_j \cdot \hat{T}_k = \left(\hat{T}_1 \cdot \hat{T}_2\right) + \left(\hat{T}_3 \cdot \hat{T}_2\right) + \left(\hat{T}_4 \cdot \hat{T}_2\right)$$

Using the associative property of the dot product, the previous sums can be combined and regrouped to yield:

$$5(\hat{T}_1 \cdot \hat{T}_2) + 2(\hat{T}_1 \cdot \hat{T}_3) + 2(\hat{T}_1 \cdot \hat{T}_4) + 3(\hat{T}_1 \cdot \hat{T}_5) + 3(\hat{T}_2 \cdot \hat{T}_3) + 3(\hat{T}_2 \cdot \hat{T}_4) + 2(\hat{T}_2 \cdot \hat{T}_5) + 4(\hat{T}_3 \cdot \hat{T}_4) + 3(\hat{T}_3 \cdot \hat{T}_5) + 3(\hat{T}_4 \cdot \hat{T}_5)$$

The multiplicity of each of the terms in the summation is directly linked to the proximity graph of the swarm. A pattern emerges in which the coefficient equals the number of edges connected to the associated pair of agents in the graph. These coefficients can be related to the Laplacian matrix using the following equation:

$$L_{ii} + L_{jj} + 2L_{ij}, \qquad \forall i, j \neq i$$

The first two numbers in the equation represent the number of edges connected to agents i and j, respectively. If the two agents are neighbors, then the third number will subtract the edge between them. Therefore, the previous equation will find the number of edges connected to any pair of agents in the swarm. Figure 75 illustrates this concept.

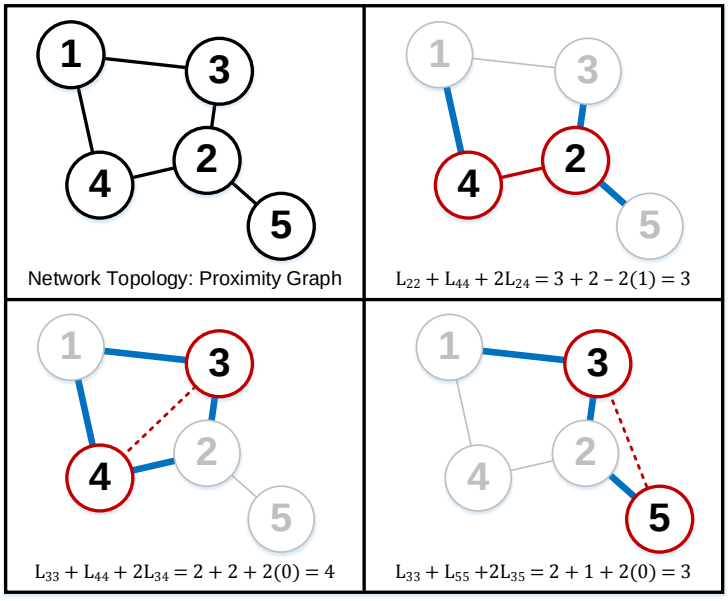

Figure 75 Number of edges connected to pairs of agents.

Page Intentionally Left Blank

Appendix B
Simulation Figures

Two-Dimensional Simulation 2

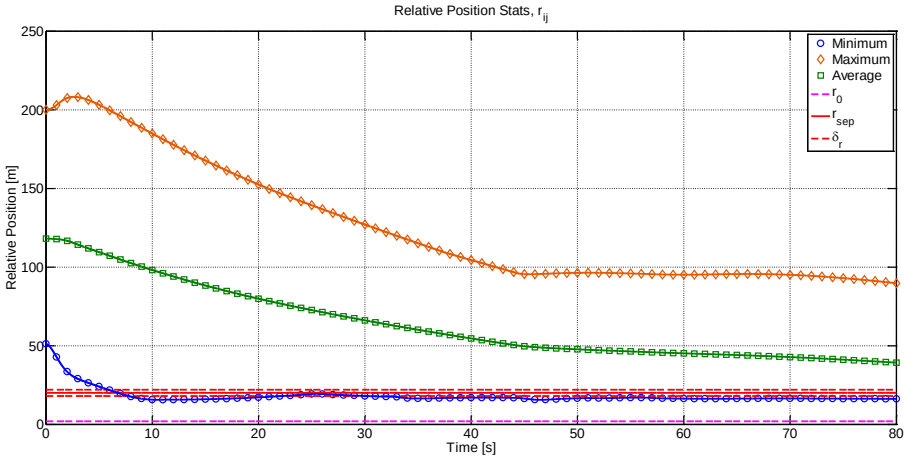

Figure 76 Relative position statistics of 15 agent swarm (r_{sen} = 150 m).

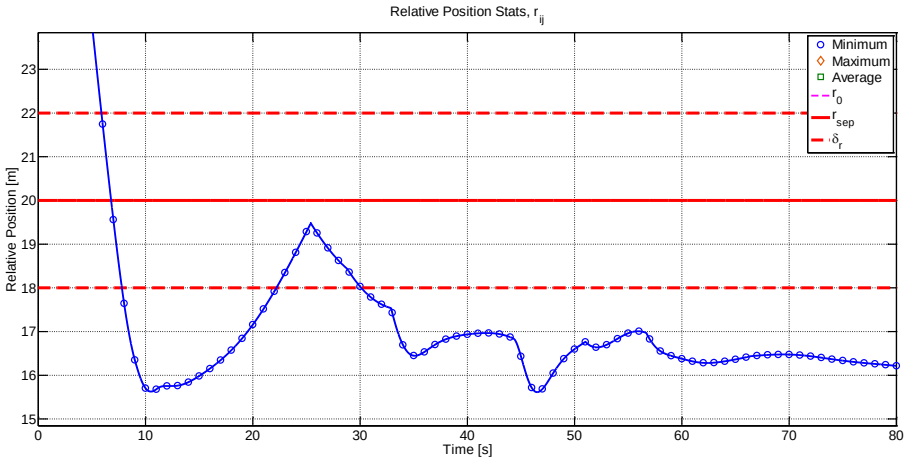

Figure 77 Minimum relative position curve for 15 agent swarm (r_{sen} = 150 m).

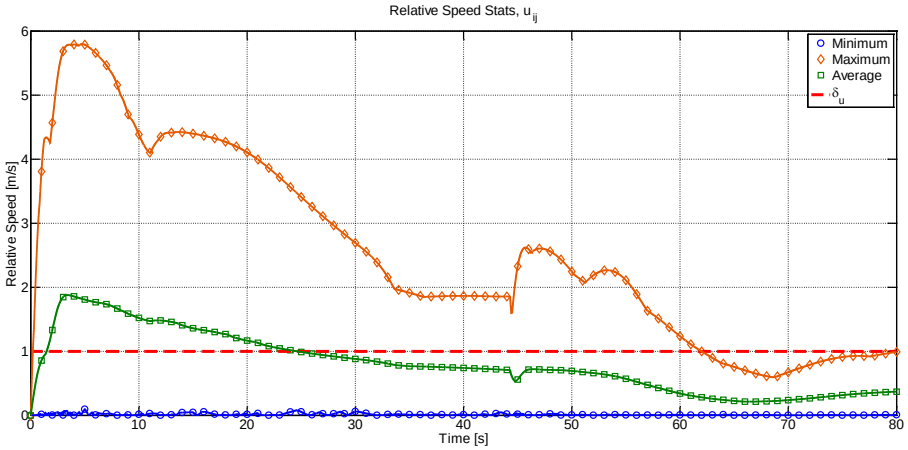

Figure 78 Relative speed statistics for 15 agent swarm (r_{sen} = 150 m).

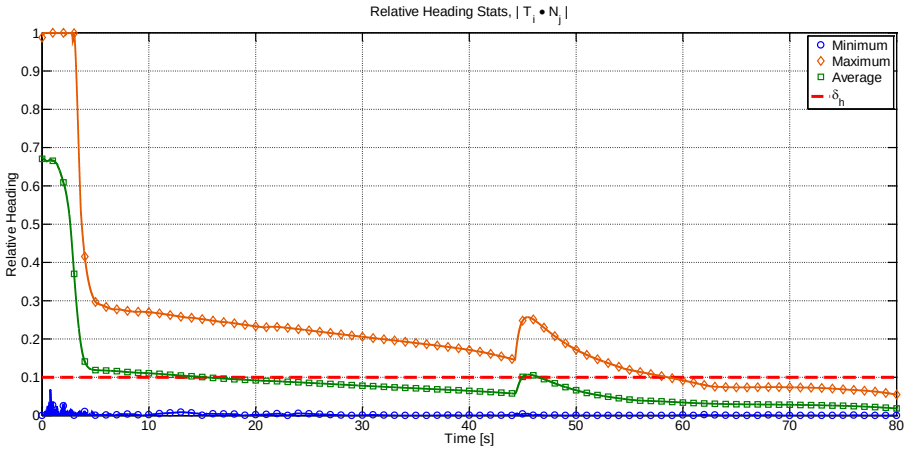

Figure 79 Relative heading statistics for 15 agent swarm (r_{sen} = 150 m).

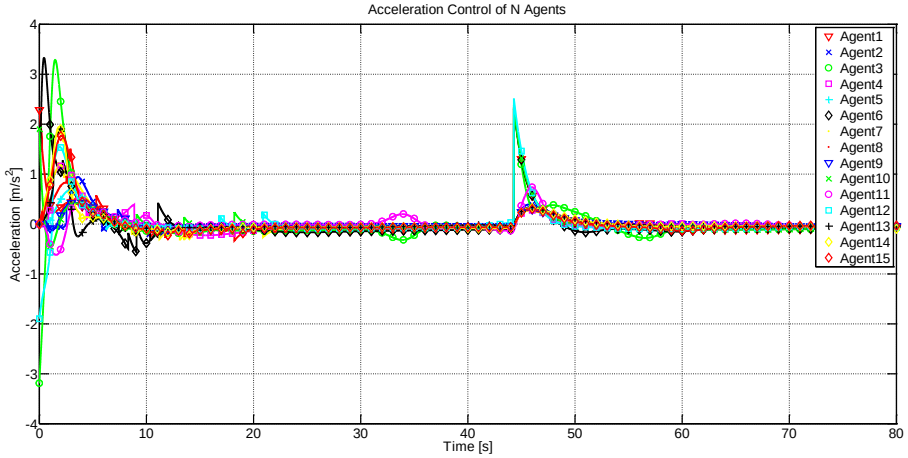

Figure 80 Acceleration control inputs of 15 agents (r_{sen} = 150 m).

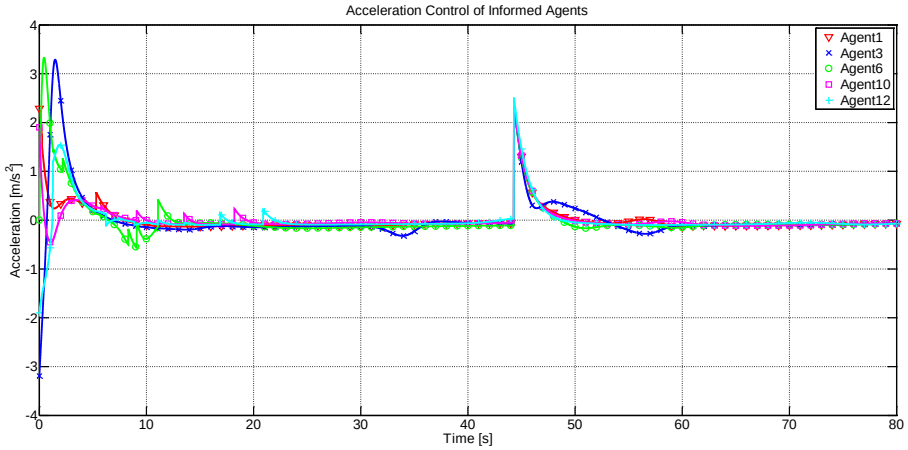

Figure 81 Acceleration control inputs of informed agents (r_{sen} = 150 m).

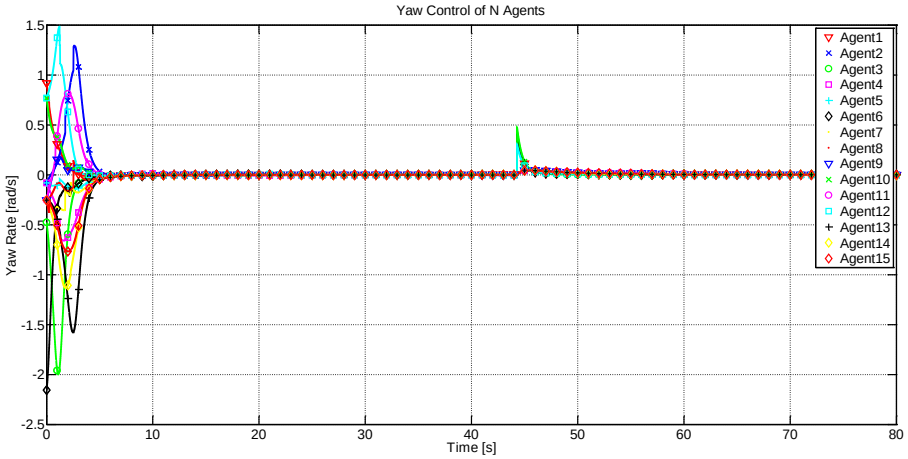

Figure 82 Yaw control inputs of 15 agents (r_{sen} = 150 m).

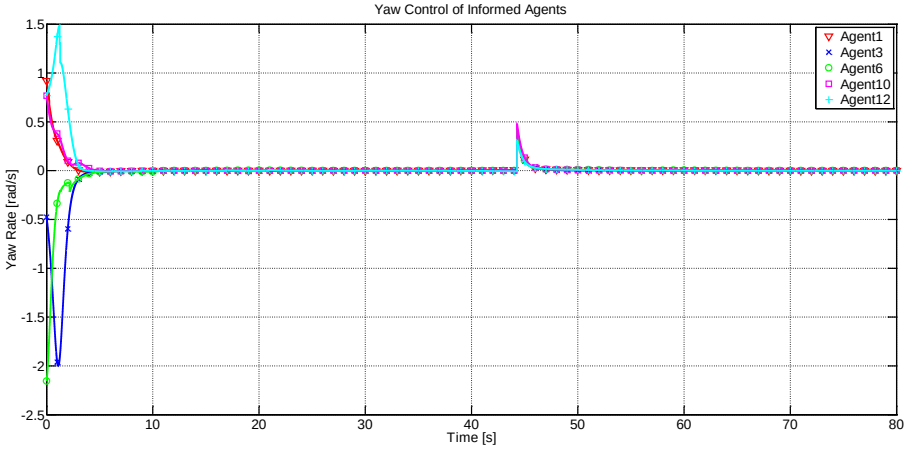

Figure 83 Yaw control inputs of informed agents (r_{sen} = 150 m).

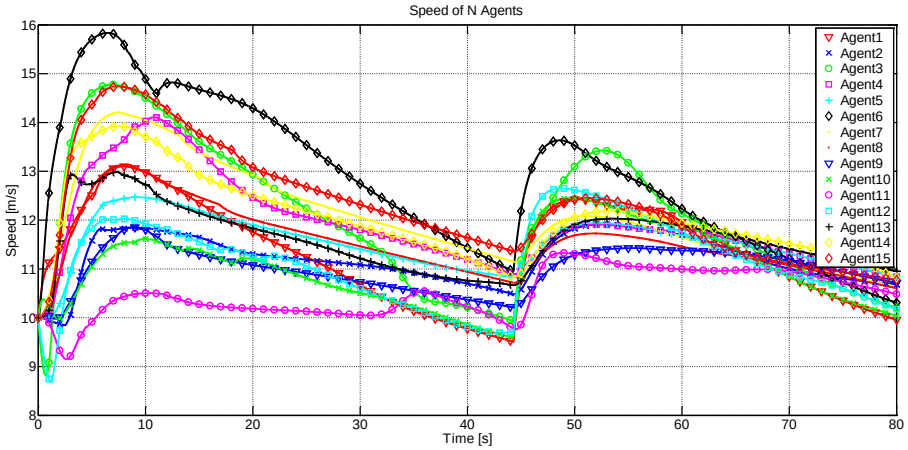

Figure 84 Speed of 15 agents (r_{sen} = 150 m).

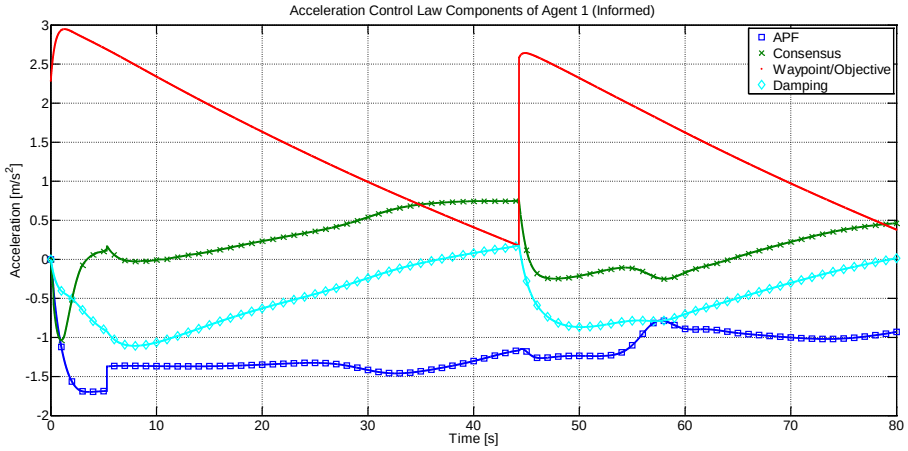

Figure 85 Components of acceleration control law of agent 1 (informed).

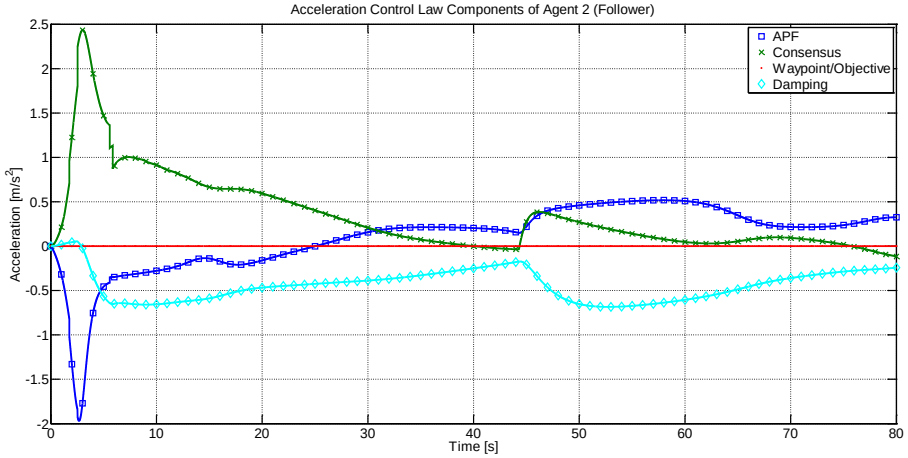

Figure 86 Components of acceleration control law of agent 2 (follower).

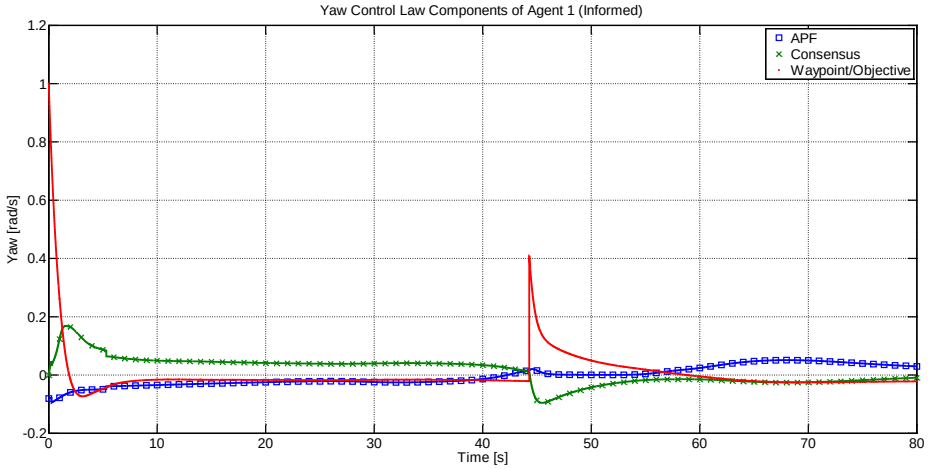

Figure 87 Components of yaw control law of agent 1 (informed).

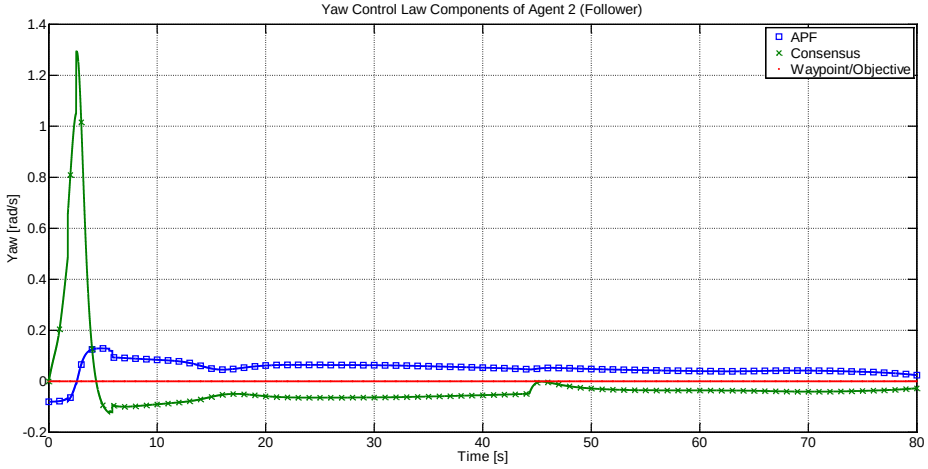

Figure 88 Components of yaw control law of agent 2 (follower).

Two-Dimensional Simulation 3

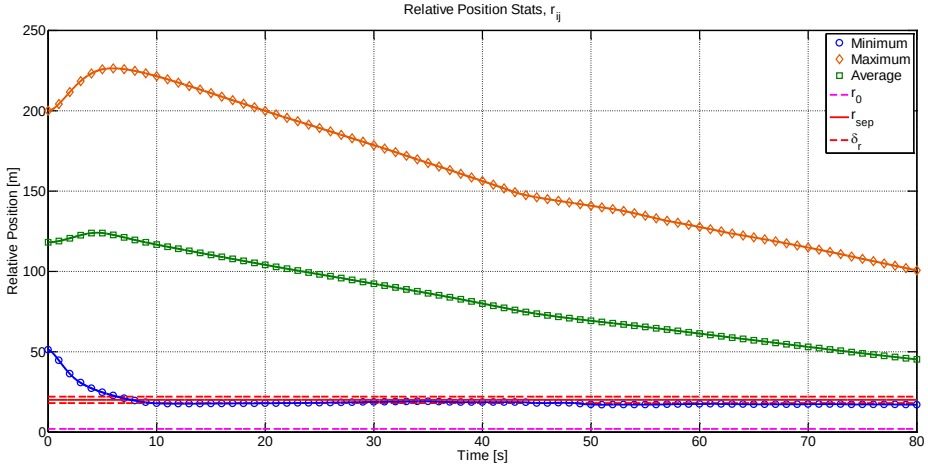

Figure 89 Relative position statistics of 15 agent swarm (r_{sen} = 80 m).

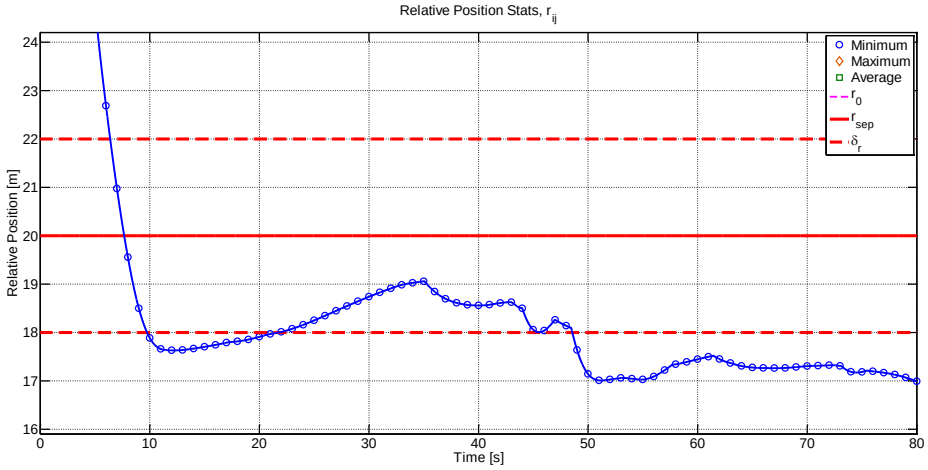

Figure 90 Minimum relative position of 15 agent swarm (r_{sen} = 80 m).

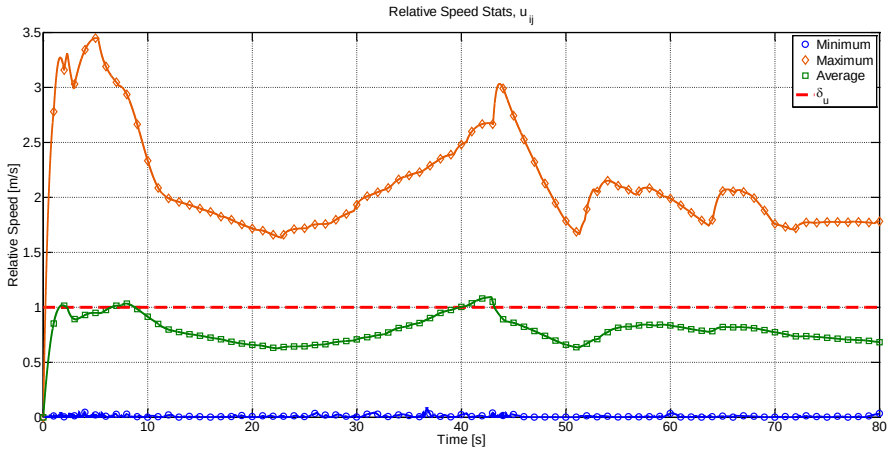

Figure 91 Relative speed statistics for 15 agent swarm (r_{sen} = 80 m).

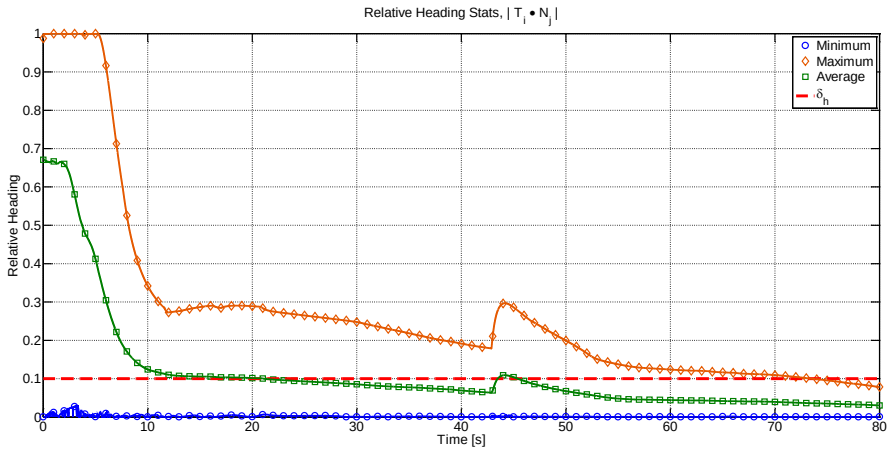

Figure 92 Relative heading statistics for 15 agent swarm (r_{sen} = 80 m).

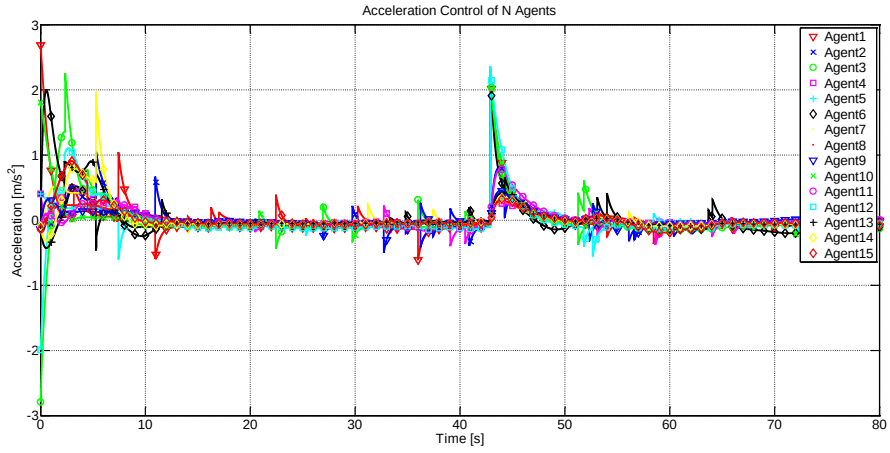

Figure 93 Acceleration control inputs of 15 agents (r_{sen} = 80 m).

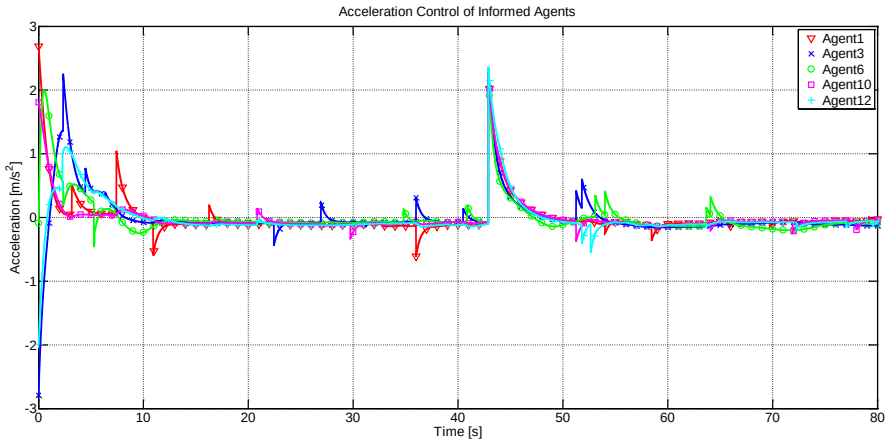

Figure 94 Acceleration control inputs of informed agents (r_{sen} = 80 m).

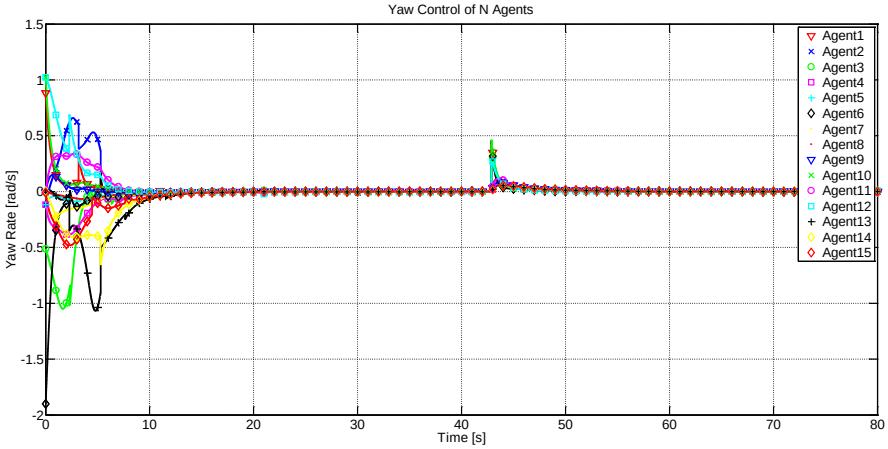

Figure 95 Yaw control inputs of 15 agents (r_{sen} = 80 m).

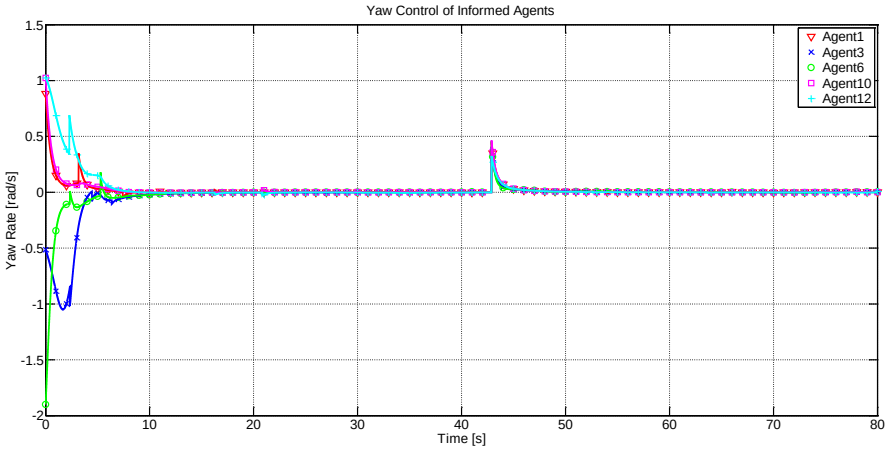

Figure 96 Yaw control inputs of informed agents (r_{sen} = 80 m).

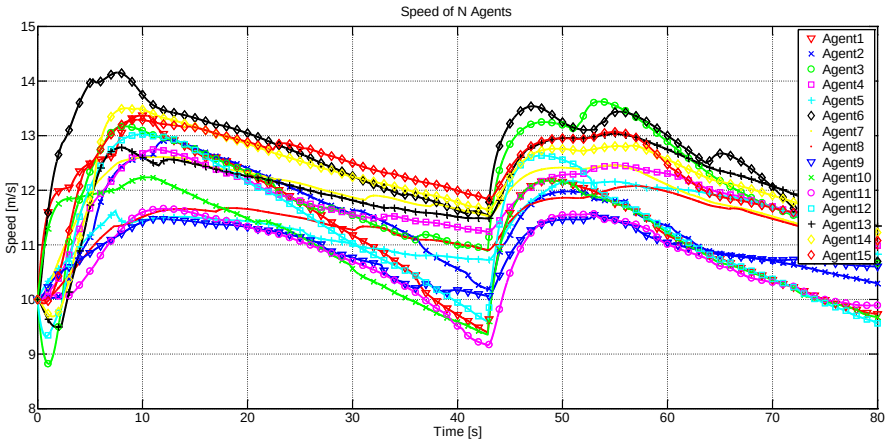

Figure 97 Speed of 15 agents (r_{sen} = 80 m).

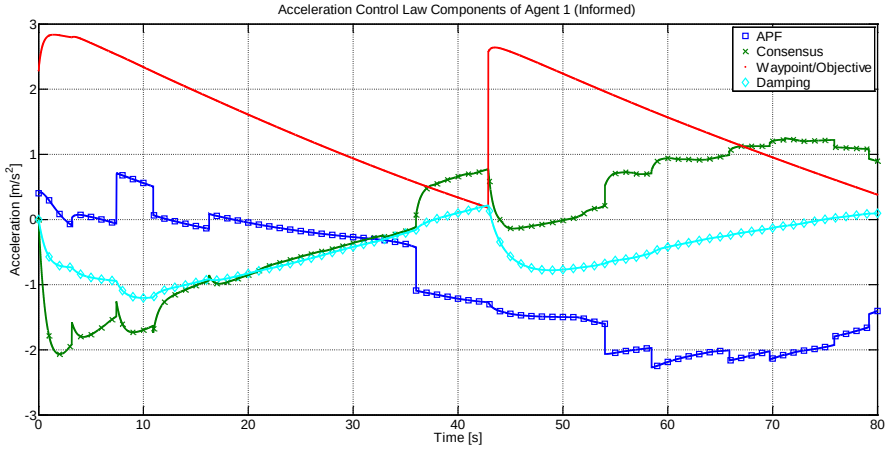

Figure 98 Components of acceleration control law of agent 1 (informed).

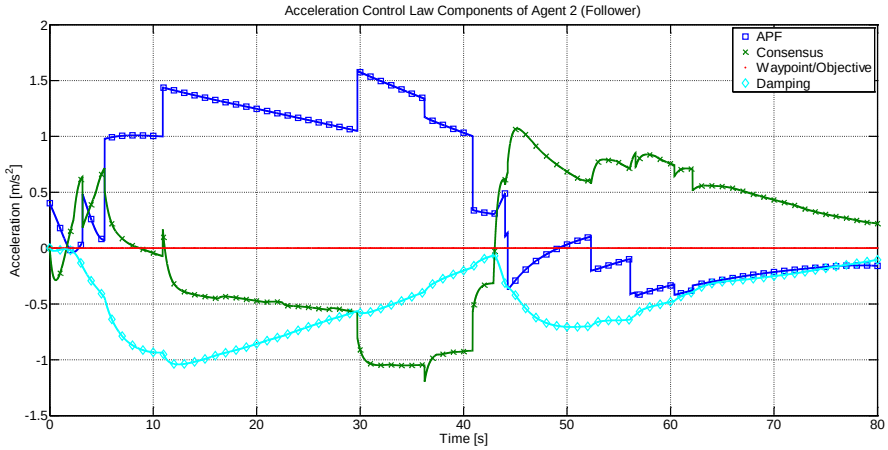

Figure 99 Components of acceleration control law of agent 2 (follower).

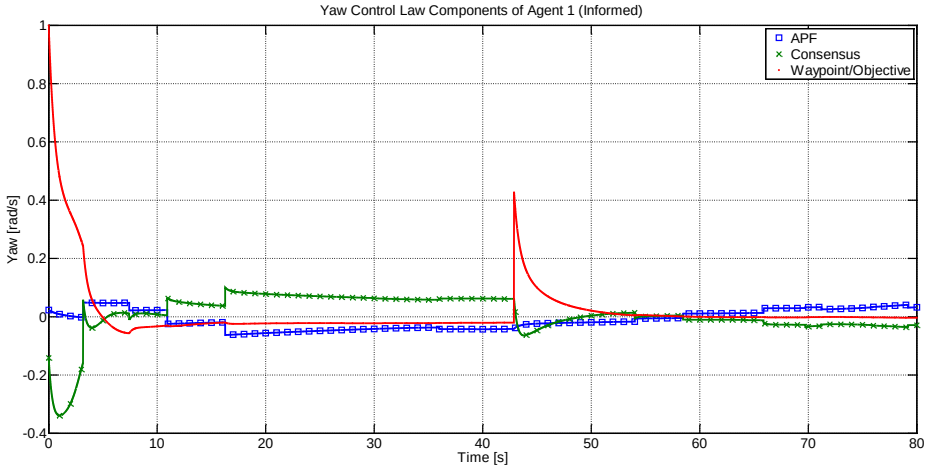

Figure 100 Components of yaw control law of agent 1 (informed).

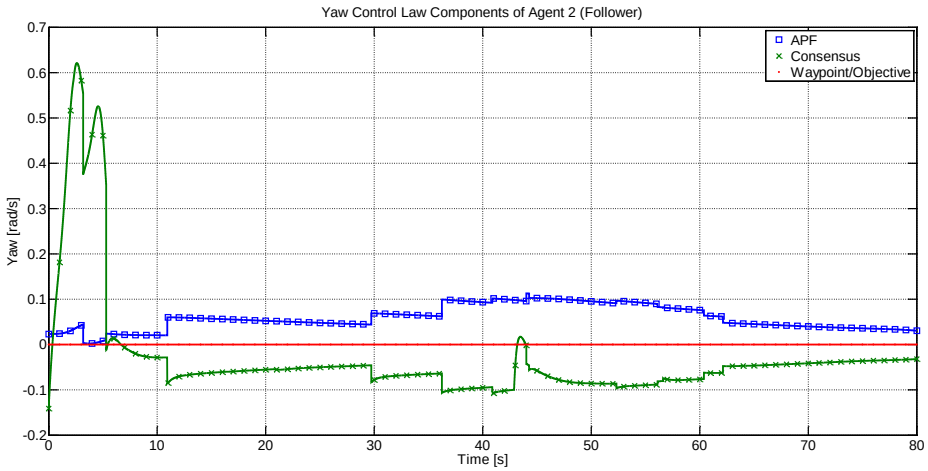

Figure 101 Components of yaw control law of agent 2 (follower).

Three-Dimensional Simulation 1

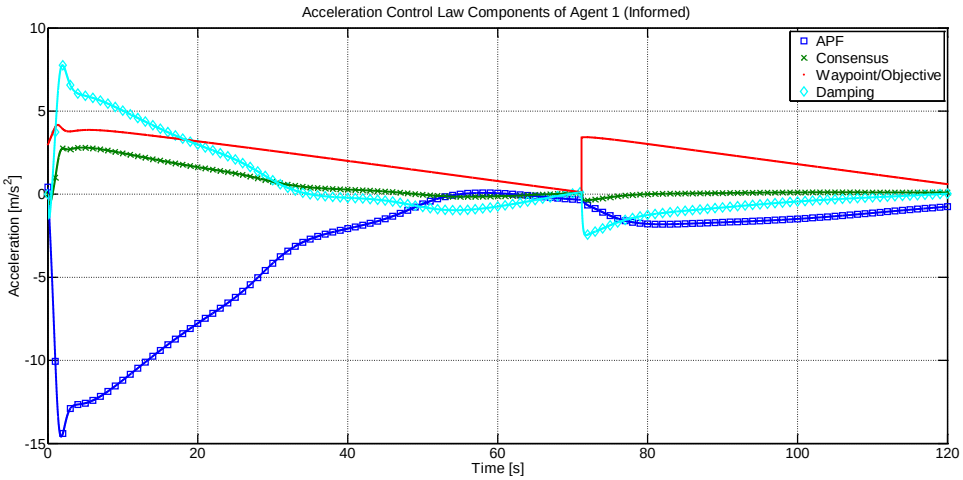

Figure 102 Components of acceleration control law of agent 1 (informed).

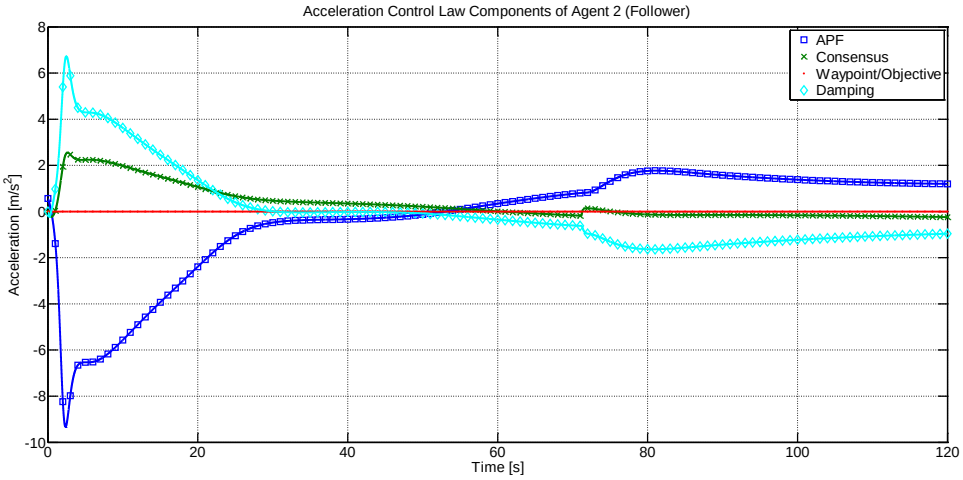

Figure 103 Components of acceleration control law of agent 2 (follower).

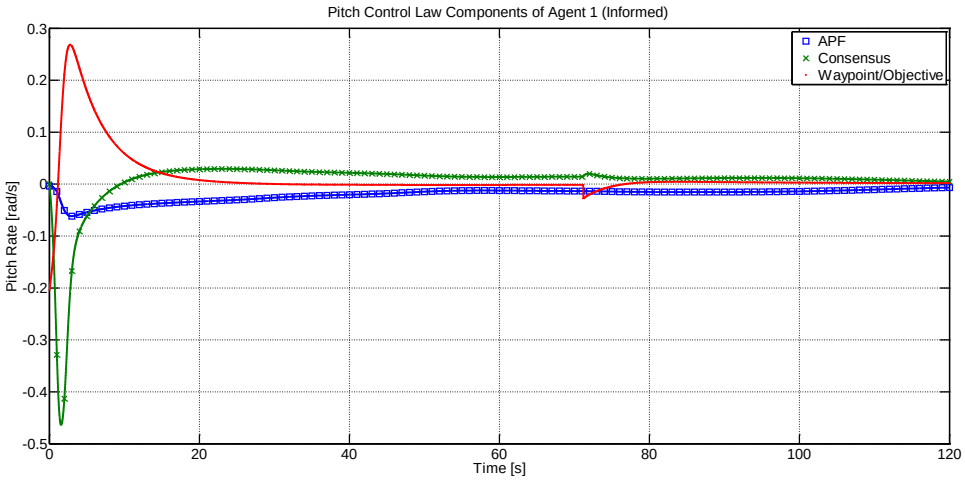

Figure 104 Components of pitch control law of agent 1 (informed).

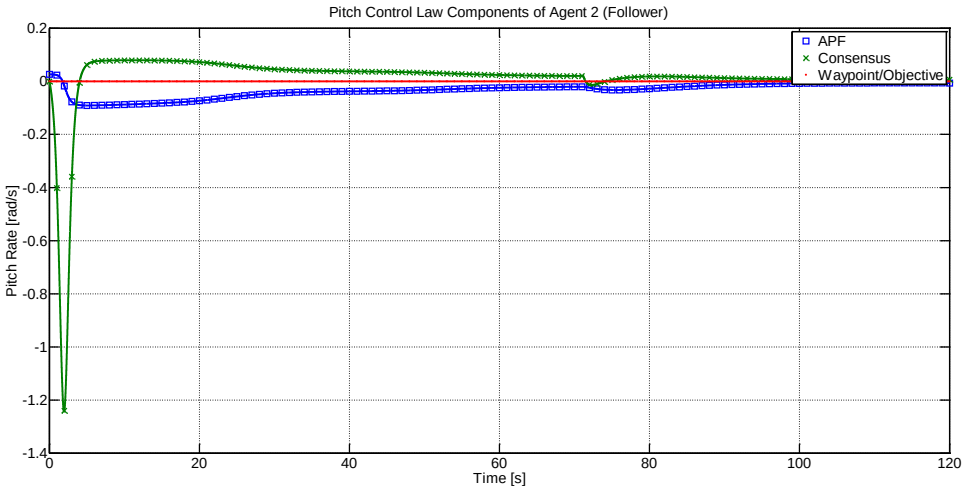

Figure 105 Components of pitch control law of agent 2 (follower).

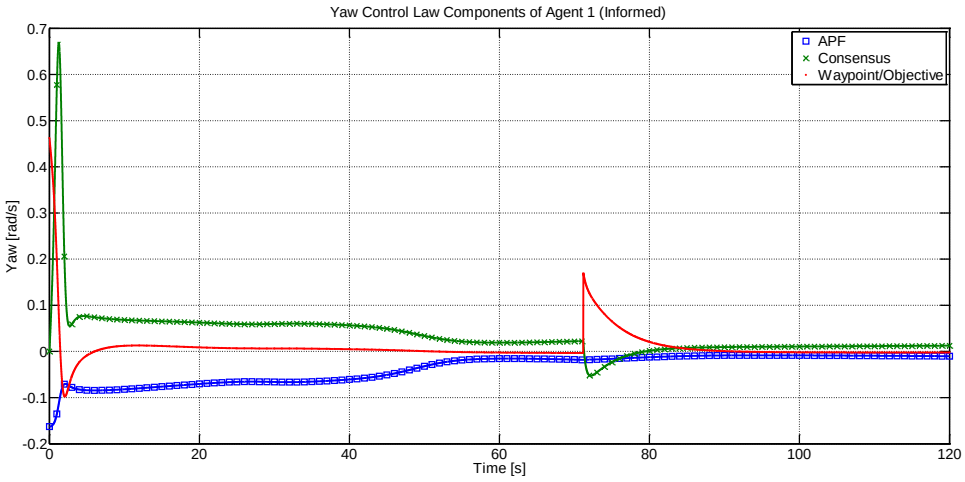

Figure 106 Components of yaw control law of agent 1 (informed).

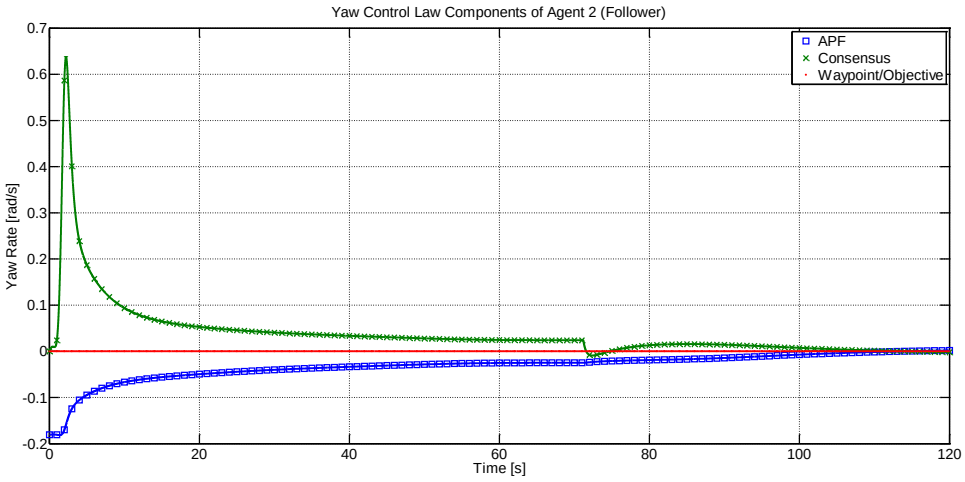

Figure 107 Components of yaw control law of agent 2 (follower).

Three-Dimensional Simulation 2

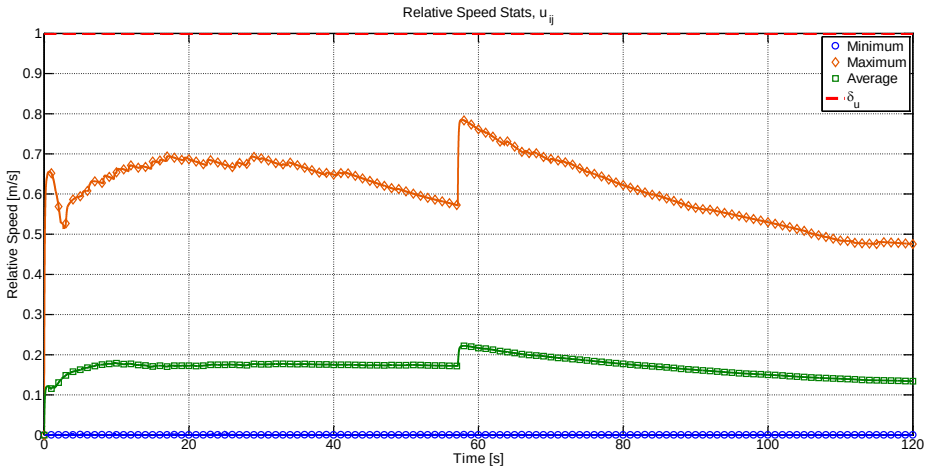

Figure 108 Relative speed statistics for 45 agent swarm.

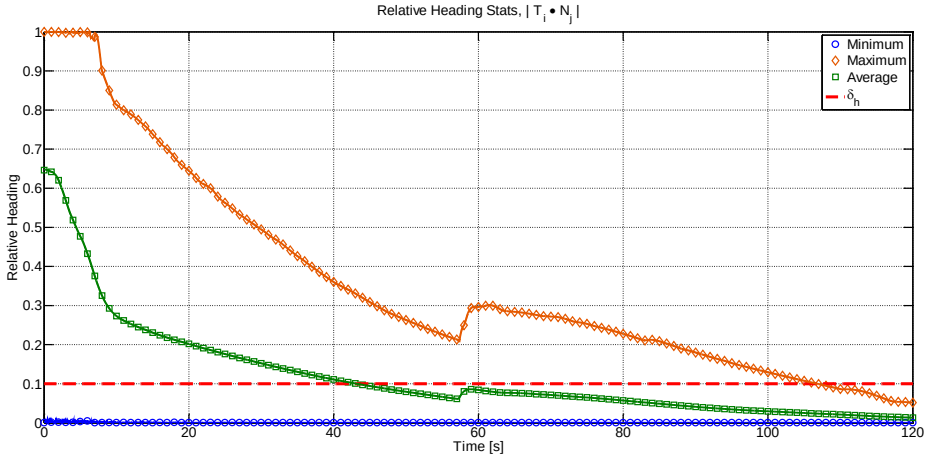

Figure 109 Relative heading statistics for 45 agent swarm.

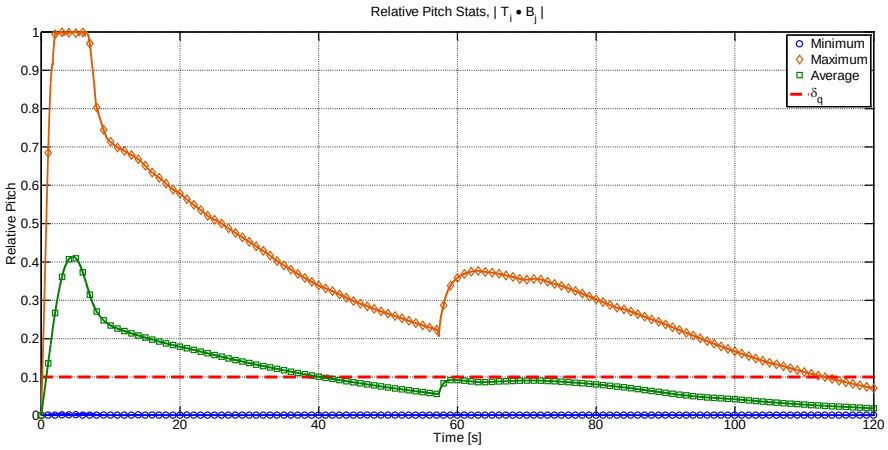

Figure 110 Relative pitch statistics for 45 agent swarm.

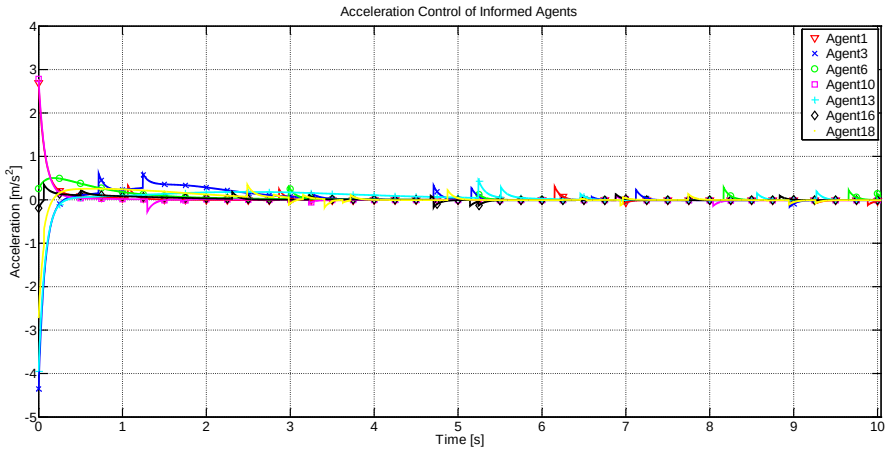

Figure 111 Acceleration control inputs for select informed agents.

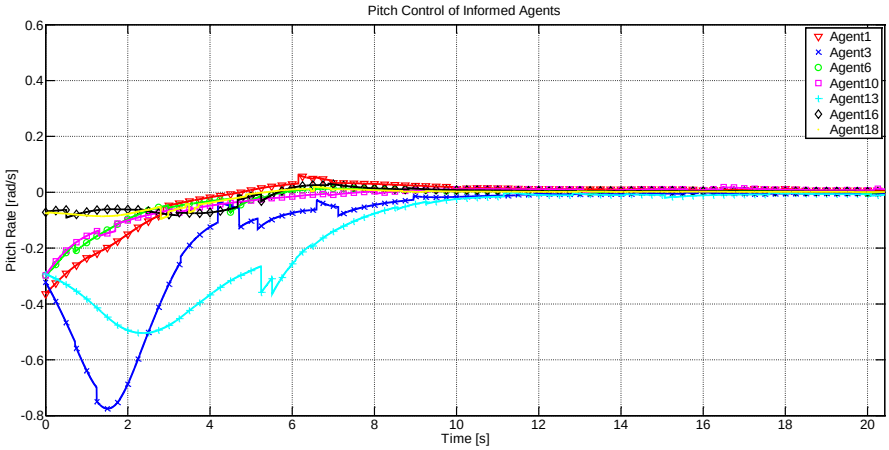

Figure 112 Pitch control inputs for select informed agents.

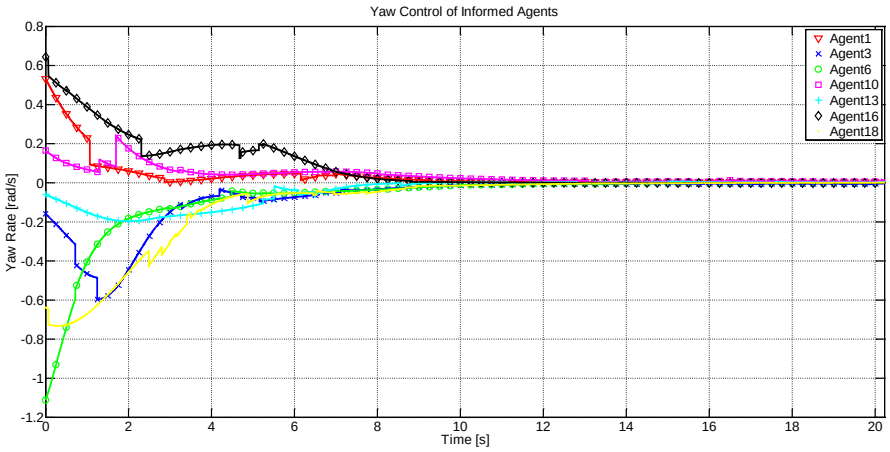

Figure 113 Yaw control inputs for select informed agents.

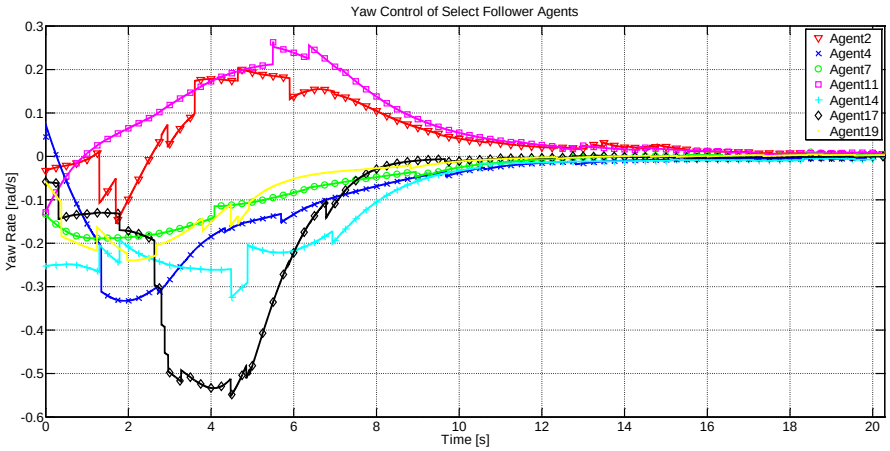

Figure 114 Yaw control inputs of select follower agents.

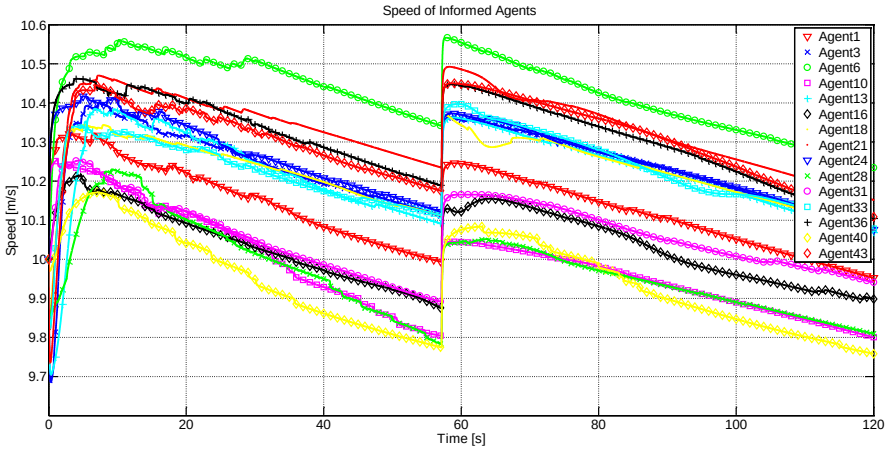

Figure 115 Speed of informed agents.

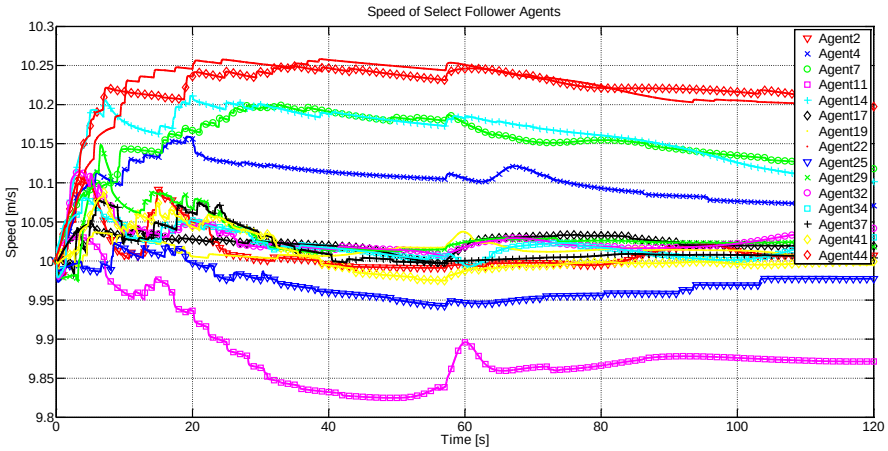

Figure 116 Speed of select follower agents.

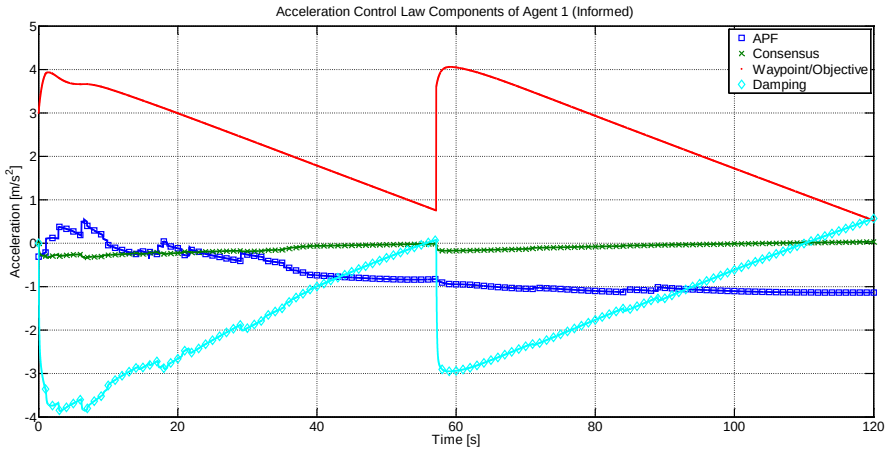

Figure 117 Components of acceleration control law of agent 1 (informed).

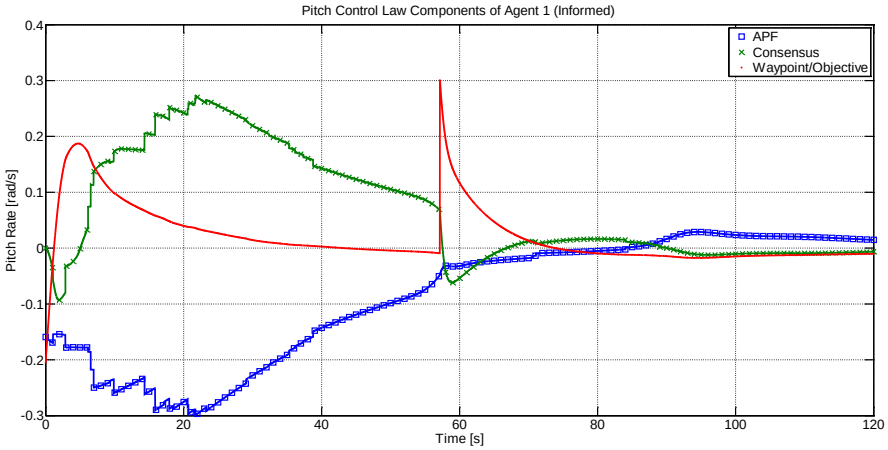

Figure 118 Components of pitch control law of agent 1 (informed).

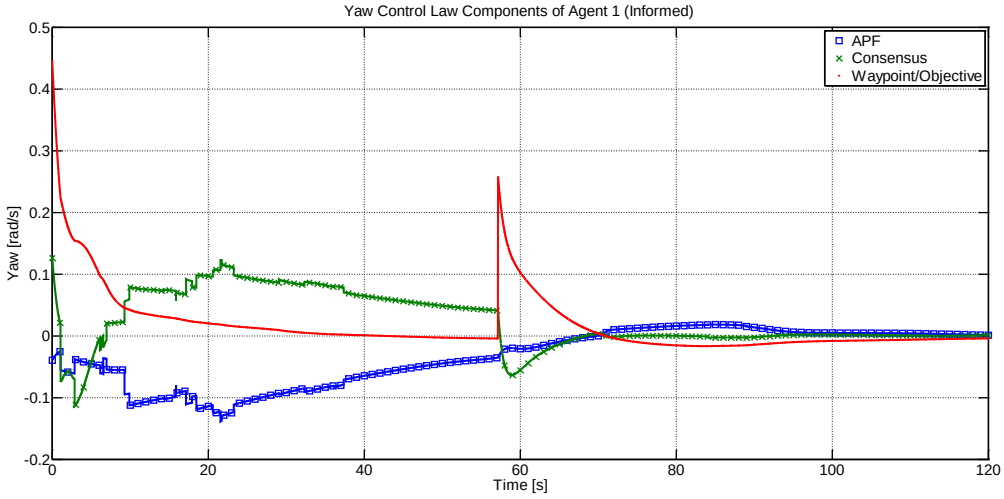

Figure 119 Components of yaw control law of agent 1 (informed).

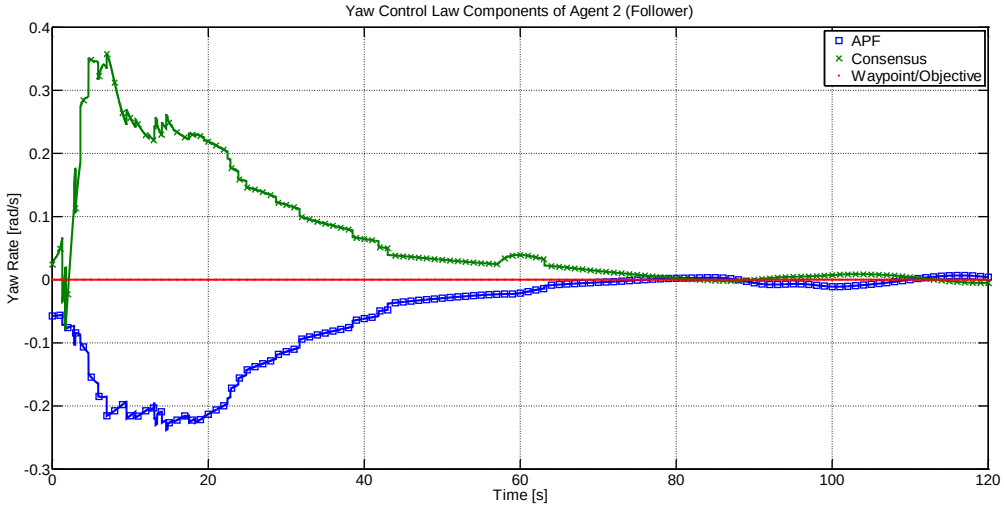

Figure 120 Components of yaw control law of agent 2 (follower).

Post-Saturation Simulation

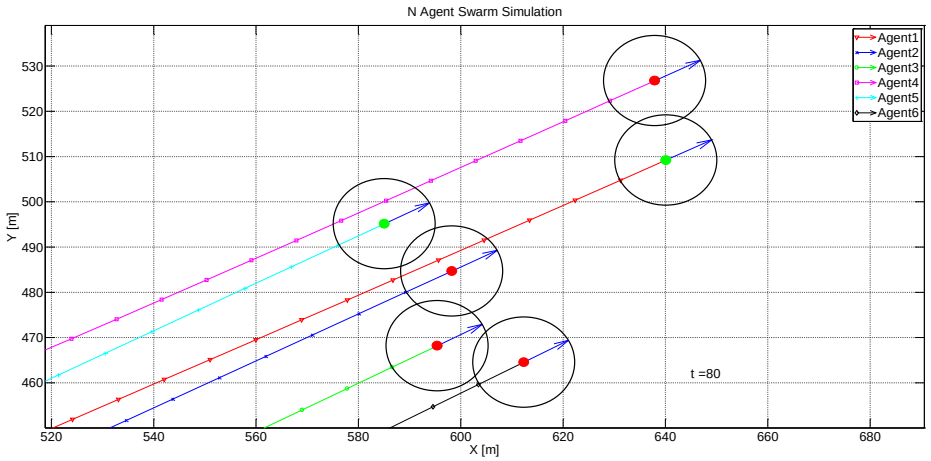

Figure 121 Final positions of 6 agent swarm (post-saturation).

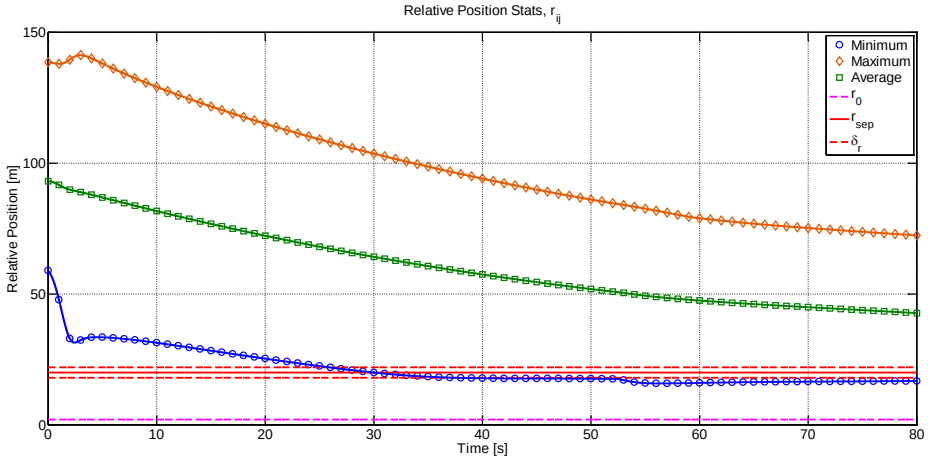

Figure 122 Relative position statistics for 6 agent swarm (post-saturation).

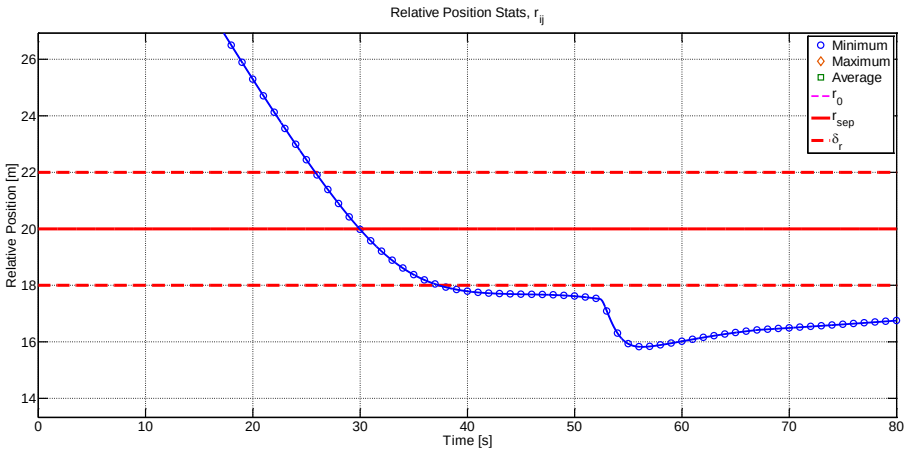

Figure 123 Minimum relative position for 6 agent swarm (post-saturation).

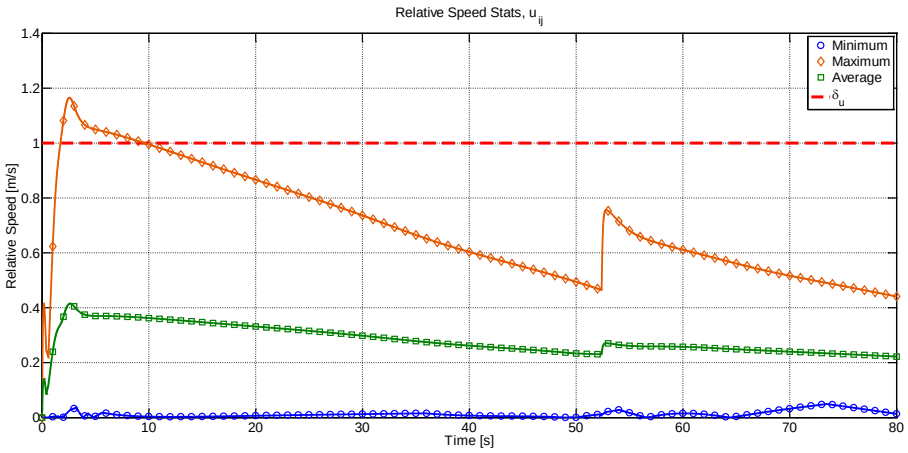

Figure 124 Relative speed statistics for 6 agent swarm (post-saturation).

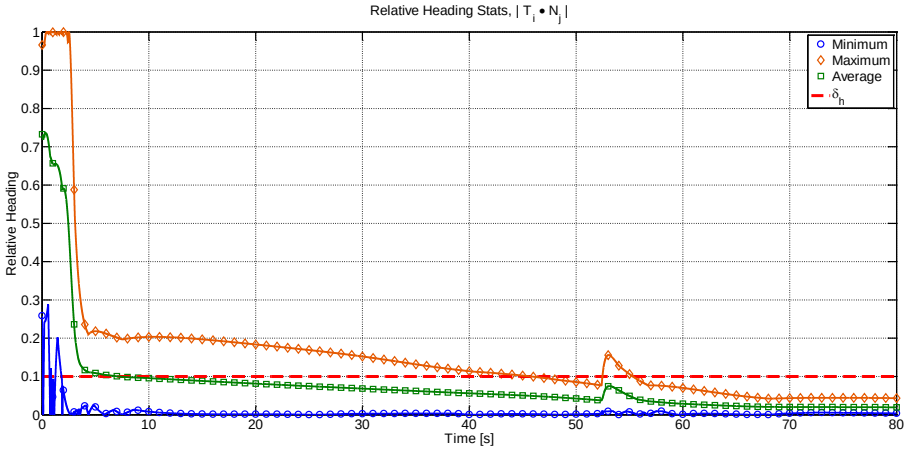

Figure 125 Relative heading statistics for 6 agent swarm (post-saturation).

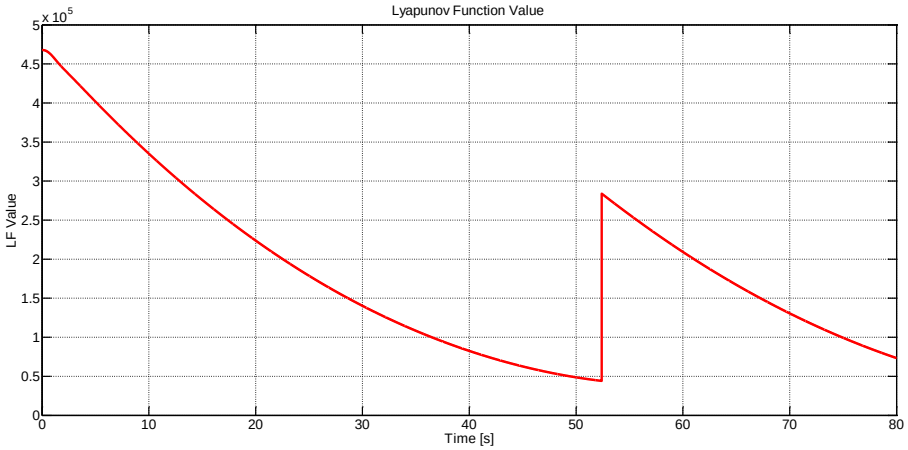

Figure 126 Lyapunov function time history of 6 agent swarm (post-saturation).

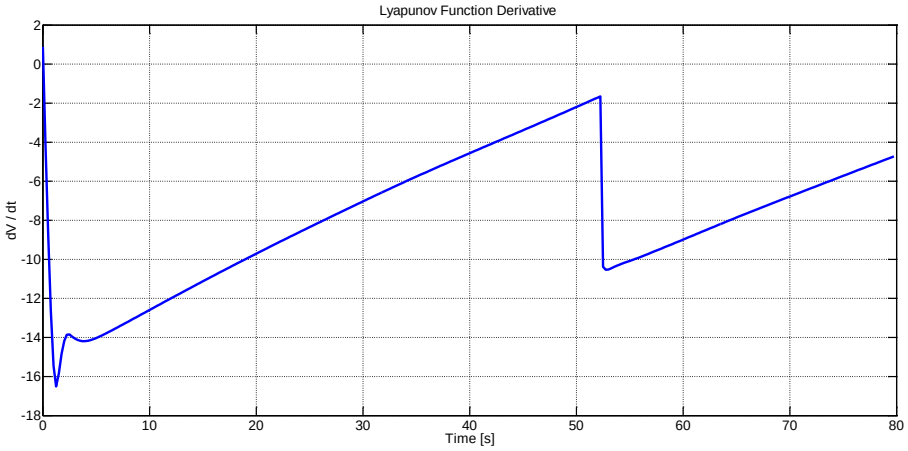

Figure 127 Lyapunov function time derivative. Points are plotted every ¼ second.

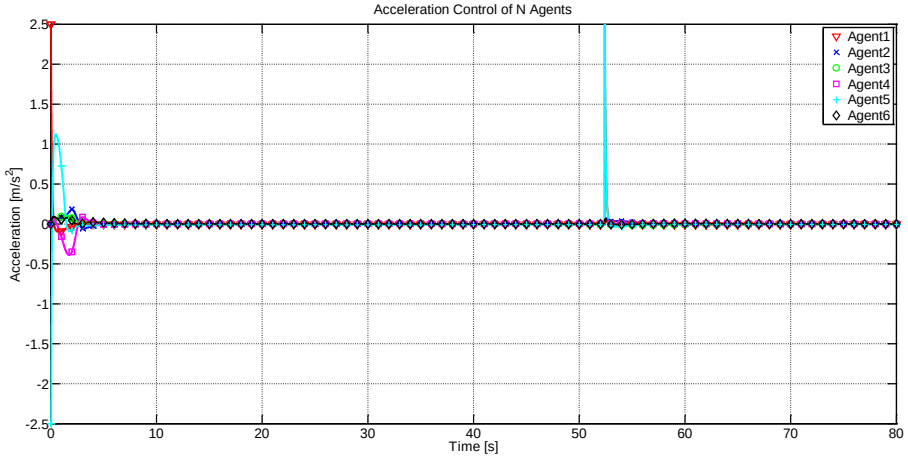

Figure 128 Acceleration control inputs of 6 agents (post-saturation).

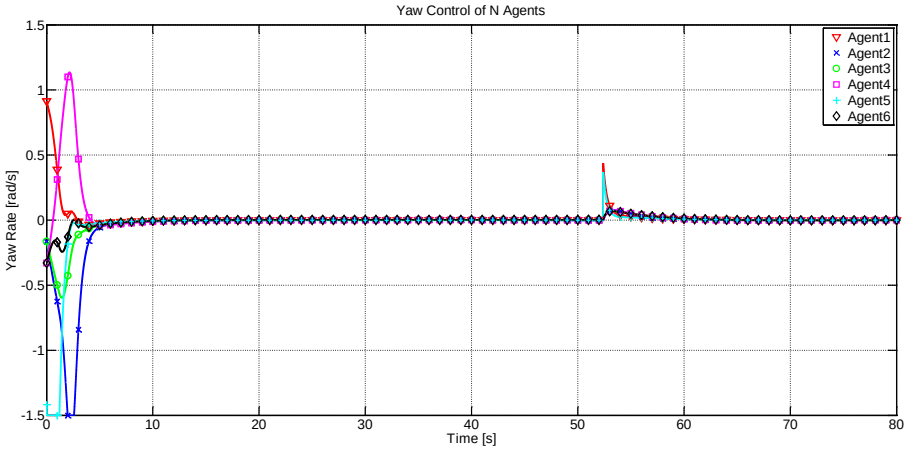

Figure 129 Yaw control inputs for 6 agents (post-saturation).

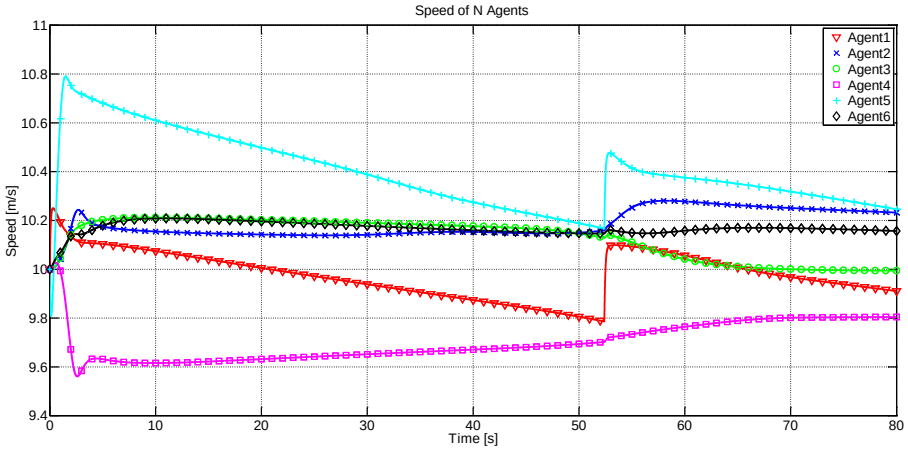

Figure 130 Speed of 6 agents (post-saturation).

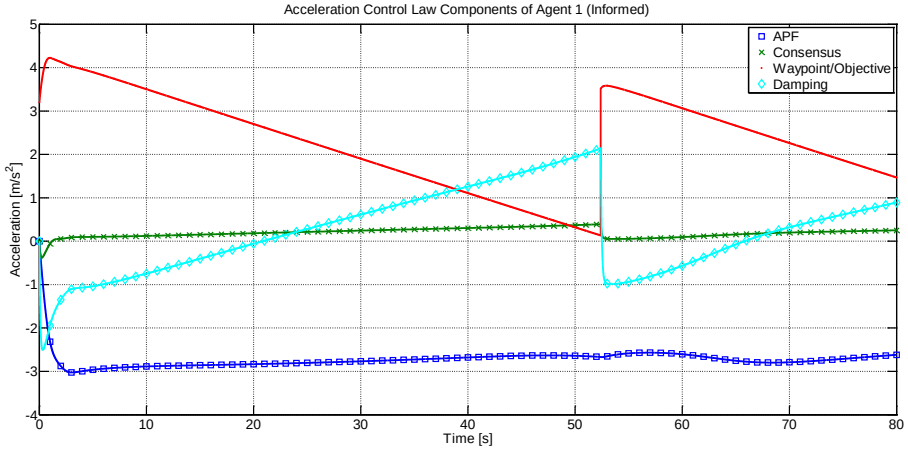

Figure 131 Components of acceleration control law of agent 1 (informed).

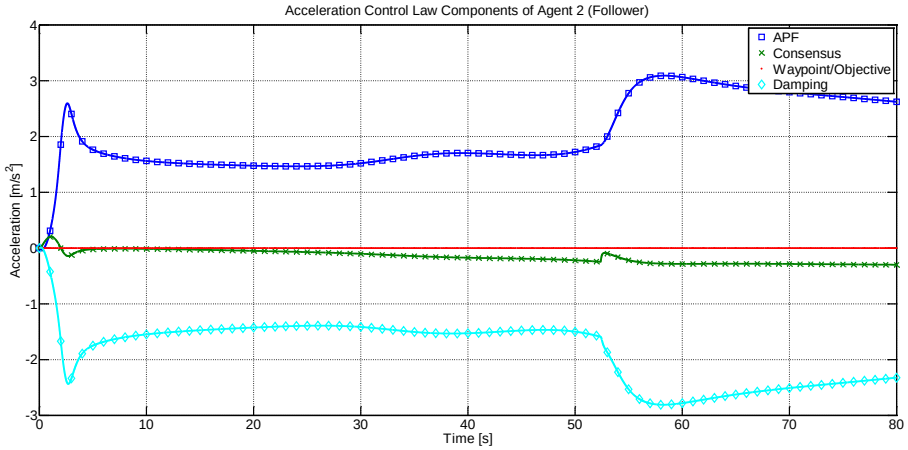

Figure 132 Components of acceleration control law of agent 2 (follower).

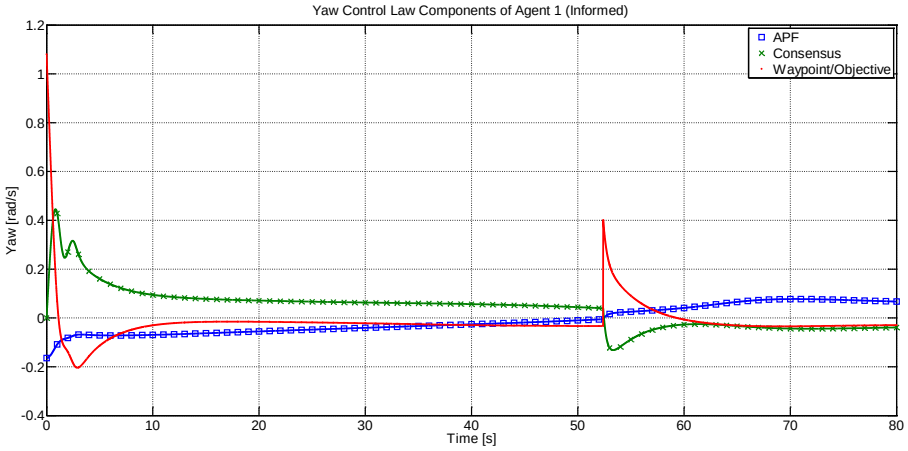

Figure 133 Components of yaw control law of agent 1 (informed).

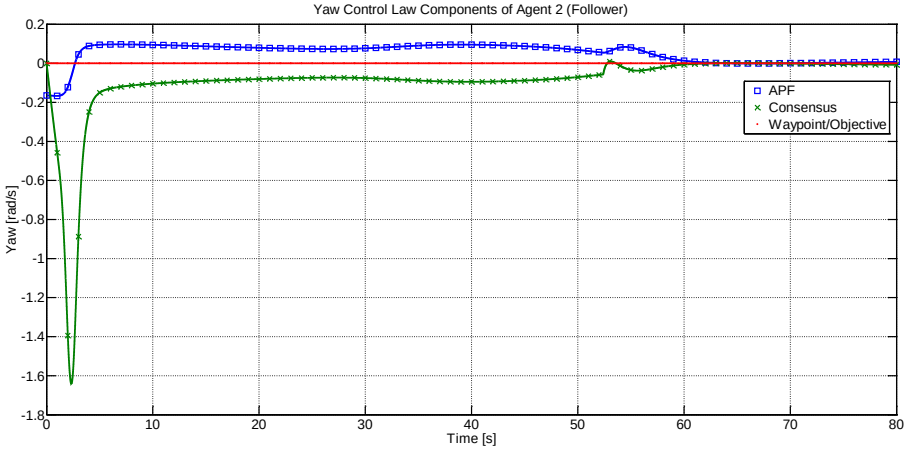

Figure 134 Components of yaw control law of agent 2 (follower).

Pre-Saturation Simulation

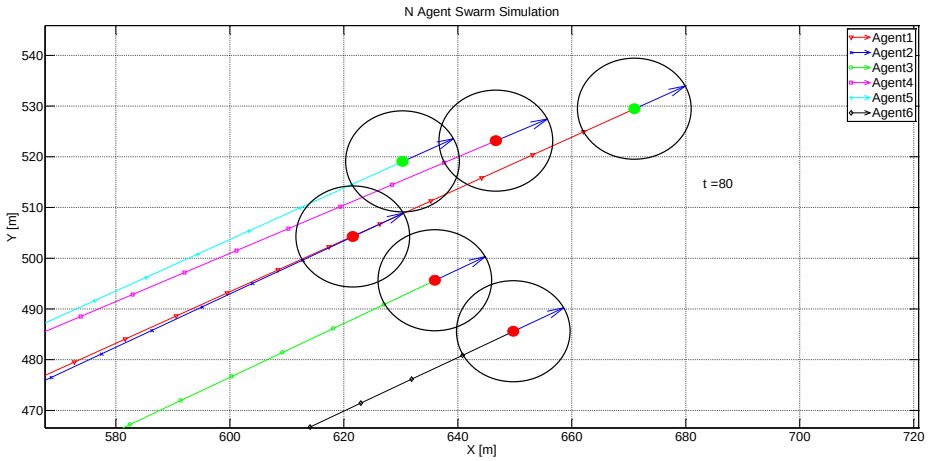

Figure 135 Final positions of 6 agent swarm (pre-saturation).

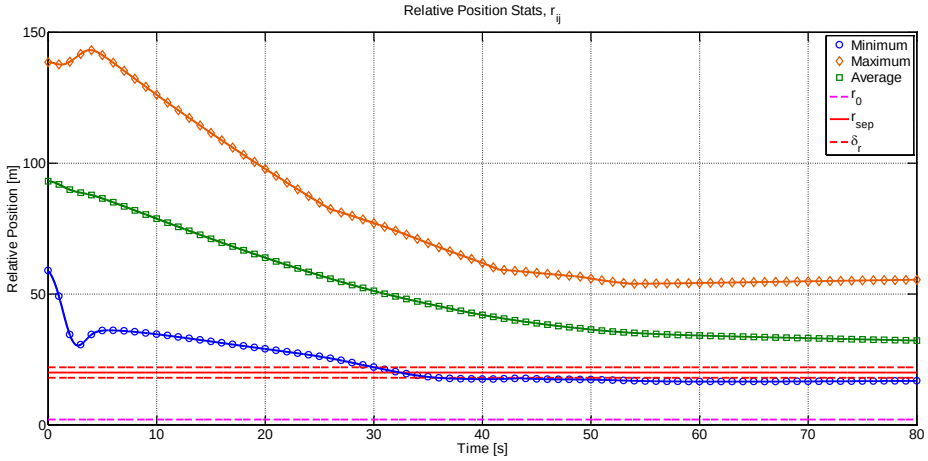

Figure 136 Relative position statistics for 6 agent swarm (pre-saturation).

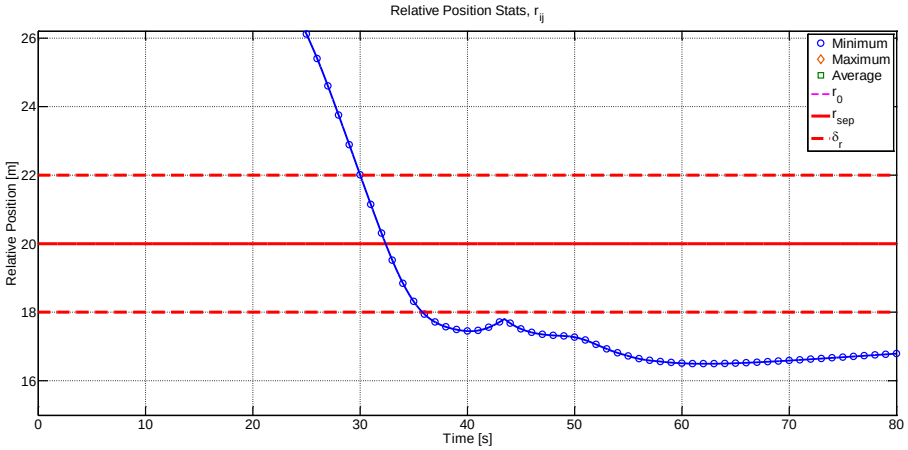

Figure 137 Minimum relative position for 6 agent swarm (pre-saturation).

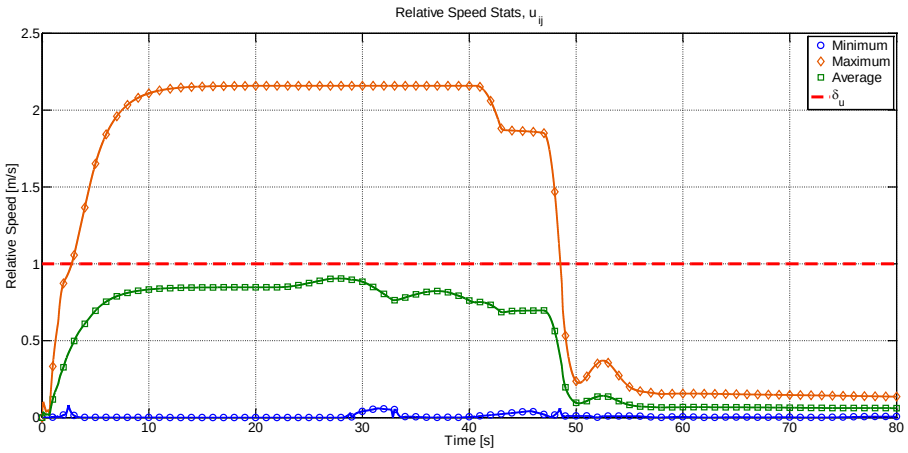

Figure 138 Relative speed statistics for 6 agent swarm (pre-saturation).

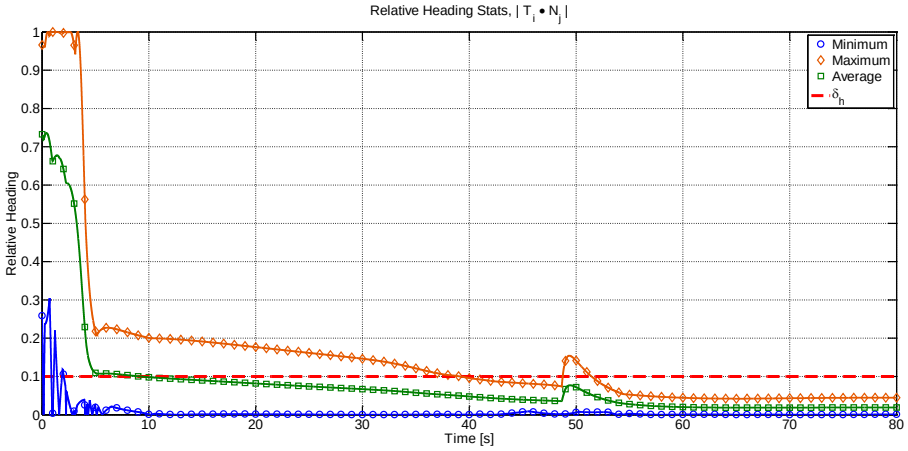

Figure 139 Relative heading statistics for 6 agent swarm (pre-saturation).

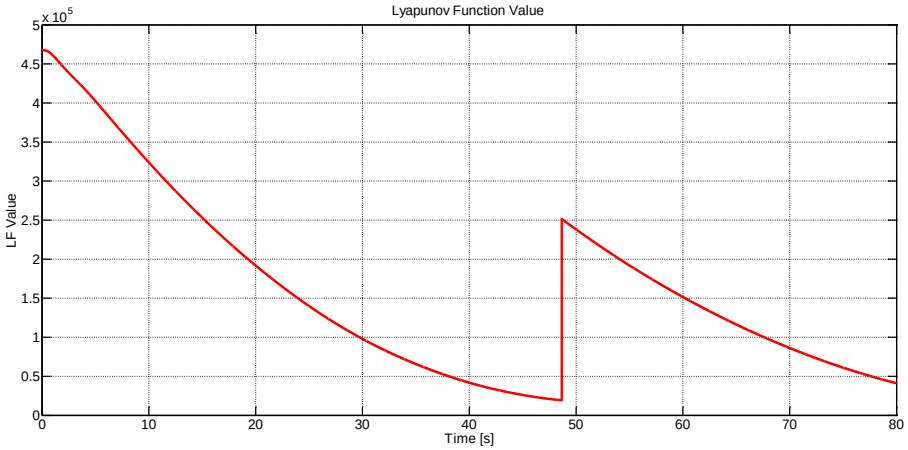

Figure 140 Lyapunov function time history of 6 agent swarm (pre-saturation).

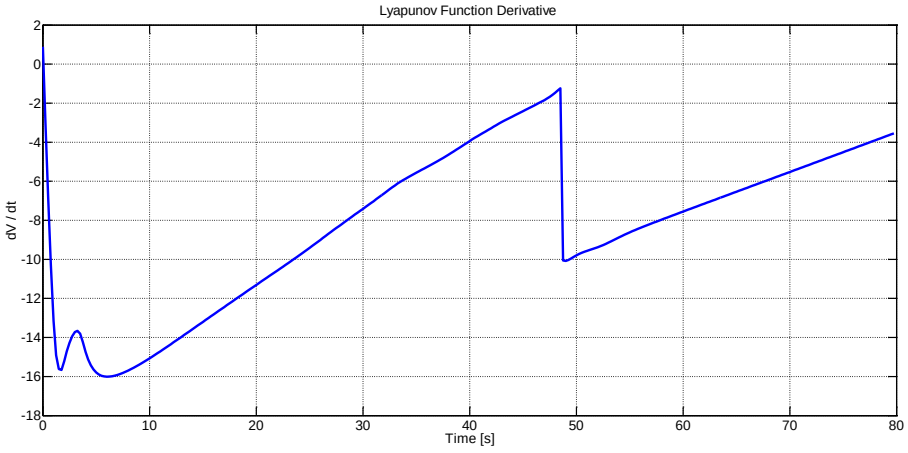

Figure 141 Lyapunov function time derivative. Points are plotted every ¼ second.

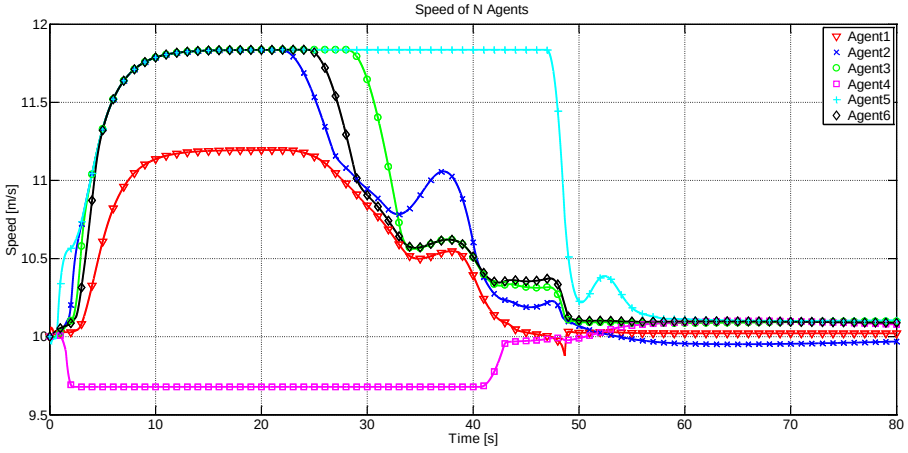

Figure 142 Speed of 6 agents (pre-saturation).

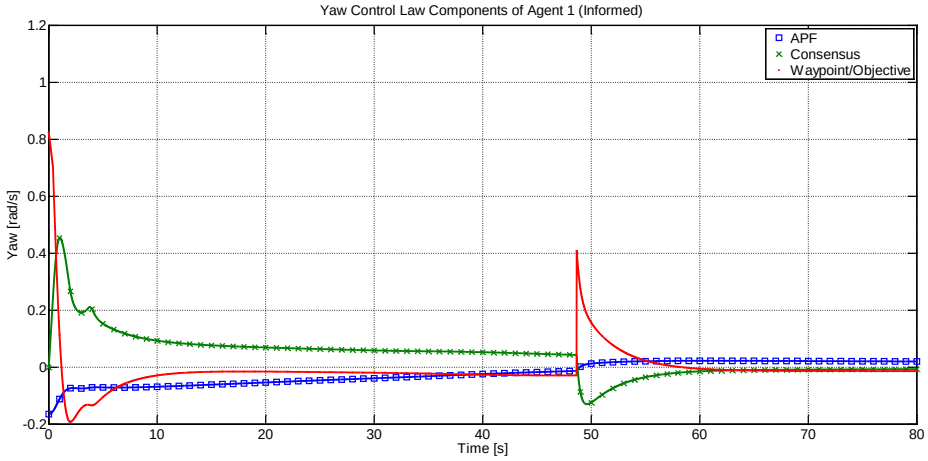

Figure 143 Components of yaw control law of agent 1 (informed).

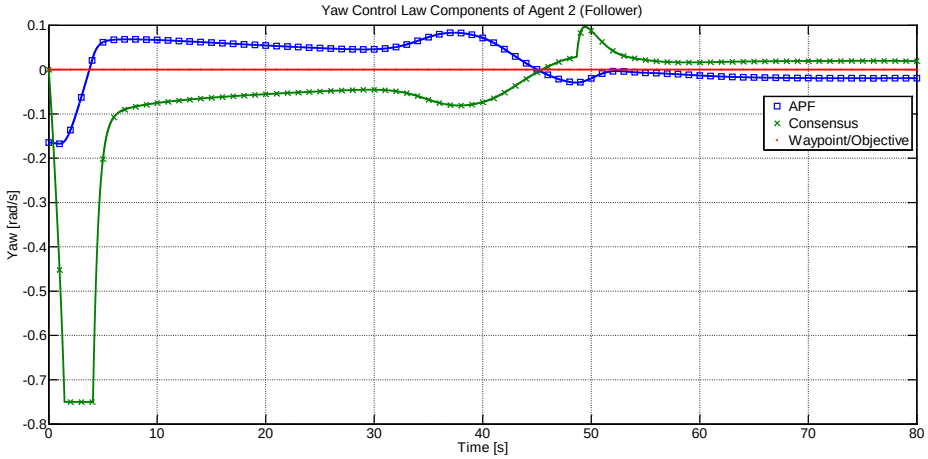

Figure 144 Components of yaw control law of agent 2 (follower).

Progressive Saturation Simulation

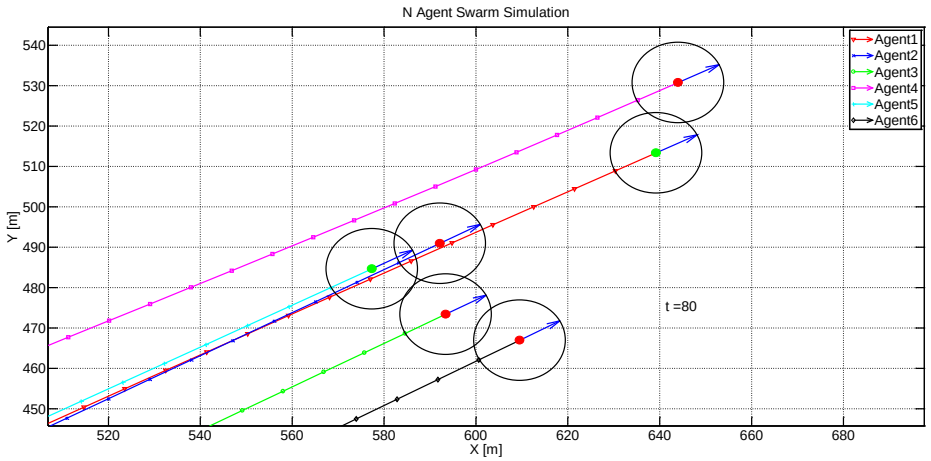

Figure 145 Final positions of 6 agent swarm (progressive saturation).

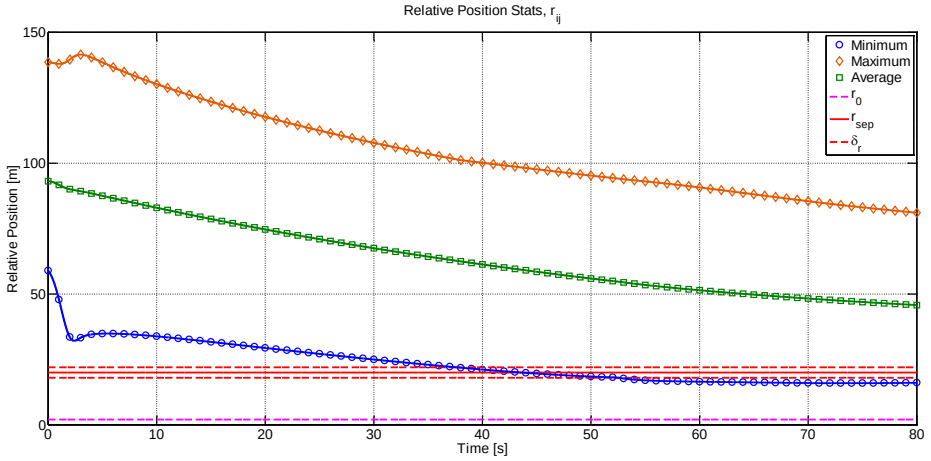

Figure 146 Relative position statistics for 6 agent swarm (progressive saturation).

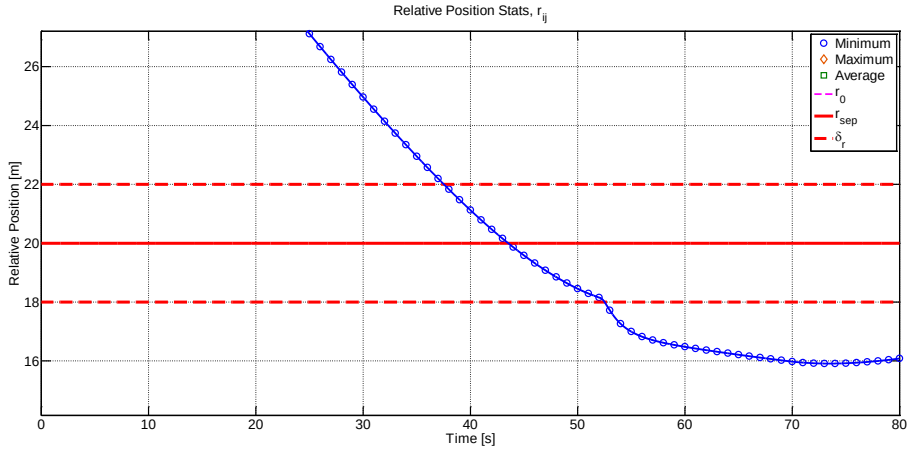

Figure 147 Minimum relative position for 6 agent swarm (progressive saturation).

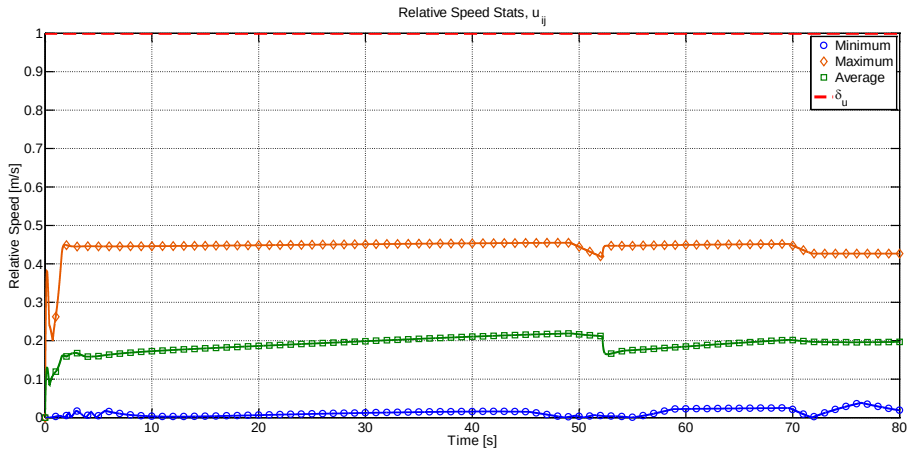

Figure 148 Relative speed statistics for 6 agent swarm (progressive saturation).

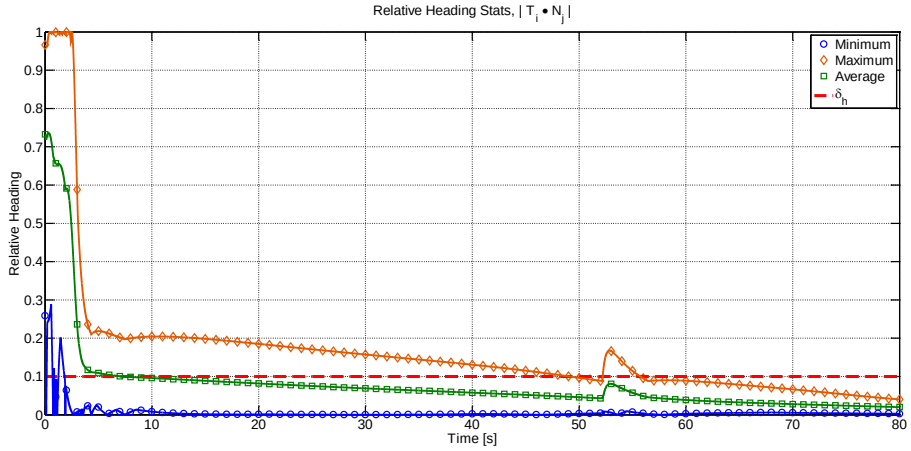

Figure 149 Relative heading statistics for 6 agent swarm (progressive saturation).

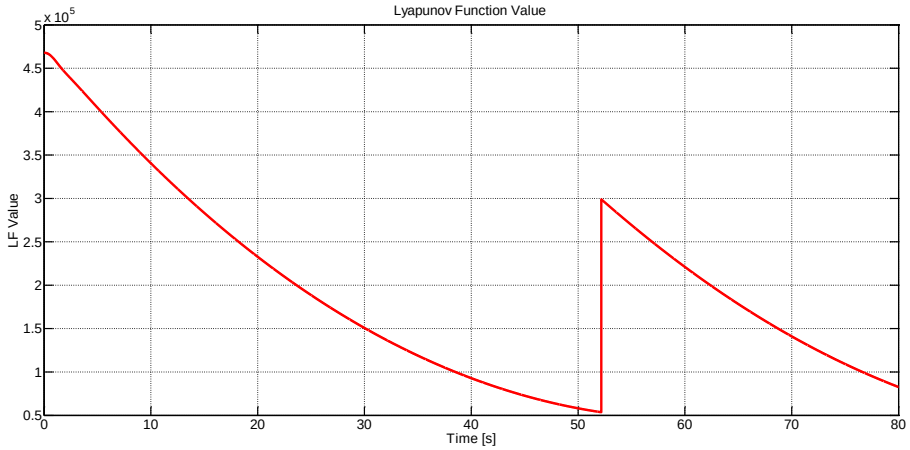

Figure 150 Lyapunov function time history of 6 agent swarm (progressive saturation).

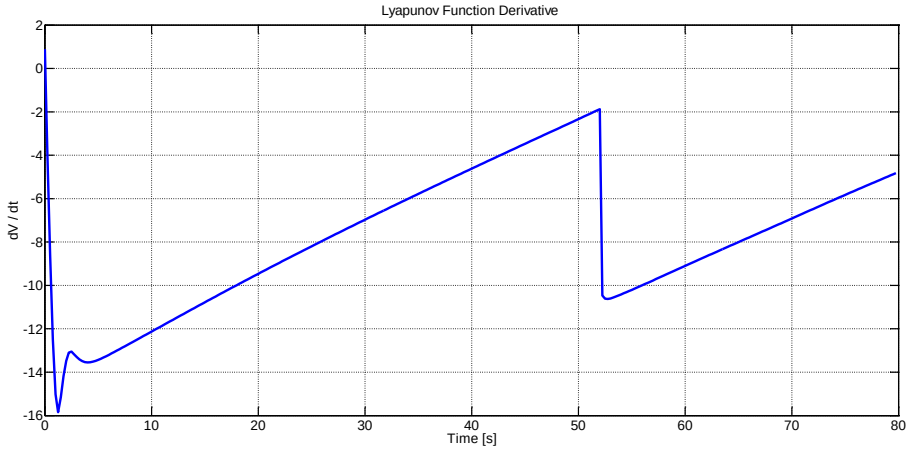

Figure 151 Lyapunov function time derivative. Points are plotted every ¼ second.

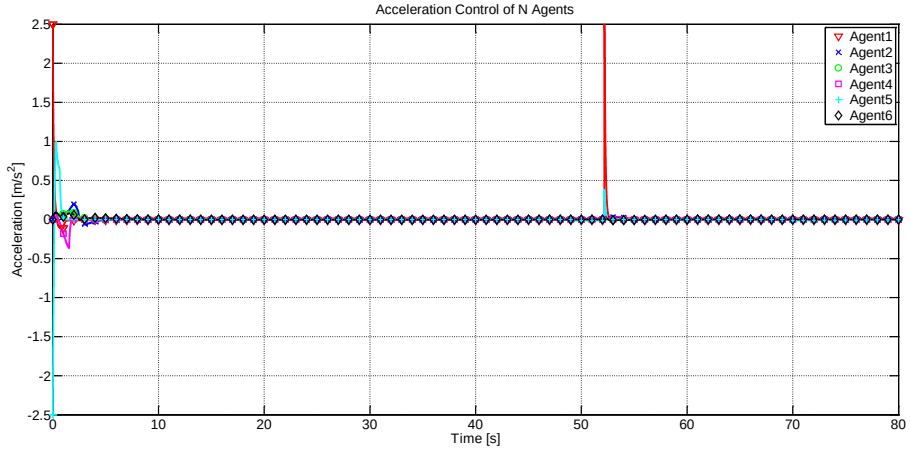

Figure 152 Acceleration control inputs of 6 agents (progressive saturation).

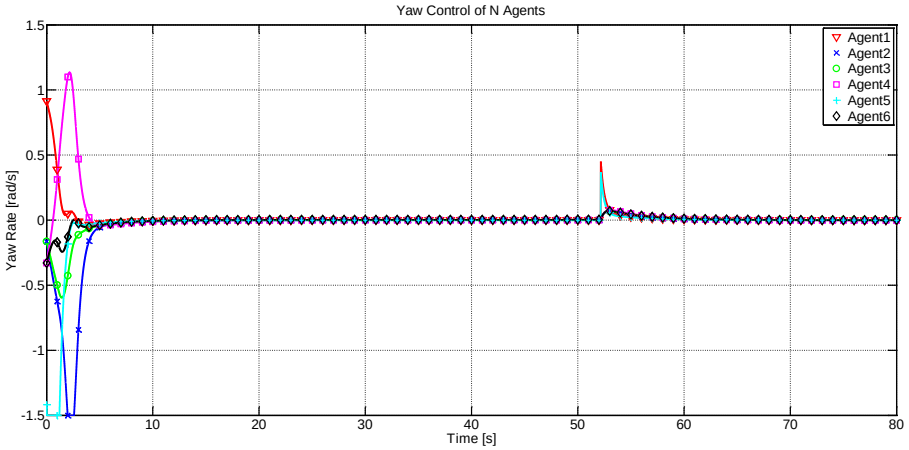

Figure 153 Yaw control inputs of 6 agents (progressive saturation).

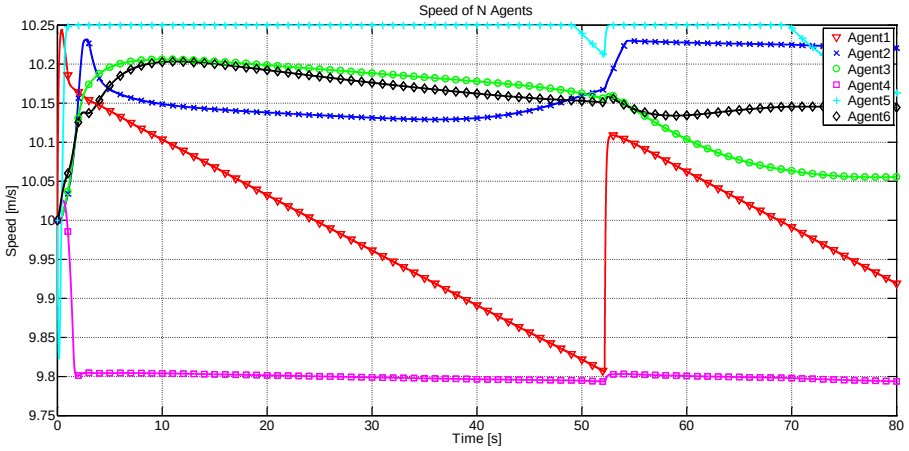

Figure 154 Speed of 6 agents (progressive saturation).

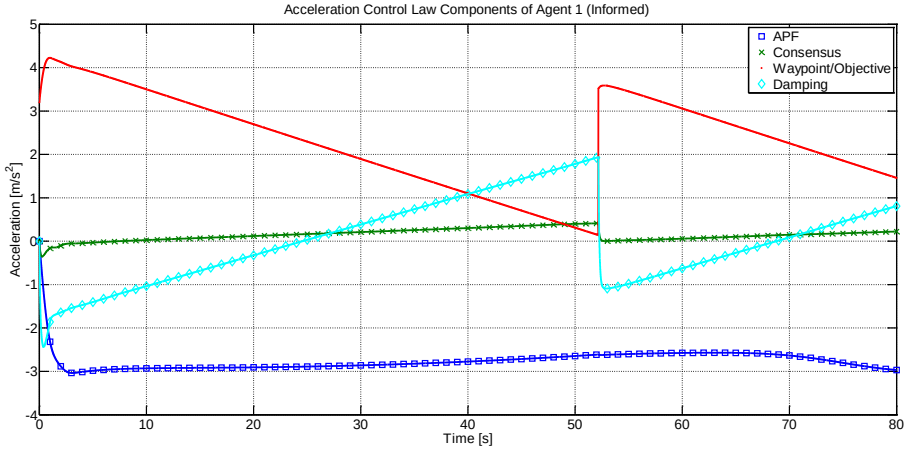

Figure 155 Components of acceleration control law of agent 1 (informed).

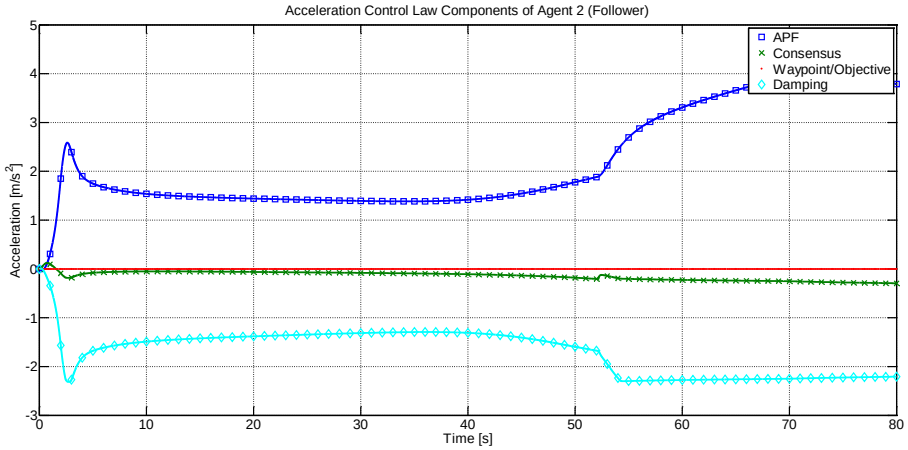

Figure 156 Components of acceleration control law of agent 2 (follower).

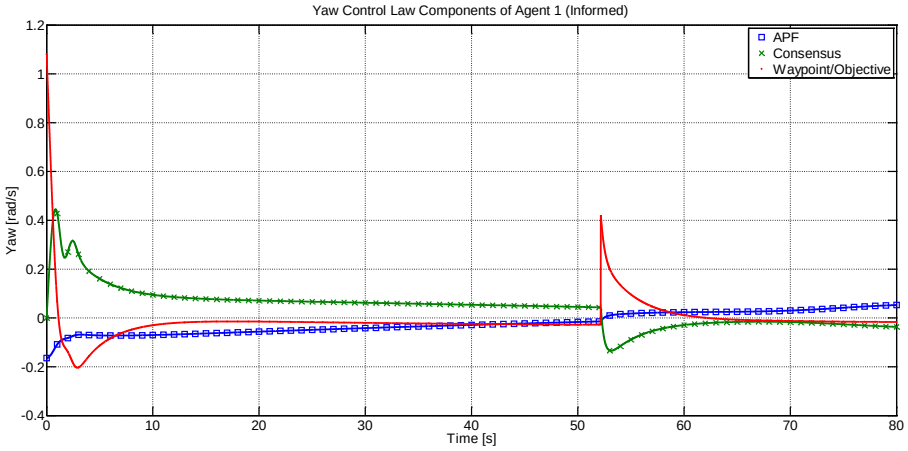

Figure 157 Components of yaw control law of agent 1 (informed).

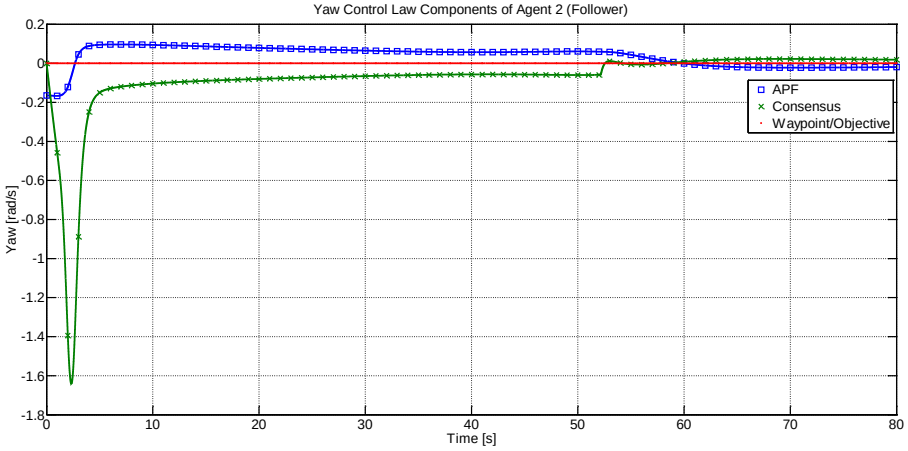

Figure 158 Components of yaw control law of agent 2 (follower).