Study of Consensus Algorithm and Decentralized Formation Control of Autonomous Vehicles
Applied to the Context of Unmanned Disaster Response System

A graduate project submitted in partial fulfillment of the requirements
For the degree of Master of Science in
Electrical Engineering

By
Alagendran Periapoilan

December 2013
The graduate project of Alagendran Periapelan is approved:

Dr. Ali Amini

Dr. Xiyi Hang

Dr. Xiaojun Geng, Chair

CALIFORNIA STATE UNIVERSITY, NORTHRIDGE
ACKNOWLEDGEMENT

I would like to express my sincere gratitude to my project advisor, Dr. Xiaojun Geng, for her invaluable support, encouragement and guidance during the course of my project.

I would also like to thank Dr. Ali Amini and Dr. Xiji Hang for serving on my project committee, providing valuable support and encouraging me throughout my project.

Last but not least, I would like to thank all my immediate family members and friends without their support this project would not have been possible.
# TABLE OF CONTENTS

SIGNATURE PAGE ....................................................................................................................... ii
ACKNOWLEDGEMENT .................................................................................................................. iii
LIST OF FIGURES ......................................................................................................................... v
ABSTRACT .................................................................................................................................... vi

CHAPTER 1: INTRODUCTION ..................................................................................................... 1
  Definition ................................................................................................................................. 1
  Challenges ................................................................................................................................. 2
  Advantages of Cooperative over Single Autonomous Vehicles ........................................... 2
  Applications ............................................................................................................................. 4
  Centralized Approach ............................................................................................................ 4
  Distributed Approach ............................................................................................................ 5
  Real-world Scenario ............................................................................................................... 6

CHAPTER 2: GRAPH THEORY ................................................................................................... 8
  Graph Definitions .................................................................................................................... 8
  Relationship between Matrices and Graphs ......................................................................... 13

CHAPTER 3: CONSENSUS ALGORITHM .................................................................................... 16
  Overview ................................................................................................................................. 16
  Consensus Algorithms .......................................................................................................... 16
    Information Consensus ........................................................................................................ 16
    Simulation Results of Consensus for three vehicles .......................................................... 20

CHAPTER 4: DECENTRALIZED FORMATION CONTROL ............................................................ 22
  Formation Control .................................................................................................................. 22
  Model for formation control ................................................................................................. 23
  Simulation Results of Formation control ............................................................................ 27

CHAPTER 5: CONCLUSIONS AND FUTURE WORK ................................................................. 30

REFERENCES ............................................................................................................................. 32

APPENDIX: SOURCE CODE ..................................................................................................... 33
LIST OF FIGURES

Fig 1.1. Division and Distribution of Tasks among individual agents. --------------4

Fig 2.1. Example for Graph. ------------------------------------------------------9

Fig 2.2. Example for undirected Graph.-----------------------------------------------10

Fig 2.3. Example for Sub graph. ---------------------------------------------------11

Fig 2.4. Regular Graph. -------------------------------------------------------------11

Fig 2.5. Connected Graph. -----------------------------------------------------------12

Fig 2.7. Sub graph Isomorphism. -----------------------------------------------------13

Fig 2.8. Degree of Digraphs. ----------------------------------------------------------13

Fig 2.9. Strongly Connected Digraph. -------------------------------------------------14

Fig 2.10. Adjacency matrix Representation. --------------------------------------------15

Fig 3.1. Communication topologies for three graphs. -----------------------------------19

Fig 3.2. Simulation results of Consensus Algorithms. -----------------------------------22

Fig. 4.1. Physical representation of vehicles in formation. ---------------------------25

Fig 4.2. Circle Formation of Vehicles from Initial to Final position. -----------------29

Fig 4.3. Trajectory of Circular Formation of Vehicles. -------------------------------29

Fig 4.4. Hexagon Formation of Vehicles from Initial to Final position. -------------30

Fig 4.5. Trajectory of Hexagonal Formation of Vehicles. -----------------------------30
ABSTRACT

Study of Consensus Algorithm and Decentralized Formation Control of Autonomous Vehicles
Applied to the Context of Unmanned Disaster Response System

By

Alagendran Periapoilan

Master of Science in Electrical Engineering

In this graduate level project, the concept of cooperative control between multiple autonomous vehicles is explored in depth. This report presents a real world scenario that will potentially require the cooperative control to execute a critical task. This report also describes a simulation of the cooperative control using consensus algorithm and decentralized formation control algorithm using MATLAB. The formation algorithm simulates a group of autonomous vehicles exchanging information between them with the intention of achieving a specific formation. The formation is achieved through decentralized feedback control. The results of the simulation are presented in this report.
CHAPTER 1: INTRODUCTION

Cooperative Control system is defined as an engineering system that can be characterized as a collection of interconnected decision-making components (systems) with limited processing capabilities, locally sensed information, and limited inter-component communications, all seeking to achieve a collective (global) objective. The cooperative control is also often referred as ‘multi-agent control’, ‘distributed control’, ‘networked control’, ‘swarms’ or ‘coordinated control’.

The control of autonomous vehicles is a recent topic of much interest and has a lot of scope. A cooperative team of multiple-vehicles is used in both civil and military applications. The coordinated vehicles have a higher success rate and better operational capability as compared to the autonomous vehicle that perform the whole task individually. Some of the potential real world applications for multi-vehicle systems are rescue systems, space-based interferometers, combat systems, surveillance and reconnaissance systems; hazardous material handling, and distributed reconfigurable sensor networks.

**Definition**

The word coordination or cooperation implies that there is an interaction between some or all of the agents and they share information to achieve the end goal. There are several aspects to the coordination or cooperation and some of them are listed below:

- a joint collaborative behavior that is directed toward some goal in which there is a common interest or reward.
- a form of interaction, usually based on communication.
• the joining together for doing something that creates a progressive result such as increasing performance or saving time.

The first aspect leads to the study of task decomposition, task allocation, and other distributed artificial intelligence issues. The second aspect underlines the requirements for communication or other common resources. Finally, the third is related to the performance measurements of cooperation, such as time to complete a task. An effective cooperative control translates into capabilities such as formation control, rendezvous, traffic control, cooperative task execution etc.

**Challenges**

Some of the theoretical and practical challenges involved in designing a cooperative control system for multiple vehicles are:

- The need to develop a system of subsystems rather than a single system for the vehicles.
- The communication links between individual vehicles are often limited and unreliable.
- Arbitration between team goals and individual goals needs to be done in real time.
- The computational resources of each individual vehicle will always be limited.
- Complexity of the entire system increases rapidly (non-linear) as more subsystems are added to the control group.

**Advantages of Cooperative over Single Autonomous Vehicles**

The cooperative multi-vehicle systems have several advantages over the single autonomous vehicle systems. Some of them are:
- Better fault tolerance than individual vehicle systems.
- Tasks that are too complex or difficult for a single system can be executed.
- Quicker response to environmental factors through distributed sensing and actuation.
- Redundancy allows for substituting cheaper components and hence cost effective.
- Easier to operate since the system is more self-reliant.

Individual vehicle in a multi-vehicle system has different tasks and applications. These tasks are usually divided into sub-tasks so that the system would be easier to comprehend, configure and control. After decomposing the tasks to sub-tasks they are distributed to the multiple vehicle agents such that all the tasks and sub-tasks combined will obtain the desired end goal. The collective behavior of the multi-vehicle system is exhibited through the communication between the individuals using an explicit communication channel or indirectly through one agent sensing the others. The overall system performance can be measured by the characteristics such as execution time, computational complexity, robustness and fault tolerance. These characteristics depend on the control architectures and strategies and typology of the whole system.

Fig 1.1. Division and Distribution of Tasks among individual agents.
Applications

The applications for multi-vehicle systems are found in a variety of fields, such as industrial robots, service robots, military vehicles, biological research and space exploration. Every application has a different objective and expected outcome. Some examples of current day multi-vehicle systems are Unmanned Aerial Vehicles (UAV), Underwater Autonomous Vehicles and Grounded Mobile Robots.

A team of vehicles or robots can be more effective than a single robot in a number of situations. A team of robots can accomplish a task more quickly than a single robot by concurrently executing several sub tasks. The team structure also allows some of the robots to acts as specialists in certain tasks and can execute the tasks more efficiently than a set of generalist robots (eg: Reconnaissance, Heavy lifting).

The applications for multi-vehicle systems are usually classified based on the types of vehicles and the mission objectives assigned to those vehicles. However, the classification could be done in broadly two ways based on the planning and coordination between the systems, namely a centralized approach and a distributed approach.

Centralized Approach

The difficulty in coordinating a team of robots to perform a single, global task leads to a centralized approach to coordinate the robot team as a unified system with varying degrees of freedom. A powerful computer acting as a centralized coordinator manages the group over a communication network to perform the complex tasks. The best available algorithms may not lead to optimal level of coordination due to exponential
complexity involved. Thus, the problem becomes intractable if the team size grows more than a few robots even for the most powerful computers available today. Additionally, this approach assumes that the communication network is reliable and the information does not change during the time when the optimal plan is put together. These assumptions usually breakdown for problems in which the environmental conditions are constantly in flux, communication network has limited reliability, and the individual robots behave in unpredictable ways from time to time. Another issue with this approach is the vulnerability to single point failure and a lack of immediate failover mechanism in case the central command unit of the team malfunctions, to avoid disabling the entire system.

**Distributed Approach**

Distributed approaches address the problems that arise with centrally controlled and coordinated systems. The distributed system allows each robot to operate independently, acting on local sensory information rather than a remote transmitted command. A robot may also localize the coordination with the other robots through proximity detection and peer to peer sub-division of tasks that may be too complex to handle by a single robot. The primary benefit of this approach is that it eliminates the computation needs at the central level, since each robot plans and executes its own activities based on the initial input. Also, much of the back and forth communication is eliminated, since the robots communicate among themselves based on proximity. The robots become more agile and responsive to changing conditions by sensing the environment locally. The approach also leads to more robustness in the system since the malfunction of the central command unit no longer affects the performance of the team adversely. However, since the robots
function in a semi-autonomous manner, the computational needs of an individual robot are higher in the distributed approach than the centralized approach. The distributed approach works best for problems that are amenable to decomposition of isolated sub tasks and the aggregation of the sub tasks into a meaningful goal. The primary issue of the distributed approach is that sometimes the decomposition of tasks may lead to suboptimal achievement of the team goal.

**Real-world Scenario**

Consider a real world disaster response situation where human first responders like firefighters make a formation around a problem area, like forest fires or urban building fires. The first responding firefighters often have limited information on the scope and size of the problem, like the intensity and range of the fire, and so suffer casualties. In the US alone, there are at least 20 firefighter fatalities per year, and in this year 2013, a single forest fire in Arizona consumed the lives of 19 firemen. [6]

A system of autonomous firefighting robots could potentially reduce the fatalities even if it does not eliminate all of the risks faced by human firefighters. The series of operations that will be performed by a cooperative multi-vehicle firefighting system is listed below:

- A multi-vehicle firefighting system receives a distress signal for a fire in an urban building with GPS location information.
- The vehicles within a certain radius calculate the time it will take to travel to the problem site.
- A set of autonomous vehicles move automatically to the location based on the pre-defined response time limit.
• Once they reach the problem area, the vehicles evaluate the situation, like radius of fire.

• Each vehicle initiates a communication with its neighboring vehicles to reach a consensus and execute a formation, in circular form or in other form, to secure a perimeter around the fire.

• From time to time, the vehicles re-evaluate their formation and adjust their positions based on the progress in controlling the fire.

Before demonstrating the results of the formation control through simulation, this report describes briefly the mathematical background behind the formation control and consensus algorithm, including the parts of graph theory used.
CHAPTER 2: GRAPH THEORY

Graph Definitions

A graph is a diagram consisting of points called vertices (nodes) which are joined by lines, called edges (arc or line) and each edge joins exactly two vertices. It has wide range of applications in real life. For example, the points could represent the train stations and the route between them could be represented by edges. [2]

A graph \( G = (V, E) \) consists of a non-empty set of vertices \( V \) and a set of edges \( E \). Each edge is a subset of \( V \) of two elements denoting a connection between two vertices. In a graph, two or more edges joining the same pair of vertices are multiple edges. An edge joining a vertex to itself is a loop or self-loop. In general, we assume \( V = \{1, 2, 3 \ldots n\} \), where \( n \) is the order of the graph. A graph with no multiple edges or no self-loops between the same pair of vertices is a simple graph. If a simple graph consists of an edge between any pair of distinct vertices is called a complete graph. The vertices \( v \) and \( w \) of a graph are adjacent vertices if they are joined by an edge \( e \). The vertices \( v \) and \( w \) are incident with the edge \( e \), and the edge \( e \) is incident with the vertices \( v \) and \( w \).

\[
V = \{1, 2, 3, 4, 5, 6, 7\} \\
|V| = 7
\]

Fig 2.1. Example for Graph.

The following are the definitions of various graphs.
**Undirected Graph**

An undirected graph is a graph in which the vertices are connected by undirected edges. An undirected arc is an edge that has no arrow. Both ends of an undirected arc are equivalent--there is no head or tail. Therefore, we represent an edge in an undirected graph as a set rather than an ordered pair.

![Diagram of an undirected graph with vertices A, B, C, D, E, F and edges (A, B), (A, E), (B, E), (C, F).]

\[ V = \{ A, B, C, D, E, F \} \]
\[ |V| = 6 \]

\[ E = \{ \{ A, B \}, \{ A, E \}, \{ B, E \}, \{ C, F \} \} \]
\[ |E| = 4 \]

**Fig 2.2. Example for undirected Graph.**

**Isomorphism**

Two graphs G and H are isomorphic if H can be obtained by relabeling the vertices of G—that is, if one-one correspondence between the vertices of G and those of H, such that the number of edges joining each pair of vertices in G is equal to the number of edges joining the corresponding pair of vertices in H. Such one-one correspondence is an isomorphism.

[2]

**Sub Graph**

A sub graph of a graph G is a graph all of whose vertices are vertices of G and all of whose edges are edges of G. In a graph, the degree of a vertex v is the number of edges incident with v, with each loop counted twice, and is denoted by deg v. [2]
Regular Graph

A graph is regular if its vertices all have the same degree. An undirected graph is connected if you can get from any node to any other by following a sequence of edges since any two nodes are connected by a path. [2]

Handshaking lemma:

In any graph, the sum of all the vertex degree is equal to twice the number of edges.

Connected Graph

A graph is connected if there is a path between each pair of vertices, and is disconnected otherwise. An edge in a connected graph is a bridge and if its removal leaves a disconnected graph. Every disconnected graph can be split up into a number graphs, called components. [2]
Bipartite Graph

A bipartite graph is a graph whose set of vertices can be split into two subsets A and B in such a way that each edge of the graph joins a vertex in A and a vertex in B. [2]

Directed Graphs or Digraphs

A digraph or directed graphs consists of a set of elements called vertices and a set of elements called arcs. Each arc joins two vertices in a specified direction. In a digraph, two or more arcs joining the same pair of vertices in the same direction are multiple arcs. An arc joining a vertex to it is a loop. A digraph with no multiple arcs or loops is a simple digraph. The vertices v and w of a digraph are adjacent vertices if they are joined by an arc e. An arc e that joins v to w is incident from v and incident to w; v is incident to e, and w is incident from e. A directed tree is a directed graph in which any two vertices are connected by exactly one simple path. A spanning tree is a subgraph of a connected graph that includes all the vertices and is also a tree. [2]

Sub graph Isomorphism

A sub digraph of a digraph D is a digraph all of whose vertices are vertices of D and all of whose arcs are arcs of D. Two digraphs C and D are isomorphic if D can be obtained by relabeling the vertices of C that is, if there is one-one correspondence between the vertices of C and those of D, such that the arcs joining each pair of vertices in C agree in
both number and direction with the arcs joining the corresponding pair of vertices in D.

[2]

G1 is a directed graph.

G2 is the sub graph of G1.

G1 and G3 are isomorphic.

\[ G1 \]
\[ G2 \]
\[ G3 \]

Fig 2.7. Sub graph Isomorphism.

Degree

In a directed graph, the out degree of a vertex v is the number of edges leaving it and the in-degree is the number of edges entering it. [2]

The out-degree sequence of a digraph is the sequence obtained by listing the out degrees of D in increasing order, with repeats as necessary. The in-degree sequence of D is defined analogously.

\[
\begin{array}{ccc}
1 & \rightarrow & 2 \\
4 & \rightarrow & 5
\end{array}
\]

The out-degree of 2 is 2 and The in-degree of 2 is 1

Fig 2.8. Degree of Digraphs.

Strongly Connected Digraph

A digraph is connected if its underlying graph is a connected graph, and is disconnected
otherwise. A digraph is strongly connected if there is a path between each pair of vertices. [2]

![Fig 2.9. Strongly Connected Digraph.](image)

**Handshaking lemma:**

In any digraph, the sum of all the out-degrees and the sum of all the in-degrees are both equal to the number of arcs.

**Relationship between Matrices and Graphs**

Graph theory is closely related to Matrix theory.

A matrix $A$ is positive definite (semi definite) if and only if all eigenvalues of $A + A^T$ are positive (non-negative).

The Kronecker product of a matrix $A$ of size $nxm$ and a matrix $B$ of size $kxl$ is defined as

$$A \otimes B = \begin{pmatrix} A_{11}B & \cdots & A_{1m}B \\ \vdots & \ddots & \vdots \\ A_{n1}B & \cdots & A_{nm}B \end{pmatrix}$$

A matrix is reducible if it can be written as

$$P \begin{pmatrix} A & B \\ 0 & C \end{pmatrix} P^T$$

Where $P$ is the permutation matrix. A matrix is irreducible if it is not reducible. If $A$ is irreducible then for any $i \neq j$, there exists $i = i_1, i_2, ..., i_k = j$, such that $A_{i_h,i_{h+1}} \neq 0$ for all $h$. Thus a digraph is strongly connected if and only if its adjacency matrix (or its
Laplacian matrix) is irreducible. [5]

**Adjacency Matrix**

Let $G$ be a graph with $n$ vertices labeled $1, 2, 3, ... n$. The adjacency matrix $A(G)$ of $G$ is the $n \times n$ matrix in which the entry in row $i$ and column $j$ is the number of edges joining the vertices $i$ and $j$. [2]

For a graph of $G = (V, E)$, it is assumed that the vertices are numbered $1, 2, ..., |V|$ in some arbitrary order. Then the adjacency matrix representation of $G$ consists of a $|V| \times |V|$ matrix $A = (a_{ij})$ such that

$$a_{ij} = \begin{cases} 
1, & \text{if } (i, j) \in E \\
0, & \text{otherwise}
\end{cases}$$

The adjacency matrix of a digraph is not usually symmetrical about the main diagonal. Also if the digraph has no loops, then each entry on the main diagonal is 0, the sum of the entries in any row is the out-degree of the vertex corresponding to that row, and the sum of the numbers in any column is the in-degree of the vertex corresponding to that column.

![Fig 2.10. Adjacency matrix Representation.](image)

The interaction graph of a matrix $A$ is defined as the graph of $A^T$. So, the interaction
graph and the graph of a matrix are reversals of each other. The interaction graph is useful in networks of coupled dynamical systems with linear coupling.

**Laplacian Matrix**

The Laplacian matrix of a graph is a zero non-negative matrix $L$ defined as $L = D - A$, where $A$ is the adjacency matrix and $D$ is the diagonal matrix of vertex out degrees.

A vertex is balanced if it’s out degree is equal to its in degree. A graph is balanced if all its vertices are balanced. A graph is balanced if and only if its Laplacian matrix has zero row sums and zero column sums.
CHAPTER 3: CONSENSUS ALGORITHM

Overview

In dynamic systems or in networks of autonomous agents, consensus means to reach an agreement regarding a certain quantity of interest that depends on the state of all agents. A consensus algorithm is an interaction rule that specifies the information exchange between an agent and all of its neighbors on the network.

In the firefighting scenario described earlier, each autonomous firefighting robot determines the coordination variables for the problem, in this case the radius of fire, the radius of the perimeter, etc., instantiates the coordination variables and shares it with the nearby robots. In order for the firefighting to be truly effective, each robot needs to position itself at a distance from the center where its effort at quenching the fire will be maximized. It will then be necessary for the robots to come to an agreement on the formation of an effective perimeter and also to be at a reasonable separation from other each other, in other words agree on the set of the coordination variables or that the differences between the coordination variables converge to a pre-specified variable. The robots will have to arrive at a consensus despite some of the constraints like imperfect sensors, communication dropout, sparse communication topologies, and noisy unreliable communication links.

Consensus Algorithms

Information Consensus

In Information consensus algorithm, the information state of each vehicle are updated and shared to the other vehicles in the network over a noisy and time varying environment. The information states are updated based on the information states of their neighbors. So,
the consensus algorithm is an update law which makes the information state of all the vehicles in the network converges to a common value so that all the vehicles have similar dynamics. The information state update of each vehicle is modeled using a differential equation, if the communication topology has continuous communication. But, if the communication data arrive in discrete packets, then the information state update is modeled using a difference equation. In this project, a basic consensus algorithm in which a scalar information state is updated by each vehicle using a first-order differential equation is used.

The basic and most common consensus algorithm is given by

\[ \dot{x}_i(t) = -\sum_{j=1}^{n} a_{ij}(t) \left( x_i(t) - x_j(t) \right), \quad i = 1, 2, \ldots, n \]  

(3.1)

where \( a_{ij}(t) \) is the (i,j) entry of the adjacency matrix of the graph at time t, and \( x_i(t) \) is the information state of the i’th vehicle. [1]

Setting \( a_{ij} = 0 \) denotes the fact that vehicle \( i \) cannot receive information from vehicle \( j \). A consequence of the above equation is that the information state \( x_i(t) \) of vehicle \( i \) is driven toward the information states of its neighbors. Although the equation ensures that the information states of the team agree, it does not dictate a specified common value.

Suppose that there are \( n \) vehicles in the team. The team’s communication topology can be represented by directed graph \( G_n = (V_n, E_n) \), where \( V_n = \{1, \ldots, n\} \) is the node set and \( E_n \) is the edge set. The communication topology may be time varying due to vehicle motion or communication dropouts. For example, communication dropouts might occur when a disaster response vehicle (MAS) banks away from its neighbor or in the case of an Unmanned Air Vehicle (UAV) flies through an urban canyon.
For example, Figure below shows three different communication topologies for three vehicles.

(a)

(b)

(c)

Fig 3.1. Communication topologies for three graphs. (a) and (b) are not strongly connected (c) is strongly connected.
In this section, we investigate conditions under which the information states in the consensus algorithm converges when the communication topology is time invariant and the gains $a_{ij}$ are constant, that is, the nonsymmetrical Laplacian matrix $L_n$ is constant.

The Laplacian matrices for the 3 graphs above are as follows:

$$L_1 = \begin{bmatrix} 1 & -1 & 0 \\ 0 & 1.5 & -1.5 \\ 0 & 0 & 0 \end{bmatrix}$$

$$L_2 = \begin{bmatrix} 1 & -1 & 0 \\ 0 & 1.5 & -1.5 \\ 0 & -2 & 2 \end{bmatrix} \quad (3.2)$$

$$L_3 = \begin{bmatrix} 1 & -1 & 0 \\ 0 & 1.5 & -1.5 \\ -2 & 0 & 2 \end{bmatrix}$$

To achieve convergence, it needs to be ensured that zero is a simple eigenvalue of $L_n$ [1]. Although, all of the nonsymmetrical Laplacian matrices above, have simple zero eigenvalues, Figs. a and b are not strongly connected while Fig c is strongly connected. The only common feature is that the directed graphs of $L_1$, $L_2$, and $L_3$ all have rooted directed spanning trees. Hence it can be shown [4] that zero is a simple eigenvalue of $L_n$ if and only if the associated directed graph has a directed spanning tree. This result implies that Equ (3.1) achieves consensus if and only if the directed communication topology has a directed spanning tree or the undirected communication topology is connected.

When multiple vehicles agree on the value of a variable of interest, they are said to have
reached consensus. Information consensus guarantees that vehicles sharing information over a network topology have a consistent view of information that is critical to the coordination task. To achieve consensus, there must be a shared variable of interest, called the information state, as well as appropriate algorithmic methods for negotiating to reach consensus on the value of that variable, called the consensus algorithms. The information state represents an instantiation of the coordination variable for the team. In the case of the autonomous firefighting robots, this include a local representation of the center and shape of a formation, the rendezvous time, the length of a perimeter being monitored, the direction of motion for a multi-vehicle swarm, and the probability that the fire will be controlled and quenched.

**Simulation Results of Consensus for three vehicles**

In our simulation, we have used the communication topology in Fig 3.1(c) of three vehicles which is strongly connected since there is a directed path between every pair of nodes. We have analyzed two cases of consensus algorithm for which the initial positions of the three vehicles $x_1, x_2$ and $x_3$ are the same (0.2, 0.4 and 0.6 respectively). In the first case, basic consensus algorithm in matrix form: $\dot{x} = -L_3 x$ is used, where $L_3$ is the Laplacian matrix of the communication topology in Fig 3.1(c). Although the graph is strongly connected, the consensus equilibrium state is not achieved at the average consensus since the graph is not balanced. In the second case, we have used the consensus form as $\dot{x} = -\text{diag}(w)L_3 x$ where $w$ is the positive column left eigenvector of $L_3$ satisfying $w^T 1 = 1$ which corresponds to the eigen value 0 and $\text{diag}(w)$ is the diagonal matrix whose diagonal entries are given by $w$. In this case, we have reached the equilibrium at the average consensus since the graph $\Gamma[\text{diag}(w)L_3]$ is
balanced as shown in Fig 3.2.

Corresponds to \( \dot{x} = -L_3 x \)

Corresponds to \( \dot{x} = -\text{diag}(w) L_3 x \)

Fig 3.2. Simulation results of Consensus Algorithms.
CHAPTER 4: DECENTRALIZED FORMATION CONTROL

Formation Control

Formation control is one of the important applications of coordinated control for a group of autonomous vehicles/robots. In many applications, a group of autonomous vehicles are required to follow a predefined trajectory while maintaining a desired spatial pattern. Moving in formations is considered over conventional systems, since it can reduce the system cost, increase the robustness and efficiency of the system [3].

In Formation control, a network of vehicles exchange information among themselves with the intention of achieving specified formation. The network achieves the formation through decentralized control or centralized control. Several information flow laws are considered in order to improve the performance of the vehicle network. Formation control for collections of vehicles is a problem that has recently generated much interest in the research community, motivated largely by applications to autonomous vehicles. Many of the feedback schemes investigated are inspired by the motion of aggregates of individuals in nature, like how flocks of birds and schools of fish achieve coordinated motions without the use of a central controlling mechanism [3]. Formation control for multiple vehicles/Robots is desired where it enables capabilities which are not possible with individual vehicles.

The decentralized cooperative formation control approach combines results from control theory, and algebraic graph theory. Formation control is closely related to the consensus seeking autonomous agents. Control to formation is in a way a consensus problem, since in order to converge to formation the vehicles have to achieve, among other things, the same velocity [3]. In addition, the vehicle dynamics in the formation problem are
governed by second-order dynamics [3]. Key features of our development include that the vehicle controls are decentralized, and each vehicle computes its own control based on information available locally to it. The communication among vehicles can be sparse and formation stability is maintained. We also define the minimum communication requirements necessary to find a stabilizing feedback. Building on our earlier work, we define formation hierarchy and show that our convergence and stability results apply in this case too. Only sub formation leaders need to communicate with the higher level of the hierarchy. In all these cases, the communication graph can be independent of the desired formation geometry. This allows, for example, the same communication links to be used for different formation geometrics. As a natural next step in our development, we investigate stability of formations when the spatial and communication structures are allowed to vary over time.

**Model for formation control**

For this model, it is assumed that there are $N$ vehicles with the same dynamics

$$\dot{x}_i = A_{veh}x_i + B_{veh}u_i, \quad i = 1, \ldots, N, \quad x_i \in \mathbb{R}^{2n}$$

where the entries of $x_i$ represent $n$ configuration variables for $i$ vehicles and their derivatives, and the $u_i$ represent control inputs.[3] For simplicity we will assume further that the matrices $A_{veh}$ and $B_{veh}$ have the form

$$A_{veh} = \text{diag}\left(\begin{pmatrix} 0 & 1 \\ a_{21}^1 & a_{22}^1 \end{pmatrix}, \ldots, \begin{pmatrix} 0 & 1 \\ a_{21}^n & a_{22}^n \end{pmatrix}\right)$$

$$B = I_n \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

where $\otimes$ denotes the Kronecker Product. The odd-numbered entries of $x = (x_1, \ldots, x_N)^T$ correspond to position-like variables and the even-numbered entries to
velocity-like variables. The following notations are used:

\[ x_p = (x_{p1}, \ldots, x_{pN})^T, \quad x_v = (x_{v1}, \ldots, x_{vN})^T \]

to denote the vectors of position-like and velocity-like variables, so

\[ x = x_p \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} + x_v \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} \]

**Definition 1**[3]: A formation is a vector \( h = h_p \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} \in \mathbb{R}^{2nN} \). The \( N \) vehicles are in formation \( h \) at time \( t \) if there are vectors \( q, w \in \mathbb{R}^n \) such that \((x_{pi})(t) - (h_{pi}) = q\) and \((x_{vi})(t) = w\), for \( i = 1, \ldots, N \). The vehicles converge to formation \( h \) if there exists \( \mathbb{R}^n \) valued functions \( q(\bullet), w(\bullet) \) such that \((x_{pi})(t) - (h_{pi}) - q(t) \to 0 \) and \((x_{vi})(t) - w(t) \to 0\), as \( t \to \infty \), for \( i = 1, \ldots, N \).

Figure below illustrates the interpretation of the vectors in the definition.

**Fig. 4.1. Physical representation of vehicles in formation.**

**Definition 2**: [3] Let \( h = h_p \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} \in \mathbb{R}^{2nN} \), let \( 1_N \) denote the all ones vector of size \( N \), and let \( e_j, j = 1, \ldots, 2n \) denote the standard basis vectors in \( \mathbb{R}^n \). Set \( W = \{1_N \otimes e_j : j = \)
We define $h$- formation space $\mathcal{F}_h$ by

$$\mathcal{F}_h = h + \text{span} W = \{ x | \exists \gamma \in \mathbb{R}^{2n} : x - h = 1_N \otimes \gamma \} \quad (4.4)$$

Notice that $x - h = 1_N \otimes \gamma$ is equivalent to $(x_p)_i - (h_p)_i = q$ and $(x_v)_i = w$ where

$$\gamma = q \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} + w \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$  

With this definition $x$ is in formation $h$ if and only if $x \in \mathcal{F}_h$ and $x(t)$ converges to formation $h$ if and only if the distance from $x(t)$ to the space $\mathcal{F}_h$ tends to zero as $t \to \infty$.

Let $G = (V, E)$ be the graph which captures the communication links between vehicles (where $V$ is a set of vertices and $E$ is the set of edges connecting the vertices). Each vertex represents a vehicle and there is a directed edge from vehicle $i$ to vehicle $j$, if there is a communication link sending information from $i$ to $j$ and vehicle $i$ is neighbor of $j$. The vehicle $j$ would then be able to use this information in a feedback formula to adjust its own state. For each vehicle $i$, $\mathcal{I}_i$ denotes the set of its neighbors. In this model, each vehicle uses only relative information about its neighbors $\mathcal{I}_i$ to capture the decentralized nature of the feedback control law. The controls $u_i$ are functions of $x_j - x_i$ and $h_j - h_i$ for each $j \in \mathcal{I}_i$.

To combine the relative information, output functions $y_i$ are defined and computed from an average of the relative displacements (and velocities) of the neighboring vehicles [3] as follows:

$$y_i = (x_i - h_i) - \frac{1}{|\mathcal{I}_i|} \sum_{j \in \mathcal{I}_i} (x_j - h_j), \quad i = 1, \ldots, N \quad (4.5)$$

where $|\mathcal{I}_i|$ indicates the number of neighbors to vehicle $i$. However, there is a possibility that a vehicle might receive no information and the corresponding vertex in the graph
would have no neighbors, which might be the case in formation with leaders.

We will define the output functions $z_i$ by

$$z_i = \sum_{j \in \mathcal{I}_i} \left( (x_i - h_i) - (x_j - h_j) \right), \quad i = 1, \ldots, N \quad (4.6)$$

As a result, the corresponding output vector $z$ can be written as $z = L(x - h)$ where $L = L_G \otimes I_{2n}$ and $L_G$ is the Laplacian matrix of the communication graph $G$. Collecting the equations for all the vehicles into a single system we obtain

$$\dot{x} = Ax + Bu \quad (4.7)$$

$$z = L(x - h) \quad (4.8)$$

with $A = I_N \otimes A_{veh}, B = I_N \otimes B_{veh}$.

We are interested in studying the existence of feedback matrices $F$ such that the solutions to the following equation converge to formation $h$.

$$\dot{x} = I_N \otimes A_{veh} x + L_G \otimes B_{veh} F_{veh} (x - h) \quad (4.9)$$

Later on we will see the relationship between existence of feedback matrices $F_{veh}$ that guarantee convergence to formation and properties of the communication graph $G\{3\}$. Before going into further details of formation control, let us see how eigenvalues of $L_G$ play a central role in formation $[3]$. Let $U$ be a matrix such that $\tilde{L}_G = U^{-1} L_G U$ is upper triangular. In particular, the eigenvalues of $L_G$ are the diagonal entries of $\tilde{L}_G$. A direct calculation using the special form of $A, B$ and $F$ gives

$$(U^{-1} \otimes I_{2n})(A + BFL)(U \otimes I_{2n}) = I_N \otimes A_{veh} + \tilde{L}_G \otimes B_{veh} F_{veh} \quad (4.10)$$

The right-hand side is block upper triangular. Its diagonal blocks are of the form:

$$A_{veh} + \lambda B_{veh} F_{veh}$$

where $\lambda$ is an eigenvalue of $L_G$. There is one block for each eigenvalue (counting multiplicity). Therefore, the eigenvalues of $A + BFL$ are those of $A_{veh} +$
\( \lambda B_{veh} F_{veh} \) for \( \lambda \) an eigenvalue of \( L_G \). [3]

**Simulation Results of Formation control**

In this example, we have explored two formations, the circle and the hexagon. The desired formation is specified as the vertices of a circle and the vertices of the hexagon. First we assume that \( a_{22} = 0 \) so each coordinate is modelled as a double integrator. For our simulation purposes, the Laplacian matrix is dynamically constructed based on the proximity between the vehicles. To construct the Laplacian matrix, the vehicles \( i \) and \( j \) are considered adjacent if the distance between those two vehicles is less than an arbitrary chosen threshold value. Using the Proposition 4.5 of [3], stabilizing pair of feedback gain values \( f_1 \) and \( f_2 \) are chosen such that \( f_1 < 0 \) and \( f_2 < 0 \) with \( f_2 \) sufficiently large in absolute value, so that the solution to the following equation converge to formation \( h \).

\[
\dot{x} = I_N \otimes A_{veh} x + L_G \otimes B_{veh} F_{veh} (x - h)
\]

In the both the circle and hexagon simulation results, the initial positions of the vehicles are marked with an 'o' and the intermediate and final positions are marked with '*'. The results are shown in Fig 4.2 and Fig 4.4 as six subplots in which each subplot represents the position of vehicles at different iterations of time. There are four intermediate iterations that are depicted in addition to the initial and final positions. As can be seen, the vehicles slowly converge to the formation around 500 iterations.

In the following trajectory graphs Fig 4.3 and Fig 4.5, the movement of the vehicles from its initial position to the final position is captured, with 'o' being the initial position and '*' being the final position. The dotted curve represents the position of vehicles at different instances of time.
Fig 4.2. Circle Formation of Vehicles from Initial to Final position.

Fig 4.3. Trajectory of Circular Formation of Vehicles.
Fig 4.4. Hexagon Formation of Vehicles from Initial to Final position.

Fig 4.5. Trajectory of Hexagonal Formation of Vehicles.
CHAPTER 5: CONCLUSIONS AND FUTURE WORK

In this project, the concept of cooperative control between multiple autonomous vehicles has been explored in-depth and applied to a real-world scenario of an autonomous disaster response system. The basics of a consensus algorithm were studied with references to directed graphs and Laplacian matrices. The information consensus algorithm was implemented using MATLAB to demonstrate the convergence of coordination variables in a time varying environment. The formation control was simulated in MATLAB using random starting points for the vehicles and the trajectories were plotted for the vehicles. The results of the simulation provide a solid foundation to extend the model to introduce various real world constraints and observe the system performance. The end goal of building an autonomous disaster response system also provides a road map for several of the characteristics that are expected out of the simulation model.

Despite the recent advancements in UAV and autonomous driving vehicles, implementing a cooperatively controlled disaster response system described in this report is still a somewhat distant possibility. However the immense cost incurred by society every year in terms of human and financial losses in dealing with the natural and man-made disasters will provide the necessary impetus to make this a reality sooner than expected. The simulation presented in this project addresses some of the core concerns of an autonomous disaster response system, namely initiating a consensus process and forming a perimeter around the disaster zone. There are several other challenging aspects to this system that could be considered towards future work.
• The size and shape of the disaster zone could be parameterized and added to the simulation as a no-go zone for the vehicles. This will limit the range of movement of the vehicles to arrive at the formation.

• The shape of the formation could be made to take into account the terrain topology like streets and hills instead of the idealized shapes like circles or squares.

• Disasters like fire rarely remain static and tend to move in the direction of the wind or the burning material availability. The formation of the autonomous vehicles could be adjusted to take into account the changes in the topology of the disaster zone, while keeping the formation intact.

• Some of the autonomous vehicles could be made unavailable in a random manner, like a loss of communication or damage due to fire, and the simulation could be enhanced to observe the impact on the active vehicles.

• Since disaster response is extremely time sensitive, the simulations could be extended to study the impact of various environmental constraints on the time to arrive at a formation.

The most exciting future work will be to potentially implement a prototype cooperative control system, probably on a set of autonomous vehicles like the Google self-driving cars and observe the results under various real world conditions.
REFERENCES

[1] Information Consensus in Multivehicle Cooperative Control
Wei Ren, Randal W. Beard, and Ella M. Atkins

Joan M. Aldous and Robin J. Wilson.

G. Lafferriere, A. Williams, J. Caughman, J. J. P. Veerman
Department of Mathematics and statistics, Portland State University


Chai Wah Wu


ADDITIONAL REFERENCES

Wei Ren, Randal W. Beard, and Ella M. Atkins

[8] Coordination Control of Multiple Mobile Robots
Filippo Arrichiello http://webuser.unicas.it/arrichiello  [Feb 13]

[9] Consensus and Cooperation in Networked Multi-Agent Systems
Reza Olfati-Saber, J. Alex Fax, and Richard M. Murray

[10] Synchronization in Complex Networks of Nonlinear dynamical Systems
Chai Wah Wu

Tony Stentz http://www.frc.ri.cmu.edu/~axs/multi-robot.html  [Mar 13]
**APPENDIX: SOURCE CODE**

Consensus Algorithm \( \frac{dx}{dt}(t) = -L(t)x(t) \)

% CONVERGENCE ANALYSIS OF CONSENSUS ALGORITHMS

%Consensus Algorithm is written as
%\( \frac{dx}{dt}(t)=-L(t)x(t) \)
%where \( L(t) \) is the Laplacian Matrix of the underlying
%communication graph and \( x(t) \)is the information state
%Initial values of Information states are 0.2,0.4,0.6 for
%the corresponding vehicles.
%x0=[0.2,0.4,0.6]'
tspan = [0 10];
[t,x]=ode45('information_state',tspan,x0);
t
plot(t,x(:,1),'r');
hold on
plot(t,x(:,2),'b');
hold on
plot(t,x(:,3),'g');
hold on
title('Consensus for three vehicles');
xlabel('Time(s)');
ylabel('Information states');

%Function "information_state" updates the information state of each
%vehicles
function xdot=information_state(t,x)
xdot=[-x(1)+x(2);-1.5*x(2)+1.5*x(3);2*x(1)-2*x(3)];

---

Consensus Algorithm \( \frac{dx}{dt}(t) = -\text{diag}(w)L(t)x(t) \)

% CONVERGENCE ANALYSIS OF CONSENSUS ALGORITHMS

%Consensus Algorithm is written as
%\( \frac{dx}{dt}(t)=-\text{diag}(w)L(t)x(t) \)
%where \( L(t) \) is the Laplacian Matrix of the underlying
%communication graph, \( w \) is the positive column left eigenvector and
%x(t) is the information state
%Initial values of Information states are 0.2,0.4,0.6 for
%the corresponding vehicles.

x0=[0.2,0.4,0.6]'
tspan = [0 10];
[t,x]=ode45('information_state1',tspan,x0);
t
plot(t,x(:,1),'r');
hold on
plot(t,x(:,2),'b');
hold on

plot(t,x(:,3),'g');
hold on
title('Consensus for three vehicles');
xlabel('Time(s)');
ylabel('Information states');

function xdot=information_state1(t,x)
L3=[1 -1 0; 1.5 -1.5; -2 0 2];
[v,a]=eig(L3');
By inspection, v(:,1) is the positive column left eigenvector
xdot=[0.7682*(-x(1)+x(2));(0.5121)*(-1.5*x(2)+1.5*x(3));(0.3841)*(2*x(1)-2*x(3))];
----------------------------------

---

Formation Control

% Source Code for Decentralized Circle Formation
% Decentralized Formation Control algorithm %
% Vertices of Final Formation ***************%

radius=3;
%circle formation (The vehicles will be positioned
%equidistant on the perimeter)
hx=[radius radius*cos(pi/6) radius*cos(pi/3) radius*cos(pi/2)
radius*cos(2*pi/3) radius*cos(5*pi/6) radius*cos(pi) radius*cos(7*pi/6)
radius*cos(4*pi/3) radius*cos(3*pi/2) radius*cos(5*pi/3)
radius*cos(11*pi/6) radius*cos(2*pi)];
hy=[0 radius*sin(pi/6) radius*sin(pi/3) radius*sin(pi/2)
radius*sin(2*pi/3) radius*sin(5*pi/6) radius*sin(pi) radius*sin(7*pi/6)
radius*sin(4*pi/3) radius*sin(3*pi/2) radius*sin(5*pi/3)
radius*sin(11*pi/6) radius*sin(2*pi)];

% Vertices of Initial Location for circle%
x=[2 5 4 8 7 3 4 6 5 3 2 1 8];
%y=[1 2 3 4 5 6 7 8 9 10 11 12];
y=[8 9 10 11 12 1 2 3 4 5 6 7];

% Number of vehicles
n=12;
% Initialize the model parameter and Output vectors
X=zeros(1,4*n);
XX=zeros(4*n,1);
h=zeros(4*n,1);
Aveh=[[0 1;0 0] zeros(2);zeros(2) [0 1;0 0]];
Bveh=kron(eye(2),[0;1]);

% Finding A=I_N(kron)A_veh, B=I_N(kron)B_veh
A=kron(eye(n),Aveh);
B=kron(eye(n),Bveh);
% Feedback Values%
f1=-1;
f2=-3;
% Feedback matrix%
\[ F = \text{kron}(\text{eye}(n*2), [f1 \ f2]); \]
\[
\text{for } i = 1:n \\
\quad \text{hp}(2*i-1) = \text{hx}(i); \\
\quad \text{hp}(2*i) = \text{hy}(i); \\
\text{end} \\
\text{display(size(hp))};
\]

%Calculation of Formation Vector%
\[ h = \text{kron}(\text{hp}', [1; 0]); \]
\[
\text{for } i = 1:n \\
\quad \text{X}(4*i-3) = x(i); \\
\quad \text{X}(4*i-1) = y(i); \\
\text{end} \\
\text{BB} = B*F; \\
\text{XX} = X'; \\
\text{tspan} = 0:0.05:50; \\
[t, XX] = \text{ode45}(@\text{formmodel}, \text{tspan}, \text{XX}, [], h, BB, A, n); \\
\]

%Plotting the formation model%
\[ \text{linex} = \text{zeros}(n+1); \]
\[ \text{liney} = \text{zeros}(n+1); \]
\[
\text{for } i = 1:n \\
\quad \text{subplot(231);} \\
\quad \text{plot(XX(1,4*i-3), XX(1,4*i-1), 'o')} \\
\quad \text{title('Formation at initial position');} \\
\quad \text{xlabel('X');} \\
\quad \text{ylabel('Y');} \\
\quad \text{hold on;} \\
\quad \text{subplot(232);} \\
\quad \text{plot(XX(40,4*i-3), XX(40,4*i-1), '*')} \\
\quad \text{title('Formation at the iteration t=40');} \\
\quad \text{xlabel('X');} \\
\quad \text{ylabel('Y');} \\
\quad \text{hold on;} \\
\quad \text{subplot(233);} \\
\quad \text{plot(XX(80,4*i-3), XX(80,4*i-1), '*')} \\
\quad \text{title('Formation at the iteration t=80');} \\
\quad \text{xlabel('X');} \\
\quad \text{ylabel('Y');} \\
\quad \text{hold on;} \\
\quad \text{subplot(234);} \\
\quad \text{plot(XX(120,4*i-3), XX(120,4*i-1), '*')} \\
\quad \text{title('Formation at the iteration t=120');} \\
\quad \text{xlabel('X');} \\
\quad \text{ylabel('Y');} \\
\quad \text{hold on;} \\
\quad \text{subplot(235);} \\
\quad \text{plot(XX(150,4*i-3), XX(150,4*i-1), '*')} \\
\quad \text{title('Formation at the iteration t=150');} \\
\quad \text{xlabel('X');} \\
\quad \text{ylabel('Y');} \\
\quad \text{hold on;} \\
\quad \text{subplot(236);} \\
\quad \text{plot(XX(500,4*i-3), XX(500,4*i-1), '*')} \\
\quad \text{title('Final Circle Formation at the iteration t=500');} \\
\quad \text{xlabel('X');} \\
\]

ylabel('Y');
linex(i) = XX(1001, 4*i-3);
liney(i) = XX(1001, 4*i-1);
if i==n
    linex(n+1) = linex(1);
    liney(n+1) = liney(1);
    line(linex, liney, 'Color','g', 'LineWidth', 2);
end

hold on;
end

% Function to determine the formation model%
function xdot = formmodel(t, XX, h, BB, A, n)

x = zeros(1, n);
y = zeros(1, n);
d = zeros(n);
s = zeros(n);
l = zeros(n);
k = zeros(1, n);
for i=1:n
    x(i) = XX(4*i-3, 1);
    y(i) = XX(4*i-1, 1);
end
r = 3.8;
% construction of laplacian matrix
for i=1:n
    for j=1:n
        if i~=j
            d(i, j) = abs(sqrt((x(i)-x(j))^2+(y(i)-y(j))^2));
            s(i, j) = (d(i, j)<=r);
        else
            d(i, j) = 0;
            s(i, j) = 0;
        end
    end
end
for i=1:n
    k(i) = 0;
    for j=1:n
        k(i) = k(i) + s(i, j);
    end
end
for i=1:n
    for j=1:n
        if i==j
            l(i, j) = k(i);
        else
            l(i, j) = -s(i, j);
        end
    end
end
% Laplacian Matrix
l=kron(l,eye(4));

% Formation model

% Plotting Trajectory of vehicles from Initial position
% to the Final Circular Position
for i=1:12
    plot(XX(length(tspan),4*i-3),XX(length(tspan),4*i-1),'*')
end

% Source Code for Decentralized Hexagon Formation
% Decentralized Formation Control algorithm
% Vertices of Final Formation

radius=3;

hx=[radius radius*cos(pi/3) radius*cos(2*pi/3) radius*cos(pi)
radius*cos(4*pi/3) radius*cos(5*pi/3) radius*cos(6*pi/3)];
hy=[0 radius*sin(pi/3) radius*sin(2*pi/3) radius*sin(pi)
radius*sin(4*pi/3) radius*sin(5*pi/3) radius*sin(6*pi/3)];

% Vertices of Initial Location for hexagon
x=[1 1 1 1 1 1];
y=[1 2 3 4 5 6];

% Number of vehicles
n=6;
%Initialize the model parameter and Output vectors
X=zeros(1,4*n);
XX=zeros(4*n,1);
h=zeros(4*n,1);

Aveh=[[0 1;0 0] zeros(2);zeros(2) [0 1;0 0]];
Bveh=kron(eye(2),[0;1]);

%Finding A=I_N(kron)A_veh,B=I_N(kron)B_veh%
A=kron(eye(n),Aveh);
B=kron(eye(n),Bveh);

%Feedback Values%
f1=-1;
f2=-3;

%Feedback matrix%
F=kron(eye(n*2),[f1 f2]);
for i=1:n
    hp(2*i-1)=hx(i);
    hp(2*i)=hy(i);
end

%Calculation of Formation Vector%
h=kron(hp',[1;0]);
for i=1:n
    X(4*i-3)=x(i);
    X(4*i-1)=y(i);
end
BB=B*F;
XX=X';
tspan=0:0.05:50;
[t,XX]=ode45(@formmodel,tspan,XX,[],h,BB,A,n);

%Plotting the formation model%
linex=zeros(n+1);
liney=zeros(n+1);
for i=1:n
    subplot(231);
    plot(XX(1,4*i-3),XX(1,4*i-1),'o')
title('Formation at initial position');
xlabel('X');
ylabel('Y');
hold on;
subplot(232);
plot(XX(40,4*i-3),XX(40,4*i-1),'*')
title('Formation at the iteration t=40');
xlabel('X');
ylabel('Y');
hold on;
subplot(233);
plot(XX(80,4*i-3),XX(80,4*i-1),'*')
title('Formation at the iteration t=80');
xlabel('X');
ylabel('Y');
hold on;
subplot(234);
plot(XX(120,4*i-3),XX(120,4*i-1),'*');
title('Formation at the iteration t=120');
xlabel('X');
ylabel('Y');
hold on;
subplot(235);
plot(XX(150,4*i-3),XX(150,4*i-1),'*');
title('Formation at the iteration t=150');
xlabel('X');
ylabel('Y');
hold on;
subplot(236);
plot(XX(500,4*i-3),XX(500,4*i-1),'*');
title('Final Circle Formation at the iteration t=500');
xlabel('X');
ylabel('Y');
hold on;
linex(i)= XX(1001,4*i-3);
liney(i)= XX(1001,4*i-1);
if i==n
    linex(n+1)= linex(1);
    liney(n+1)= liney(1);
    line(linex,liney,'Color','g','LineWidth',2);
end
hold on;
end

% Function to determine the formation model%
function xdot = formmodel(t,XX,h,BB,A,n)

x=zeros(1,n);
y=zeros(1,n);
d=zeros(n);
s=zeros(n);
l=zeros(n);
k=zeros(1,n);
for i=1:n
    x(i)=XX(4*i-3,1);
y(i)=XX(4*i-1,1);
end
r=3.8;
%construction of laplacian matrix
for i=1:n
    for j=1:n
        if i==j
            d(i,j)=abs(sqrt((x(i)-x(j))^2+(y(i)-y(j))^2));
s(i,j)= ( d(i,j)<=r);
        else
            d(i,j)=0;
s(i,j)=0;
        end
    end
end
for i=1:n k(i)=0;
    for j=1:n
        k(i)=k(i)+s(i,j);
    end
end
for i=1:n
    for j=1:n
        if i==j
            l(i,j)=k(i);
        else
            l(i,j)=-s(i,j);
        end
    end
end

Laplacian Matrix%
l=kron(l,eye(4));

Formation model%
xdot=A*XX+BB*L*(XX-h);

-----------------------------------------------------------------------------------------------------
% Plotting Trajectory of vehicles from Initial position %
% to the Final Hexagon Position *****************************************************%
for i=1:6
    plot(XX(length(tspan),4*i-3),XX(length(tspan),4*i-1),'*')
    title('Trajectory of vehicles');
    xlabel('time');
    ylabel('position of vehicles');
    hold on
end
for i=1:6
    plot(x(i),y(i),'o')
    hold on
end
for j=1:length(tspan)
    plot(XX(j,1),XX(j,3),'k')
    plot(XX(j,5),XX(j,7),'k')
    plot(XX(j,9),XX(j,11),'k')
    plot(XX(j,13),XX(j,15),'k')
    plot(XX(j,17),XX(j,19),'k')
    plot(XX(j,21),XX(j,23),'k')
    hold on
end

-----------------------------------------------------------------------------------------------------