Design of a Guidance Controller Using Network Topology

by

Clay Jackson Robertson

A dissertation submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Auburn, Alabama
December 10, 2016

Keywords: Autonomous Control, Consensus, Graph Laplacian, Linear-Quadratic-Tracker

Approved by

Andrew Sinclair, Chair, Associate Professor Aerospace Engineering
David Bevly, Professor of Mechanical Engineering
David Cicci, Professor of Aerospace Engineering
John Cochran, Professor of Aerospace Engineering
Emily Doucette, Research Engineer, Air Force Research Laboratory

Abstract

In dynamic environments, the control of a formation of unmanned vehicles remains a significant control problem on both network and guidance dynamics levels. Currently, formations of unmanned vehicles require multiple ground operators to supervise the movement of a formation to effectively execute a task or mission objective. By designating a single vehicle as the lead agent of the formation, control over the entire formation reduces to a distribution of information through a single vehicle, thereby reducing the number of required ground operators. In such architectures, the location of the leader and the manner in which the control is distributed amongst the decentralized formation must suit the desired behavior requested by the ground operator. This work centers on the implementation of a single-leader formation of unmanned vehicles following a desired trajectory. Incorporating the desired formation behavior provided by a ground operator, the locally implemented distributed controllers satisfy a previously established global performance index, leveraging only local information exchange in the decentralized formation.

Consider a fleet of unmanned vehicles moving through an environment under the direction of a single lead agent communicating with a ground operator. The topology of the communication structure is unknown to the operator, knowing only that it communicates with a single vehicle in the formation and provides a desired behavior for the formation as it moves along a series of predetermined waypoints. The formation establishes its leader based on the desired response from the ground operator and the knowledge of the formation structure, as determined exclusively through nearest-neighbor communication.

The contributions of this work are twofold. First, a novel approach to the decentralized selection of a formation leader establishes a leader best suited to follow a desired trajectory while adhering to the behavioral constraints requested by the ground operator. Second, an

optimal control approach to a distributed controller design incorporates trajectory tracking and formation keeping through the underlying communication topology of the unmanned formation. This distributed design provides the leader with feedback from its followers, thereby reducing control usage by the followers to retain the formation structure and allows the leader to track the desired trajectory from the ground operator.

Acknowledgments

The author would like to acknowledge and sincerely thank Dr. Andrew J. Sinclair for his guidance, persistence, and encouragement throughout his time in graduate school. Also, the author sincerely appreciates the faculty of the Aerospace Engineering Department at Auburn University for their years of advisement and teaching opportunities granted and the Munitions Directorate of Air Force Research Laboratory for the inspiration and opportunity to explore this work. Also, the sincerest appreciation and thanks to Dr. Emily A. Doucette for her unwavering support throughout his time at the AFRL REEF.

The author would also like to thank all the friends he's met along the way, too many to mention, without whom this work would have been complete two years earlier. Thank you.

Dedication

To my Mom and Dad. You have been there for me from day one. Thank you for the love, support, encouragement, dedication, trust, and most of all, friendship you have given without bound my entire life. To my sister and oldest friend, Jourdan. Thank you for being my confidant, protector and sounding board throughout this entire journey.

This work is dedicated to you.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

$\bar{\mathcal{A}}$	Normalized adjacency matrix of graph $\mathcal{G}$
$\Delta(\mathcal{G})$	Degree matrix of graph $\mathcal{G}$
$\gamma(i,j;\mathcal{G})$	Geodesic distance from node i to node j
λ	Eigenvalue
$\mathcal{A}$	Adjacency matrix of graph $\mathcal{G}$
$\mathcal{G}$	Undirected graph
$\mathcal{N}_i(\mathcal{G})$	Neighborhood of node i in graph $\mathcal{G}$
$B_i^n(\tau)$	Bernstein polynomial
$C(\tau)$	Bézier curve
$C_c(\mathcal{G})$	Graph closeness centrality
$C_c(i;\mathcal{G})$	Nodal closeness centrality
$C_d(\mathcal{G})$	Graph degree centrality
$C_d(i;\mathcal{G})$	Nodal degree centrality
d_i	Degree of node i in graph $\mathcal{G}$
$L(\mathcal{G})$	Graph Laplacian
n	Size of formation
$P^B(\mathcal{A})$	Bonacich centrality vector

$P^E(\mathcal{G})$	Eigenvector centrality vector
$P^K(\mathcal{G})$	Katz centrality vector
$P^{K2}(\mathcal{G})$	Second Katz centrality vector
P_i	Bézier control point
v	Eigenvector
W	Waypoint
$X(t)$	Consensus state
DAA	Decentralized Adjacency Algorithm
DLC	Dynamic Laplacian Centrality
DLQGT	Decentralized Linear-Quadratic-Gaussian-Tracker
LQT	Linear Quadratic Control

Chapter 1

Introduction

With the advent of increased computational ability of onboard computers and the potential to maintain large teams of unmanned vehicles simultaneously, decentralized distributed control of formations has gained sizable interest over the past years. In an effort to develop flexible formations that handle changing environments, attention has focused on creating robust methods that adjust to vehicle attrition and environmental uncertainties. This dissertation develops a guidance controller for a formation of unmanned vehicles using characteristics of the underlying communication structure. This controller exploits inherent properties of network topology, utilizing the communication structure, desired formation behavior from a ground operator, and other intrinsic attributes as design variables to satisfy a global performance index using only local information exchange.

A multi-agent formation of autonomous vehicles is tasked to maneuver through an obstructed environment, following a set of sequential waypoints provided by a ground operator. The agents share a global performance index, but the control laws considered herein are decentralized, since the agents are unaware of the commanded trajectory. This permits them to only sense the state of their nearest neighbors while the waypoints that describe the trajectory are known exclusively to the global leader. This trajectory is supplied by a ground operator sharing a single link with the formation leader who is elected by the decentralized formation. Sensor data of each agent is processed locally using a Kalman filter so that the state of the leader may be indirectly observed.

This work develops a decentralized control strategy based on the desired response of the formation to a single exogenous input provided by a ground operator and the communication capabilities inherent to the previously established interactions amongst the unmanned

agents. The controller allows an agent variable dependence on its leaders and followers, ranging from a leader-follower to a virtual-structure control architecture. In a leader-follower control strategy, a hierarchy of leaders and followers are designated and the control on the leader is not affected by the state of the followers. A virtual-structure control strategy treats the formation as a rigid body without any detailed hierarchy. In this application, the aforementioned architectures act as a bound on a control continuum such that a leader shares a dependence on the state of its followers, and vice versa. The location of the leader in the topology elicits certain types of behavior from the formation in terms of its response to an exogenous input, thereby restricting the bounds of the allowable control to stabilize the formation. Balancing these constraints with the desired response of the network, as requested by a ground operator, the individual agents determine their own control by using limited information of the network to best satisfy the wants of the user and constraints of the graph topology.

This dissertation centers around the control of a single-leader formation that reflects the wants of the ground operator. Given that a ground operator's command will not exclusively align with the leader-follower or virtual-structure control strategies necessitates the design of a controller that exists between these two bounds. Section 1.1 provides an account of previous controller designs for formations of unmanned vehicles. The inclusion of graph theory to aid in solving these problems motivates a brief overview of graph theoretic provided in Section 2.1. The problem addressed requires the selection of a single leader, best suited to reflect the wants of the ground operator and the needs of the formation structure. With this, an account of various centrality measures used to identify key players in a network is provide in Chapter 2. This section reveals the need for a new centrality measure that incorporates the dynamic capabilities of the formation while considering only nearest-neighbor interaction. A formal definition of this centrality measure, termed the dynamic Laplacian centrality, displays its applicability to the problem considered in identifying agents dynamically capable of leading the formation.

As previously stated, since the formation needs to adapt to attrition, a method of identifying changes to the formation is required. Chapter 3 illustrates a recursive method of checking the formation for changes. This method, termed the decentralized adjacency algorithm, coupled with the leadership parameter described in Chapter 2, leads to the development of an inherent hierarchy. Section 4.3 provides the coupling of these procedures and the instantiates a hierarchy into the decentralized formation. The ground operator provides the formation leader with a series of waypoints to follow such that the manner in which the formation approaches the waypoints becomes a design parameter of the mission. Chapter 5 presents a novel approach to determining the desired track through these waypoints with the application of Bézier curves which provides the ground operator with an additional level of control of the behavior of the formation along the track. Finally, Chapter 6 presents the optimal control approach to designing the distributed control architecture for the decentralized formation. A brief description of optimal control and how it applies to this problem provides the foundation for the controller design which incorporates the wants of the ground controller, the restrictions of the communication structure while retaining the formation structure, and follows a desired trajectory.

1.1 Literature Review

The centralized control problem involving predetermined sets of leaders and followers for distributed control has long since been solved for a fully connected information exchange network [1, 2, 3]. Recognizing the robust appeal of decentralized formation control, a recent push in multi-agent control has expanded upon these classic examples to include localized formation control for more sparsely connected networks [4]. Coordinated control of multi-agent systems has received great attention due to their improved productivity in completing complex tasks over their individual counterparts. Coordinated decentralized control opens the door to multiple fields of study including space, terrestrial and oceanic explorations, spacecraft formation flying, cooperative surveillance, and sensor networks [5, 6]. When establishing

a hierarchy, separate sets of leaders and followers work to solve the n-trailer problem for a convoy of vehicles through the local sensing of position and velocity [7, 8, 9]. Alternatively, a consensus approach to virtual-structure control is typically implemented when coordinating vehicle formations for directed and undirected communication topologies because of its application to graph theory and the analytic capabilities for stability analysis [10, 11]. In the application herein, elements of the consensus approach to formation control help to retain the formation structure while relaxing the predetermination of a leader-follower hierarchy.

The intermingling of graph theory and control theory has a rich history, dating back to applications of decentralized controllers investigated in the 1970s [12]. Modeling the interconnections of a formation through graph theoretic matrices allows the application of algebraic tools in analyzing stability and the application of topologies in decentralized formations. The computational ease of these methods allows them to be utilized onboard flexible formations topologies and in dynamic mission environments. To mitigate discrepancies between network and formation dynamics, efforts by [13], [14], [15],and [16] work to control the dynamics of the network to complement the motion of the formation. Original work by Fiedler found that certain graph theoretic matrices provided topological descriptions of the connectivity and, in turn, the convergence properties of a network. This measure of connectivity, named the Fiedler value, marked the onset of investigations linking communication dynamics to the spectral properties of the network topology [17]. Other applications of the Fiedler value surfaced in the analysis of networks for data mining, consensus agreement and data clustering where it provided an ad hoc estimation of an otherwise computationally expensive grouping of network nodes based on the topological structure [18]. With this, elements of graph theory help bridge the gap between the restriction of the communication network and the control capabilities of the formation.

The use of unmanned vehicles in decentralized settings led to combining methods of formation control and graph theory to determine regions of stability based on the communication characteristics of the network. Fax and Murray used tools from algebraic graph theory

to relate the underlying communication structure to the formation stability [19]. Using a Nyquist criterion, the eigenvalues of the graph Laplacian helped prove formation stability. To avoid restrictions inherent in directed graphs, the proposed controller estimated the location of the formation center such that delays or limited information of individual vehicles did not hinder the convergence of the formation. The advent of these methods postulated that further extensions into nonlinear systems with variable time delay and controllers cable of stabilizing command over convoys, or strings, of vehicles relied on controllers that better utilized the topology of the communication network [20]. An application of these findings will aid in the development of the guidance controller that depends on the stability and topology of the network.

Several aspects of cooperative control of unmanned vehicles have explored various control schemes in which limited bandwidth and communication present complications for decentralized control. In 2007, Sorensen and Ren exploited nearest-neighbor communication in formation networks, arriving at a formation control using either a leader-follower or virtual-structure framework [10]. Through consensus, an estimation of the formation center allowed the vehicles to estimate their location in the formation based on nearest-neighbor information exchange. For directed networks where the leader must occupy the root of a spanning tree, a selected formation control could generate, what Ren referred to as "simple" followers who lacked direct communication to the formation leader, resulting in disjoint formations. Other approaches to combining the leader-follower and virtual-structure formations by Zhong-Hai et al. used clustering to find the location of multiple leaders in underwater formations groups such that the leaders of each formation used a leader-follower formation types while the individual clusters of followers used virtual-structure control to follow their local leader [21]. In Ren and Zhong-Hai's works, controllers for single and double-integrator dynamics utilized the feedback capabilities in virtual-structure formations and the implications of limited inter-vehicle information feedback in leader-follower formations proved unfavorable because the follower networks may become disjoint and the formation may falter. By first addressing

the topology of the network, identifying locations for control inputs, and designing around the predisposed network limitations a formation may operate without becoming disjoint.As seen from the previously mentioned application of graph theory in the construction of viable control laws for decentralized formations, the next chapter provides a brief overview of the basic graph theory essential to the development of the later formulated distributed control law.

1.2 Contributions

This dissertation addresses the feasibility of manipulating a decentralized formation of unmanned vehicles from its inception to its completion by utilizing only the limited observation capabilities of each decentralized agent. Throughout this investigation, many novel and interesting contributions surfaced as a result of the need to address this single-leader scenario in a new way. A new centrality measure is introduced that leverages the dynamic characteristics of the decentralized network and characterizes the individual nodes by their contributions as a viable formation leader. This centrality measure then contributes to a novel approach for hierarchy development and illustrates the iterative method of establishing it through neighbor-to-neighbor communication exchange. The formation leader is provided with a set of waypoints to intercept during the mission. This work presents a novel variation of waypoint following by implementing Bézier curves and controlling the shape of the desired trajectory through information provided by the ground operator. Lastly, a decentralized application of the linear-quadratic-tracker provides an novel alternative to formation keeping and trajectory tracking problems. The locally implemented controller exploits an agent's aggregated knowledge of the formation structure to locally determine its control in satisfying a global performance criterion. Additional variations of this decentralized controller further limit the influence non-neighboring members have on an agent's locally implemented control, thereby stabilizing the formation through purely decentralized methods. Cumulatively,

these contributions provide a new approach to decentralized control and present them in a realistic environment for effective implementation.

Chapter 2

Network Centrality

Elements of graph theory provide a logical correlation between the communication structure of the unmanned vehicles and a mathematical representation of the data. With this, the mechanical nature in which the agents receive or transmit data, the modes of communication, and other physical traits associated with the exchange of information become irrelevant in the description of the underlying communication structure. Using a graphical representation of the network as to only describe the direction of communication within the underlying topology thereby avoids these physical abstractions altogether. This investigation intends to expand upon the application of graph theory to the selection of a single leader in a network to accomplish a mission objective, not to provide an exhaustive description of graph theory. To this end, a brief description of basic graph theory should adequately provide the tools needed to implement the content herein.

2.1 Preliminaries

A network, or graph, structure $\mathcal{G}$ consist of vertices V (nodes, players, agents, etc.). Consider a set of 2-element subsets, denoted as $[\cdot]^2$. Therefore, an edge set is defined as $E \subseteq [V]^2$ such that E is a particular subset of $[V]^2$. With this, the graph, $\mathcal{G}$, is defined as the pair $\mathcal{G} = (V, E)$, given in Fig. 2.1 [22, 23].

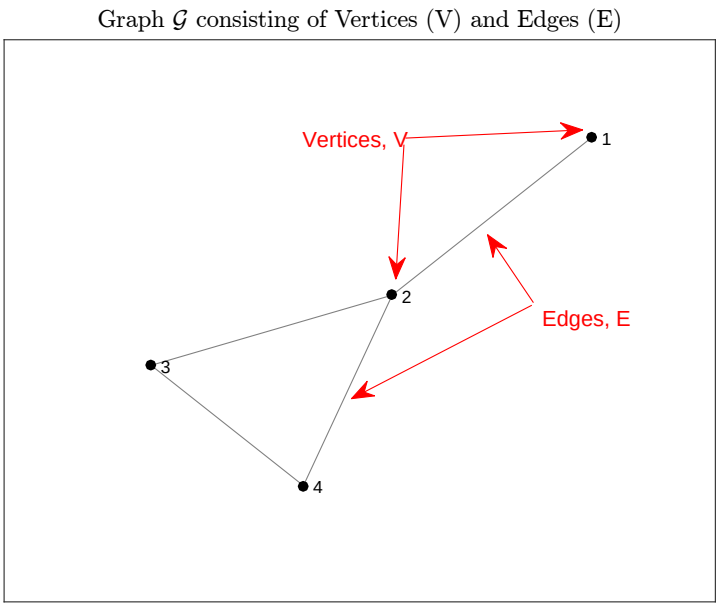


Figure 2.1: Example of an undirected, connect graph consisting of vertices (V) and edges V.

The vertices make up a vertex set denoted as $V = \{v_1, v_2, \ldots, v_n\}$, representing n elements of a set. In terms of the problem in question, each vertex represents an unmanned vehicle such that each element of the set has a specific identification. Edges of a network represent communication between vehicles. By convention, elements of the edge set $E = \{v_i v_j\}$ states that the communication transmits from vertex i to vertex j such that $i, j = 1, \ldots, n$. On a graph, points represent agents while arrows represent communication originating from the transmitting agent and terminating at the receiving agent. If subsets $\{v_i v_j\} \in E$ and $\{v_j v_i\} \in E$, then the edge represents bidirectional, or undirected, communication where both agents transmit and receive information along the graph edges [24]. Unless otherwise stated, edges without arrows indicate undirected communication.

When describing the relationship between vertices and edges of a graph, the adjacency, degree, and graph Laplacian matrices help to represent the graph algebraically. The adjacency matrix provides a row ordered depiction of each node's first-degree neighbors such

that a single edge connects the nodes.

$$\mathcal{A}_{ij}\left(\mathcal{G}\right) = \begin{cases} 1, \text{ if } (i,j) \text{ is an edge in } \mathcal{G} \\ 0, \text{ otherwise} \end{cases} \tag{2.1}$$

Equation (2.1) describes a binary matrix $\mathcal{A}$ such that the sum of the rows defines the degree of connection, $d_i\left(\mathcal{G}\right)$, for each ordered node i of the graph, $\mathcal{G}$, and each filled element of a column j indicates an existing communication link between nodes i and j.

$$\Delta\left(\mathcal{G}\right) = \text{diag}\left(\left[d_1\left(\mathcal{G}\right)\ d_2\left(\mathcal{G}\right)\ldots\ d_n\left(\mathcal{G}\right)\right]\right) \tag{2.2}$$

Equation (2.2) defines the diagonal degree matrix whose diagonal elements contain the row sum of the adjacency matrix $\mathcal{A}$, or the degree, $d_i\left(\mathcal{G}\right)$ of each node, i. Using the example graph in Fig. 2.1, the accompanying degree matrix is given in Eq. (2.3).

$$\Delta\left(\mathcal{G}\right) = \text{diag}\left(\left[1,\ 3,\ 2,\ 2\right]\right) \tag{2.3}$$

Different combinations and normalizations of the adjacency and degree matrices form the basis of graph theory in that the two matrices provide an algebraic description of a physical network [25]. One such manipulation of the adjacency matrix comes from its normalization by the degree of each node, $d_i\left(\mathcal{G}\right)$.

$$\bar{\mathcal{A}}_{ij}\left(\mathcal{G}\right) = \begin{cases} \frac{1}{d_i(\mathcal{G})}, \text{ if } (i,j) \text{ is an edge in } \mathcal{G} \\ 0, \text{ otherwise} \end{cases} \tag{2.4}$$

Equation (2.4) incorporates the structure of the network along with the degree of each connection to fully describe the topology of the graph. A comparison of the adjacency and normalized adjacency matrices is given in Eq. (2.5) for the example graph provided in Fig. 2.1, where the rows are ordered by the unique identification of the nodes in ascending

order.

$$\mathcal{A} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \qquad \bar{\mathcal{A}} = \begin{bmatrix} 0.000 & 1.000 & 0.000 & 0.000 \\ 0.333 & 0.000 & 0.333 & 0.333 \\ 0.000 & 0.500 & 0.000 & 0.500 \\ 0.000 & 0.500 & 0.500 & 0.000 \end{bmatrix} \tag{2.5}$$

The nonnegative construction of $\bar{\mathcal{A}}$ holds specific spectral properties realizable through the theory of nonnegative matrices, particularly the Perron-Frobenius theory for irreducible, or strongly connected, graphs. The strongly connected stipulation requires that each node be reachable from all other nodes in the network through a finite number of edges. Satisfying this constraint through the use of undirected graphs, the Perron-Frobenius theorem applies directly.

Theorem 2.1 *(Perron-Frobenius) Let $\mathcal{F}$ be a nonnegative, irreducible matrix and let $\rho(\mathcal{F})$ be the spectral radius of $\mathcal{F}$ such that $\rho(\mathcal{F})$ is the supremum of the absolute value of the elements in the spectrum. The following are true:*

(1) $\rho(\mathcal{F}) > 0$

(2) *$\rho(\mathcal{F})$ is a simple eigenvalue of $\mathcal{F}$, and any eigenvector of $\mathcal{F}$ of the same modulus is simple*

(3) *$\mathcal{F}$ has a positive eigenvector x corresponding to $\rho(\mathcal{F})$ such that every element of the eigenvector is positive*

This spectral description of nonnegative matrices aids in connecting the graph theoretic properties of the network to the dynamic properties of the unmanned formation. The construction of the graph Laplacian matrix branches the application of the network graph for the communication dynamics to the unmanned formation, thereby relating a communication topology to the guidance controller. Define a matrix L based on the adjacency of neighboring nodes in the network.

$$L(\mathcal{G}) = \Delta(\mathcal{G}) - \mathcal{A}(\mathcal{G}) \tag{2.6}$$

For a connected graph, the graph Laplacian has a few properties that help bound the spectrum of the graph which are covered extensively in Chung's monograph on the spectrum of the graph Laplacian [26]. Using the same approach to normalizing the adjacency matrix in Eq. (2.4), an extension of Eq. (2.6) reveals a simple connection to nonnegative graphs.

$$\mathcal{L}(\mathcal{G}) = I_n - \bar{\mathcal{A}}(\mathcal{G}) \tag{2.7}$$

Equation (2.7) finds the difference between the $n \times n$ identity matrix, I_n, and the normalized adjacency matrix to formulate the normalized graph Laplacian matrix. The unit shift in Eq. (2.7) also shifts to the eigenvalues of the graph Laplacian such that the eigenvalues, λ, of $\mathcal{L}$ are equal to $(1 - \rho)$ of $\bar{\mathcal{A}}$. Given the spectral properties set forth by the Perron-Frobenius theorem, the following conclusions may be formed about the graph Laplacian.

Proposition 2.2 *L has a simple eigenvalue of zero and associated eigenvector $v = \mathbf{1}$, where $\mathbf{1}$ represents the ones vector of length n.*

Proposition 2.3 *If $\mathcal{G}$ is strongly connected $0 = \lambda_0 < \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_k$ where $k = 0, 1, \ldots, n-1$, and λ_k denotes the eigenvalues of L ordered by increasing magnitude.*

Proposition 2.4 *If $\mathcal{G}$ is undirected, then all the eigenvalues of L are real.*

Proposition 2.4 comes directly from the Perron-Frobenius theorem for nonnegative matrices which localizes the eigenvalues of the normalized graph Laplacian between zero and two, such that the eigenvalues lie on a disk of radius one (referred to as the Perron disk), centered at $1 + 0j$ where $j = \sqrt{-1}$. Given Proposition 2.4 and the bounds provided by the Perron disk, the normalized eigenvalues of the graph Laplacian will lie on the real axis between zero and two. Propositions 2.2 and 2.3 state that the lowest eigenvalue of the connected graph's Laplacian matrix has a magnitude of zero, associated with the ones eigenvector of length n. This characteristic of the graph Laplacian, along with the increasing magnitude of the eigenvalues, will assist in describing the convergence properties of the first-order dynamics of the formation.

2.2 Network Centrality

To designate a node as significant, or deserving of leadership, implies a process of identifying a qualitative notion of importance. This process would observe the qualities of all the nodes and weight them accordingly for a given objective, or mission, as to select the node most capable in the network. Therefore, this measure of centrality has many definitions such that the method of determining centrality characterizes the implied interpretation of leadership. Using elements of graph theory, the proximity of a node, degree of connection, and importance of neighboring nodes contribute to different measures of centrality. A measure of proximity defined as the greatest distance provides a different approximation of the centrality measure than a proximity based on the shortest distance, thereby altering the interpretation and application of the centrality measure. Most centrality measures define distance as the geodesic, or shortest path, between nodes but if the communication does not travel the shortest path, the initial assumption will result in a misrepresentation of the centrality measure originally intended. Unless otherwise stated, the networks considered herein model unweighted lines of communication such that the cost of traveling along any edge takes the value of one, making the distance equal to the number of edges traveled along the path between nodes.

Different centrality measures take precedence over others when restricting the direction of information flow along communication lines and across nodes of the network. If the topology has branching trees or cycles, the possibility of stagnation points or information replication at a node will affect the selection of the centrality measure used. The direction of information flow can also change the centrality meaning. Some measures assume the information takes the shortest path (geodesic) yet if the information randomly selects the route to move, much like a Markovian process, certain centrality measures do not reflect the information expected from its metric due to misinterpretation between definition and application [27].

In a directed network, the information flows in a one direction. Networks without cycles broadcast information along distinct nodes and edges, reaching nodes only once. This directed trajectory describes graph-theoretic paths, trails, and geodesics. Communication in directed networks with cycles still have a restricted trajectory along its links but information may revisit the nodes. This creates a challenge in information update such that a node receives outdated information, much like attitudes or opinions in social networks. Because of the cyclic nature in the information exchange of the network, centrality measures must take into account the communication structure, as well as the topology, when applying a centrality measure for leadership selection. Metrics that weight a degree of connection over the importance of an agent's connections better characterize a leader for directed networks.

Conversely, undirected graphs enable information to traverse a node or edge multiple times such that the centrality description relies more on the importance of a node's neighbors rather than the immediate connections of the node. With information able to flow unrestricted along an edge, a neighbor's influence carries more weight, in terms of leadership qualities, yet its level of communication should continue to bear weight in the centrality ranking. With this, measures that determine a node's prestige in a neighborhood account for the importance of a node's neighbors, as well as, the node's location in the network structure. A brief presentation of centrality measures typically implemented in the following sections will provide an understanding of proximity and prestige centralities, to later define a new centrality measure through the graph Laplacian.

2.3 Degree Centrality

Degree centrality provides a very basic description of a node in a network based on its level of connection. From a graph-theoretic perspective, calculation of the degree, $d_i(\mathcal{G})$, of a node simply amounts to the row sum of the network adjacency matrix. Another way to describe this measure is by the cardinality of node i's neighborhood, $\mathcal{N}_i(\mathcal{G})$, defined as the

set of nodes adjaecent to node i, such that $d_i(\mathcal{G}) = |\mathcal{N}_i(\mathcal{G})|$ [28].

$$C_d(i;\mathcal{G}) = \frac{d_i(\mathcal{G})}{n-1} = \frac{|\mathcal{N}_i(\mathcal{G})|}{n-1} \tag{2.8}$$

By normalizing the centrality measure by the largest number of connections possible, $(n-1)$, the degree centrality ranks a node between zero and one, shown in Eq. (2.8). This centrality measure relies only on the number of first-degree connections in the network, rather than the trajectory of the information flow. In terms of leadership, deal makers who exchange information with others have a high degree centrality. With this, selection of a leader based on the degree centrality implies that a powerful node can reach a large density of the network through its first-degree connections.

The ability to immediately communicate with a large density of the network describes a suitable leadership quality of an agent. Given that degree centrality measures the quality of a node by its degree cardinality, situations and mission objectives that require access to a large density of the network immediately by an exogenous input lend themselves to a leader with a high degree centrality rank. This does not account for the structure of the network outside of first-degree connections, making this measure a local description of the node in the network, restricted to its neighborhood.

Applying this concept of degree centrality for nodes on the level of the network graph, a comparison of networks across varying topologies and size helps identify networks that suit themselves to degree centrality based leadership [28].

$$i^* = \arg\max_i C_d(i;\mathcal{G}) \tag{2.9}$$

$$C_d(\mathcal{G}) = \frac{\sum_{i=1}^{n}\left[C_d(i^*;\mathcal{G}) - C_d(i;\mathcal{G})\right]}{n-2} \tag{2.10}$$

Equation (2.10) finds the average difference between the greatest degree centrality ranking, $C_d(i^*;\mathcal{G})$, and all other degree centrality measures, $C_d(i;\mathcal{G})$, normalizing by the largest

number of connections possible for a pair of nodes, $(n-2)$. The resulting degree centrality of the graph values the level of connection of the entire network. Homogeneous graphs, where the degree of connection regularly distributes throughout the graph, result in a lower graph degree centrality rankings. Complete graphs that utilize all possible connections, or k-regular graphs with each node sharing k connections, convey low graph degree centralities suggesting a more homogeneous topological construct in the absence of nodal clusters. With this, graph degree centrality provides a first approach to investigating the existence of a dominating player in the network. Networks returning a low $C_d(\mathcal{G})$ value distribute connections uniformly in the topology such that applying a nodal degree centrality to locate a leader will not result in an unanimous decision by the network on a predominant leader.

2.4 Closeness Centrality

Closeness centrality measures a node's proximity in terms of its distance to all other nodes in the network such that the closeness centrality measures a nodes level of independence in the network. An independent node does not rely on intermediaries to influence other nodes to communicate with the entire network because they require less assistance in communicating with all other nodes in the network [29]. Accordingly, an agent's independence in a network provides a suitable leadership quality based on its location and proximity to all other nodes in the network.

$$C_c(i;\mathcal{G}) = \frac{n-1}{\sum_{j\neq i}\gamma(i,j;\mathcal{G})} \tag{2.11}$$

The closeness centrality, $C_c(i;\mathcal{G})$, of a node i, given in Eq. (2.11), defines the measure as a function of distance $\gamma(i,j;\mathcal{G})$, from the node i to node j. Letting the distance γ define a geodesic, then the denominator of Eq. (2.11) provides the accumulation of the "farness" node i has to all other nodes in the network. Consequently, the inverse of the farness measure provides a closeness measure of node i to all other nodes.

To correctly interpret centrality measures with proximity considerations, trajectory and distance must have a clear definition prior to calculation. Geodesics provide a natural measure of distance since a broadcast signal moves along every connection from the input node, including the shortest path. Therefore, the ambiguity of distance does not become a factor for the intended application herein due to communication delays occurring at the nodes, not in propagation along the edges.

Closeness centrality of a graph relates to the compactness of a graph. Using the same approach as in the case of degree centrality, a ratio of the gross difference in node closeness centrality, $C_c(i, \mathcal{G})$, to the maximum possible difference provides a means of comparison between graphs of varying topology.

$$i^* = \arg\max_i C_c(i; \mathcal{G}) \tag{2.12}$$

$$C_c(\mathcal{G}) = \frac{(2n-3)\sum_{i=1}^{n}\left[C_c(i^*; \mathcal{G}) - C_c(i; \mathcal{G})\right]}{(n-2)(n-1)} \tag{2.13}$$

Equation (2.13) accumulates the difference between the maximum closeness measure, $C_c(i^*; \mathcal{G})$, and compares it to all other nodal measures, $C_c(i; \mathcal{G})$. To interpret the graph closeness centrality, $C_c(\mathcal{G})$, consider two extremes of connected graphs: the star and complete graph. A star graph, or complete bipartite graph, consist of a single node sharing a connection with all other nodes with degree one. This represents the most compact structure such that the graph closeness centrality receives a maximum value of one. In a complete graph, each node shares a unique connection to all other nodes of the network thereby utilizing every possible connection. This represents a homogeneous graph, in all respects, resulting in a closeness centrality measure of zero [29]. Used as a first approach in applying a centrality metric, high graph centrality rankings suggest a higher likelihood of locating clusters and dominate figures in the network. From a social networking standpoint, individuals near the center of network communities have a higher closeness centrality ranking. They occupy the shortest paths of information spread, making them important locally, yet not typically recognized as

dominant figures in the network [30]. To this end, leadership situations that require shortest paths of information spread to increase time response of a network, may lend themselves to a closeness centrality ranking. In the event that a network exists in clusters, a nodal centrality ranking indicates the location of the communities carrying more influence in the network.

The closeness centrality looks more to the topology of the network than the degree centrality in determining the importance of the individual nodes yet it does have some setbacks. Nodes close to dense populations and far from outlying nodes will receive a measure similar to nodes with a moderate distance to all nodes of the network. With this, more information about the connections of a node's first-degree neighbors and the overall topology of the network helps better define the hierarchy of the network structure.

2.5 Prestige Centrality

Prestige centrality takes into account a richer range of direct and indirect influences in a network on individual nodes. The centrality measures considered insofar have provided information about a node's immediate connections or their relative position in the network with respect to all other nodes. The ranking of these measures does not take into account the aptitude or importance of surrounding nodes. Used as a measure of influence in undirected communication, and power in exchange networks, prestige centralities use the importance of a node's neighbors to rank its own importance [28]. Loosely mirrored through a combination of the degree and closeness centralities, an understanding of an agent's influence on its neighbors and how that influence translates to the network provides a suitable leadership quality of an agent. The type of prestige centrality used depends on the topology of the network, flow characteristics, and the type of information desired about the importance of a node. Abstaining from an exhaustive review of all prestige centralities, this investigation identifies centralities that obtain qualities for a single network leader.

2.5.1 Eigenvector Centrality

Eigenvector centrality serves as a mathematical framework to characterize agents in a network. Communication graphs visualized through the use of nodes and edges naturally characterize individuals based on their neighboring connections. By expressing these interactions in a matrix, eigenvalues naturally characterize nodes through a set of characteristic values. One example uses the eigenvalues and eigenvectors of the adjacency matrix.

$$\alpha P^E = \mathcal{A} P^E \tag{2.14}$$

$$\alpha_1 \geq \alpha_2 \geq \ldots \geq \alpha_n \tag{2.15}$$

Ordering the eigenvalues of the adjacency matrix, given in Eqs. (2.14) and (2.15), the largest magnitude eigenvalue, α_1, points to the eigenvector used as the centrality vector, P^E, where the superscript E denotes the vector P as the eigenvector centrality. The elements of P^E measure the prestige of the corresponding nodes, with higher magnitudes indicating more central nodes in the network [27, 31]. A high eigenvector centrality works to identify the powerful players in the network. These nodes have positions of power and influence the network as a whole [32]. With the task of identifying an agent for leadership, eigenvector centrality points to agents with greater global influence. In situations where the network should act cohesively or need a dominant agent, eigenvector centrality produces suitable leaders. The problem with this centrality lies in the way it handles peripheral nodes residing outside the community of the high centrality ranked agents. Using the structure of the network as the mechanism for its ranking, the magnitude of a node's importance reflects its access to the leader. With this, centrality rankings with a wide spectrum present situations where outlying nodes remain difficult to influence. By accompanying the eigenvector centrality with a graph centrality measure such as degree or closeness centrality to determine the homogeneity of the network, a justification to losing access to peripheral nodes via an eigenvector centrality leader becomes available.

2.5.2 Katz Centrality

Katz centrality, an extension of the eigenvector centrality, accounts for the degree of connection for each node and their influence in the network. As with eigenvector centrality, walks model the information trajectory, using nodes and edges multiple times to move around the network. To measure not only the importance of location and distance from all nodes of the network, but weight them based on their respective degree of connection, the Katz centrality makes use of the normalized form of the adjacency matrix, $\bar{\mathcal{A}}$.

$$\bar{\mathcal{A}}_{ij} = \frac{\mathcal{A}_{ij}}{d_j\left(\mathcal{G}\right)} \tag{2.16}$$

Equation (2.16) normalizes each column of the adjacency matrix with respect to the in-degree of the node. For undirected matrices, row or column normalizations remain equivalent due to the symmetry of the adjacency matrix. The Katz centrality provides a more in-depth look at directed networks and the effect directed communication has on the influence of a node in the network [28].

$$P^K\left(\bar{\mathcal{A}}\right) = \bar{\mathcal{A}}P^K\left(\bar{\mathcal{A}}\right) \tag{2.17}$$

Equation (2.17) provides the Katz centrality, P^K, where the superscript K denotes the vector P as the Katz centrality vector, which uses the normalized adjacency matrix. As with the eigenvector centrality, the eigenvalue of the greatest modulus corresponds to the centrality vector whose elements rank their associated node's prestige in the network. With this, an eigenvalue of one corresponds to the centrality vector, due to the normalization of the adjacency matrix. Correcting the measure by each node's cardinality provides insight into the amount of access a node's neighbors have with their common node. If node i shares a connection with node j and node j has a higher degree of connection than node i, node i's prestige value reduces due to the limited access it has to node j. For undirected communication graphs, the normalization with respect to degree reduces the Katz centrality to that of the degree centrality.

From a leadership perspective, degree of connection should factor into a node's eligibility. The eigenvector centrality chose an endogenous prospective of the network, primarily using walks as the delineating factor in the ranking metric. Normalizing by the degree of connection, Katz centrality handles prestige measurements better for heterogeneous topologies where degree of connection may vary greatly. Using both eigenvector and Katz centrality simultaneously shows the cohesiveness of the network, providing a direction of investigation based on the presence of communities. Also, Katz centrality corrects for some misleading information inherent in the eigenvector centrality since degree of connection can change a node's influence with respect to other nodes in the network.

2.5.3 Second Katz Centrality

The prestige centralities considered insofar have considered the number of walks between nodes such that each walk of any length influences the centrality of the node equally. The second Katz centrality regulates the dependence of the centrality on the length of the walk by damping the contribution of long walks to the centrality ranking. Using powers of the adjacency matrix, walks of increasing length weight according to a damping factor α.

$$P^{K2}(\mathcal{A}) = (I_n - \alpha\mathcal{A})^{-1}\,\alpha\mathcal{A}\mathbf{1} \tag{2.18}$$

The eigenvalue problem in Eq. (2.18) includes an addition damping factor of α used to attenuate the effect of walks of increasing length have on the centrality measure, P^{K2}, where the superscript $K2$ denotes the vector P as the second Katz centrality vector.In Eq. (2.18), I_n and $\mathbf{1}$ represent the $n \times n$ identity matrix and $n \times 1$ vector of ones, respectively. Expressing Eq. (2.18) as a power series, the relationship between the damping factor and the power of the adjacency matrix becomes apparent.

$$P^{K2}(\mathcal{G}, \alpha) = \alpha\mathcal{A}\mathbf{1} + \alpha^2\mathcal{A}^2\mathbf{1} + \alpha^3\mathcal{A}^3\mathbf{1} + \ldots \tag{2.19}$$

Equation (2.19) shows that direct connections acquire more weight in the centrality measure and connections of greater radial distance decay in influence. The introduction of this design parameter provides a variable radius of influence. For small values of α, the contribution of longer paths attenuate faster, such that direct connections and node cardinality influence the centrality ranking more than the extended topology of the network. Conversely, as higher values of α devalue longer paths smoothly, centrality scores reflect a node's importance based more on the endogenous topology than direct connections. To ensure convergence of the power series, the largest value of α should not exceed the reciprocal of the greatest magnitude eigenvalue, λ, of the adjacency matrix [28].

$$\lambda^* = \max |\lambda_k| \tag{2.20}$$

$$\alpha < \left(\frac{1}{\lambda^*}\right) \tag{2.21}$$

Here, $k = 0, 1, \ldots, n-1$ for n nodes in the graph $\mathcal{G}$. The second Katz centrality provides a design factor enabling the rank of individual nodes to reflect the wants of a user. Should the user want a leader focused on the needs of its immediate connections, a lower value of α helps refine a selection more than the first Katz centrality or a degree-only centrality ranking. If the leader should interact with the entire network evenly, choosing a higher value of α depicts a leader more readily than a pure closeness or eigenvector centrality. To this end, the second Katz centrality appears superior to other centrality measures yet, similar to applying proximity based centralities, the validity of the second Katz centrality lies in the application of the damping factor to the intended mission. With this, the range of convergent damping factors depends on the spectral radius of each network's adjacency matrix such that the relevance of one damping factor in a network does not correlate to other networks with different topologies. The damping factor provides flexibility for the user, blending between a local and global understanding of influence in the network, but needs tailoring between topologies to ensure that the ranking reflects the expected value of the centrality measure.

2.5.4 Bonacich Centrality

The Bonacich centrality acts as an extension of the second Katz centrality. As a measure of prestige, or power, this centrality makes use of the eigenvector problem in determining the node's importance and influence in a network based on predetermined weights on first-degree connections and path lengths emanating from the node. The first design parameter, α, sets a base value for first-degree connections, similar to the second Katz centrality. A second parameter, β, permits a separate weighting on the path length such that the association of a base value becomes uncorrelated to the influence of path length on the centrality measure.

$$P^B(\mathcal{G}, \alpha, \beta) = (I_n - \beta\mathcal{A})^{-1} \alpha\mathcal{A}\mathbf{1} \tag{2.22}$$

The combination of the parameters reflect different characteristics of the network topology in terms of the centrality of a node. The base value parameter, α, serves as a weighting factor for all first-degree connections, such that its value simply scales the centrality vector. The interpretation of β and the centrality vector, P^B, where the superscript B denotes the vector P as the Bonacich centrality vector, helps delineate the second Katz centrality and the Bonacich centrality. Viewing β as a probability that a communication sent will transmit, by the receiving node, to any of its contacts, P^B represents the expected number of paths in a network activated directly or indirectly by each individual. Using a power series iterative process to find the centrality vector, initializing from above to ensure that the centrality vector corresponds to the principal eigenvalue, the parameter β can range from zero to the reciprocal of the principal eigenvalue, previously stated in Eq. (2.21), to ensure convergence of the series, such that $0 \leq \beta \leq \alpha$. With this, the magnitude of β should reflect the degree to which communication transmits locally or to the structure as a whole. Small values of β heavily weight the local structure whereas larger values account for the topology of the network structure. To this end, β defines a radius of influence for the node. For a single path in the network, $(1 - \beta)^{-1}$ defines the expected length of any single path emanating

from positions in the network. Therefore, $(1-\beta)^{-1}$ defines the radius investigated by the centrality measure [33].

For networks that have a clear sense of hierarchy, the second Katz centrality and Bonacich centrality can provide different interpretations of leadership. For networks where hierarchy, or the presence of a dominant agent, has little affect on the communication or performance of the network, the second Katz centrality selects nodes with higher cardinality over nodes whose prestige depends on the importance of their neighbors. Conversely, if a dominate node resides in the network such that topology directly reflects hierarchy, Bonacich centrality identifies the more prestigious nodes. With this, the leadership qualities desired for a single agent directly affect the selection of the two design parameters for Bonacich centrality. The base parameter α reflects the importance of direct connections while β determines the importance of the network on the selection of the leader. Bonacich centrality provides a radius of investigation, giving the measure a tangible interpretation of influence on a node, lending itself to applications involving variation in direct and indirect communication.

Chapter 3

Decentralized Adjacency Algorithm

In a decentralized network of unmanned vehicles, the agents need to communicate their perspective of the network to their neighbors. In doing so, the network performs a status update, identifying any possible changes to the previously understood topology. Ideally, given a finite bandwidth between communicating agents, an agent only transmits the minimal amount of information to their neighborhood as to feasibly perform the operation online and in real time. By limiting the computational expense of the process, the network may update faster than the dynamics of the physical formation, ensuring the stability of system. Letting each agent transfer and receive the locally understood adjacency matrices of their neighboring agents, the connected topology of the network allows the construction of the network through each iteration locally.

The sharing of information among agents within their respective neighborhoods involves several processes to ensure that the information passed contains the most updated view of the network. To remain flexible to topological changes throughout a mission timeline, an agent must check on their neighborhood, efficiently gather the needed information, and convey their perspective of the network to their neighbors. The next sections detail the implementation of the decentralized adjacency algorithm (DAA), outlined in Fig. 3.1. Throughout the mission timeline, each agent runs the DAA protocol to ensure that any changes to the global formation structure is understood at the local level in a finite amount of time steps. This algorithm allows each agent to not only learn the network through the local exchange of information, but to later implement a leadership selection so that both a local and global hierarchy is established.

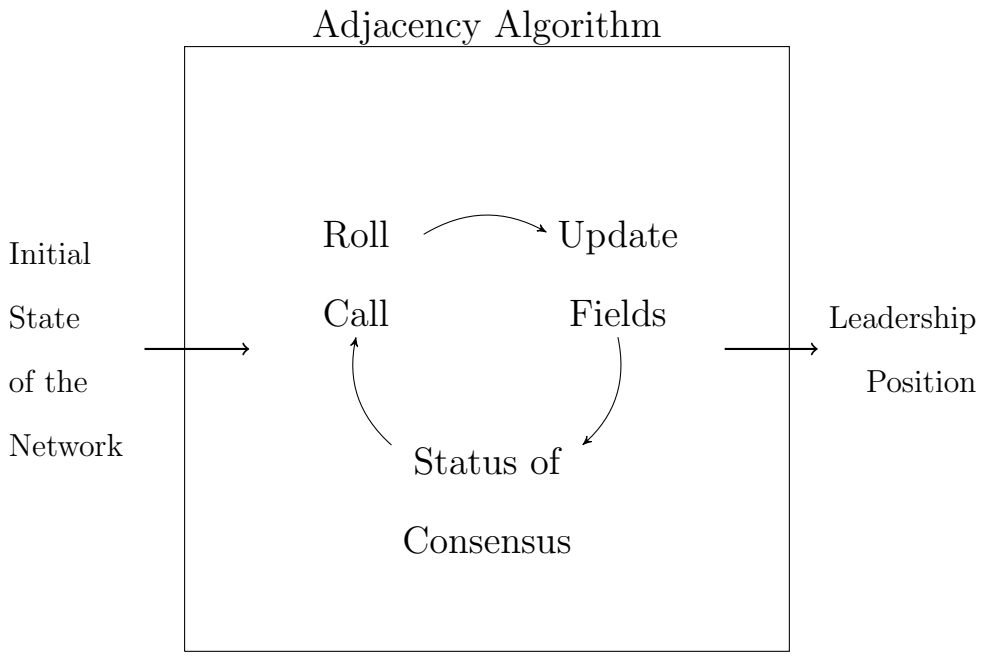


Figure 3.1: High level organization of the decentralized adjacency algorithm.

3.1 Initializing the Network

To begin the process of learning the network, the formation must first establish a connected global topology, which is only initially known to a global observer. The connected constraint of the problem ensures the reachability of each agent by all other agents in the network, therefore making each agent a viable candidate for leadership, acting as the possible location of the exogenous input from the ground operator. The initialization of the network state derives from an agent's ability to sense their neighboring network connections and exchange accurate information about their individual understanding of the network. Given a heterogeneous network with identifiable agents, a few basic characteristics help each agent effectively influence the network and learn the topology.

Consider an agent i in a connected network, shown in Fig. 3.2. The agent holds a personal identification number (ID) which all other informed agents know. This agent carries with it an ordered list (ascending) of its neighboring IDs that contribute to agent i's initial

understanding of the network structure. Knowing only its first-degree connections, agent i believes it lies at the center of a star, or tree network, surrounded by p agents where $p = |\mathcal{N}_i(\mathcal{G})|$, with $\mathcal{N}_i$ representing the neighborhood of agent i within the locally understood perspective of the network graph. From the initial understanding of the network, agent i constructs its adjacency matrix, $\mathcal{A}_i(\mathcal{G}_i)$, using the indexes of the ordered adjacency list $\ell_i(\mathcal{G}_i)$, holding the IDs of the agents in the network graph, $\mathcal{G}_i \subseteq \mathcal{G}$, as best understood by agent i.

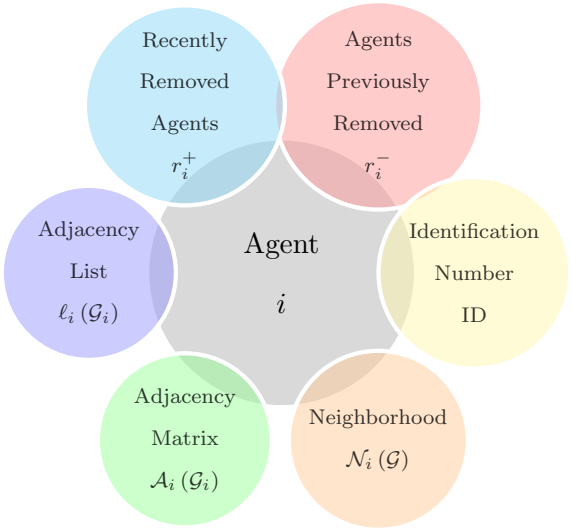


Figure 3.2: Properties of an agent i.

During each iteration, an agent calls on their neighbors to gather information. During this process, if the agent does not respond, the agent realizes the loss of a neighbor and stores their ID in the list r_i^+. This list notifies the remaining neighbors of the loss in order to update their information during the next iteration. Finally, each agent retains a list of the agents lost along the mission timeline, r_i^-. As a network grows closer to consensus, the removal of an agent may not be recognized by extended neighbors for many iterations, allowing old information to feed back into an agent's local understanding of the network. In the next sections, a few methods explain how these lists help an agent learn the global network topology through limited resources provided only through local information exchange.

3.2 Roll Call

In the process termed "Roll Call" in Algorithm 1, each agent i gathers information from their neighborhood, $\mathcal{N}_i$. Neighboring agent j, with $j = 1, \ldots, |\mathcal{N}_i|$, provides their local understanding of the network, $\mathcal{A}_j$, and its accompanying adjacency list, ℓ_j. This information then merges with agent i's adjacency matrix, $\mathcal{A}_i$, and accompanying adjacency list, ℓ_i, by keeping the order of the list and the correlated indexes of the adjacency matrix in ascending order according to the IDs of the agents (line 14). Section 3.2.1 provides a more in-depth description of this process. If a neighboring agent does not respond, agent i removes agent $\mathcal{N}_i(j)$ from their neighborhood and adds their ID to the list of agents to remove, r_i (line 17).

After the agent calls on all its available neighbors for information about the network (lines 3-19), the agent checks its list of agents to remove, r_i, from the newly created adjacency matrix and list (lines 20-26). This allows an agent to accumulate all the information from their neighbors and rationalize for itself whether the agent was removed, added, or remains. In the final steps of the Roll Call process, line 27, the agent finds the set difference between the list of agents removed during the current iteration, t, and all past removals, r_i^-. Doing so, the agent creates a list to notify its neighbors about the most current change to the network topology. Line 28 adds the most recently removed agents to agent i's memory of lost agents during the mission, r_i^-.

Repeating this process for all agents in the network allows each agent to pull information from their respective neighborhoods and have the entire network update simultaneously at each iteration. Throughout this process, new instances of the agent's adjacency matrix, adjacency list, and neighborhoods (denoted with an asterisk) carry into the update phase of the adjacency algorithm, present next in Section 3.2.1. This method preserves the information from each iteration needed to move iteratively from one update to the next, restricting each agent to only learn of network changes from its first-degree connections.

Algorithm 1 Updating agent information at iteration `t`

1: **procedure** RollCall
2: **Input:** $\mathcal{G}$
3: **for** $i \leftarrow 1, |V(\mathcal{G})|$ **do**
4: $\mathcal{A}_i$ ▷ Adjacency matrix of agent i
5: $\mathcal{A}_i^*$ ▷ New adjacency matrix of agent i
6: ℓ_i ▷ Adjacency list of agent i
7: ℓ_i^* ▷ New adjacency list of agent i
8: $\mathcal{N}_i$ ▷ Neighborhood of agent i
9: $\mathcal{N}_i^*$ ▷ New neighborhood of agent i
10: r_i ▷ List of IDs for agent i to remove during iteration t
11: r_i^- ▷ List of IDs removed by agent i during iteration $(t-1)$

12: **for** $j \leftarrow 1, |\mathcal{N}_i|$ **do**
13: **if** $N_i(j) \in V(\mathcal{G})$ **then** ▷ Gathering information from agent j
14: **Call** $[\mathcal{A}_i^*, \ell_i^*] =$ MergeAdjacency$(\mathcal{A}_i^*, \ell_i^*, \mathcal{A}_j, \ell_j)$
15: **else**
16: $\mathcal{N}_i^* \leftarrow \mathcal{N}_i^* \setminus \mathcal{N}_i(j)$ ▷ Removing agent j from $\mathcal{N}_i^*$
17: $r_i \leftarrow r_i \cup \mathcal{N}_i(j)$ ▷ Add missing agent to r_i, list to remove
18: **end if**
19: **end for**

20: **if** $|r_i| > 0$ **then** ▷ Removing agents from the update
21: **for** $j \leftarrow 1, |r_i|$ **do**
22: $z \leftarrow [r_i(j) : \ell_i^*]$ ▷ Index of $r_i(j)$ in the new list ℓ_i^*
23: $\mathcal{A}_i^*(z, z) \leftarrow \emptyset$ ▷ Remove from adjacency matrix
24: $\ell_i^*(z) \leftarrow \emptyset$ ▷ Remove from adjacency list
25: **end for**
26: **end if**

27: $r_i^+ \leftarrow r_i \setminus r_i^-$ ▷ Agents for neighborhood to remove
28: $r_i^- \leftarrow r_i^- \cup r_i$ ▷ Agents removed by agent `i`
29: **end for**
30: **return** $\mathcal{G}$
31: **end procedure**

3.2.1 Merging Matrices

The process of merging adjacency matrices begins by finding the union of the ID list for agent i and j (line 3). The newly merged list, ℓ^*, holds the ID of every agent present in

both agent i and agent j's understanding of the network. The index of the two subgroups, z_i (line 4), finds where the ID list of agent i intersects ℓ^*, thereby indicating the location of the elements of agent i's adjacency matrix in the new adjacency matrix, $\mathcal{A}^*$. Line 6 describes the same procedure of finding the location of the indices of agent j's ID list in ℓ^*. Since z_j has the same number of elements as $\mathcal{A}_j$, lines 8 through 17 search for nonzero elements of $\mathcal{A}_j$ to move into the elements of $\mathcal{A}^*$ at the indices of z_j.

Algorithm 2 Merging adjacency matrices for agent i

```
1: procedure MergeAdjacency
2:     Input: A_i, ℓ_i, A_j, ℓ_j
3:     ℓ* ← ℓ_i ∪ ℓ_j                          ▷ Union of the two list, ascending order
4:     z_i ← [ℓ_i : ℓ*]                        ▷ Index of ℓ_i in the new list ℓ*
5:     A*(z_i, z_i) ← A_i                      ▷ Place A_i in new adjacency matrix A*
6:     z_j ← [ℓ_j : ℓ*]                        ▷ Index of ℓ_j in new list ℓ*
7:     kk ← 0
8:     for k ← z_j do                          ▷ Place A_j in new adjacency matrix A*
9:         kk ← kk + 1
10:        mm ← 0
11:        for m ← z_j do
12:            mm ← mm + 1
13:            if A_j(kk, mm) == 1 then
14:                A*(k, m) ← 1
15:            end if
16:        end for
17:    end for
18:    return A*, ℓ*                           ▷ Return new adjacency and new list
19: end procedure
```

3.3 Agent Update Procedure

After the agents have acquired their new information about the structure of the network, each agent updates their information and relays any needed information to their neighborhood. Each agent must update their adjacency matrix, $\mathcal{A}$, its corresponding adjacency list, ℓ, and their newly updated neighborhood, $\mathcal{N}$ from iteration $(t-1)$ to t. Having the agents acquire information at iteration $(t-1)$, while retaining their previous perception of the

topology, allows the entire network to update simultaneously. If the agents updated their information in parallel to gathering new information, old information may feed back into the system and the agents would fail to recognize changes to the topology.

Algorithm 3 waits until every agent gathers information from their respective neighborhoods. Lines 4-6 show the update from iteration $(t-1)$ to t of the adjacency matrix $\mathcal{A}_i$, adjacency list ℓ_i, and neighborhood $\mathcal{N}_i$ of agent i. Line 7 checks agent i's list of IDs recently removed during iteration $(t-1)$, denoted as r_i^+. If occupied, agent i notifies all of its neighbors, $\mathcal{N}_i$, of its update to the network topology so that they may remove those agents the next iteration, adding to their r_j list. Algorithm 1 references this list in line 17 such that after updating the agent fields, this list accumulates all current updates by the agent, as well as the updates from its neighbors from the previous iteration. Finally, the list of agents currently removed by i becomes an empty set in line 11. This process repeats for all agents in the vertex set V.

Algorithm 3 Updating agent information at iteration `t`

```
1: procedure AgentUpdate
2:     Input: 𝒢
3:     for i ← 1, |V(𝒢)| do
4:         𝒜_i ← 𝒜_i^*                          ▷ Update adjacency matrix
5:         ℓ_i ← ℓ_i^*                          ▷ Update adjacency list
6:         𝒩_i ← 𝒩_i^*                          ▷ Update neighbor list
7:         if |r_i^+| ≥ 0 then                  ▷ Checking for changes to the network
8:             for j ← 1, |𝒩_i| do
9:                 r_j ← (r_i^+ ∪ r_j) \ r_j^-
10:            end for
11:            r_i^+ ← ∅
12:        end if
13:    end for
14:    return 𝒢
15: end procedure
```

3.3.1 Status of Consensus

DAA provides the network with a means of determining the network topology through first-degree communication. Before allowing an agent to determine a hierarchical structure, the agent needs to reach a consensus within their respective neighborhood on the agreed upon structure of the formation. Through each iteration, agents exchange limited information including their local adjacency matrix and its accompanying list. By reaching agreement on their understanding of the network, they determine the status of the network consensus at the local level. Therefore, with every agent reaching consensus within their local network, the global network reaches consensus as a whole.

Algorithm 4 compares the adjacency matrices and ID list of each agent within an agent's neighborhood to determine the status of the network consensus. The adjacency list and matrix of each agent must be equal to the adjacency list and matrix of its neighbors. The necessary condition that both sets be identical accounts for the occurrence of symmetric structures in a network, guaranteeing the unique comparison between the agent and its neighborhood. Comparing information in line 6 finds the status of the consensus as either true or false. Once a neighborhood achieves consensus, they may then begin to determine the hierarchical structure of the network based on the wants of the user and the restrictions established by the realizable internal network dynamics.

Algorithm 4 Algorithm to determine the consensus of the global network.

```
1:  procedure StatusOfConsensus
2:      Input: 𝒢
3:      for i ← 1, |V(𝒢)| do
4:          tf_i ← 1                                  ▷ Status of consensus for agent i
5:          for j ← 1, |𝒩_i| do
6:              if 𝒜_i ≠ 𝒜_j or ℓ_i ≠ ℓ_j then        ▷ Comparing adjacency matrix and list
7:                  tf_i ← 0
8:                  return tf_i                       ▷ If any neighbor disagrees, consensus is not met
9:              end if
10:         end for
11:     end for
12:     return 𝒢
13: end procedure
```

Through the implementation of the DAA, a decentralized agent is capable of determining the structure of the formation and adjust to changes in the topology, paving the way for local and global leadership establishment through a previously determine metric of leadership. This process provides a realistic procedure to learn the network of a decentralized formation through local information exchange. DAA alleviates the needs for a central observer and provides a feasible approach to develop a decentralized controller, implemented locally, for each vehicle in the formation.

3.3.2 Convergence of the DAA

As a demonstration of the convergence of the DAA for learning, consider the global network given in Fig. 3.3. Table 3.1 describes the iterative process of learning the topological structure for each agent in the network. Each row of the table indicates the perception of the network as seen by the local agent, indicated by their respective columns. Initially, the agents only know of their first-degree neighbors such that iteration 1 consist of individual star graphs with the respective agents as the network centers. At the next iteration, the neighbors exchange their adjacency matrices from the last iteration. For example, Agent 1 obtains the adjacency matrix of Agent 2 from the previous iteration, thereby learning of Agent 3's relationship to Agent 2. The well connected nodes of the network learn of the entire topology quickly; namely Agent 3 who learns the network in two iterations. Outlying nodes require more iterations to develop a better understanding of the topology. As the iterations progress, each agent learns the full topology of the global network. This process continues at every iteration as to recognize further changes to the network structure along the mission timeline. As each agent makes alterations to their last perspective of the network, they check their understanding of the network with that of their neighbors to determine if consensus has been reached.

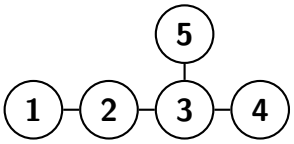


Figure 3.3: Global network topology initially unknown to local agents.

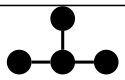
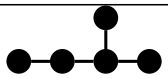
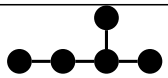
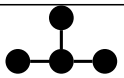
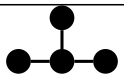
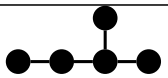
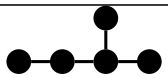
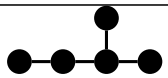
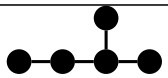
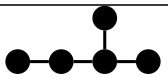
	Nodes				
Iteration	1	2	3	4	5
1					
2					
3					

Table 3.1: Iterative description of the DAA for a small network utilizing limited information exchange to build a local understanding of the global topology.

The next chapter defines a new metric for leadership selection in the decentralized formation. With each agent knowing this metric *a priori*, they each have the ability to determine their local and global hierarchy through the implementation of the DAA. A more detailed example of this application will show how the network converges onto a local leader through the limited exchange communication topology. As consensus is reached locally within their respective neighborhoods, the network reaches consensus globally and the identification of the formation leader is learned in a decentralized manner.

Chapter 4

Dynamic Laplacian Centrality

This dissertation introduces a new centrality measure developed by addressing a decentralized control problem for a formation of unmanned vehicles. Motivated to limit the control of the formation through a single lead vehicle, the guidance controller implemented varies the leader's dependence on its follower's states, using the leader-follower and virtual-structure as the boundaries of a formation continuum. With communication and command relayed exclusively through an agent's local communication, the location of the leader in the network topology has a direct affect on the response of the formation to an exogenous command. Typically, the identification of important nodes in a network arrives through topological centrality considerations such as the degree, closeness and prestige of a node, as given in Chapter 2. In this application, leadership derives from the dynamic benefit of selecting one agent over another in terms of the response of the formation as an extension of the prestige centralities. Using certain convergent qualities realizable through the graph-theoretic Laplacian matrix, the convergence rate of the network becomes a quantitative measure of centrality in describing a formation's reaction to an exogenous command through a single lead agent.

The use of the graph Laplacian has shown considerable interest in terms of spectral clustering and consensus dynamics in network communication applications [18, 34, 35]. As an extension to a guidance controller, this application utilizes the spectral properties as a domain in which to select a leader for a desired formation response. In this section, a definition of the implemented centrality, along with simulations showing the effects of its selection on a formation utilizing first-order dynamics, demonstrates its usefulness. Next, a quantitative description of the formation response based on the entire spectrum of the

graph Laplacian shows the effectiveness of this method in selecting a formation leader that produces a predetermined desired response.

In agreement, or consensus protocols, a network of agents work to achieve a unanimous state of agreement such that each agent's initial state converges to a network whole. In a connected network, this resolves into each agent finding agreement within their respective neighborhoods, and as a result, the agreement of a node moves under error dynamics within its neighborhood.

$$\dot{X}_i(t) = -\sum_{j \in \mathcal{N}_i(\mathcal{G})} (X_i(t) - X_j(t)) \tag{4.1}$$

Equation (4.1) finds the difference between the consensus state, $X_i(t)$, for agent i and its neighborhood, $\mathcal{N}_i(\mathcal{G})$, such that a matrix representation of the consensus dynamics reduces to the graph Laplacian [23].

$$\dot{X}(t) = -L(\mathcal{G})X(t) \tag{4.2}$$

From Section 2.1, the nonnegative, real eigenvalues of a connected graph show that the consensus dynamics in Eq. (4.2) converge asymptotically. The graph Laplacian considered herein are symmetric matrices. Therefore the eigenvectors of the graph Laplacian are orthogonal and the graph Laplacian may be decomposed, using spectral factorization, along each eigen-axis.

$$X(t) = e^{-\lambda_0 t}(v_0^T X_0)v_0 + e^{-\lambda_1 t}(v_1^T X_0)v_1 + \cdots + e^{-\lambda_k t}(v_k^T X_0)v_k \tag{4.3}$$

In (4.3), λ_k denotes the eigenvalues of the respective eigenvectors, v_k, for $k = 0, 1, \ldots, (n-1)$. Consider the selection of an initial condition with one component of X_0 equal to one and the other components equal to zero. Further consider the node with nonzero initial conditions to be the leader. By Propositions 2.2 through 2.4, the zero eigenvalue of the graph Laplacian, λ_0, corresponds to the ones eigenvector, v_0, such that the first term in Eq. (4.3) becomes the average of the initial condition X_0, i.e. the consensus value. The behavior of the

remaining modes will depend on how much they are excited by the initial condition vector. For example, the slowest non-constant mode, described by λ_1, is excited in proportion to the quantity $v_1^T X_0$. Selecting a leader so that this quantity is minimized will help speed up the convergence of the network to the consensus value. If the value of λ_1 is repeated, then there exist an entire subspace along which the response behaves at the same rate. Therefore, it would be desirable to minimize the projection of the initial condition into this subspace.

Alternatively, consider an exogenous signal applied to one node, i.e. the leader, attempting to drive the entire network to a certain state and elicit a desired response. Consider an exogenous signal taking the form of an impulse at the initial time applied to any one node of the network.

$$U = \delta(t_0) X_d \tag{4.4}$$

The signal is then incorporated into the consensus dynamics.

$$\dot{X}(t) = -L(\mathcal{G}) X(t) + BU \tag{4.5}$$

$$B_i = \begin{cases} 1 & \text{if } i = \text{leader} \\ 0 & \text{otherwise} \end{cases} \tag{4.6}$$

This results in the following time response.

$$\begin{aligned} X(t) &= e^{-L(t-t_0)} X_0 + \int_{t_0}^{t} e^{-L(t-\tau)} B\delta(t_0) X_d d\tau \\ &= e^{-L(t-t_0)} X_0 + e^{-L(t-t_0)} B X_d \\ &= \left(e^{-\lambda_0 (t-t_0)} v_0 v_0^T + e^{-\lambda_1 (t-t_0)} v_1 v_1^T + \cdots + e^{-\lambda_k (t-t_0)} v_k v_k^T \right) (X_0 + B X_d) \end{aligned}$$

Because $\lambda_0 = 0$ and $v_0 = \mathbf{1}$, the steady-state response of the control signal will converge to $(X_0 + BX_d)$, since the impulse input at the initial time changes the initial conditions of the homogeneous response. With this, the system converges to the average of the new initial

conditions, regardless of the choice of leadership. However, wanting to produce a desired response from the network as it converges onto the steady-state value, the objective remains to appropriately select the location of the input node. To achieve this, the impulse should excite the node capable of producing the desired response from the network. For instance, in order to increase the rate of convergence, the impulse should excite the second-slowest mode as little as possible, thereby minimizing $v_1^T B$. With this, the definition of the dynamic Laplacian centrality is constructed.

Definition 4.1 (The Dynamic Laplacian Centrality) *Consider a given graph* $\mathcal{G} = (V, E)$ *with graph Laplacian* $L(\mathcal{G})$*, which has eigenvalues* $\lambda_0 < \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_k$ *and corresponding eigenvectors* $v_0, v_1, \cdots, v_k$ *such that* $k = 0, \cdots, n-1$*, for* n *nodes in the network. If* λ_1 *is unique, then the dynamic Laplacian centrality (DLC) vector is given by* $|v_1|$*. If* λ_1 *is repeated* r *times, such that* $\lambda_1 = \lambda_2 = \cdots = \lambda_r$*, where* $1 < r < (n-2)$*, then the DLC vector is given by* $\sqrt{v_1^2 + v_2^2 + \cdots + v_r^2}$*. Here,* $|v|$*,* $\sqrt{v}$ *and* v^2 *indicate the element-by-element absolute value, square root, and square of the vector* v*, respectively.*

4.1 Special Graphs

Although simplistic in nature, symmetries inherent to special graphs result in repeated values of λ_1 such that care must be taken when evaluating the DLC. The following sections provide examples of common special graphs to demonstrate how the DLC uncovers topological ambiguities in the presence of symmetry and homogeneity. The definition of the centrality in the previous section states that the DLC vector is given by $\sqrt{v_1^2 + v_2^2 + \cdots + v_r^2}$ for r repeated values of λ_1. This, the Euclidean norm, denoted by $\|v\|$, determines the overall affect of the contributing nodes to the dynamics of the network.

4.1.1 Star Graphs

A star, or wheel, graph has a center node that shares a connection with every other node of the graph. This graph represents the most central case of graph topologies. Using

the degree and closeness centrality measures as an example, the center node holds a value of unity for both measures, making it the most influential node of the formation. The outlying nodes share the same centrality measures, as expected, due to their symmetry about the center node. Looking at the graph Laplacian eigenvalues, λ, $(n-2)$ repeated ones follow the simple zero and the spectrum is bounded from above by $\lambda = n$.

When applying the DLC, it should reflect that the center node holds the most influence over the network, while the outlying nodes reflect the symmetry of the graph. The DLC vector is computed from the eigenvectors of the repeated eigenvalues. Tables 4.1 and 4.2 indicated the repeated eigenvalues and their associated eigenvectors. The last column of the tables compute the Euclidean norm, thereby determining the DLC for the star networks.

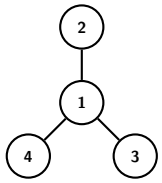


Figure 4.1: Star graph, four nodes.

Ordered Eigenvalues
0.000
1.000
1.000
4.000

ID	$\lvert v_0 \rvert$	$\lvert v_1 \rvert$	$\lvert v_2 \rvert$	$\lvert v_3 \rvert$	$\lVert v \rVert$
1	0.500	0.000	0.000	0.866	0.000
2	0.500	0.671	0.465	0.289	0.817
3	0.500	0.067	0.814	0.289	0.817
4	0.500	0.738	0.349	0.289	0.817

Table 4.1: Ordered eigenvalues and associated eigenvectors of the star graph given in Fig. 4.1 with an even number of nodes.

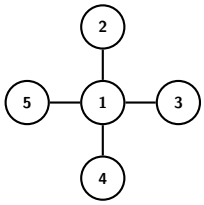


Figure 4.2: Star graph, 5 nodes.

Ordered Eigenvalues
0.000
1.000
1.000
1.000
5.000

ID	$\lvert v_0 \rvert$	$\lvert v_1 \rvert$	$\lvert v_2 \rvert$	$\lvert v_3 \rvert$	$\lvert v_4 \rvert$	$\lVert v \rVert$
1	0.447	0.000	0.000	0.000	0.894	0.000
2	0.447	0.789	0.211	0.289	0.224	0.866
3	0.447	0.577	0.577	0.289	0.224	0.866
4	0.447	0.211	0.789	0.289	0.224	0.866
5	0.447	0.000	0.000	0.866	0.224	0.866

Table 4.2: Ordered eigenvalues and associated eigenvectors of the star graph given in Fig. 4.2 with an odd number of nodes.

4.1.2 Circle Graphs

A circle graph connects the ending nodes of a line graph, creating a cycle within the network. Each node has the same degree of connection and sits identically between nodes compared to all other nodes of the network. Using the degree and closeness centrality measures as comparisons, the measures label each node equally important in the network such that discriminating between homogeneous nodes proves impossible. The DLC should therefore reflect the same result.

Applying the DLC to the circle network, the eigenvalues of the graph Laplacian have repeated values of λ following the simple zero eigenvalue, given in Tables 4.3 and 4.4. This characterizes special graphs and alludes to the presence of symmetries in the network. The last column of each table shows that the centrality value of each node is equivalent, as expected from the degree and closeness centrality measures. Table 4.4, for a circle graph with an odd number of nodes, has two sets of repeated values. As directed, the Euclidean norm of λ_1, with $r = 2$, determines the value of the DLC.

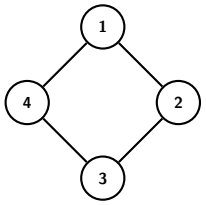


Figure 4.3: Circle graph, 4 nodes.

Ordered Eigenvalues
0.000
2.000
2.000
4.000

ID	$\|v_0\|$	$\|v_1\|$	$\|v_2\|$	$\|v_3\|$	$\|\|v\|\|$
1	0.500	0.707	0.000	0.500	0.707
2	0.500	0.000	0.707	0.500	0.707
3	0.500	0.707	0.000	0.500	0.707
4	0.500	0.000	0.707	0.500	0.707

Table 4.3: Ordered eigenvalues and associated eigenvectors of the circle graph given in Fig. 4.3 with an even number of nodes.

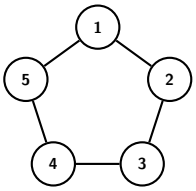


Figure 4.4: Circle graph, 5 nodes.

Ordered Eigenvalues
0.000
1.382
1.382
3.618
3.618

ID	$\|v_0\|$	$\|v_1\|$	$\|v_2\|$	$\|v_3\|$	$\|v_4\|$	$\|\|v\|\|$
1	0.447	0.587	0.236	0.457	0.437	0.633
2	0.447	0.406	0.485	0.113	0.622	0.633
3	0.447	0.336	0.536	0.275	0.570	0.633
4	0.447	0.614	0.154	0.557	0.299	0.633
5	0.447	0.043	0.631	0.627	0.085	0.633

Table 4.4: Ordered eigenvalues and associated eigenvectors of the circle graph given in Fig. 4.4 with an odd number of nodes.

4.1.3 Complete Graphs

Each node of the complete graph shares a connection with all other nodes in the graph. With this, the degree and closeness centrality of each node equals unity, implying that each node caries the same influence within the network. Looking at the graph Laplacian

eigenvalues, λ, in Tables 4.5 and 4.6, each take the same form of a simple zero followed by a repeated second eigenvalue, with $r = (n-1)$. As with the application of the degree and closeness centrality, the DLC value of each node should carry the same amount of influence in the network. The last column of the tables verifies the equivalence of each node in terms of the DLC value. This type of special graph, along with the circle graph previously discussed, do not benefit from a centrality analysis such that the dynamic response of the graph is indistinguishable when varying the location of the exogenous input.

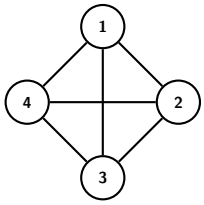


Figure 4.5: Complete graph, 4 nodes.

Ordered Eigenvalues
0.000
4.000
4.000
4.000

ID	$\lvert v_0 \rvert$	$\lvert v_1 \rvert$	$\lvert v_2 \rvert$	$\lvert v_3 \rvert$	$\lVert v \rVert$
1	0.500	0.154	0.236	0.819	0.866
2	0.500	0.598	0.350	0.519	0.866
3	0.500	0.785	0.277	0.237	0.866
4	0.500	0.033	0.863	0.063	0.866

Table 4.5: Ordered eigenvalues and associated eigenvectors of the complete graph given in Fig. 4.5 with an even number of nodes.

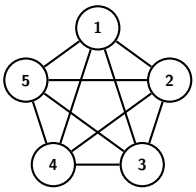


Figure 4.6: Complete graph, 5 nodes.

4.1.4 Tree Graphs

The ability of the DLC vector to locate the symmetry within the graph makes it a viable tool for graphs of unknown topology. Tree graphs help illustrate this case by looking at a

Ordered Eigenvalues
0.000
5.000
5.000
5.000
5.000

ID	$\lvert v_0 \rvert$	$\lvert v_1 \rvert$	$\lvert v_2 \rvert$	$\lvert v_3 \rvert$	$\lvert v_4 \rvert$	$\lVert v \rVert$
1	0.447	0.319	0.707	0.276	0.349	0.894
2	0.447	0.319	0.707	0.276	0.349	0.894
3	0.447	0.097	0.000	0.866	0.201	0.894
4	0.447	0.836	0.000	0.312	0.054	0.894
5	0.447	0.295	0.000	0.001	0.844	0.894

Table 4.6: Ordered eigenvalues and associated eigenvectors of the complete graph given in Fig. 4.5 with an odd number of nodes.

symmetric and asymmetric topology. Figure 4.7 shows an example of a symmetric graph, where node 1 acts as a pivot point. Using λ_1 in Table 4.7, the associated eigenvector v_1 represents the DLC vector. With this, node 1 has a DLC value of zero, meaning it acts as the pivot point in the graph's topology, given in Fig. 4.7. This zero value of the DLC vector appears in bipartite graphs that exhibit a clear hierarchy in nodal centralities.

An asymmetric case of this graph, given in Fig. 4.8, is analyzed in the same manner. Using v_1 in Table 4.8 as the DLC vector, the absence of any zero elements designate the network as asymmetric; a conclusion derived solely through the spectral properties of the graph Laplacian. Also, seen in previous cases with repeated values of λ_1, repeated DLC values identify similarities between nodes in the network.

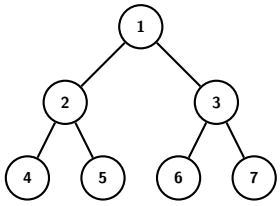


Figure 4.7: Tree graph, symmetric.

Ordered Eigenvalues
0.000
0.268
1.000
1.000
1.586
3.732
4.414

ID	v_0	v_1	v_2	v_3	v_4	v_5	v_6
1	0.378	0.000	0.000	0.000	0.794	0.000	0.476
2	0.378	0.325	0.000	0.000	0.164	0.628	0.575
3	0.378	0.325	0.000	0.000	0.164	0.628	0.575
4	0.378	0.444	0.209	0.676	0.281	0.230	0.168
5	0.378	0.444	0.209	0.676	0.281	0.230	0.168
6	0.378	0.444	0.676	0.209	0.281	0.230	0.168
7	0.378	0.444	0.676	0.209	0.281	0.230	0.168

Table 4.7: Ordered eigenvalues and associated eigenvectors of the symmetric tree graph given in Fig. 4.7.

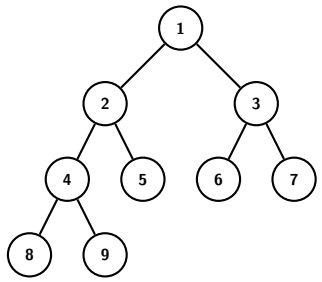


Figure 4.8: Tree graph, asymmetric.

Ordered Eigenvalues
0.000
0.183
0.572
1.000
1.000
1.509
3.000
4.044
4.691

ID	$\lvert v_0\rvert$	$\lvert v_1\rvert$	$\lvert v_2\rvert$	$\lvert v_3\rvert$	$\lvert v_4\rvert$	$\lvert v_5\rvert$	$\lvert v_6\rvert$	$\lvert v_7\rvert$	$\lvert v_8\rvert$
1	0.333	0.134	0.171	0.000	0.000	0.750	0.272	0.292	0.347
2	0.333	0.120	0.320	0.000	0.000	0.229	0.544	0.156	0.632
3	0.333	0.363	0.076	0.000	0.000	0.138	0.272	0.753	0.302
4	0.333	0.324	0.142	0.000	0.000	0.042	0.544	0.404	0.550
5	0.333	0.146	0.747	0.000	0.000	0.450	0.272	0.051	0.171
6	0.333	0.444	0.178	0.688	0.165	0.272	0.136	0.247	0.082
7	0.333	0.444	0.178	0.688	0.165	0.272	0.136	0.247	0.082
8	0.333	0.397	0.332	0.165	0.688	0.083	0.272	0.133	0.149
9	0.333	0.397	0.332	0.165	0.688	0.083	0.272	0.133	0.149

Table 4.8: Ordered eigenvalues and associated eigenvectors of the asymmetric tree graph given in Fig. 4.8.

4.2 Application of the Dynamic Laplacian Centrality

As described in Chapter 4, the first nonzero eigenvalue of the graph Laplacian matrix locates the eigenvector whose absolute value determines the DLC measure of the individual nodes. The determination of the input location, through the application of the DLC, has a direct effect on the response of the network. The next section highlights some of the difficulties in selecting a single node as the input node when looking for an optimal response for the case of formation keeping. After which, the behavior of a formation to an exogenous input is demonstrated for various network topologies and the implications of leadership location is illustrated through the DLC.

Consider a vehicle performing a consensus protocol using simple Laplacian dynamics.

$$\dot{x}(t) = -L(\mathcal{G})x(t) + Bu(t) \tag{4.7}$$

In Eq. (4.7), the form of the graph Laplacian is known for a given formation's topology but the form of the input vector, B, is unknown. To simplify the analysis in determining the optimal form of the input vector, the control input is chosen such that $u(t) = 1 \; \forall \; t > 0$. The next section describes the form of the input vector required to reduce the state separation of the formation when provided with an exogenous input by using an optimal control approach. Later, this result is compared to the simplified process of leadership selection by applying the DLC to determine the form of the input vector to best reduces the state separation of the formation

4.2.1 Optimal Leadership Selection for Formation Keeping

The behavior of the formation depends on the topological location of the exogenous input. In networks where a set of nodes act as unconstrained leaders while the remaining agents execute a consensus protocol, the convergent characteristics of strategically placing the location of the input demonstrates the importance of leadership selection [36]. In this application, the dynamics of the leader are influenced by their neighborhood such that their motion is no longer unconstrained.

$$J = \frac{1}{2} \int_0^{t_f} x^T(\tau) L(\mathcal{G}) x(\tau) d\tau \tag{4.8}$$

The performance index in Eq. (4.8) penalizes the spacing between the states of the formation, leveraging the communication relationships modeled through the graph Laplacian. By reducing the spacing between states of the formation, the structure remains intact as the formation receives an input. Consider the case where the vehicles move under the dynamics described by Eq. (4.7) such that the agents implement a consensus protocol where an agent's dynamics are influenced only by their neighbor's states. Substituting the constant control input, an augmented cost function is formulated by combining the performance index with

the constraints through the use of the multiplier function $\mu(t)$.

$$J_a = \int_0^{t_f} \frac{1}{2} x^T L x + \mu^T(-Lx + B - \dot{x}) d\tau \tag{4.9}$$

The variation of the augmented cost function is determined by varying x, $\dot{x}$, μ, and for the application of this optimal control problem, the input vector, B.

$$\delta J = \int_0^{t_f} \left[\left(x^T L - \mu^T L \right) \delta x - \mu^T \delta \dot{x} + \left(-x^T L + B^T - \dot{x}^T \right) \delta \mu + \mu^T \delta B \right] d\tau \tag{4.10}$$

Integrating by parts to remove the variation of $\dot{x}$, the variation of the augmented cost function is separated into three elements.

$$\delta J = -\mu^T \delta x \mid_0^{t_f} + \int_0^{t_f} \left[\left(x^T L - \mu^T L + \dot{\mu}^T \right) \delta x + \left(-x^T L + B^T - \dot{x}^T \right) \delta \mu + \mu^T \delta B \right] d\tau \tag{4.11}$$

The three elements in the integral of Eq. (4.11) represent the three necessary conditions for optimality are restated as follows:

$$\begin{aligned} \dot{x} &= -Lx + B \\ \dot{\mu} &= -Lx + L\mu \\ 0 &= \textstyle\int_0^{t_f} \mu(\tau) d\tau \end{aligned} \tag{4.12}$$

The initial states are fixed in Eq. (4.12), with $x_0 = 0$, such that $\mu_0 = 0$. When $t = t_f$, the states are free so that the final value of μ must equal zero. To begin, the solution to the state equation is found in terms of the input vector.

$$\begin{aligned} x(t) &= e^{-Lt} x_0 + e^{-Lt} \textstyle\int_0^{t_f} e^{L\tau} B d\tau \\ &= e^{-Lt} \left[\textstyle\int_0^{t_f} e^{L\tau} d\tau \right] B \end{aligned} \tag{4.13}$$

Using the same approach with the costate equation and substituting the solution from Eq. (4.13):

$$\begin{aligned} \mu(t) &= e^{Lt}\mu_0 - e^{Lt}\int_0^{t_f} e^{-L\tau}Lx(\tau)d\tau \\ &= e^{Lt}\mu_0 - e^{Lt}\left[\int_0^{t_f} e^{-2L\tau}L\left[\int_0^{t_f} e^{L\tau_1}d\tau_1\right]d\tau\right]B \end{aligned} \tag{4.14}$$

It is important to note that the inverse of L does not exist due to the semi-positive definite nature of the graph Laplacian. For this application, the inverse is not evaluated, due to the costate's dependence upon the graph Laplacian in Eq. (4.14). Therefore the inner integral may be evaluated.

$$\begin{aligned} \mu(t) &= e^{Lt}\mu_0 - e^{Lt}\left[\int_0^{t_f} e^{-2L\tau}\left[e^{L\tau} - I_n\right]d\tau\right]B \\ &= e^{Lt}\mu_0 + e^{Lt}\left[\int_0^{t_f} e^{-L\tau}\left[e^{-L\tau} - I_n\right]d\tau\right]B \end{aligned} \tag{4.15}$$

The graph Laplacian represents the undirected communication between the agents such that $L(\mathcal{G})$ is symmetric. With this, the eigenvectors of the graph Laplacian are orthogonal, allowing the matrix to be decomposed along each eigen-axis using spectral factorization. Defining the matrix $V = [v_0, v_1, \ldots, v_k]$ as the modal matrix containing the eigenvectors of the graph Laplacian, v_i, and $\Lambda = \text{diag}([\lambda_0, \lambda_1, \ldots, \lambda_k])$ where $k = 0, 1, \ldots, n-1$ for n nodes in the network. From the properties of the graph Laplacian previously defined in Section 2.1, the eigenvalues of the graph Laplacian can be arranged in ascending order where $0 = \lambda_0 < \lambda_1 \leq \ldots \leq \lambda_{n-1}$, noting that the lowest eigenvalue for a connected, undirected network is identically zero. Therefore, using the decomposition, $L = V\Lambda V^T$, the exponential of the graph Laplacian may be written in terms of the graph Laplacian's eigenvalues and eigenvectors.

$$\begin{aligned} e^{Lt} &= e^{(V\Lambda V^T)t} \\ &= Ve^{\Lambda t}V^T \\ &= \textstyle\sum_{i=0}^{n-1} e^{\lambda_i t}v_i v_i^T \\ &= v_0 v_0^T + \textstyle\sum_{i=1}^{n-1} e^{\lambda_i t}v_i v_i^T \end{aligned} \tag{4.16}$$

Applying this identity and integrating, the costate equation is written in terms of the input vector and spectrum of the graph Laplacian.

$$\mu(t) = e^{Lt}\mu_0 + e^{Lt}\left[\sum_{i=1}^{n-1}\left[\frac{1}{2\lambda_i} - \frac{1}{\lambda_i}e^{-\lambda_i t}\left(1 - \frac{1}{2}e^{-\lambda_i t}\right)\right]v_i v_i^T\right]B \tag{4.17}$$

The final necessary condition listed in Eq. (4.12) notes that the integral of the costate should be zero. The inner term of the summation in Eq. (4.17) is in terms of the constant properties of the formation structure. Therefore, the form of the input vector remains as the only variable to optimize the system. As stated, the eigenvalues are arranged in ascending order. With this, the contributions of λ_i decreases as i increases and the function of the costate becomes dependent upon the dominate mode of the graph Laplacian, namely v_0 which is encapsulated in the e^{Lt} terms of Eq. (4.17). Equation (4.12) states that, with the final value of the costate equal to zero, the integral of μ must equal zero. Therefore, the optimal solution is given by $\mu_0 = 0$ and B being proportional to v_0. Intuitively, this coincides with the logical solution where optimal formation keeping would be achieved by providing the input equally to all nodes. Another interesting aspect of B being dependent upon the constant mode is that the magnitude of the optimal B does not affect the solution, only the direction. In reference to centrality, this is counter intuitive. Centrality determines nodes of importance, isolating them as optimal locations for an input where as the optimal solution demands the input to be provided equally to all nodes.

The actual problem definition of this work requires that the input be placed at a single node, designated as the leader. Therefore the optimal solution is not feasible so that a suboptimal form of the input vector must be found. For a single input, the B vector takes the form of a binary vector with a single nonzero element corresponding to the leader's position in the formation. Looking at Eq. (4.17) and setting it to zero, the derivative of the costate is set to zero in order to find the form of B that keep the costate as close to zero as

possible.

$$0 = \left[\sum_{i=1}^{n-1} \left(e^{-\lambda_i t} - e^{-2\lambda_i t} \right) v_i v_i^T \right] B \tag{4.18}$$

The ascending order of the eigenvalues make the contributions of v_i decreases as i increases. With this, the modified input vector B should be in the null space of the outer product of the next dominant mode of the graph Laplacian, v_1. Looking to excite this mode the least as to best satisfy the cost function in Eq. (4.8), the nonzero element of the B vector should correspond to the lowest element of v_1 such that the combination $v_1^T B$ is minimized.

$$B(j) = 1 \quad \text{where} \quad \min_j v_1(j) \tag{4.19}$$

By adding the constraint on the input vector, the results of the optimal control problem match that of the DLC implementation. Therefore, by determining the DLC vector of the formation from the graph Laplacian, the mode located corresponds to the mode B should be projected into in order to minimize the contribution of the slowest mode of the graph Laplacian. The case of minimizing the separation of the formation states required that the minimum of v_1 be selected as the input location, a conclusion that the also drawn from the application of the DLC measure through observation of the spectral properties of the formation's graph Laplacian.

4.2.2 Finding the Formation Leaders

Selection of a single agent to control a purely leader-follower or virtual-structure formation amounts to finding the bounds of the DLC vector. The results in this section will focus on the virtual-structure controller such that the only significant aspect of the leadership hierarchy is the determination of the overall formation leader. Selecting a leader with a small component of the DLC vector prevents excitation of slowly converging motion, and allows quick convergence of the formation to the reference motion. It is also shown that

selecting a leader with a large component of the DLC vector excites slowly converging motion, meaning at least some of the agents of the formation may be slow to converge to the reference motion. However, that trade-off is that this selection allows the leader itself to more quickly converge to the reference motion. Additionally, the reference motion is chosen as $x_r = 1$. The figures that follow note the formation error as the separation distance from the formation's centroid. Defined as the difference between the states of the agents and the formation center, the formation error is given in Eq. (4.21).

$$\epsilon = \left(\frac{\mathbf{1}^T x(t)}{n} \right) \mathbf{1} - x(t) \tag{4.20}$$

$$e(t) = \sqrt{\epsilon^T \epsilon} \tag{4.21}$$

Consider the toy network given in Fig. 4.9. Choosing the first nonzero eigenvalue of the graph Laplacian, as seen in Table 4.9, the corresponding eigenvector shows the ranking of the nodes in terms of their contributions to the communication dynamics. Selecting the agent with the lowest centrality measure, Agent 3, the positions of the agents as they converge onto a common state are given in Fig. 4.10.

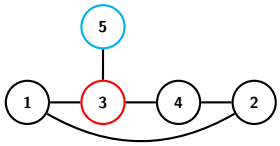


Figure 4.9: Toy network.

Ordered Eigenvalues
0.0000
0.8299
2.0000
2.6889
4.4812

ID	$\lvert v_0 \rvert$	$\lvert v_1 \rvert$	$\lvert v_2 \rvert$	$\lvert v_3 \rvert$	$\lvert v_4 \rvert$
1	0.4472	0.2560	0.7071	0.2422	0.4193
2	0.4472	0.4375	0.0000	0.7031	0.3380
3	0.4472	0.1380	0.0000	0.5363	0.7024
4	0.4472	0.2560	0.7071	0.2422	0.4193
5	0.4472	0.8115	0.0000	0.3175	0.2018

Table 4.9: Spectrum of the graph Laplacian for the toy graph in Fig. 4.9.

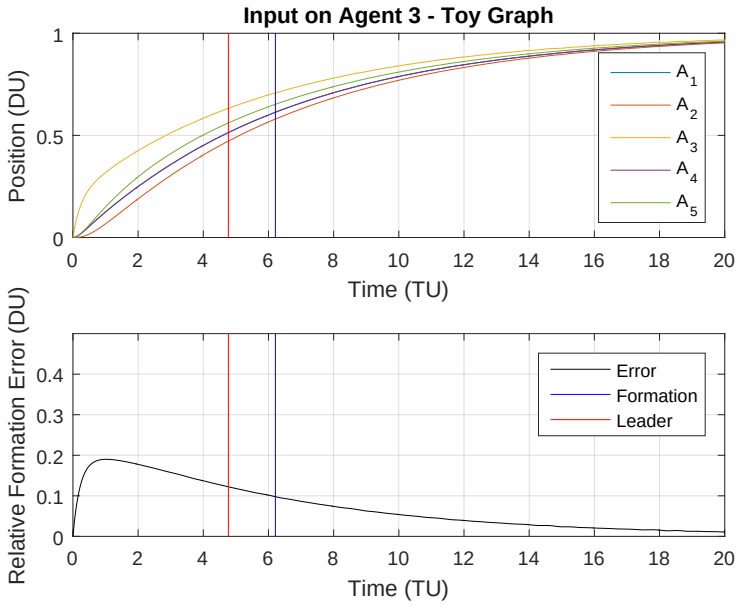


Figure 4.10: Agent states as they converge onto the desired input placed at Agent 3.

The first plot in Fig. 4.10 shows the movement of each agent as they progress onto the reference trajectory. The vertical lines in each window mark the convergence threshold where the leader and the entire formation come within 63.2% of the final value, marked as red and blue, respectively. By placing the exogenous input at the node with the lowest centrality measure, the formation reduces its relative error, keeping the agents close together as they progress towards the target. Moving the location of the leader to Agent 5, shown in Fig. 4.11, the correlation between the centrality measure the and the characteristics of the response

becomes evident. The response time of the leader (Agent 5) in Fig. 4.11 decreases such that the leader pursues the desired trajectory more closely, leaving the formation behind. With this, the relative error of the formation increases with the increase in time to converge of the formation.

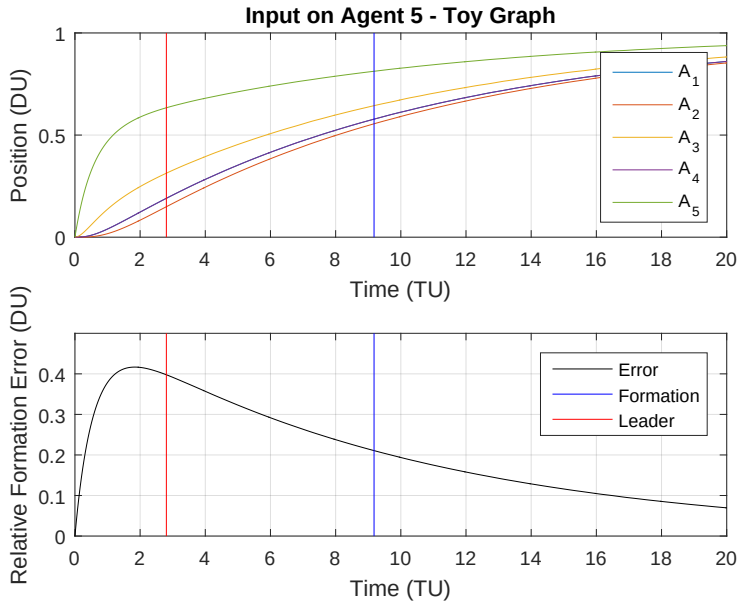


Figure 4.11: Agent states as they converge onto the desired input placed at Agent 5.

The DLC points to the slowest mode of the homogeneous Laplacian dynamic, which governs the formation convergence rate. The values within this mode correlate to the extent in which the slowest mode can be excited. By exciting nodes with low DLC values, the formation converges quickly. The eigenvectors also describe how much each state is excited within each mode. Selecting a leader with a low DLC value means the leader responds quickly to the homogeneous formation dynamics, helping to keep the formation intact.

Alternatively, selecting nodes with large DLC values provides high excitation of the slow mode and slows the formation response. The trade-off for this is that the leader also responds slowly to the homogeneous formation dynamics, allowing it to respond more quickly to the exogenous input. This makes the leader more responsive to the reference trajectory.

The above discussion provides the relationship between the communication dynamics of the network and the guidance dynamics of the formation. In terms of formation types, by selecting a leader with a lower DLC value, its location will support a virtual-structure type controller, providing the formation with the opportunity to reduce formation error and retain the structure. At the other end of the continuum, selecting an agent with a higher DLC value allows that leader to better follow the desired trajectory and decrease its own response time.

4.2.3 Finding Formation Leaders - Special Graphs

The presence of a repeated value of λ_1 and other repeated eigenvalues implies symmetry in the graph about certain nodes, particularly inherent to the special graphs defined in Section 4.1. Looking at Fig. 4.12, the outlying nodes are symmetrically distributed around the center node.

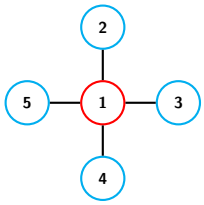


Figure 4.12: Star graph, 5 nodes.

Ordered Eigenvalues
0.0000
1.0000
1.0000
1.0000
5.0000

ID	$\lvert v_0 \rvert$	$\lvert v_1 \rvert$	$\lvert v_2 \rvert$	$\lvert v_3 \rvert$	$\lvert v_4 \rvert$	$\lVert v \rVert$
1	0.4472	0.0000	0.0000	0.0000	0.8944	0.000
2	0.4472	0.7887	0.2113	0.2887	0.2236	0.866
3	0.4472	0.5774	0.5774	0.2887	0.2236	0.866
4	0.4472	0.2113	0.7887	0.2887	0.2236	0.866
5	0.4472	0.0000	0.0000	0.8660	0.2236	0.866

Table 4.10: Spectrum of the graph Laplacian for the star graph in Fig. 4.12.

The DLC vector reflects the symmetry of the network. Agents 2-5 have equal DLC values indicating similar responses from the selection of any of the these agents as the network leader. In Fig. 4.13, with the input located at the center of the star formation (Agent 1), the relative position error between the agents reduces.

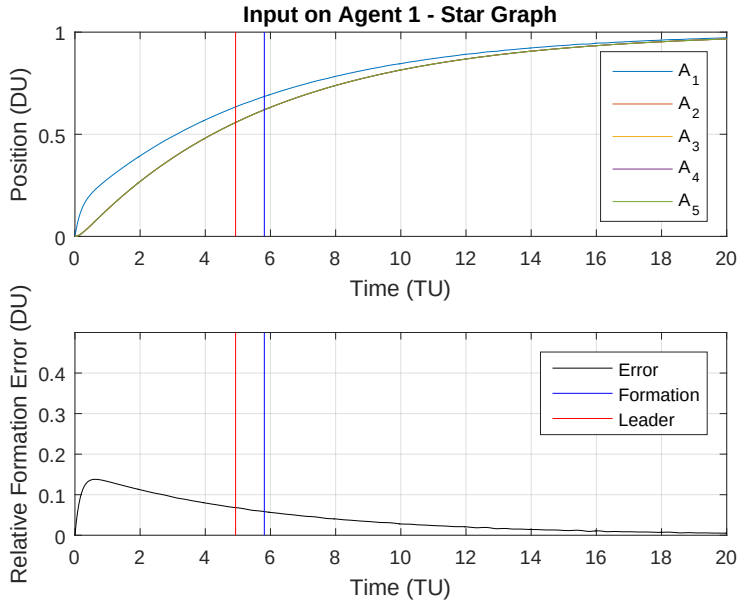


Figure 4.13: Agent states as they converge onto the desired input placed at Agent 1.

The DLC value corresponding to the center agent, noted in Table 4.10, occupies the minimum value of the centrality vector such that the lower centrality values correspond to virtual-structure type formations for special graphs with repeated values of λ_1, aligning with the analysis given in Section 4.2.2. The outlying nodes, Agents 2-5, retain equivalent DLC values by the method described in Definition 4.1 and their dynamic response to an exogenous input associates with a leader-follower formation response.

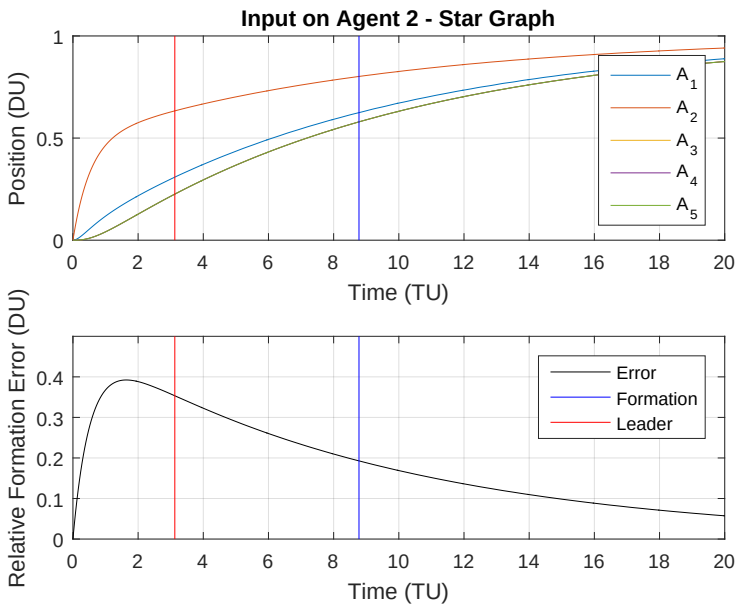


Figure 4.14: Agent states as they converge onto the desired input placed at Agent 2.

Figure 4.14 shows the faster response of the leader and larger separations in the formation. This amounts to selecting a node with a larger DLC value. The above analysis extends to cases where only the spectral properties of the network are available. With this, by knowing the characteristics of the graph's spectrum, a description of the topology leads to information about the heterogeneous nature of the graph, as well as the existence of certain symmetries [37].

4.3 Agent Hierarchy

The DLC provided a global leader for the decentralized network of unmanned vehicles. For certain control formulations a more detailed chain of command must be established within the network. Designating a set of leaders and followers within each agent's local neighborhood permits the proposed controller to fade between leader-follower and virtual-structure architectures depending on the reliance of an agent on these sets. Consider a hypothetical relationship between neighboring agents that allows the depends on their local

leader's and follower to vary.

$$u_i(t) = -\frac{k_i^L}{|\mathcal{N}_L(i)|}\sum_{j\in\mathcal{N}_L(i)}\left(x_i(t) - x_j^L(t)\right) - \frac{k_i^F}{|\mathcal{N}_F(i)|}\sum_{\ell\in\mathcal{N}_F(i)}\left(x_i(t) - x_\ell^F(t)\right) \tag{4.22}$$

The control gains k_i^L and k_i^F would vary the dependence the control u_i has on its leader set, $\mathcal{N}_L(i) \subset \mathcal{N}(i)$, and follower set, $\mathcal{N}_F(i) \subset \mathcal{N}(i)$, respectively. The implementation of such a controller would require definitive sets of leader's and followers for each agent such that agent i could determine their own control locally. The proposed guidance controller will feedback on the estimates of the Kalman filters such that Eq. (4.22) provides a notional illustration of how the control works, described in more detail in Chapter 6. An additional variation of this guidance controller leverages only an agent's measured states of their local neighborhood to stabilize the global formation, thereby make use of an instantiated hierarchy. Therefore, a local appointment of leadership is required. Through the DAA, each agent is aware of the existence of the entire formation, and knowledge of this existence will be used in determining the agent hierarchy, but knowledge of the states of the entire formation will not be used to evaluate the control.

The difference in neighboring agent states, $x(t)$, extends the application of the DLC formulation such that an addendum to its leadership application follows a description of the network hierarchy formulation. The DLC values of each agent in the formation provide a numerical description of their contribution to the dynamic characteristics of the network and are designated as node weights. By locally determining the DLC values in the network, each agent knows the DLC value of every node in the neighborhood, therefore alleviating further computation after the leadership selection process. For topologically homogeneous formations where all nodes have similar degrees of connection, neighboring agents may hold equivalent DLC values. With this, by initially assigning each edge a unit weight, no edge will result in a value of zero when comparing neighboring node weights, i.e. neighboring DLC values. Figure 4.15 describes the process of weighting two neighboring nodes. Each

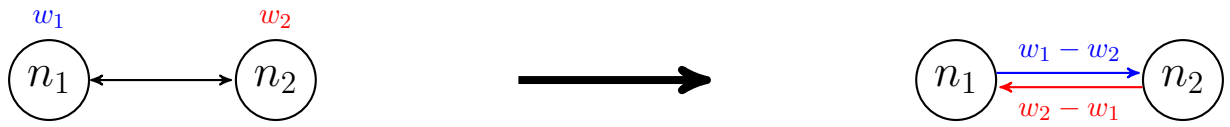


Figure 4.15: Weighting procedure between neighboring nodes. The difference between the transmitting node and the receiving node defines the weighting with an added factor of one.

agent holds a DLC value, indicated as w_i above each node, n_i. The cost, or weight, to travel from one node to the next translates into the relative difference from moving from one DLC value to the next. The arrows originating at the transmitting node and terminating at the receiving node indicates the directedness of the weight such that moving from node n_1 to n_2 has a different cost than moving from node n_2 to node n_1, assuming that each node has a unique DLC value.

Continuing this process for every node pair in the network defines the decentralized weighting process to determine hierarchy. By imposing weights, the cost to move from one node to the next creates a chain of command within the network. Through the application of Dijkstra's method of determining the least weighted path to all nodes of the network, each node then determines its leader and follower sets [38].

4.4 Implementation

Consider the example formation given in Fig. 4.16 where each node and edge represents an unmanned vehicle and communication link, respectively. As previously described, the direction of the arrows in the figure represent the orientation of the transmitting node to the receiving node, accompanied by its respective edge weighting. The implementation of Dijkstra's method requires the designation of a source, or leader, in the network. Placing the reference input, $r(t)$, at node "A", and applying Dijkstra's method to determine the least weighted path to each node, the reorientated network shows the chain of command in the formation. Figure 4.17 shows this hierarchy, using solid arrows to indicate the subordinate to each node. Dijkstra's method provides a unique relationship between nodes, allowing each

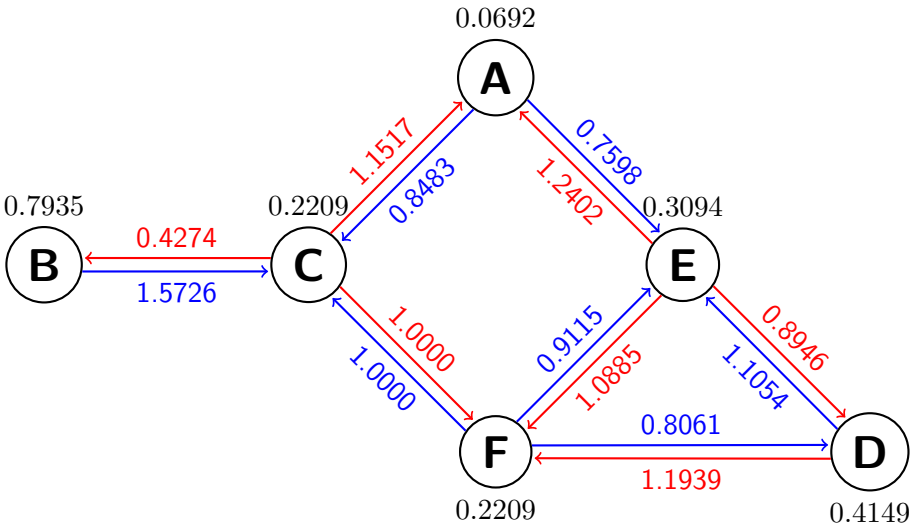


Figure 4.16: Edge weights for the directed graph.

agent a single leader. Doing so excludes two of the existing lines of communication connecting "C" to "F" and "D" to "F". Allowing a mutual dependence along the communication link, this edge represents a follower edge where each agent along the edge views its counterpart as a follower.

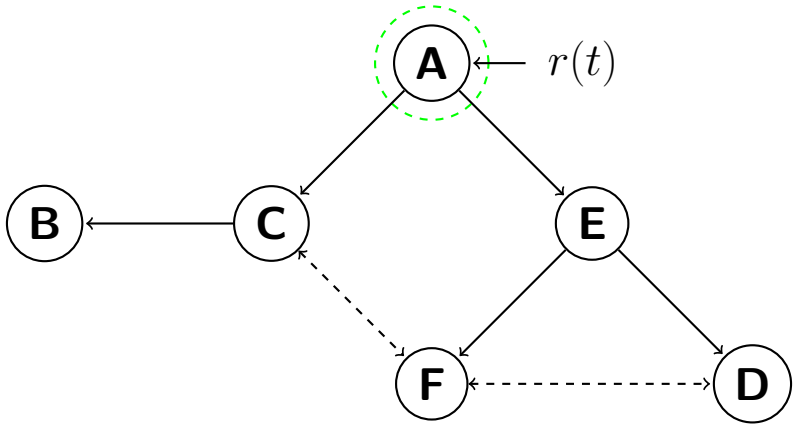


Figure 4.17: Hierarchy using node A as the source.

Figure 4.18 displays the same communication topology with the exogenous input place at node "F". Here, Dijkstra's method determines that node "F" should reach node "A" through node "E", which would incur the same cost as reaching "A" through node "C". Therefore,

this method handles ties and decision making in the development of the hierarchy, making this method appealing to rapid construction of the decentralized hierarchy.

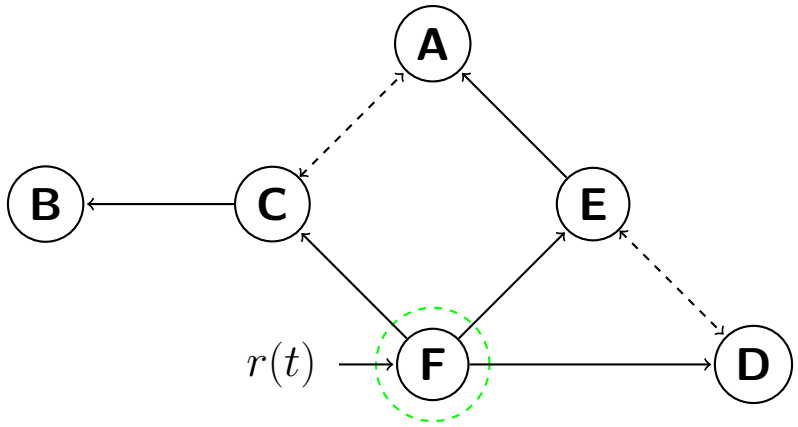


Figure 4.18: Hierarchy using node F as the source.

As the control on each agent fades from a virtual-structure ($k_i^L = k_i^F$) to a leader-follower ($k_i^L = 1$, $k_i^F = 0$) for the configurations in Figs. 4.17 and 4.18, the hierarchy resolves into the least weighted paths between the source and the nodes as determined by Dijkstra's method. The follower edges lose their dependence on their counterparts, thereby creating a controller capable of transitioning continuously between the two bounds of the spectrum.

4.5 Benefits of Approach

Implementing this hierarchical methodology comes with many benefits. Allowing each agent to understand the hierarchy locally, as developed on-board, prevents information delay typically incurred through formalizing a hierarchy in a centralized approach. Building upon the leadership selection process, as the agents learn of the global leader, they become aware of their leader and follower sets locally. Determining their local control based on their respective neighborhoods, this hierarchical development abstains from collecting unneeded information about the extended network. Dijkstra's method consistently handles ties and decisions based on its rigorous formulation such that the decentralized formation remains

self-reliant in terms of chain of command and structure keeping. Also, the ease at which Dijkstra's method computes the least weighted path for networks of relative size makes it a robust, on-line application to the decentralized control problem. Throughout the remainder of this problem, the leadership selection will produce a global leader capable of commanding the entire formation. From this, the above development of the hierarchy instantiates the chain of command within the decentralized formation.

4.6 Implementation of the DLC

The DAA was introduced in Chapter 3 in order for the decentralized network of agents to learn the global formation structure through the local exchange of limited information. The local implementation of the DLC allows each agent to determine not only their local leader and hierarchy within their respective neighborhoods, but to determine the global formation leader. By incorporating the DLC into the DAA method, the local agents iteratively learn the network and determine leadership in a decentralized manner while simultaneously accessing any changes to the topological structure.

To demonstrate the application of the DLC within the DAA, consider the network structure given in Fig. 4.19. Agent 560, indicated by the solid green node, denotes the global leader of the formation, of which the decentralized formation is working to converge onto. The knowledge of this node as being the global leader is unknown to the decentralized agents. Each agent is initialized as the center of their own star network, building their adjacency matrix in ascending order of their neighbor's ID. Figure 4.20 shows the implementation of the DLC by every local agent. With each agent initially situated at the center of their own star network, the DLC determines that they all are the leader of the formation, highlighted by red circles. Next, every agent compares their perspective of the network with their neighbors, determine that their neighbors have new information about the network, and require an update of their local adjacency matrix, denoted in blue circles.

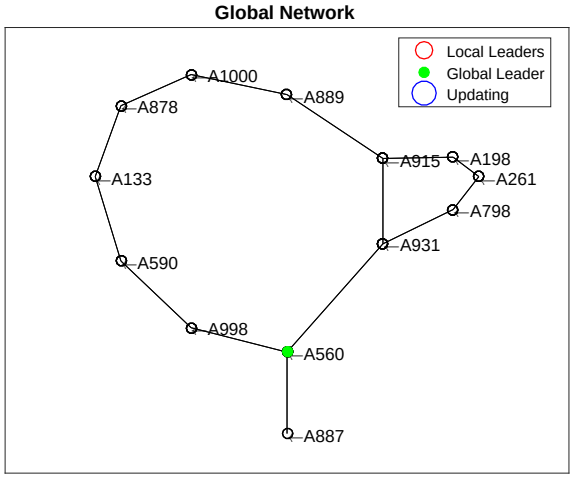


Figure 4.19: Decentralized network.

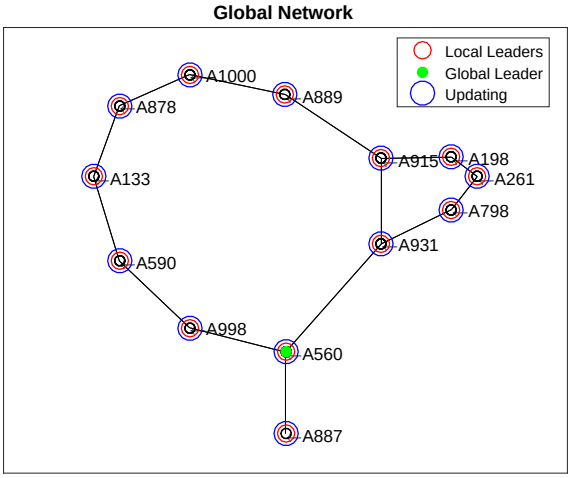


Figure 4.20: Iteration 1.

The next iteration allows each agent to reevaluate their understanding of the network and implement the DLC to determine the formation leader. Figure 4.21 shows how the formation agrees on two different agents for leadership: Agent 560 and Agent 878. The network compares their understanding of the network with their neighbors and those who notice a change from their last iteration perform an update. Figure 4.22 notes the movement of the update from the well connected cluster who learned the network quickly, to the line of agents who previously came to consensus on their understanding of the network, determining Agent 878 as their leader.

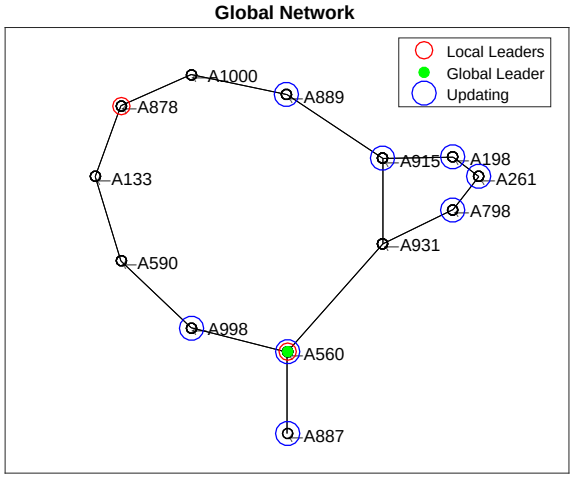


Figure 4.21: Iteration 2.

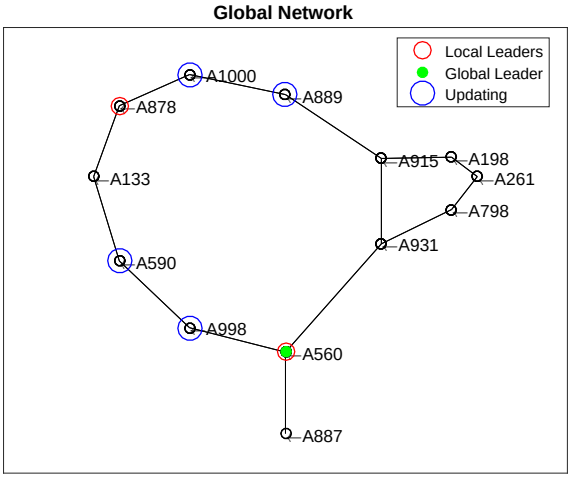


Figure 4.22: Iteration 3.

The continued observation of the agent's local neighborhood is illustrated in Fig. 4.23 where the agents who previously determined Agent 878 as leader begin to notify their surrounding agents of another, more capable leader, in the network. Finally, Fig. 4.24 shows the convergence of the network onto a single leader, Agent 560. A network of thirteen decentralized agents converged onto a single leader in six iterations by adjusting their local adjacency matrices to learn the structure of the formation through local exchange of limited information and implementing the DLC leadership protocol.

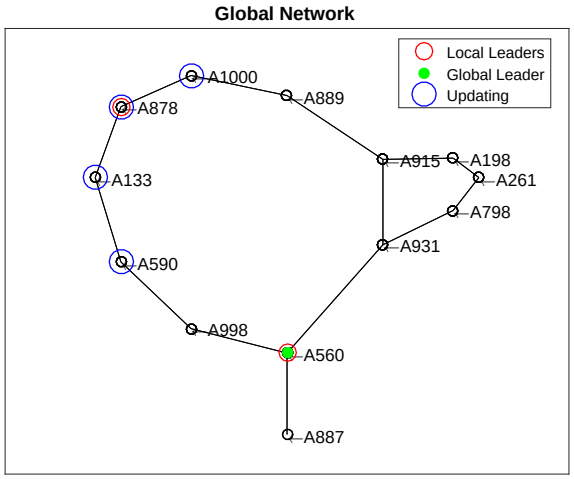


Figure 4.23: Iteration 4.

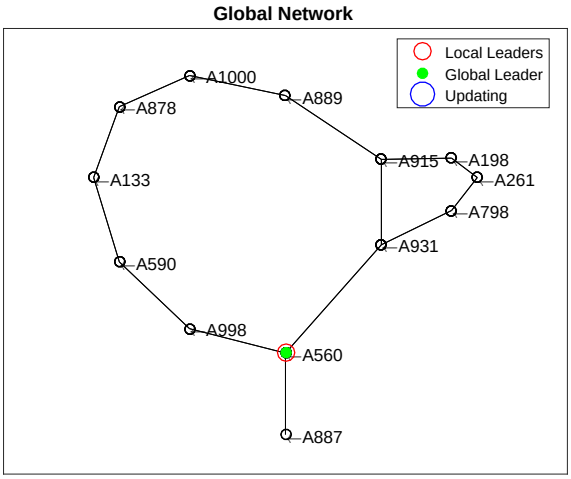


Figure 4.24: Iteration 5.

4.7 Conclusions of the Dynamic Laplacian Centrality

Centrality measures aim to locate important, or powerful, players in a network. To this end, the notion of importance lies in the application such that a key player in one sense may prove secondary to another player for a different application. When applying centrality measures to formation control, looking for a single leader to best orchestrate the movement of the formation becomes a function of the underlying communication. Asking a controller to compensate for the limitations of the established communication structure will hinder the performance of the formation or produce undesirable behaviors. By declaring an agent powerful because of its ability to convey, or influence, the states of the network, leadership selection becomes a function of the communication dynamics rather than a topological location in the network. The dynamic Laplacian centrality identifies key players, exploiting the first-order dynamics of the communication, and acts as a first approach to predicting the behavior of formations using higher order dynamics. The DLC measure identifies nodes of the network, based on their natural response to an exogenous input, and classifies them as exhibiting leader-follower and virtual-structure tendencies. In terms of formation response, this information can lead to selecting an input location that better supports a network with limited communication ability, or utilizing agents with the ability to converge quickly onto the desired states.

Chapter 5

Waypoint Following

Here it is assumed that the ground controller provides the leader with a series of successive waypoints to visit, allowing the lead agent to look ahead in the path to determine its required control. The manner in which the lead vehicle approaches each waypoint defines the characteristics of the generated trajectory. In an effort to incorporate every waypoint along the path, Choi implemented cubic Bézier curves to successively connect segments between waypoints as the ground vehicles maneuvered within a bounded highway system [39]. This work presents a variation of Choi's approach by allowing more freedom in the path planning that is better suited for aerial vehicles.

In this current application for aerial vehicles, intercepting each point in a point-by-point approach leads to extensive position error and control use. A least square method provides a better alternative over a point-by-point progression, particularly when accounting for irregular GPS waypoints [40]. Although the least square method has its merits when dealing with measured waypoints, the generated path is not guaranteed to fit to the initial and final waypoints without applying additional weighting to enforce the end conditions. Other approaches often incorporate splines of lower order to adjoin segments along the path. Splines match both the slope and the curvature at interior points, providing a smooth transition between waypoints along the generated path. Often, the endpoints of the path assume a clamped condition in order to solve for the trajectory, thereby restricting the available shape of the path. This constraint may lead to unobtainable trajectories in bounded environments.

Opposed to using splines to connect segments of waypoints, Bézier curves allow further shaping of the interior points of the segments, which is particularly helpful in applications

involving collision avoidance and path planning within bounded corridors [41]. Also, Bézier curves allow the ground controller to supply additional waypoints during the mission, providing more flexibility to changing environments and mission objectives. This work provides an alternative to Choi's approach through the use of radii around the waypoints and demonstrates their ability to adjoin successive segments within a mission environment.

5.1 Bézier Curves

Pierre Bézier, a French automotive engineer, invented Bézier curves in order to design intricate body shapes over smooth surfaces. Greater success of Bézier curves materialized in their application to graphic design due to their ability to define complicated surfaces with minimal computational effort. In this application, cubic Bézier curves form the path links between the given waypoints, providing the lead vehicle with a smooth path as it moves through the mission environment. For Bézier curve $C(\tau)$ of degree n, Eq. (5.1) defines a parametric equation in terms of τ where $0 \leq \tau \leq 1$.

$$C_{[\tau_0,\tau_1]}(\tau) = \sum_{i=0}^{n} B_i^n(\tau) P_i \tag{5.1}$$

The Bernstein polynomial, $B_i^n(\tau)$ is defined using the binomial coefficient, $\binom{n}{i}$, in Eq. (5.2).

$$B_i^n(\tau) = \binom{n}{i} \tau^i (1-\tau)^{n-i} \quad \text{with} \quad \binom{n}{i} = \frac{n!}{i!(n-i)!} \tag{5.2}$$

As described in [42], the instability of higher order Bézier curves suggest solutions of lower order to have desirable characteristics. An order of three permits the matching of both slope and curvature at the interior waypoints such that by substituting $n = 3$ into Eq. (5.1), the cubic form of the Bézier curve is provided.

$$C_{[\tau_0,\tau_1]}(\tau) = (1-\tau)^3 P_0 + 3\tau(1-\tau)^2 P_1 + 3\tau^2(1-\tau) P_2 + \tau^3 P_3 \tag{5.3}$$

Writing Eq. (5.3) in matrix form by expanding the coefficients, the parametric curve becomes a function of its control points, P_i, computed using the compact form in Eq. (5.4).

$$C_{[\tau_0,\tau_1]}(\tau) = \begin{bmatrix} P_0 & P_1 & P_2 & P_3 \end{bmatrix} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \tau^3 \\ \tau^2 \\ \tau \\ 1 \end{bmatrix} \tag{5.4}$$

The control points, P_i, are column vectors of two or three dimensions in the Cartesian frame so that the generated track may be provided in two or three dimensions. These curves are parametric curves that have three useful properties for trajectory generation [39].

1. They always pass through P_0 and P_3
2. They are always tangent to the lines $P_0 \rightarrow P_1$ and $P_2 \rightarrow P_3$
3. They always lie within the convex hull consisting of their control points

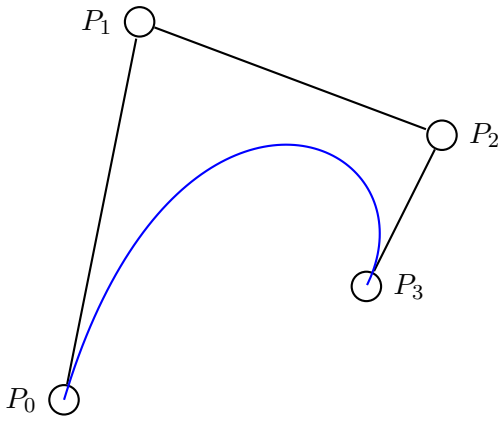


Figure 5.1: Properties of Bézier curves.

The first and second properties help construct continuously smooth curves between the waypoints while the third property bounds the path, as demonstrated in Fig. 5.1. The exterior points of each segment, defined as control points P_0 and P_3, represent the waypoints

and provide two of the four required control points for the cubic Bézier curve. Choosing the interior control points, P_1 and P_2, such that the path remains contained within the defined boundaries will have a direct impact on the control input needed to follow the generated path. With this, the next section illustrates how to select the interior control points and demonstrates the computational efficiency of this approach to path generation.

5.2 Determining Interior Control Points

The defining characteristics of Bézier curves lend themselves to path generation due to their ability to connect a series of waypoints without discontinuities and allow a level of control by the user to shape the trajectory. In [39], the borders of a highway restricted the path of the ground vehicles such that rectangles defined the control areas. The third property, stated previously, contains the path within a polygon with the control points defining the convex hull. In this application, each waypoint has an accompanying radius, as shown in Fig. 5.2, that designates how the vehicles approach the waypoint, W_i. Allowing the user to pick the radii around each waypoint adds another level of control over the formation and permits the user to indirectly design the generated path for the lead vehicle.

With each set of waypoints, a segment defines the path between consecutive waypoints. Each segment consists of four control points where the waypoints themselves define the exterior control points, P_0 and P_3. The selection of the interior point, indicated in Fig. 5.2 as (P_x, P_y), remains as the objective of this section. To begin, the intersection point of the tangent lines to the circles, (x_p, y_p), must be located using Eqs. (5.5)-(5.6), where r_j is assumed to be greater than r_i.

$$x_p = \frac{r_i W_{jx} - r_j W_{ix}}{r_i - r_j} \tag{5.5}$$

$$y_p = \frac{r_i W_{jy} - r_j W_{iy}}{r_i - r_j} \tag{5.6}$$

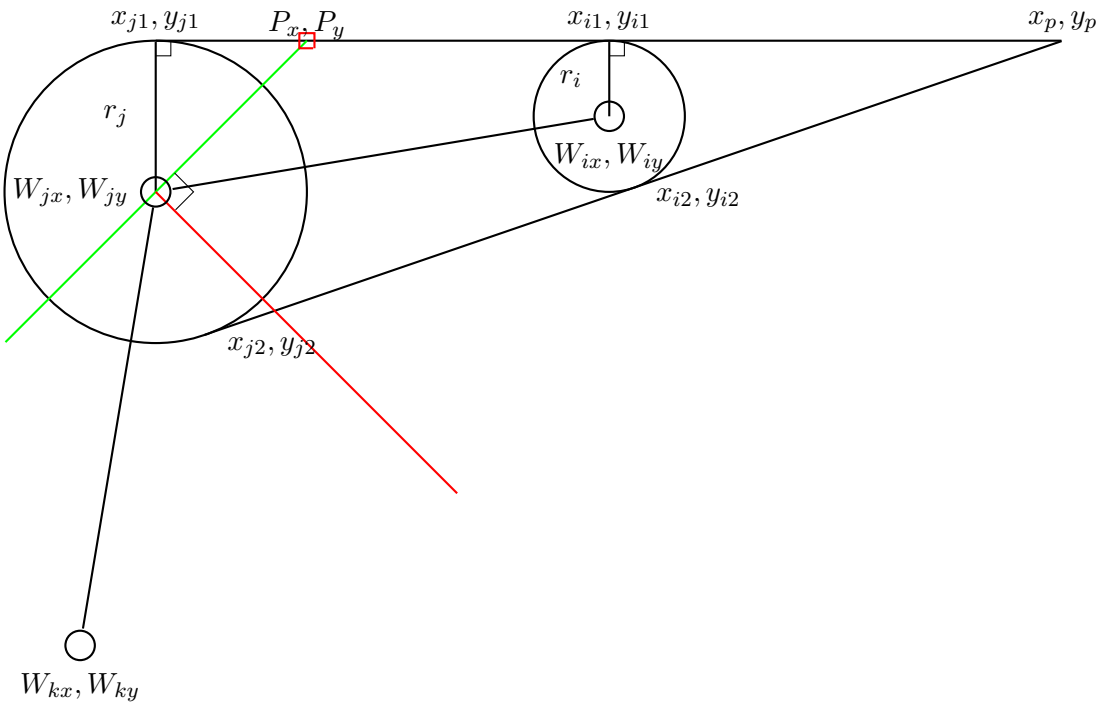


Figure 5.2: Determining the interior control point.

The location of the tangent points connecting the two circles are found using Eqs. (5.7)-(5.8).

$$x_{j(1,2)} = \frac{r_j^2(x_p - W_{jx}) \pm r_j(y_p - W_{jy})\sqrt{(x_p - W_{jx})^2 + (y_p - W_{jy})^2 - r_i^2}}{(x_p - W_{jx})^2 + (y_p - W_{jy})^2} + W_{jx} \tag{5.7}$$

$$y_{j(1,2)} = \frac{r_j^2(y_p - W_{jy}) \pm r_j(x_p - W_{jx})\sqrt{(x_p - W_{jx})^2 + (y_p - W_{jy})^2 - r_i^2}}{(x_p - W_{jx})^2 + (y_p - W_{jy})^2} + W_{jy} \tag{5.8}$$

Each circle will have two sets of tangent points, indicated as $x_{i,(1,2)}$ for the i^{th} circle. As indicated in Fig. 5.2, the tangent line to the circle and the radial line from the waypoint to the tangent point should be normal. This is used to identify the correct combination of coordinates on each circle. The points on the smaller circle, r_i, are found in a similar fashion through substitution in Eqs. (5.7)-(5.8). With the location of the tangent points determined on each circle, two separate equations are formed to describe the lines connecting the circles,

given in Eqs. (5.9)-(5.10).

$$y = \frac{y_p - y_{j1}}{x_p - x_{j1}}(x - x_{j1}) + y_{j1} \tag{5.9}$$

$$y = \frac{y_p - y_{j2}}{x_p - x_{j2}}(x - x_{j2}) + y_{j2} \tag{5.10}$$

The bisection of the angle, V_b, connecting waypoint W_i to W_k is found using Eq. (5.11).

$$v_b = |u|\, v - |v|\, u \tag{5.11}$$

where

$$u = W_j - W_i \quad \text{and} \quad v = W_k - W_j \tag{5.12}$$

With the bisection found in Eq. (5.11), the equation of the line perpendicular to the bisection, v_t, is determined (for a two dimensional problem) in Eq. (5.13).

$$v_t = \left\{ -v_{b,y} \quad v_{b,x} \right\} \tag{5.13}$$

In Eq. (5.13), $v_{b,x}$ and $v_{b,y}$ indicate the first and second elements of v_b corresponding to the x and y coordinates, respectively. With this, the equation of the perpendicular line is found using the vector v_t.

$$y_t = m_t\,(x - W_{jx}) + W_{jy} \quad \text{with} \quad m_t = \frac{v_{t,y}}{v_{t,x}} \tag{5.14}$$

Solving Eq. (5.9) and (5.14) simultaneously, the control point (P_x, P_y) is found.

$$P_x = \frac{m_t W_{jx} + y_{j1} - W_{jy} - \frac{y_p - y_{j1}}{x_p - x_{j1}}}{m_t - \frac{y_p - y_{j1}}{x_p - x_{j1}}} \tag{5.15}$$

$$P_y = m_t\,(P_x - W_{jx}) + W_{jy} \tag{5.16}$$

5.3 Application to Waypoint Following

The process of generating the trajectory, given that the ground controller systematically provides a least two future points to visit, is demonstrated in this section. Figure 5.3 indicates the visited waypoint and the future waypoints provided by the ground operator in order to generate the desired trajectory. Each waypoint has a set radius that may be used to shape the curve between the past and future waypoints along the path. By increasing the radius around a waypoint, the curvature at that waypoint increases, providing a more gradual turn. Conversely, decreasing the radius around a waypoint provides sharp turns. This additional design characteristic of the waypoints supplies the ground controller with the ability to manipulate the trajectory of the path, maneuver the formation around known obstacles, and restrict the inter-vehicle spacing without directly manipulating the control on the individual vehicles in the formation.

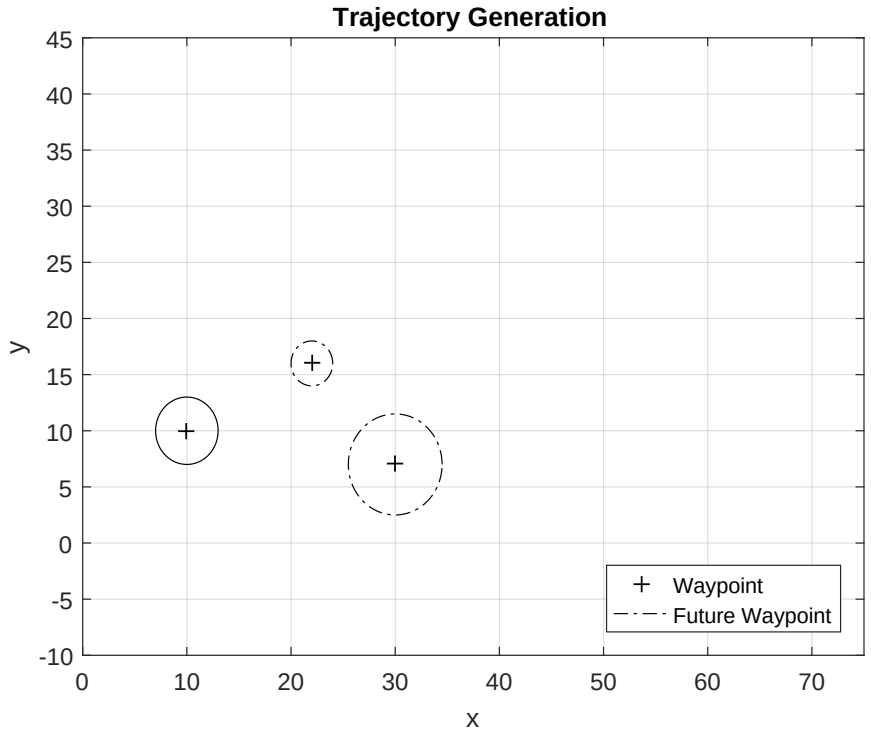


Figure 5.3: Providing the leader with two future way points. The black radius indicates the visited waypoint and the dashed radii indicate the future waypoints to visit.

This method requires four control points to generate a segment between waypoints. The two exterior points, P_0 and P_3, are defined by the two waypoints the Bézier curves are connecting such that the remaining control points lie between these endpoints of the individual segment. In order to smoothly adjoin consecutive segments, an additional waypoint is required for the agent to look ahead and adjust the current segment to transition smoothly at the next waypoint. Given that the first and last segments of a track are not required to meet a curvature condition at the waypoint, a position within the segment must be selected to begin the first segment and end the last segment of a trajectory.

$$P_1(\tau) = (1 - \eta)P_0 + \eta P_3 \tag{5.17}$$

Wanting the lead agent to start in the direction of the next waypoint, P_1 of the first segment is chosen along the line connecting the first two waypoints using the parametric equation in Eq. (5.17), with η being the distance along the parametric curve. The first segment in Fig. 5.4 is generated by incorporating the last visited waypoint (solid radius) and the two future waypoints provided by the ground controller (dashed radii). The absence of a previous waypoint requires that P_1 of the first segment lies along the line connected the first pair of waypoints, located using Eq. (5.17). The second interior point, P_2 is found as described in Section 5.2 and indicated using a red square.

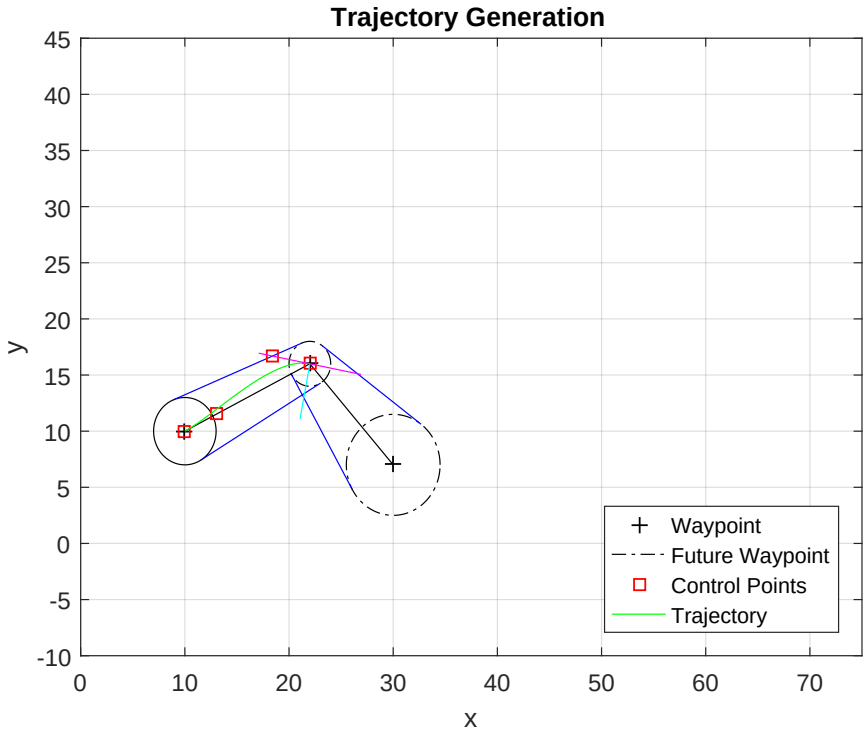


Figure 5.4: Generating the trajectory within the first segment of the sequential waypoints.

As the agent approaches the second waypoint, it is provided with the fourth waypoint as to generate the trajectory in the second segment, given in Fig. 5.5. For segments lying within the terminal points of a mission, the interior control points are determined from the past and future waypoints. The four control points of the second segment, shown in Fig. 5.5, are generated and implemented prior to the agent passing the last waypoint so that each segment may be connected smoothly. The stipulation that the line perpendicular to the bisection created by successive waypoints contain control points P_2 of the previous segment and P_1 of the current segment ensures that the transition between segments is continuous.

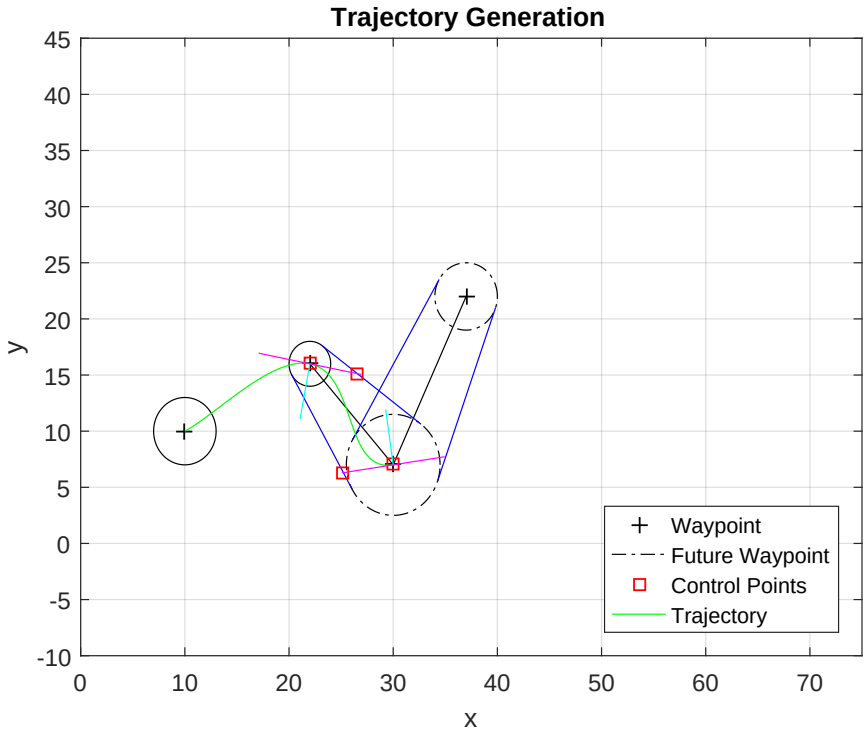


Figure 5.5: Generating the trajectory within the second segment of the sequential waypoints.

By adding another waypoint, seen in Fig. 5.6, this process continues as the ground controller applies more waypoints to visit along the mission timeline. If the agent approaches a waypoint and is only provided with a single look-ahead waypoint, it is assumed that the agent is approaching the final segment of the mission. Similar to the initial segment, the agent should finish along the direction of the last waypoint such that P_2 of the final segment is selected along the line connecting the last pair of waypoints using Eq. (5.18).

$$P_2(\tau) = \eta P_0 + (1 - \eta) P_3 \tag{5.18}$$

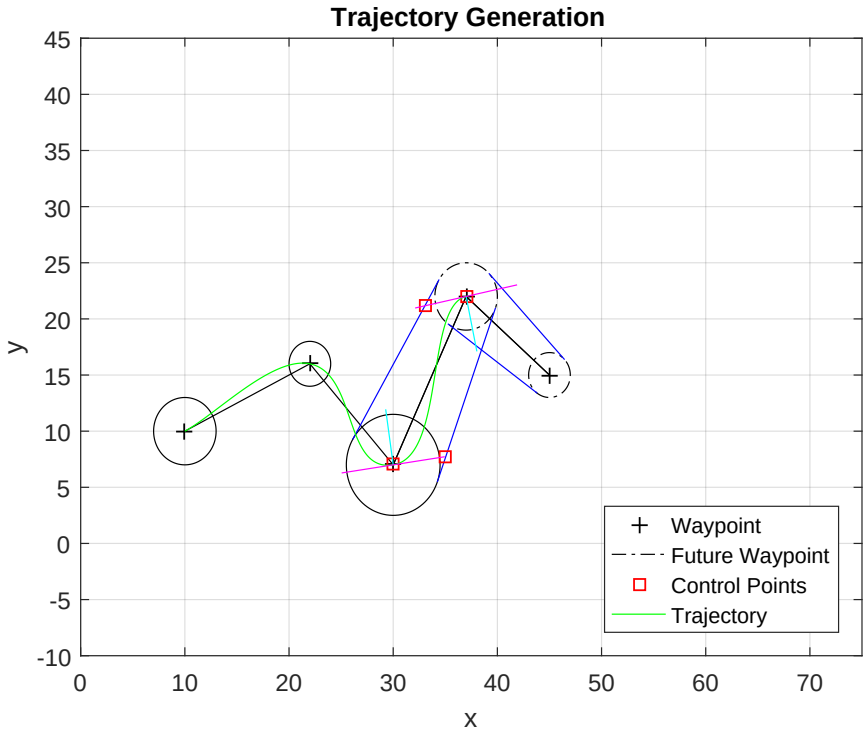


Figure 5.6: Generating the trajectory within the third segment of the sequential waypoints.

The benefit of employing this method of trajectory generation lies in the ability of the agent to determine its own track and provides the ground controller the option of supplying as little as two waypoints in the future for the formation to follow. Also, a batch of waypoints may be provided to the agent to effectively generate longer tracks, as shown in Fig. 5.7.

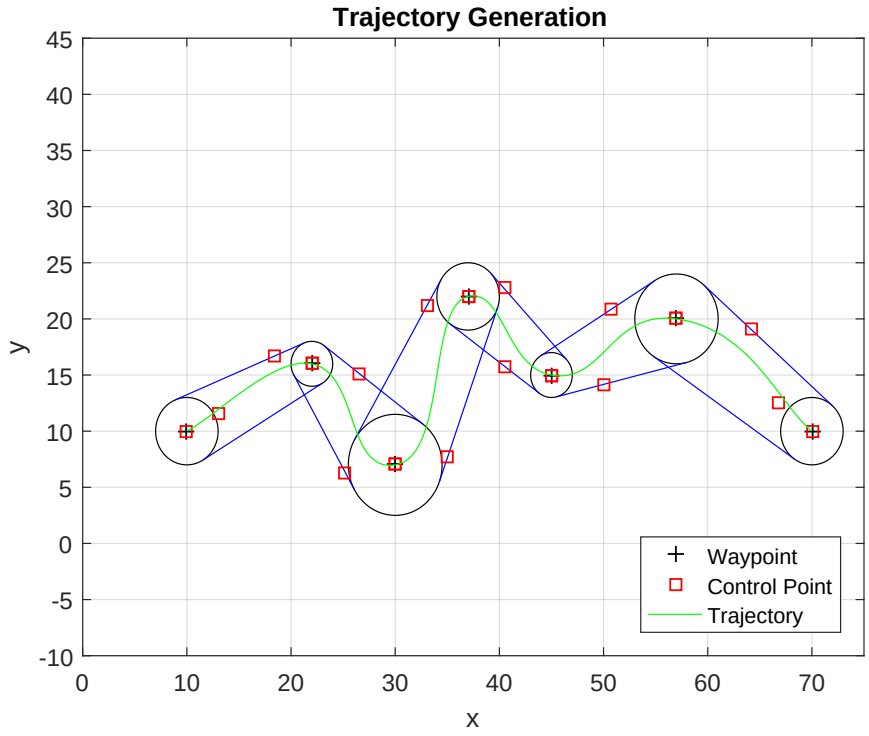


Figure 5.7: Constructing the path using the control points and connecting all waypoints.

With multiple waypoints provided, the agent employs the same method, evaluating each segment individually and allowing the Bézier curve implementation to satisfy the endpoint conditions using a low order approach. This method of path generation was chosen so that each waypoint is incorporated into the mission. Also, by allowing the user the flexibility of controlling the size of each waypoint's radius, the ground controller further influences the formation, retaining the ability to plan paths around obstacles and challenging environments. The addition element of including radii around each waypoint provides a means of adjusting how the vehicle approaches the waypoint. For the application herein with aerial vehicles travel at higher rates that require wider breadth to make gradual turns, this novel variation of Choi's application to waypoint following tailors the use of Bézier curves to vehicles of more challenging dynamics. The methods used herein match those found in [39] when the radii are equivalent at each waypoint. Unless otherwise indicated, the size of the radii are left to

the discretion of the user and a value of $\eta = 1/4$ is used in the initial and final segments of the generated paths.

Chapter 6

Linear-Quadratic-Tracker with Graph Laplacian

A multi-agent formation of autonomous vehicles is tasked to maneuver through an obstructed environment, following a set of sequential waypoints provided by a ground operator. The agents share a global performance index but the control laws considered herein are decentralized since the agents are unaware of the commanded trajectory, allowing them to only sense the states of their nearest neighbors. The ground operator shares a single link with the formation through the formation leader. Sensor data of each agent may be processed locally using a Kalman filter so that the state of the formation may be indirectly observed.

The centralized control problem involving predetermined sets of leaders and followers for distributed control has long since been solved for a fully connected information exchange network [1, 2, 3]. As the formation size grew, thereby creating more intricate communication structures with high bandwidth requirements of moving all information to a central processor, research moved towards decentralized cooperative control. Recognizing the robust appeal of decentralized formation control, a recent push in multi-agent control has expanded upon these classic examples to include localized formation control for more sparsely connected networks [4]. Coordinated control of multi-agent systems has received great attention due to their improved productivity in completing complex tasks over their individual counterparts. Coordinated decentralized control opens the door to multiple fields of study including space, terrestrial and oceanic explorations, spacecraft formation flying, cooperative surveillance, and sensor networks [5, 6]. When establishing a hierarchy, separate sets of leaders and followers work to solve the n-trailer problem for a convoy of vehicles through the local sensing of position and velocity [7, 8, 9]. Alternatively, a consensus approach to virtual-structure control is typically implemented when coordinating vehicle formations for directed

and undirected communication topologies because of its application to graph theory and the analytic capabilities for stability analysis [10, 11]. In the application herein, elements of the consensus approach to formation control help to retain the formation structure while relaxing the predetermination of a leader-follower hierarchy.

This work develops a decentralized cooperative control strategy based on the desired response of the formation to a single exogenous input provided by a ground operator and the communication capabilities inherent to the previously established interactions amongst the unmanned agents. The controller allows an agent variable dependence on its leaders and followers, ranging from a leader-follower to a virtual-structure type architecture. In a leader-follower control strategy, separate sets of leaders and followers are designated and the control on the leader is not affected by the state of the followers. A virtual-structure control strategy treats the formation as a rigid body without any detailed hierarchy. In this application, the aforementioned architectures act as a bound on a control continuum such that a leader shares a dependence on the state of its followers, and vice versa. The location of the leader in the topology elicits certain types of behavior from the formation in terms of its response to the exogenous input, as detailed previously in Chapter 4, thereby restricting the bounds of the allowable control to stabilize the formation. Balancing these constraints with the desired behavior of the network, as requested by a ground operator, the individual agents determine their own control utilizing limited information of the network to best satisfy the wants of the user and the constraints of the graph topology, i.e. a global objective function.

In a system where a formation of unmanned vehicles must pursue a desired trajectory provided through a single leader, complications arise when providing state feedback for the LQT control. Maintenance of the formation structure, variance in behavior of the formation to the selection of the formation leader, and leveraging local information to satisfy a global performance index all pose their own difficulties. This dissertation presents a novel approach to addressing this problem by incorporating the formation constraints of the graph topology

into the design of the LQT using two separate approaches to controlling the formation. First, each agent will implement their own decentralized LQT utilizing the hierarchy previously developed in Section 4.3. Their local controller continues to utilize the graph Laplacian of the global formation, but their control implementation remains only as a function of their neighboring states. Secondly, each local agent determines the state of the entire formation using a Kalman filter. The form of the LQT requires full-state feedback so each agent is tasked with estimating the state of their non-neighboring agents of the connect formation. These states are realizable through the graph Laplacian, incorporated by the performance index, such that the extended neighbors are observable and formation keeping is obtainable. This chapter presents the centralized solution to the LQT that incorporates the wants of a ground operator, the topological structure of the formation, and the desired trajectory in the formulation of the locally implemented control of each agent in the formation and various methods of decentralizing the problem are investigated.

6.1 Construction of the LQT

Consider a discrete linear dynamic system.

$$x_{k+1}^i = A^i x_k^i + B^i u_k^i \tag{6.1}$$

In Eq. (6.1), x^i is the n-vector of the state variables for agent i, u^i is the p-vector of the inputs for agent i, A^i is the $(n \times n)$ state matrix of agent i, and B^i is the $(n \times p)$ input matrix of agent i, where $i = 1, \ldots, N$. The states of the vehicles are concatenated into an ordered vector, x_k, such that $x_k^i \in x_k$.

$$x_{k+1} = f^k(x_k, u_k) = Ax_k + Bu_k \tag{6.2}$$

The system state matrix, A, is a $(Nn \times Nn)$ block diagonal matrix consisting of the individual systems such that $A = \text{diag}([A_1,\ A_2, \ldots,\ A_N])$. Similarly, B is the $(Nn \times Np)$ block

diagonal input matrix consisting of the individual systems such that $B = \text{diag}([B_1, B_2, \ldots, B_N])$. To determine the inputs, u_k, at each iteration, such that $u_k^i \in u_k$, the formation needs to satisfy an objective function that reflects the wants of the ground operator, retains the formation structure, and tracks the exogenous input. To this end, a LQT is used to optimize the performance index subject to Eq. (6.2).

$$J = \frac{1}{2}\alpha_T^T L x_T + \frac{1}{2}(1-\alpha)(y_T - r_T)^T P (y_T - r_T) + \frac{1}{2}\sum_{k=i}^{T-1} \alpha x^T L x + (1-\alpha)(y_k - r_k)^T Q (y_k - r_k) + u_k^T R u_k \quad (6.3)$$

In Eq. (6.3), (P, Q) are semi-positive definite symmetric matrices, R is a positive definite matrix and L is the semi-positive definite symmetric graph Laplacian representation of the formation interaction. The first term in the summation of the performance index leverages the graph Laplacian in representing the interaction of the agents in the formation. Therefore, this first term determines the cost of formation keeping. The second term regulates the difference between the provided reference input r_k and the linear combination of the states y_k such that $y_k = Cx_k$ where C is the $(m \times Nn)$ measurement matrix for m measurements of the leader's state. The sliding parameter α, where $0 \leq \alpha \leq 1$, represents the commanded behavior from the ground operator. By increasing the value of α, the performance index stresses the importance of formation keeping while reducing the importance of trajectory tracking, as determined by the ground operator. Similarly, by reducing the value of α, the formation puts greater emphasis on trajectory tracking when determining the control. With this, control parameter α directly affects the behavior of the formation. The solution of the LQT is obtained by a standard procedure set forth in [43] and briefly described here for its application to the decentralized problem. The cost function at each iteration, k, and the terminal cost may be rewritten as $\mathcal{L}(x_k, u_k)$ and $\phi(T, x_T)$ in Eq. (6.4), respectively, where

x_T is the final state.

$$\begin{aligned} \mathcal{L}(x_k, u_k) &= \tfrac{1}{2}\left[\alpha x_k^T L x_k + (1-\alpha)(y_k - r_k)^T Q (y_k - r_k) + u_k^T R u_k\right] \\ \phi(T, x_T) &= \tfrac{1}{2}\left[\alpha x_T^T L x_T + (1-\alpha)(y_T - r_T)^T P (y_T - r_T)\right] \end{aligned} \tag{6.4}$$

With this, the performance index in Eq. (6.3) is rewritten in a reduced form.

$$J = \phi(T, x_T) + \sum_{k=i}^{T-1} \mathcal{L}(x_k, u_k) \tag{6.5}$$

In order to optimize Eq. (6.5), subject to Eq. (6.2), an augmented cost function is formulated using the multiplier function λ. The Hamiltonian is then defined as the adjoint of the cost function and the system constraint in Eq. (6.6).

$$\mathcal{H}^k = \mathcal{L}^k + \lambda_{k+1}^T f^k(x_k, u_k) \tag{6.6}$$

As presented in [44], the optimal control equations may now be defined.

State Equation:

$$\begin{aligned} x_{k+1} &= \tfrac{\partial \mathcal{H}^k}{\partial \lambda_{k+1}} \\ &= Ax_k + Bu_k \end{aligned} \tag{6.7}$$

Costate Equation:

$$\begin{aligned} \lambda_k &= \left(\tfrac{\partial \mathcal{H}^k}{\partial x_k}\right) = \left(\tfrac{\partial f^k}{\partial x_k}\right)^T \lambda_{k+1} + \tfrac{\partial \mathcal{L}^k}{\partial x_k} \\ &= A^T \lambda_{k+1} + \alpha L x_k + (1-\alpha) C^T Q (C x_k - r_k) \end{aligned} \tag{6.8}$$

Stationary Condition:

$$\begin{aligned} 0 &= \tfrac{\partial \mathcal{H}^k}{\partial u_k} = \left(\tfrac{\partial f^k}{\partial u_k}\right)^T \lambda_{k+1} + \tfrac{\partial f^k}{\partial u_k} \\ &= B^T \lambda_{k+1} + R u_k \end{aligned} \tag{6.9}$$

Boundary Condition:

$$\begin{aligned} \lambda_T &= \tfrac{\partial \phi}{\partial x_T} \\ &= \alpha L x_T + (1-\alpha)\, C^T P \left(C x_T - r_T\right) \end{aligned} \tag{6.10}$$

The control at k is found from the stationary condition in Eq. (6.9).

$$u_k = -R^{-1} B^T \lambda_{k+1} \tag{6.11}$$

Using Eq. (6.11), the state and costate equations can be expressed as coupled equations.

$$\begin{bmatrix} x_{k+1} \\ \lambda_k \end{bmatrix} = \begin{bmatrix} A & -BR^{-1}B^T \\ \alpha L + (1-\alpha)\, C^T Q C & A^T \end{bmatrix} \begin{bmatrix} x_k \\ \lambda_{k+1} \end{bmatrix} + \begin{bmatrix} 0 \\ -(1-\alpha)\, C^T Q r_k \end{bmatrix} \tag{6.12}$$

The boundary conditions for the system in Eq. (6.12) are split such that the costate equations needs to be expressed in an implementable form for online application of the LQT. Looking at the boundary condition in Eq. (6.10), an assumed form of the costate equation for all $k \leq T$ is given in Eq. (6.13).

$$\lambda_k = S_k x_k - v_k \tag{6.13}$$

The assumed form of the costate equation contains the $(n \times n)$ auxiliary function S_k and the $(n \times 1)$ auxiliary function v_k which will now be formulated using the state and costate equations previously defined. Substituting Eq. (6.13) into the state of Eq. (6.11), the state may be expressed in terms of x_k, S_{k+1} and v_{k+1}.

$$x_{k+1} = A_k - BR^{-1}B^T S_{k+1} x_{k+1} + BR^{-1}B^T v_{k+1} \tag{6.14}$$

$$x_{k+1} = \left(I + BR^{-1}B^T S_{k+1}\right)^{-1} \left(A x_k + BR^{-1}B^T v_{k+1}\right) \tag{6.15}$$

Equation (6.15) expresses the state in terms of the auxiliary functions. Applying Eq. (6.15) and (6.13) to Eq. (6.12).

$$S_k x_k - v_k = \alpha L x_k + (1-\alpha)\, C^T Q C x_k + A^T S_{k+1} \left(I + BR^{-1}B^T S_{k+1}\right)^{-1} \times \left(Ax_k + BR^{-1}B^T v_{k+1}\right) - A^T v_{k+1} - (1-\alpha)\, C^T Q r_k \quad (6.16)$$

$$\left[-S_k + A^T S_{k+1}\left(I + BR^{-1}B^T S_{k+1}\right)^{-1} A + (1-\alpha)\, C^T Q C\right] x_k + \left[v_k + A^T S_{k+1}\left(I + BR^{-1}B^T S_{k+1}\right)^{-1} BR^{-1}B^T v_{k+1} - A^T v_{k+1} - (1-\alpha)\, C^T Q r_k\right] = 0 \quad (6.17)$$

The condition in Eq. (6.16) must hold at each iteration, requiring the bracketed terms in Eq. (6.17) to equal zero at every iteration. Applying the matrix inversion lemma to Eq. (6.17), expressions for the costate equations are found.

$$S_k = A^T \left[S_{k+1} - S_{k+1}B\left(B^T S_{k+1} B + R\right)^{-1} B^T S_{k+1}\right] A + \alpha L + (1-\alpha)\, C^T Q C \quad (6.18)$$

$$v_k = \left[A^T - A^T S_{k+1} B\left(B^T S_{k+1} B + R\right)^{-1} B^T\right] v_{k+1} + (1-\alpha)\, C^T Q r_k \quad (6.19)$$

Comparing the boundary condition from Eq. (6.10) to the assumed form of the costate in Eq. (6.13), the boundary conditions of the auxiliary functions is found at iteration T.

$$\begin{aligned} S_T &= \alpha L + (1-\alpha)\, C^T Q C \\ v_T &= (1-\alpha)\, C^T Q r_k \end{aligned} \quad (6.20)$$

The control can now be determined using the assumed form of the costate equation.

$$\begin{aligned} u_k &= -R^{-1}B^T\lambda_{k+1} \\ &= -R^{-1}B^T S_{k+1}x_{k+1} + R^{-1}B^T v_{k+1} \\ &= -R^{-1}B^T S_{k+1}\left(Ax_k + Bu_k\right) + R^{-1}B^T v_{k+1} \\ &= \left(B^T S_{k+1}B + R\right)^{-1} B^T\left(-S_{k+1}Ax_k - v_{k+1}\right) \end{aligned} \tag{6.21}$$

The feedback gain operating on the state x_k may be redefined as K_k.

$$K_k = \left(B^T S_{k+1}B + R\right)^{-1} B^T S_{k+1}A \tag{6.22}$$

The feedforward gain, K_v^k, operates on the auxiliary function v_{k+1} such that the feedforward allows the controller to follow the reference trajectory r_k.

$$K_k^v = \left(B^T S_{k+1}B + R\right)^{-1} B^T \tag{6.23}$$

With this, the control u_k may be separated into its feedback and feedforward elements, given in Eq. (6.24).

$$u_k = -K_k x_k + K_k^v v_{k+1} \tag{6.24}$$

The solution to the centralized tracking problem is presented in Eq. (6.25).

$$\begin{aligned} K_k &= \left(B^T S_{k+1}B + R\right)^{-1} B^T S_{k+1}A \\ S_k &= A^T S_{k+1}\left(A - BK_k\right) + \left(1-\alpha\right)C^T QC + \alpha L, \quad S_T = \alpha L + \left(1-\alpha\right)C^T QC \\ v_k &= \left(A - BK_k\right)^T v_{k+1} + \left(1-\alpha\right)C^T Q r_k, \qquad v_T = \left(1-\alpha\right)C^T Q r_T \\ K_k^v &= \left(B^T S_{k+1}B + R\right)^{-1} B^T \\ u_k &= -K_k x_k + K_k^v v_{k+1} \end{aligned} \tag{6.25}$$

This two-point boundary problem requires a sweeping method to determine the control u_k which consist of both feedback, K_k, and feedforward, K_k^v, elements. The feedback element

is linearly dependent upon the state while the feedforward element depends on the output v_{k+1} of the adjoint to the closed-loop plant driven by the exogenous reference trajectory r_k [43]. This aspect of the LQT requires prior knowledge of the reference trajectory to determine the control online, requiring the followers to have knowledge of their intended reference trajectory. The centralized approach to this problem requires a global observer who determines the control for each vehicle in the formation. Although simplistic in theory, the transmission of information from a fusion center makes the centralized approach increasingly problematic as the formation grows. Also, the centralized approach develops a single point of failure in the system which is unfavorable in attritional environments.

Eliminating the need for a centralized observer and appointing a single leader presents an alternative to centralized cooperative control while creating several issues in observing both the reference trajectory and state of the formation. The remainder of this work addresses these complications in satisfying a global performance index using only local information exchange in the decentralized formation. Although more complicated than the centralized implementation of the LQT, the robust characteristics of decentralized control and low bandwidth benefits provide a feasible, online alternative to its centralized counterpart.

6.1.1 Gain Matrix Analysis

In Section 4.3, a hypothetical controller was presented, provided in Eq. (6.26) for convenience.

$$u_i(t) = -\frac{k_i^L}{|\mathcal{N}_L(i)|}\sum_{j\in\mathcal{N}_L(i)}\left(x_i(t) - x_j^L(t)\right) - \frac{k_i^F}{|\mathcal{N}_F(i)|}\sum_{\ell\in\mathcal{N}_F(i)}\left(x_i(t) - x_\ell^F(t)\right) \tag{6.26}$$

Used to formulate the hierarchical framework of the network, this controller varies the dependence each agent has on its leader and followers such that $k_i^L + k_i^F = 1$. Similar to Eq. (6.3), increasing k_i^L emphasizes the controller's dependence on the leader, analogous to decreasing α. Comparatively, increasing the dependence on the follower by increasing k_i^F is analogous

to increasing α. The controller in Eq. (6.26) may be partitioned into its contributing parts corresponding to both position and velocity, K^p and K^v, respectively. With this, the controller is reconstructed, defining x_i as the vector of position states and $\dot{x}_i$ as the vector of velocity states:

$$u_i(t) = -K^v \dot{x}_i - K^p x_i \tag{6.27}$$

The square gain matrices in Eq. (6.27) may be separated into their symmetric and skew-symmetric parts.

$$u_i(t) = -(K^v_{sym} + K^v_{skew})\dot{x}_i - (K^p_{sym} + K^p_{skew})x_i \tag{6.28}$$

Consider the three vehicle convoy operating under this control law. Applying Eq. (6.26) to this formation, the gain matrices, $K^{p,v}$, take the form given in Eq. (6.29).

$$K^{p,v} = \begin{bmatrix} k_1^F & 0 & -k_1^F & 0 & 0 & 0 \\ 0 & k_1^F & 0 & -k_1^F & 0 & 0 \\ -k_2^L & 0 & k_2^L + k_2^F & 0 & -k_2^F & 0 \\ 0 & -k_2^L & 0 & k_2^L + k_2^F & 0 & -k_2^F \\ 0 & 0 & -k_3^L & 0 & k_3^L & 0 \\ 0 & 0 & 0 & -k_3^L & 0 & k_3^L \end{bmatrix} \tag{6.29}$$

When $k_i^L = k_i^F$, the gain matrix becomes symmetric, non-negative definite, with $K^{p,v}_{skew} = 0$, such that the dependence of an agent on their leader and followers is balanced and asymptotic stability is possible [45, 46]. This corresponds to a virtual structure where the formation responds as a rigid body. As the balance shifts between leader and followers, $K^{p,v}$ becomes asymmetric with $K^{p,v}_{skew} \neq 0$.

In the case of the centralized controller, where the global observer determines the control for each vehicle based on the state of the formation, the gain matrix reproduces the behavior of the hypothetical controller in Eq. (6.26). The centralized global gain matrix, K, may be partitioned into its contributing parts corresponding to both position and velocity,

represented in Eq. (6.27) as K^p and K^v, respectively, for $\alpha = 1$.

$$K^p = \begin{bmatrix} 0.2671 & 0.0000 & -0.1570 & 0.0000 & -0.1101 & 0.0000 \\ 0.0000 & 0.2671 & 0.0000 & -0.1570 & 0.0000 & -0.1101 \\ -0.1570 & 0.0000 & 0.3141 & 0.0000 & -0.1570 & 0.0000 \\ 0.0000 & -0.1570 & 0.0000 & 0.3141 & 0.0000 & -0.1570 \\ -0.1101 & 0.0000 & -0.1570 & 0.0000 & 0.2671 & 0.0000 \\ 0.0000 & -0.1101 & 0.0000 & -0.1570 & 0.0000 & 0.2671 \end{bmatrix} \tag{6.30}$$

$$K^v = \begin{bmatrix} 0.6501 & 0.0000 & -0.3632 & 0.0000 & -0.2869 & 0.0000 \\ 0.0000 & 0.6501 & 0.0000 & -0.3632 & 0.0000 & -0.2869 \\ -0.3632 & 0.0000 & 0.7264 & 0.0000 & -0.3632 & 0.0000 \\ 0.0000 & -0.3632 & 0.0000 & 0.7264 & 0.0000 & -0.3632 \\ -0.2869 & 0.0000 & -0.3632 & 0.0000 & 0.6501 & 0.0000 \\ 0.0000 & -0.2869 & 0.0000 & -0.3632 & 0.0000 & 0.6501 \end{bmatrix} \tag{6.31}$$

This LQT configuration represents a virtual-structure, analogous to the case of $k_i^L = k_i^F$ such that both K^p and K^v are symmetric, non-negative definite. Reducing the value of α such that the controller considers the reference trajectory, the centralized $K^{p,v}$ matrix becomes asymmetric, with $K_{skew}^{p,v} \neq 0$. For example, applying a value of $\alpha = 0.75$, the symmetric and skew-symmetric portions of the position gain matrix are given in Eq. (6.32) and (6.33).

$$K_{sym}^p = \begin{bmatrix} 0.3758 & 0.0000 & -0.0899 & 0.0000 & -0.0521 & 0.0000 \\ 0.0000 & 0.3758 & 0.0000 & -0.0899 & 0.0000 & -0.0521 \\ -0.0899 & 0.0000 & 0.3238 & 0.0000 & -0.1302 & 0.0000 \\ 0.0000 & -0.0899 & 0.0000 & 0.3238 & 0.0000 & -0.1302 \\ -0.0521 & 0.0000 & -0.1302 & 0.0000 & 0.2651 & 0.0000 \\ 0.0000 & -0.0521 & 0.0000 & -0.1302 & 0.0000 & 0.2651 \end{bmatrix} \tag{6.32}$$

$$K^p_{skew} = \begin{bmatrix} 0.0000 & 0.0000 & -0.0247 & 0.0000 & -0.0205 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & -0.0247 & 0.0000 & -0.0205 \\ 0.0247 & 0.0000 & 0.0000 & 0.0000 & -0.0005 & 0.0000 \\ 0.0000 & 0.0247 & 0.0000 & 0.0000 & 0.0000 & -0.0005 \\ 0.0205 & 0.0000 & 0.0005 & 0.0000 & 0.0000 & 0.0000 \\ 0.0000 & 0.0205 & 0.0000 & 0.0005 & 0.0000 & 0.0000 \end{bmatrix} \tag{6.33}$$

The upper triangular portion of K^p_{skew} shows the attraction of the controller to the followers, as expected for $\alpha = 0.75$ where the controller emphasis the relationship between the followers over that of the leader. This relationship was predicted in Eq. (6.26) where the upper triangular portion of the gain matrix corresponded to the dependence of the controller on the follower. This analysis has shown that the centralized LQT controller reproduces the behavior expect from the hypothesized controller presented in Section 4.3. Furthermore, the controller in Eq. (6.26) was inspired by the neighbor-to-neighbor interaction within the formation, as modeled by the graph Laplacian. The centralized controller reproducing this behavior is encouraging such that the first-order assumption made in the leadership selection process has carried through to the formulation of the LQT controller.

6.1.2 The Mission Environment

The mission environment for the formation of unmanned vehicles is presented in Fig. 6.1. The vehicles are tasked to maneuver around the obstructions, indicated as GPS jamming regions, by way of a set of waypoints. The desired trajectory is generated onboard the lead vehicle as the ground operator supplies at least two future waypoints along the track, as described in Chapter 5. Each following agent updates their control every iteration and assumes the control is constant between iterations while the leader updates its desired trajectory upon update from the ground operator. The simulations in the following sections begin with the formation only sensing their local neighbors. When the formation reaches

consensus and identifies the formation leader, they develop their control and egress onto the desired trajectory under the direction of their formation leader.

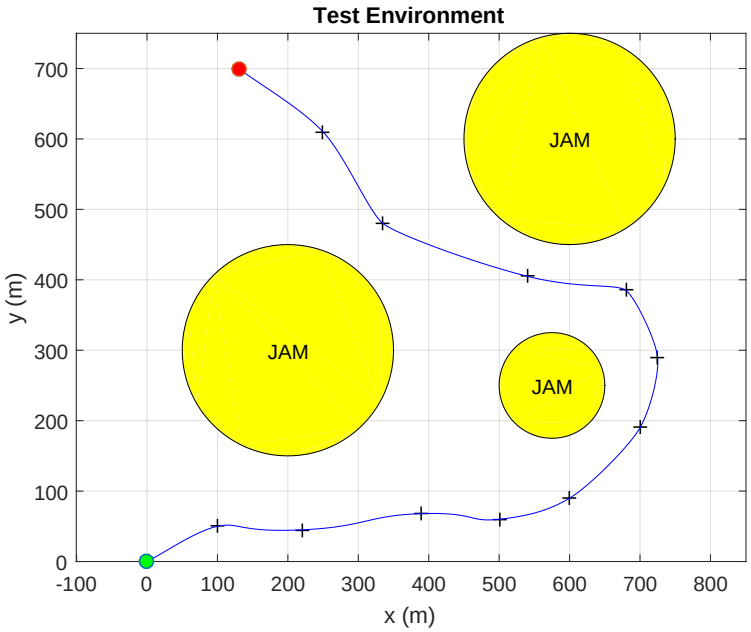


Figure 6.1: Mission environment containing the desired track and waypoints for the unmanned formation. The green and red points signify the beginning and ending of the track, respectively, and the obstructions to maneuver around are marked in yellow.

The vehicle model considered in this section is a double-integrator model. Referring to the building of the system model in Eq. (6.1), the state and input matrices have the form given in Eq. (6.34).

$$x_{k+1}^i = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} x_k^i + \begin{bmatrix} \frac{1}{2}\Delta t^2 & 0 \\ 0 & \frac{1}{2}\Delta t^2 \\ \Delta t & 0 \\ 0 & \Delta t \end{bmatrix} u_k^i \tag{6.34}$$

The time step, Δt, marks the time between each iteration, k. A rate of 1 Hz is used for each simulation to provided a conservative update rate. It is assumed that the exchange

information through the implementation of the DAA updates faster than the agents update their controller in order to ensure that the formation reaches consensus upon a global leader and remains stable to topological changes along the mission timeline.

6.2 Method 1 - Decentralized LQT Using a Reduced Order Gain Matrix

It has been shown that by designating individual sets of leaders and followers, the leaders may pursue the reference trajectory while the followers implement a consensus protocol to retain the formation structure [47, 48, 19]. Here, the leader's motion is unconstrained such that it is incumbent upon the followers to retain the formation structure and maintain their connection to the leader. In this application, the hierarchy of the decentralized formation is leveraged in applying a reduced order decentralized controller. The DLC developed this hierarchy in Section 4.3, providing each local agent with knowledge of their local leader and followers. By isolating each neighborhood of the global formation, the agents maintain their position in their neighborhood by simultaneously tracking their local leader and maintaining their local neighborhood structure. The contributions from outlying members of the formation are observed indirectly through the dynamics of their neighboring states. This approach alleviates the need to estimate the state of the entire formation, relying only on the local measurement of the neighborhood to determine an agent's control.

The formation leader receives the waypoint information from the ground operator, defining the desired trajectory for the decentralized formation to follow. Upon each update from the ground operator, the leader develops its control input based on the generated trajectory and the state of its local neighborhood. The followers of the formation must update their control at every iteration. The form of the LQT in Eq. (6.25) requires a reference trajectory to determine the control effort needed by each agent. Therefore, the local agents must project the state of their local leader to the next iteration, generate their own reference trajectory, and compute their control. The sliding parameter, α, varies the importance of trajectory tracking ($\alpha \to 0$) and formation keeping ($\alpha \to 1$). At one end of this boundary, where $\alpha = 0$,

the leader tracks the reference trajectory and the remaining agents are stationary. With this, the selection of the leader is inconsequential and the application of the DLC in selecting one node over another becomes arbitrary. At the other end of the boundary, where $\alpha = 1$, the priority of the controller is to retain the formation such that the entire formation will remain stationary, ignoring the commanded trajectory all together. The formation keeping problem, addressed in Section 4.2, showed the dependence upon the input location to the convergence rate of the formation. Therefore, the dependence of a formation's behavior upon the location of the leader increases as the value of α increases.

Consider the convoy of three vehicles tasked with tracking a desired trajectory provided by the ground operator in Fig. 6.2. Each agent determines the state of their system by directly observing the state of their neighborhood. With this, the first agent observes a single follower, the second agent observes a leader and a follower, and the third agent observes a leader. The third agent in the convoy does not have access to the leader's states or the intended trajectory so it must assume that any movement of their neighborhood is a consequence of the formation leader.

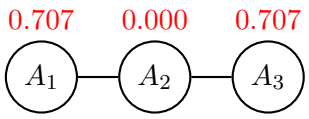


Figure 6.2: Three agent convoy with accompanying DLC measures given in red.

Agent 3 assumes a double-integrator model of its leader, Agent 2, and determines its own reference trajectory based on the projected state of Agent 2. This treatment of each neighborhood as an isolated system, with each agent determining its own control, attributes to the cost of satisfying the global performance index.

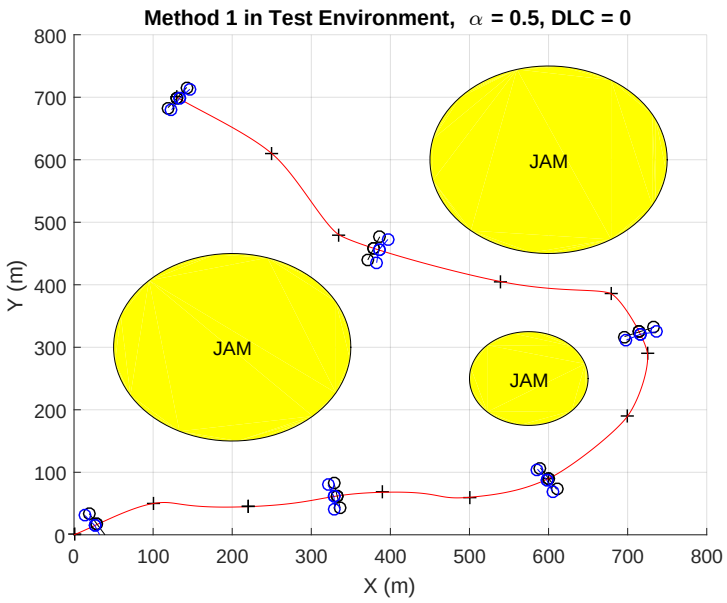


Figure 6.3: Application of Method 1, using Agent 2 as the formation leader from Fig. 6.2. The centralized solution, presented in black, precedes the decentralized solution, presented in blue.

Figure 6.3 shows the result at six different iterations along the desired path. The centralized solution, presented here in black, represents the case where a centralized observer determines the control of each vehicle. The formation in blue represents the implementation of the decentralized solution, where each agent determines their control by projecting the state of the local leader and applying the LQT. The difference in spacing between the centralized and decentralized solutions is attributed to the feedback of the formation on the control of the leader. With $\alpha = 0.50$, the control law provides a balance between the cost of formation keeping and trajectory tracking. Placing the input at Agent 2, with the DLC measure of 0, both agents have access to the leader's true state but their respective controllers are unaware of one another.

Moving the leadership position to a higher DLC location, Agent 1 is able to respond to the reference trajectory quickly but at the expense of losing the cohesiveness of the formation center, shown in Fig. 6.4.

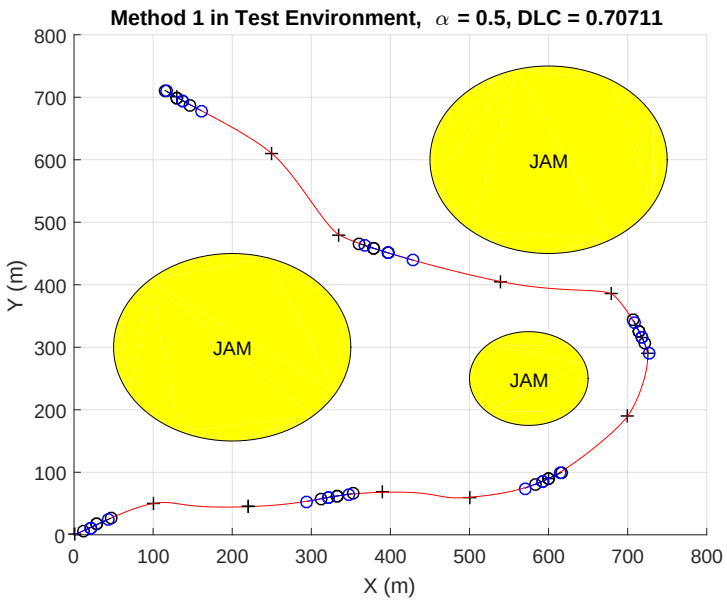


Figure 6.4: Application of Method 1, using Agent 1 as the formation leader from Fig. 6.2. The centralized solution, presented in black, precedes the decentralized solution, presented in blue.

Agent 3 determines its control from the state of Agent 2, exclusively, which attributes to the increases separation between the agents along the trajectory. This reduce order method provides a means of retaining the formation structure but as an agent's geodesic distance from the leader grows the performance of the controller deteriorates. In order to mitigate this performance loss, the update rate of the controller should be increases, providing the each agent the ability to project over shorter intervals and achieve more accurate estimates of the desired trajectory within their local neighborhoods.

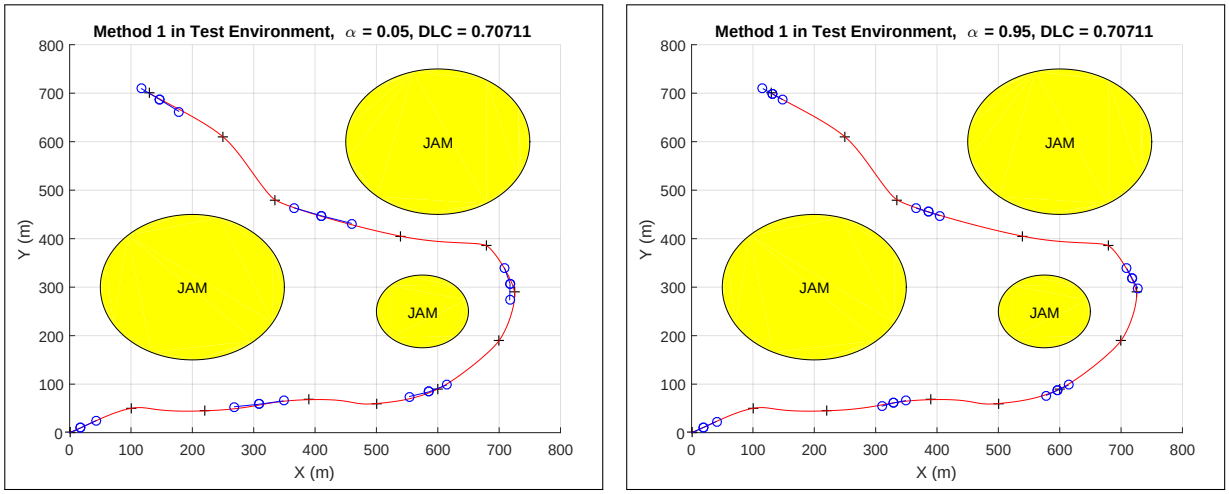


Figure 6.5: Designating Agent 1 as the leader and varying the value of the sliding parameter, α for Method 1. Image on the left reduces α to provided better trajectory following. The right image increases α to emphasize formation keeping.

The sliding parameter, α, allows the ground operator to vary the dependence of the controller on tracking the reference trajectory and retaining the formation. Figure 6.5 notes the effect of the sliding parameter on the formation behavior. For reduced values of α, the formation leader remains close to the reference trajectory. The implications of doing so result in the formation becoming more displaced, thereby losing the cohesiveness of the formation as the length of the formation grows. Conversely, by applying a larger value of α, the formation better retains its structure, but at the expense of lagging the reference trajectory. These two cases present different scenarios where the ground operator can request different behavior from the formation through adjusting α. Values of low α appeal to missions where time-on-target requirements outweigh the need for a strict formation structure. In limited sensing environments where the spacing between vehicles must be kept to a minimum, applying a higher value of α helps maintain the formation structure.

6.2.1 The Sliding Parameter

The weights of the two contributing elements of the cost function could be modified to describe a one-to-one comparison between formation keeping and trajectory tracking,

although this would not reflect the required control effort needed by the formation to retain the structure. The trajectory tracking cost accounts for a single relationship between the leader and the generated trajectory while the formation keeping accounts for $|\Delta(\mathcal{G})|/2$ relationships, where $\Delta(\mathcal{G})$ is the degree matrix of the formation graph $\mathcal{G}$. With this, asking the controller to provide equal cost in a one-to-one relationship between the leader/trajectory and formation contributions would be to reject the effort needed to retain an entire formation. Therefore, the sliding α term defines the relationship between their contribution to the overall cost of the controller. The following simulations show the effect of varying the α term for the overall performance of the controller. Comparisons between the sliding term and the DLC measure help identify the benefits of leadership selection.

Consider the example formation given in Fig. 6.6. The image on the left shows the node names for each node and their local connections. The image on the right indicates their respective DLC measures.

Figure 6.6: Identifications of seven nodes in a formation (left) with their corresponding DLC measurements (right).

Recall that low DLC measures indicate input locations that promote the retention of the formation structure, deject disturbances quickly, and whose response is dominated by the homogeneous dynamics of the formation. Conversely, larger DLC measures indicate agents more sensitive to the exogenous input, usually at the expense of formation keeping. With this, Agents 1 and 7 exemplify these bounds of the DLC measures, respectively.

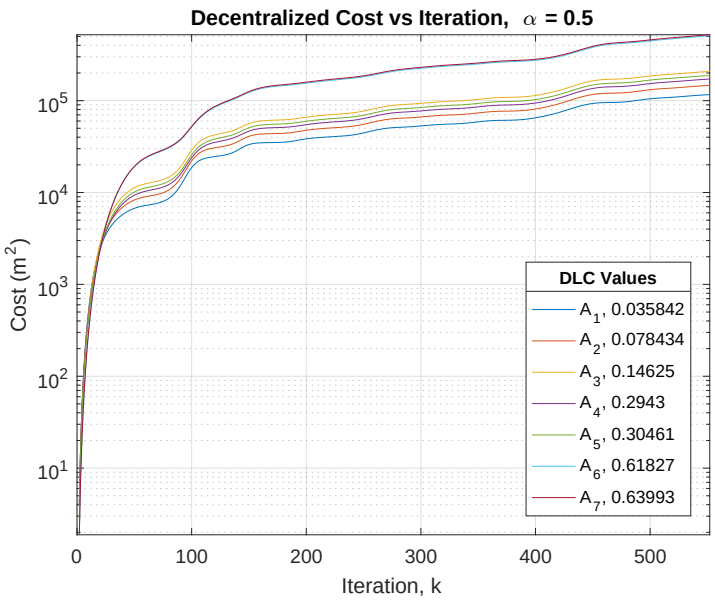


Figure 6.7: Cost function for the seven node formation by applying the reduced order LQT controller using $\alpha = 0.5$.

Figure 6.7 shows the accumulated cost, using $\alpha = 0.50$, over each iteration of the mission as the vehicles move through the mission environment using N reduced order controllers. Each line indicates the topological relocation of the leader in the formation. For example, the blue line indicating the lowest cost over each iteration, corresponds to appointing Agent 1 as the formation leader. In the centralized case where a global observer has access to the state of each agent in the formation, as well as the desired reference trajectory, the performance cost is orders of magnitude lower than the decentralized solution. Therefore, decentralizing the problem greatly impacts the performance of the controller, particularly in this where the information and access to the reference trajectory are limited.

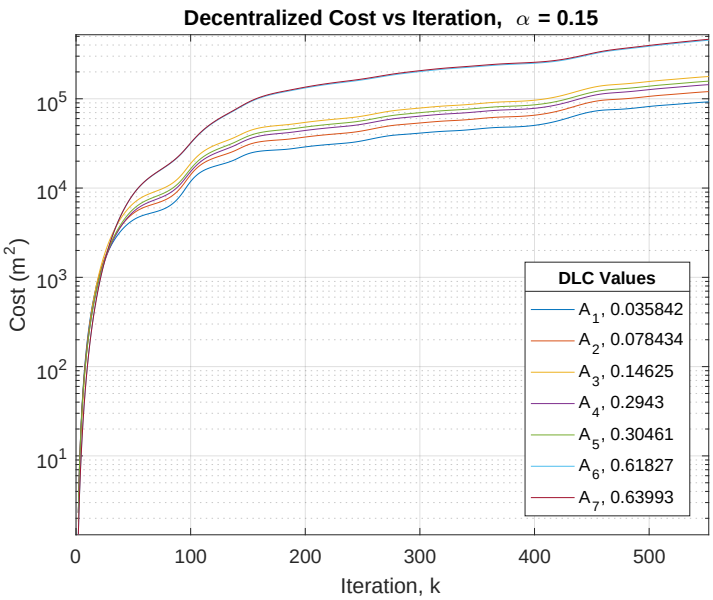


Figure 6.8: Cost functions for the seven node formation by applying the reduced order LQT controller using $\alpha = 0.15$.

As seen previously, lowering the value of α allows the leader to track the reference trajectory closer at the expense of the formation structure. Figure 6.8 indicates that the total cost increases as α decreases. Because the leader is more capable of separating from the formation to track the trajectory, the formation cost increases substantially, resulting in a large increase in the total cost on the system.

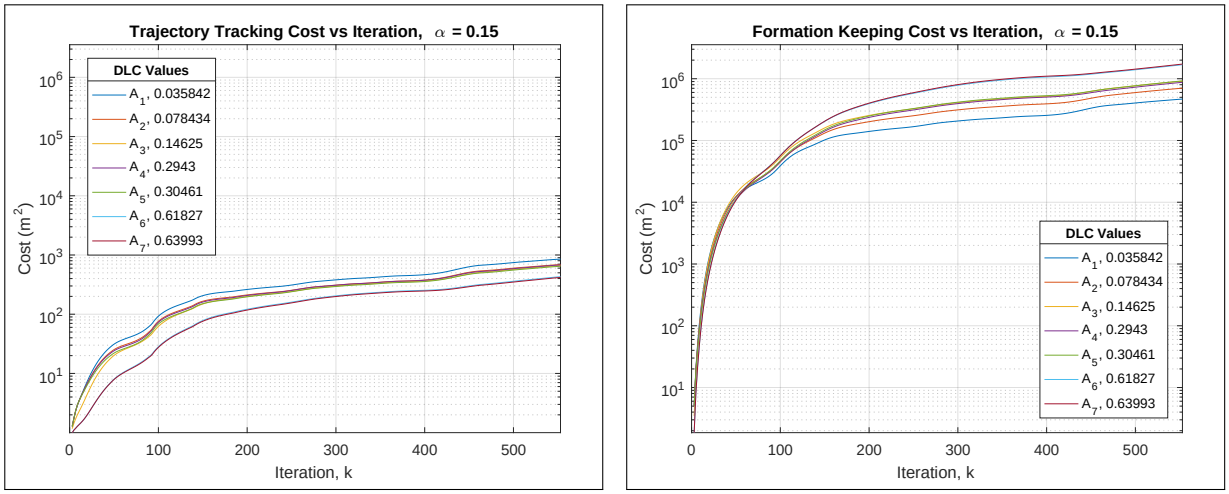


Figure 6.9: Trajectory tracking and formation keeping cost of the seven node formation in Fig. 6.6 for Method 1 using $\alpha = 0.15$ and varying the location of the leader.

Looking further into the cost function, the two contributing elements may be separated to identify the role of leadership selection in the design process of the controller. Figure 6.9 provides the trajectory tracking and formation cost accumulated over every iteration using a value of $\alpha = 0.15$ and varying the location of the leader for every mission. As expected, the trajectory tracking cost is reduced by selecting a leader with a high DLC value, indicated as Agent 7 in the left image. As highlighted in the description of the DLC that by using a low DLC measure, most notably Agent 1, the formation keeping cost is reduced throughout the mission. Therefore, the predominance of the formation keeping cost in Fig. 6.9 is attributed to the added effort needed to account for the numerous connections within the formation and the separation created by the formation leader pursuing the reference trajectory closer with less attention on the state of the formation.

Requesting a more stringent formation keeping behavior using $\alpha = 0.85$, Fig. 6.10 indicates how paying more attention to the formation structure reduces the total cost across all DLC measures during the mission timeline along with the accompanying separation of the cost function in Fig 6.11.

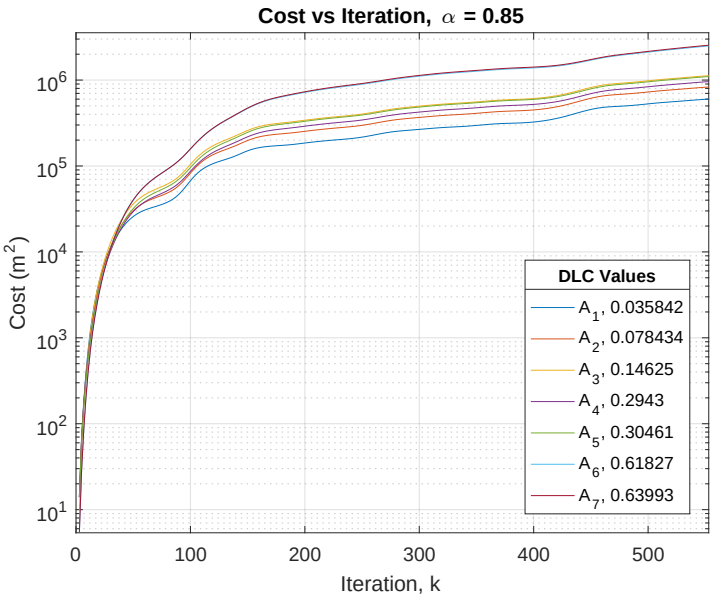


Figure 6.10: Total cost versus iteration for Method 1 using $\alpha = 0.85$ and varying the leader position.

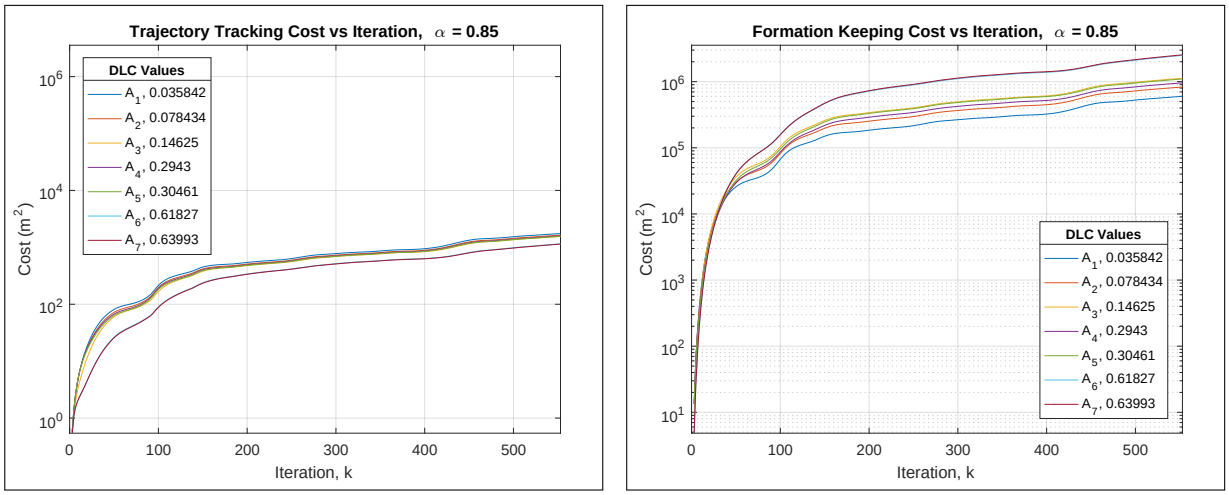


Figure 6.11: Trajectory tracking and formation keeping cost of the seven node formation in Fig. 6.6 for Method 1 using $\alpha = 0.85$ and varying the location of the leader.

Comparing the selection of the DLC for $\alpha = 0.15$, 0.5, and 0.85, the agent holding the lowest DLC value prevails as the most capable leader of the formation. It would appear that, through analysis of the cost function, the leadership selection has negligible affects on the performance of the formation. This is true from a total cost perspective yet the behavior

of the formation changes drastically when leadership changes through the network, as seen in Fig. 6.5.

This first method, utilizing a LQT for formation keeping and trajectory tracking in a decentralized formation of unmanned vehicles, proved to be effective in retaining the formation as it pursued the desired trajectory. This method allowed the control law to be applied in a decentralized formation where the control is locally implemented by each agent. The implications of leadership selection surfaced through the analysis of the individual contributions to the overall cost function. An important result from the above observations is that the expected behavior derived from applying the DLC, from a purely algebraic analysis of the formation's graph Laplacian, is preserved when applying the LQT. Therefore, a ground operator may request a leader by applying the DLC centrality and developing a sense for how the formation will respond. Chapter 4 developed a centrality measure for a formation operating under consensus dynamics, equivalent to applying a value of $\alpha = 1$ in the proposed LQT controller. The decentralized LQT controller proposed here revealed that the selection of any agent will result in trajectory tracking while the behavior of the formation along the mission timeline is reflected through the DLC implementation. The next section introduces a variant of the reduced order controller to incorporate the entire formation state using a Kalman filter.

6.3 Method 2 - Decentralized LQT Using a Global Gain Matrix

As an alternative to implementing reduced order controllers with limited observability of the network structure, the state of the formation may be estimated through the use of Kalman filter. In this approach, each agent leverages their acquired knowledge of the formation structure using the graph Laplacian and estimates the non-neighboring states of the formation. The graph Laplacian introduces the interactions of non-neighboring members of the formation and provides the LQT controller with a description of the entire formation, rather than a single neighborhood. This approach designs a global gain matrix with N

separate Kalman filters running onboard each vehicle, allowing the decentralized agents to determine their own perspective of the formation state.

$$\begin{aligned}
\hat{x}_k^- &= A\hat{x}_{k-1} + Bu_{k-1} \\
\bar{P}_k^- &= A\bar{P}_{k-1}A^T + \bar{Q} \\
\bar{K}_k &= \bar{P}_k^- H^T \left(H\bar{P}_k^- H^T + \bar{R}\right)^{-1} \\
\hat{x}_k &= \hat{x}_k^- + \bar{K}_k \left(z_k - H\hat{x}_k^-\right) \\
\bar{P}_k &= \left(I - \bar{K}_k H\right)\bar{P}_k^-
\end{aligned} \tag{6.35}$$

The only measurements available to the local agent are from their first-degree neighbors. For this problem, the agents are assumed to gather perfect information from their neighbors, leaving the remaining states of the formation to be estimated. Using a Kalman-filter based observer, the LQT receives the required full-state feedback for the system in order to determine the control. As the problem description requires, only the formation leader knows the desired trajectory as supplied by the ground operator. Therefore, the remaining agents of the formation must infer that if the state of their neighborhood changed, the state of the formation leader changed as well. This relationship is observed through the vehicles interactions modeled by the graph Laplacian, existing as the only reference local agents have of the global formation structure. At the problem inception, the leader determines its gain matrices, K_k and K_k^v, as it generates the desired trajectory from the provided waypoints. The followers, on the other hand, must project the estimated state of the formation leader until the next iteration, determine their own reference trajectory from the short projection, and implement the LQT controller over the short interval.

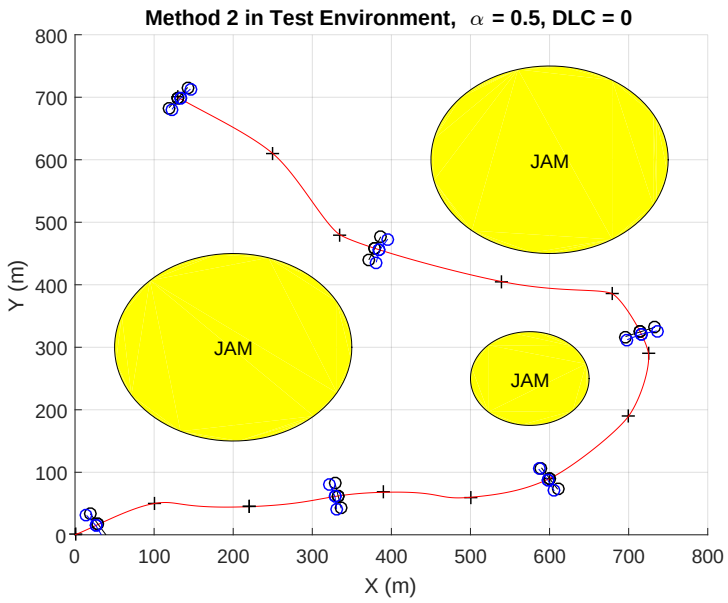


Figure 6.12: Application of Method 2, using Agent 2 as the formation leader from Fig. 6.2. The centralized solution, presented in black, precedes the decentralized solution, presented in blue.

This decentralized linear-quadratic-Gaussian-tracker (DLQGT) form of the LQT accounts for the missing state information at the local level. The individual agents determine their own perspective of the global gain matrix through the Kalman filter implemented onboard, such that the Gaussian portion of the control is decentralized and unique to the local agent. Applying this controller to the three agent convoy given in Fig. 6.2 and appointing the second node as leader, the formation effectively maintains the structure as it moves along the desired trajectory. This result coincides with that presented in Fig. 6.3, with $\alpha = 0.50$ and a DLC value of 0. In this configuration, both following agents observe the true state of the formation leader and project its future location using a double-integrator vehicle model. Opposed to the first method using a reduced order controller, the agents leverage the information stored in the graph Laplacian to observe the non-neighboring states of the formation.

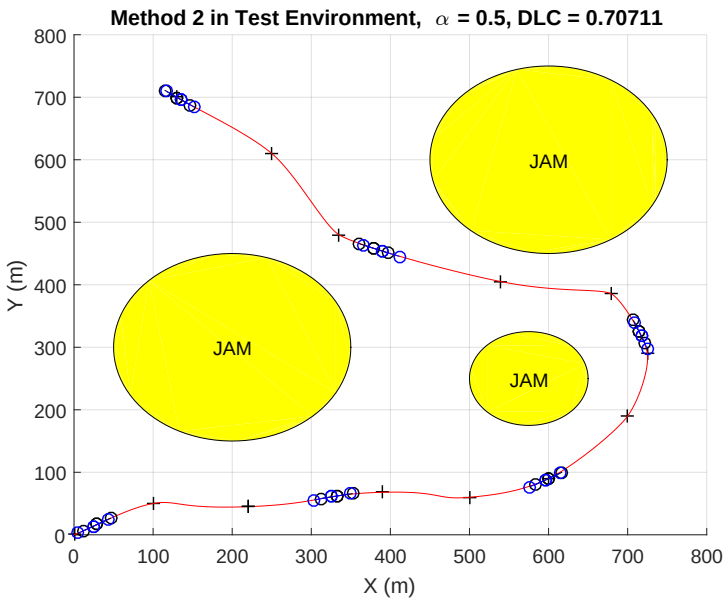


Figure 6.13: Application of Method 2, using Agent 1 as the formation leader from Fig. 6.2. The centralized solution, presented in black, precedes the decentralized solution, presented in blue.

Moving the input location to Agent 1, the ability of the DLQGT to retain the formation structure is apparent, when compared with Fig. 6.4. With Agent 3 estimating the state of Agent 1 (also the acting formation leader), the spacing within the formation is reduced, even when placing the input at a higher DLC value. In formations with less access to the formation leader, such as convoys or sparse topologies, this method retains the formation structure better through the observation of the entire network, rather than through the limited scope of the local neighborhood.

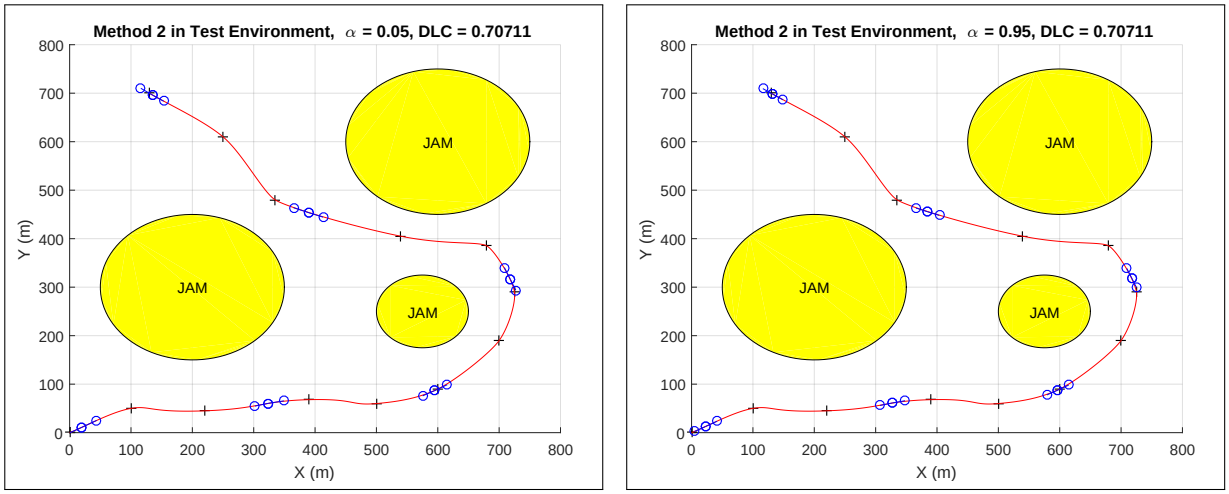


Figure 6.14: Designating Agent 1 as the leader and varying the value of the sliding parameter, α, for Method 2. Image on the left reduces α to provided better trajectory following. The right image increases α to emphasize formation keeping.

The sliding parameter, α, has a similar affect on the system as in the reduced order method. The separation of the three agent convoy is reduced through the use of the DLQGT. As the value of α reduces, the trajectory tracking error reduces, keeping the leader close to the desired path. With the inter-vehicle separation and the trajectory tracking error reduced over the range of α values, this DLQGT method outperforms the reduced order method, as expected.

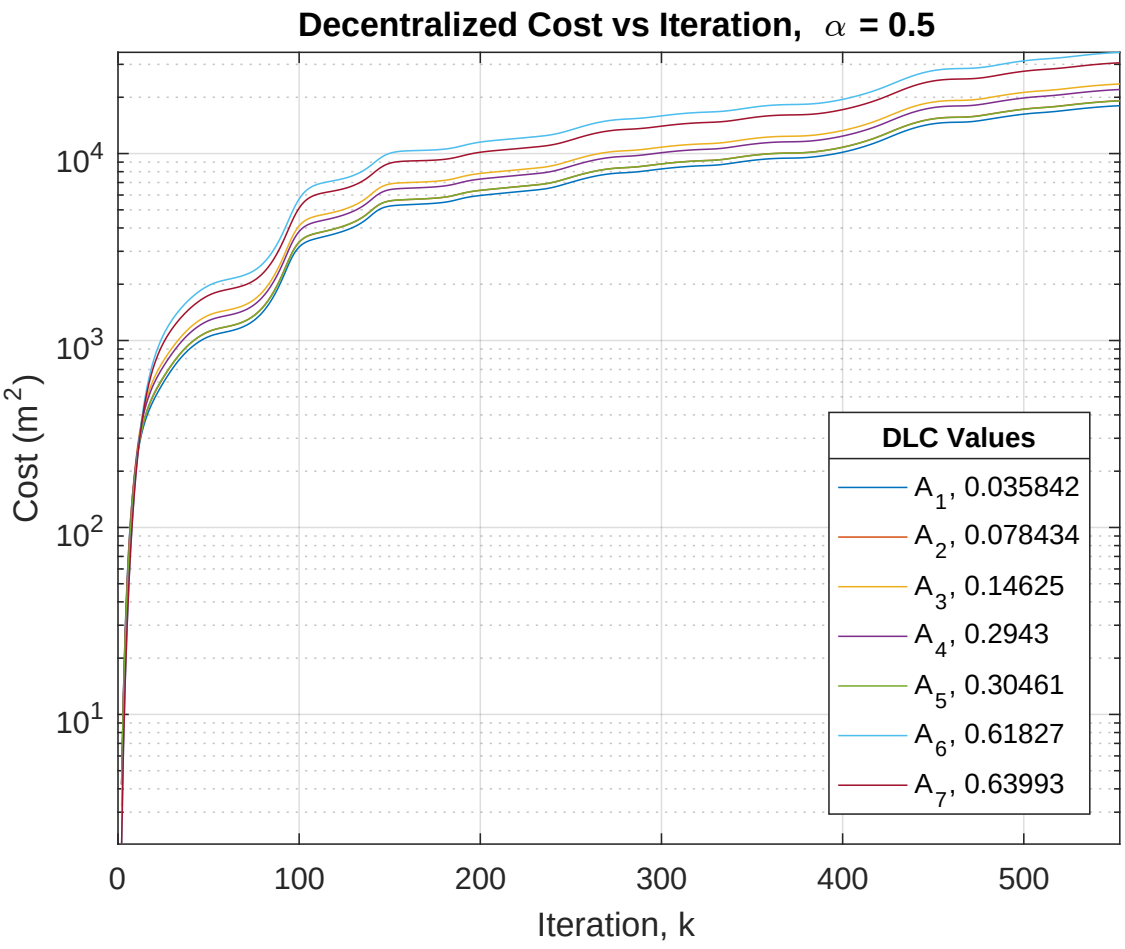


Figure 6.15: Cost functions for the seven node formation by applying the DLQGT using $\alpha = 0.5$.

Applying the DLQGT controller to the seven node formation given in Fig. 6.6, its performance is presented in Fig. 6.15 with $\alpha = 0.5$. Each line corresponds to a separate mission using a different agent as the formation leader. Here, the agents with lower DLC measurements incur less cost through each mission, coinciding with the results found in Section 6.2. The cost, as compared to the reduced order controller, is less for each leader of the formation.

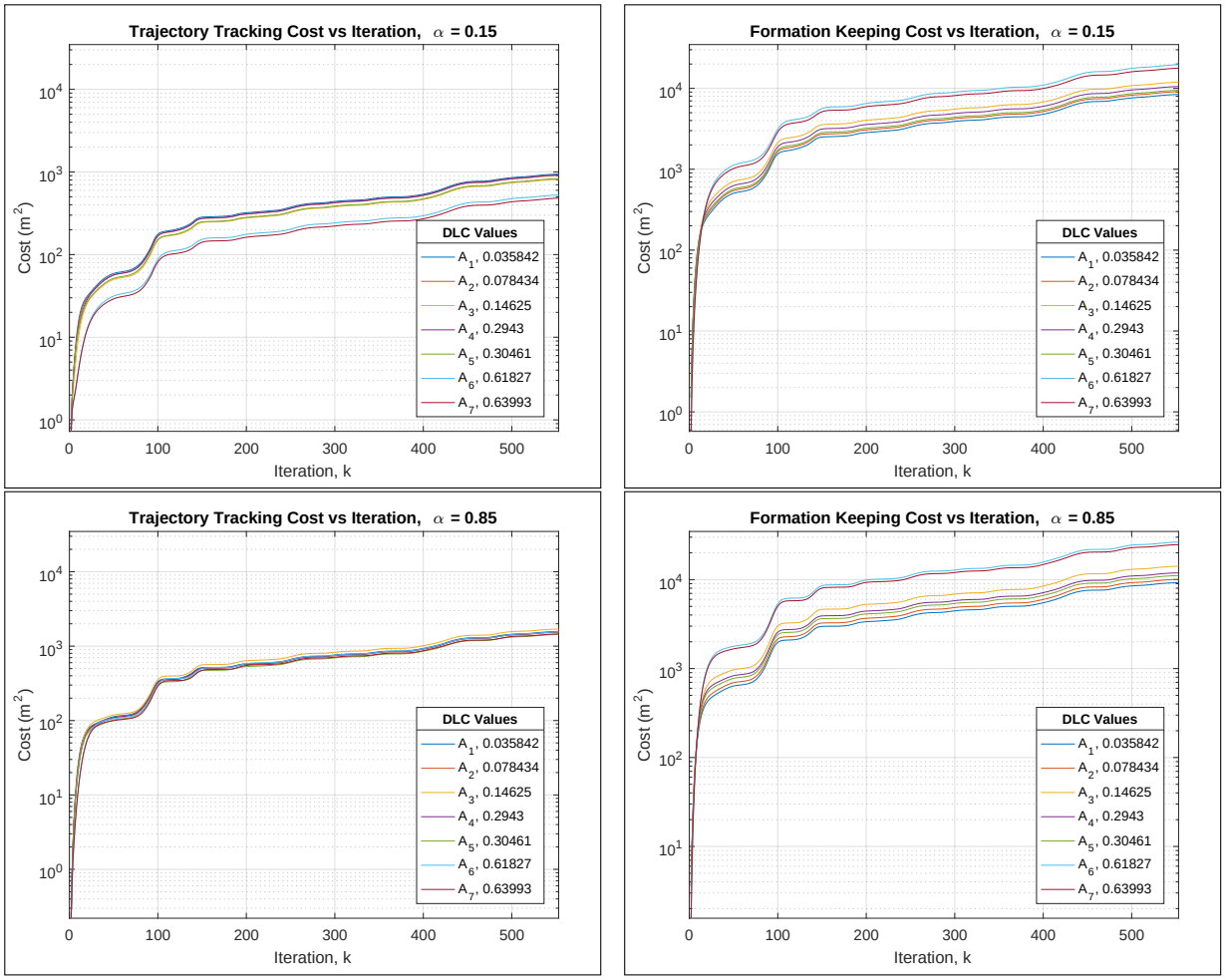


Figure 6.16: Trajectory tracking and formation keeping cost of the seven node formation in Fig. 6.6 for Method 2 using $\alpha = 0.15$ and 0.85, varying the location of the leader.

A comparison of the trajectory tracking and formation keeping elements of the cost function are shown in Fig. 6.16. The predicted behavior of the controller through the application of the DLC is confirmed by the reduced trajectory tracking cost for higher DLC measures, and reduced formation keeping cost for lower DLC measures. As stated in Section 6.1.2, as the value of α increases, the importance of leadership selection increases. The dominance of the cost by the formation keeping when $\alpha = 0.85$ denotes this importance of leadership selection.

6.4 Conclusions of the LQT

This chapter presented two separate approaches to designing a decentralized controller for a formation of unmanned vehicles. The first approach utilized a locally understood global hierarchy by applying N reduced order LQT controllers. The second approach required each agent to locally estimate the global formation and directly apply the LQT controller using their understanding of the formation state. Both applications perform well using various topologies of varying connectivity and size. The differences and limitations of the controllers arise in the onboard capabilities of the unmanned vehicles.

The first method ignores the contributions from agents outside of the local neighborhood of the control such that each agent moves according to their own locally determined control and assume that their neighbors move in the same perception of the formation state. This produces excessive error in the projection of the local leader's state. With this, the vehicles must be capable of updating their control at a higher rate in order to effectively estimate their locally determined reference trajectory. The second method, incorporates the extended states of the formation into the local controller such that the system remains stable under update rates. The application of the controllers must also consider the computation expense of the two designs. The reduced order LQT controller does not require a Kalman filter in its application, such that only the states of the neighboring agents are considered. This is computationally more efficient than each agent design a global gain matrix, as in the DLQGT method. The consequence of reducing the computational cost lies in the performance of the controller.

For a convoy of vehicles, representing a sparse formation topology, the addition of more vehicles to the formation requires agents topologically further away from the leader to have a more global perspective of the formation in order to maintain its position in the formation. With this, as the formation topology becomes more sparse, the performance of the reduced order LQT deteriorates and the DLQGT method should be implemented. In the case of the well connected formation where a majority of the agents have access to the formation

leader's states, the computationally less expensive reduced order approach performs well under conditions where vehicles have limited capabilities.

Chapter 7

Conclusions

This work explored the development of a decentralized cooperative controller able to incorporate input from a ground operator and utilize the dynamic characteristics of the multi-agent formation of unmanned vehicles. Limiting the formation to a single lead agent, the task of identifying an agent in a predefined communication topology capable of accommodating the input command and leveraging the formation structure was a significant challenge in this work. Previous approaches to this multi-agent problem involved designating individual sets of leaders and follower. The leaders would move unconstrained and manipulate the followers, who were implementing a consensus protocol to retain the formation structure. Other methods optimized the structure of the communication topology, tailoring the network to a particular application. This work provided a complement to these approaches, allowing any agent to be the formation leader and tailoring the controller to the predetermine network topology. In achieving this, various aspects of the decentralized controller had to be address individually.

The agents begin the mission with only local knowledge of the formation structure such that a method of learning the network in a decentralized fashion was needed. This led to the development of the DAA, through which the formation's topology was systematically constructed using a recursive algorithm. Next, a leader was defined as an agent capable of providing a formation response that fit the wants of the ground operator. By addressing the algebraic properties of the formation's graph Laplacian, and through the implementation of the DAA, a decentralized method of determining leadership was developed. With the leadership established, the formation was tasked with following a desired trajectory, only known to the formation leader. Using a waypoint following approach, the desired trajectory

was generated using a variation of a Bézier curve application to better complement the dynamics of the vehicles intended. Finally, the predetermined formation structure, provided with a single formation leader, satisfied a global objective function by locally implementing a LQT controller. This controller incorporated the wants of the user and utilized the location of the leader to effectively move on the desired trajectory. This controller applied feedback of a Kalman filter estimate of the state of the formation, implemented using only local measurements to satisfy a global objective.

The DAA provided an avenue for learning the network in a decentralized manner. The constraints of the problem restrict the communication to neighbor-neighbor interactions such that information exchange at every update had to continuously reinforce the topological state of the formation. This method incorporated the concept of consensus such that the network would converge onto a common understanding of the formation as the individual neighborhoods reached consensus. Through its application, the decentralized formation was able to recognize changes in the network structure and account for the status of the formation while operating in attritional environments. This approach to learning the network made the formation more robust in challenging environments, providing a level of flexibility to the controller design. Also, it built an avenue for the local agents to observe the global formation structure later needed for the decentralized controller to satisfy a global objective.

With each local agent obtaining a global perspective of the formation, the task of determining leadership was accomplished through the implementation of the DLC. This centrality measure utilized the neighbor-to-neighbor interactions in the formation to describe the benefits of different input locations. A combination of the DLC method for identifying leaders and the DAA method of relaying a topological description of the formation, the decentralized leadership selection was shown to converge on to a single agent as recognized by all members of the formation. A secondary application of the DLC allowed the formation to determine its own hierarchy after selecting a leader. Using the centrality measures from the DLC as

indications of the chain of command, Dijkstra's method of determine the least weighted developed a hierarchy within the decentralized system. This approach provided a structured solution to hierarchy development, able to be implemented locally due to each agent running the DAA method and using the DLC to identify the global formation leader locally. The algorithms introduced run efficiently onboard every vehicle and are computationally cost effective such that the decentralized leadership selection process can be executed online during the mission.

The leader of the formation received a set of waypoints from the ground operator and generated a desired trajectory onboard and in real time. Through the use of Bézier curves, each waypoint was incorporated into the trajectory of the formation, providing a novel approach to waypoint following. The ground operator provided a radius around each waypoint which controlled how the formation approached the waypoint. The computational efficiency of lower order Bézier curves and the simple geometric application to the problem permitted the trajectory generation to happen onboard the lead vehicle. Also, the continuity of Bézier segments lends itself to this problem, giving the operator freedom to update the path and maneuver the formation through challenging environments.

The steps leading up to the development of the decentralized cooperative control law provided the conditions for the formation to operate in a decentralized environment. The imposed assumptions aided in fully describing the design environment in order to illustrate a logical implementation of these methods. Utilizing the information gathered by each decentralized agent inspired the form of the decentralized controller. Leadership selection, hierarchy development, and trajectory generation were all required by the formation to operator autonomously through the mission environment. Armed with these tools, the decentralized controller was built around their framework. The first approach to the LQT design leveraged the instantiated hierarchy within the formation. The wants of the ground operator were realized through the sliding parameter, which complemented the decentralized leadership selection. Secondly, the DLQGT presented an alternative to the reduced

order controller providing a global perspective of the formation through the graph Laplacian framework. Incorporating a Kalman filter onboard each vehicle to process the local measurements and estimate the state of the formation, the LQT controller was locally applied by each agent. The decision to implement one controller over another remains a function of the vehicle capabilities. The decentralized reduced order controller requires a higher update rate in order accurately project the states of the local neighborhood leaders. As this rate decrease, the controller suffers increases errors in projecting the state. The DLQGT controller provided a complement to this design, allowing for lower update rates at a higher computational cost. In the event that a formation topology is unknown, the DLQGT allows for lower update rates and greater stability, tolerating more complicated topologies and missions though attritional environments.

Bibliography

[1] Speyer, J. L., "Computation and Transmission Requirements for a Decentralized Linear-Quadratic-Gaussian Control Problem," *IEEE*, Vol. 24, No. 2, April 1979.

[2] Geromel, J. and Bernussou, J., "Optimal Decentralized Control of Dynamic Systems," *Automatica*, Vol. 18, No. 5, 1982, pp. 545 – 557.

[3] Athans, M., "The Role and Use of the Stochastic Linear-Quadratic-Gaussian Problem in Control System Design," *IEEE*, Vol. 16, No. 6, December 1971.

[4] Ren, W. and Beard, R. W., *Distributed consensus in multi-vehicle cooperative control : theory and applications*, Communications and control engineering, Springer, London, 2008.

[5] Hernndez-Martnez, E. G. and Aranda-Bricaire, E., "Marching control of unicycles based on the leader-followers scheme," *Industrial Electronics, 2009. IECON '09. 35th Annual Conference of IEEE*, Nov 2009, pp. 2265–2270.

[6] Olfati-Saber, R., Fax, J. A., and Murray, R. M., "Consensus and Cooperation in Networked Multi-Agent Systems," *Proceedings of the IEEE*, Vol. 95, No. 1, Jan 2007, pp. 215–233.

[7] Penaloza-Mendoza, G. R., Hernandez-Mendoza, D. E., and Aranda-Bricaire, E., "Time-varying formation control for multi-agent systems applied to n-trailer configuration," *Electrical Engineering Computing Science and Automatic Control (CCE), 2011 8th International Conference on*, Oct 2011, pp. 1–6.

[8] Gonzlez-Sierra, J. and Aranda-Bricaire, E., "Design of a virtual mechanism for trajectory tracking of convoys of mobile robots," *Electrical Engineering, Computing Science and Automatic Control (CCE), 2013 10th International Conference on*, Sept 2013, pp. 364–368.

[9] Li, Z., Ren, W., Liu, X., and Xie, L., "Distributed containment control of linear multi-agent systems with multiple leaders and reduced-order controllers," *2011 50th IEEE Conference on Decision and Control and European Control Conference*, Dec 2011, pp. 771–776.

[10] Sorensen, N. and Ren, W., "A Unified Formation Control Scheme with a Single or Multiple Leaders," *American Control Conference, 2007. ACC '07*, July 2007, pp. 5412–5418.

[11] Jadbabaie, A., Lin, J., and Morse, A. S., "Coordination of groups of mobile autonomous agents using nearest neighbor rules," *IEEE Transactions on Automatic Control*, Vol. 48, No. 6, June 2003, pp. 988–1001.

[12] Corfmat, J. and Morse, A., "Decentralized control of linear multivariable systems," *Automatica*, Vol. 12, No. 5, 1976, pp. 479 – 495.

[13] Mesbahi, M. and Hadaegh, F. Y., "Graphs, Matrix Inequalities, and Switching for the Formation Flying Control of Multiple Spacecraft," *Proceedings of the American Control Conference*, 1999, pp. 4148–4152.

[14] Beard, R., Lawton, J., and Hadaegh, F., "A coordination architecture for spacecraft formation control," *Control Systems Technology, IEEE Transactions on*, Vol. 9, No. 6, Nov 2001, pp. 777–790.

[15] Mesbahi, M., "On State-dependent dynamic graphs and their controllability properties," *Automatic Control, IEEE Transactions on*, Vol. 50, No. 3, March 2005, pp. 387–392.

[16] Saber, R. and Murray, R., "Consensus protocols for networks of dynamic agents," *American Control Conference, 2003. Proceedings of the 2003*, Vol. 2, June 2003, pp. 951–956.

[17] Fiedler, M., "Algebraic connectivity of graphs," *Czechoslovak Mathematical Journal*, Vol. 23, No. 2, 1973, pp. 298–305.

[18] Luxburg, U., "A Tutorial on Spectral Clustering," *Statistics and Computing*, Vol. 17, No. 4, Dec. 2007, pp. 395–416.

[19] Fax, J. and Murray, R. M., "Information Flow and Coopeative Control of Vehicle Formations," *IEEE*, Vol. 49, No. 9, September 2004, pp. 1465–1476.

[20] Swaroop, D. and Hedrick, J., "String Stability of Interconnected Systems," *IEEE Transactions on Automatic Control*, Vol. 41, No. 3, March 1996, pp. 349–357.

[21] Zhong-Hai, Z., Jian, Y., Wen-Xia, Z., and Jin-Ping, Z., "Virtual-leader-follower structure and finite-time controller based cooperative control of multiple autonomous underwater vehicles," *Control and Decision Conference (CCDC), 2012 24th Chinese*, May 2012, pp. 3670–3675.

[22] Borgatti, S. and Everett, M., "A Graph-theoretic perspective on centrality," *Social Networks*, 2006.

[23] Mesbahi, M. and Egerstedt, M., *Graph Theoretic Methods in Multiagent Networks*, Princeton University Press, 41 William Street, Princeton, New Jersey, 2010.

[24] Bondy, J.-A. and Murty, U. S. R., *Graph theory*, Graduate texts in mathematics, Springer, New York, London, 2007.

[25] Diestel, R., *Graph Theory*, Springer, 4th ed., 2010.

[26] Chung, F. R., *Spectral Graph Theory*, American Mathematical Society, 2009.

[27] Borgatti, S. P., "Centrality and Network Flow," *Social Networks*, Vol. 27, 2005, pp. 55–71.

[28] Rusinowska, A., Berghammer, R., de Swart, H., and Grabisch, M., "Social networks: Prestige, centrality, and influence (invited paper)," June 2011.

[29] Freeman, L., "Centrality in Social Networks Conceptual Clarification," *Social Networks*, 1978.

[30] Jackson, M. O., *Social and Economic Networks*, Princeton University Press, 6 Oxford Street, Woodstock, Oxfordshire, 2008.

[31] Meyer, C., "Perron-Frobenius Theory of Nonnegative Matrices," *Matrix Analysis and Applied Linear Algebra*, 2000.

[32] Straffin, P. J., "Algebra in Geography: Eigenvectors of Networks," *Mathematics Magazine*, Vol. 53, No. 5, November 1980, pp. 269–276.

[33] Bonacich, P., "Power and Centrality: A Family of Measures," *American Journal of Sociology*, Vol. 92, No. 5, March 1987, pp. 1170–1182.

[34] Pauls, D. and D, R., "A measure of centrality based on the spectrum of the Laplacian," *Physical Review E*, Vol. 85, No. 6, June 2012, pp. 66127.

[35] Qi, X. and Fuller, E., "Laplacian centrality: A new centrality measure for weighted networks," *Information Sciences*, Vol. 194, 2012, pp. 240–253.

[36] Rahmani, A., Ji, M., Mesbahi, M., and Egerstedt, M., "Controllability of Multi-Agent Systems from a Graph-Theoretic Perspective." *SIAM J. Control and Optimization*, Vol. 48, No. 1, 2009, pp. 162–186.

[37] Robertson, C., Sinclair, A., and Doucette, E., "A New Centrality Measure Based on Formation Response," *AIAA Guidance, Navigation, and Control Conference*, American Institute of Aeronautics and Astronautics, 2016.

[38] Dijkstra, E. W., "A Note on Two Problems in Connexion with Graphs," *NUMERISCHE MATHEMATIK*, Vol. 1, No. 1, 1959, pp. 269–271.

[39] w. Choi, J., Curry, R., and Elkaim, G., "Path Planning Based on Bezier Curve for Autonomous Ground Vehicles," *World Congress on Engineering and Computer Science 2008, WCECS '08. Advances in Electrical and Electronics Engineering - IAENG Special Edition of the*, Oct 2008, pp. 158–166.

[40] Jeon, S. J., Kang, C. M., Lee, S. H., and Chung, C. C., "GPS waypoint fitting and tracking using model predictive control," *2015 IEEE Intelligent Vehicles Symposium (IV)*, June 2015, pp. 298–303.

[41] *The 2005 DARPA Grand Challenge*, Springer, 2007.

[42] Skrjanc, I. and Klancar, G., "Optimal cooperative collision avoidance between multiple robots based on Bernstein-Bezier curves." *Robotics and Autonomous Systems*, Vol. 58, No. 1, 2010, pp. 1–9.

[43] Lewis, F., Vrabie, D. L., and Syrmos, V. L., *Optimal Control: Third Edition*, John Wiley and Sons, 1 2012.

[44] Lewis, F. L. and Syrmos, V. L., *Optimal control*, J. Wiley, New York, 1995, A Wiley-Interscience publication.

[45] Hughes, P. C., *Spacecraft Attitude Dynamics*, Dover Publications, 2004.

[46] Huseyin, K., *Vibrations and Stability of Mulitple Parameter Systems*, Noordhoff International Publishing, 1978.

[47] Ji, M., Muhammad, A., and Egerstedt, M., "Leader-based multi-agent coordination: controllability and optimal control," *2006 American Control Conference*, June 2006, pp. 6 pp.–.

[48] Lawton, J. R. T., Beard, R. W., and Young, B. J., "A decentralized approach to formation maneuvers," *IEEE Transactions on Robotics and Automation*, Vol. 19, No. 6, Dec 2003, pp. 933–941.