Distributed unmanned aerial vehicles formation with built-in safety systems

Wei, Dong

2016

Wei, D. (2016). Distributed unmanned aerial vehicles formation with built-in safety systems.
Master's thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/69086>

<https://doi.org/10.32657/10356/69086>

**Distributed Unmanned Aerial Vehicles
Formation with Built-in Safety Systems**

Wei Dong

School of Electrical and Electronic Engineering

A thesis submitted to the Nanyang Technological

University

in partial fulfillment of the requirement for the degree of

Master of Engineering

2016

Acknowledgments

I would like to express my gratitude and appreciation to the people who have assisted me during the preparation of this thesis.

I would like to appreciate my thesis supervisor, Professor Xie Lihua, for his patient guidance and kind help during my research and development period. Without his excellent suggestions and strong supports, I could not reach my destination that far easily and smoothly.

I want to thank Mr. Ma Xiao, Mr. Yuen Sien Huan who gave me strong technical supports in experiments. Besides, many thanks to Mr. Huynh Minh Quan for his helps whenever needed.

Lastly but not the least I would like to express my gratitudes to my wife Ms Zhang Songyuan and my parents. Their infinite love and support carried me through difficulties and enabled me to focus on my research.

Contents

Contents	ii
List of Figures	ix
List of Tables	xiii
1 Introduction	1
1.1 Motivation	1
1.2 Objective	2
1.3 Contributions	3
1.4 Organization	4
2 Literature Review	5
2.1 UAV Formation Configuration	6
2.1.1 Centralized Formation	7
2.1.2 Distributed Formation	7
2.1.3 Formation Configuration Selection	8
2.2 UAV Formation Control Schemes	8

2.2.1	Leader/Follower	8
2.2.2	Virtual Structure	9
2.2.3	Behavioral	9
2.2.4	Formation Scheme Selection	10
2.3	Distributed Multiple UAV Formation	11
2.4	UAV Safety	12
2.5	Summary	13
3	Distributed Formation Control Algorithms	14
3.1	Preliminary	14
3.1.1	Graph Theory and Rigidity	14
3.1.2	Azimuthal Equidistant Projection	17
3.2	UAV Modeling	20
3.3	Positive Potential Function	20
3.4	Distributed Formation Controller	21
3.4.1	Distributed Formation with Proportional Controller .	22
3.4.2	Distributed Formation with Proportional Integral Con- troller	29
3.5	Map Projection	34
3.6	Summary	36
4	Development of Distributed Formation Testbed	37
4.1	Multicopter Autopilot Architecture	37

4.1.1	Multicopter Control	37
4.1.2	Data Fusion	39
4.2	Quadcopter Frame	41
4.2.1	Frame Selection	42
4.2.2	Motor and Propeller	44
4.2.3	Battery	45
4.2.4	Electronic Speed Controller (ESC)	46
4.3	Quadcopter Avionics	47
4.3.1	Autopilot	47
4.3.2	Power Module	48
4.3.3	Inertial Management Unit (IMU)	49
4.3.4	Global Positioning System (GPS)	50
4.3.5	Telemetry Communication	51
4.3.6	Remote Control	52
4.4	Assembling	53
4.5	Summary	54
5	Autopilot Software Development	56
5.1	Autopilot Architecture	56
5.2	Distributed Formation Control	61
5.2.1	System Architecture Modification	61
5.3	Main Function Realization	63
5.3.1	Coordinate Projection	63

5.3.2	Distributed Formation Controller	65
5.3.3	Safety and Precaution	66
5.4	Distributed Formation Simulation	69
5.5	Signal and Program Interfacing	70
5.5.1	Flight Mode	70
5.5.2	Communication Link	71
5.6	Summary	72
6	UAV Safety Implementation	73
6.1	Hardware Platform	74
6.2	System Architecture	75
6.3	Preliminary	75
6.3.1	Fault Tree Analysis	75
6.3.2	Onboard Autopilot Sensors	76
6.4	Pre-flight Check	78
6.4.1	Sensor Stacks Checked during Pre-flight	79
6.4.2	Recursive Means and Variance Algorithm	81
6.5	In-flight Check	82
6.5.1	Mode Priority	84
6.6	Inertial Management Unit Online Swap	84
6.7	GCS Health Monitoring	86
6.8	Experimental Results	89
6.9	Summary	89

7	Integration Test and Discussion	94
7.1	Hardware in the Loop Simulation	95
7.2	PID Tuning	96
7.3	Distributed Formation Flight Experiment	99
7.4	Safety Precaution Mechanism	105
7.4.1	Communication Latency Test	105
7.4.2	Velocity Command	110
7.5	Summary	110
8	Conclusion and Recommendations	112
8.1	Conclusion	112
8.2	Recommendations	114
8.2.1	External Reference Velocity Flight Experiment	114
8.2.2	Increase Number of Formation UAVs	114
8.2.3	Risk Assessment Analysis	114
8.2.4	Elimination of Communication Latency Effects . . .	115
	Bibliography	117

Summary

Due to its high maneuverability, portability, high payload capability and good expandability, Unmanned Aerial Vehicle (UAV) is capable of accomplishing varieties of tough missions. In particular, cooperative multiple UAVs can significantly increase the mission efficiency and effectiveness. In some applications with complex environments such as narrow canyons, distributed formation with adjustable shapes is required. Moreover, a reliable autopilot safety system is mandatory so that the flight does not pose threats to the surrounding environment.

In this thesis, distributed formation control of multiple UAVs with built-in comprehensive safety features for operations in complex environments is studied. Real time in air communication and on-line trajectory generation are two tasks of distributed formation control.

Distributed formation control has been studied in an existing work by integrating proportional control into its path planner. There exist, however, steady state errors when an external reference velocity is applied to a certain UAV. In order to improve the formation control and eliminate the steady-state errors, a proportional-integral (PI) controller is proposed. It is shown that the proposed distributed PI controller can stabilize the system and achieve the zero steady-state error. Numerical and hardware in the loop

(HIL) simulations and flight tests are conducted to verify the effectiveness of the proposed approach.

The new proposed PI controller is a Multi Input Multi Output (MIMO) system. The integral gain in this controller is a 3×3 matrix. Therefore, finding an appropriate integral gain is very time consuming. Hence, search is conducted to obtain a suitable integral controller.

Since distributed formation is controlled by an on-board autopilot, failures may happen due to system or hardware faults. Hence, safety and robustness of the on-board autopilot become very important. Therefore a comprehensive approach to ensure safe operations for multiple UAVs formation in outdoor environments is proposed. A pre-flight UAV status inspection schedule and a hierarchical three layers in-flight safety architecture are developed in our system.

When flying multiple UAVs, a robust communication, and some appropriate safety control precautions are required. Latency of data link is discussed to improve system robustness and neighbor UAVs' position estimation.

List of Figures

3.1	Global Coordinates Before Azimuthal Earth Projection . . .	18
3.2	Projected Coordinates After Azimuthal Earth Projection . .	19
3.3	Distributed Formation Controller	24
3.4	Distributed Formation with Proportional Controller of $k_P=1$	25
3.5	Distributed Formation with Proportional Controller of $k_P=2$	26
3.6	Distributed Formation with Proportional Controller of $k_P=3$	26
3.7	Distributed Formation Potential Result with Proportional Con- troller of $k_P=1$	27
3.8	Distributed Formation Potential Result with Proportional Con- troller of $k_P=2$	28
3.9	Distributed Formation Potential Result with Proportional Con- troller of $k_P=3$	28
3.10	PI Controller Distributed Formation	33
3.11	Distributed Formation Potential Result with PI Controller .	34
4.1	Multicopter System Architecture	38

4.2	Flame Wheel 450mm Arm Length	43
4.3	Flame Wheel 450mm Central Board	43
4.4	Normal Power Distribution Board	43
4.5	Motor-Propeller Performance Sheet	45
4.6	Battery	46
4.7	Electronic Speed Controller	46
4.8	Distributed Avionics System	47
4.9	Pixhawk Autopilot	48
4.10	Pixhawk Power Module	49
4.11	Pixhawk uBlox GPS with compass	50
4.12	Telemetry System	51
4.13	Telemetry Setup	52
4.14	Remote Control Receiver	53
4.15	Remote Control Transmitter	53
4.16	Assembled Testbed for Distributed Formation	54
5.1	Autopilot System Architecture	57
5.2	Micro Object Register Broker Messages	60
5.3	Distributed Formation uORB Message	61
5.4	Autopilot System Architecture for Distributed Formation . .	62
5.5	Distributed Formation Control Program File List	63
5.6	Distribute Formation Simulator Diagram	70
5.7	UAV Flight Mode Diagram	71

5.8	MAVLink Communication Interface	72
6.1	Hardware UAV System	74
6.2	System Architecture	76
6.3	Fault Tree Analysis	77
6.4	Pixhawk	78
6.5	Preflight Check Sequence	79
6.6	Fault Tree Analysis Diagram	82
6.7	Mode Priority State Machine.	86
6.8	GCS field setup.	88
6.9	GCS health monitoring window	91
6.10	Formation Performance 8 UAVs	92
6.11	Formation Performance 14 UAVs	92
6.12	Safety System Testing Result	93
7.1	Distributed Formation Simulation Program Flow	95
7.2	Distributed Formation Simulation Program Flow	96
7.3	Distributed UAV Flight Area	98
7.4	Distributed UAV PID Tuning	99
7.5	Distributed UAV Ground Initialization	100
7.6	Distributed Formation State Machine	101
7.7	Distributed UAV Flight Test	102
7.8	Distributed UAV Flight Test Path Log	103

7.9	Distributed UAV Flight Test Distance Log	104
7.10	Distributed UAV Formation Error	104
7.11	Distributed Communication Network	105
7.12	Distributed Program Communication Ground Test Three UAVs	106
7.13	Distributed Program Communication Ground Test Four UAVs	107
7.14	Distributed Communication Network Four UAVs	108
7.15	Distributed Communication Network Four UAVs	109

List of Tables

2.1	Formation Scheme Comparison	10
3.1	Finding Integral Term Result	32
6.1	In-flight Check Triggered Conditions.	85
6.2	Inertial Management Unit Dead Variance Threshold (sam- pling of ten readings).	87
7.1	Distribute Formation Simulation Initialization File	97

Chapter 1

Introduction

1.1 Motivation

Unmanned Aerial Vehicles (UAVs) are popular in many fields such as aerial photography [1], search and rescue operations [2, 3] and even entertainment technology showcase. Multiple UAVs formation is a new research area. Some studies such as quadcopter [4] type UAV formation [5] and vision based UAV formation [6] have been conducted impressively. The advantages of multiple UAVs formation are its mission efficiency and flexibility. However, there are many complex environments that constrain this application such as narrow canyons, forests and even tunnels. The UAVs have to change their formation shapes accordingly to adapt to the constraints in the operating environments. Another remarkable aspect of the distributed formation is that UAVs are not coordinated or controlled by a central station or a spe-

cific UAV but every single UAV is able to carry out path planning by itself. It is safer and more robust compared to a centralized control formation. One example of distributed formation applications is the mission in forest when UAV's data link to ground control station is not stable, and there is a demand of changing flexible formation shapes based on variable terrains.

A coin has two sides, UAV could cause serious injuries and damages to public assets if operated improperly. Local government has to introduce rules to regulate the operations of UAVs in Singapore airspace [7]. Currently, there are few commercial UAV products integrated with safety features. It is necessary to have a regulated safety integration system.

The aim of this study is to develop a distributed UAV system that is capable of performing distance constrained formation with safety considerations. This research can be used in search and rescue, military and etc. with the requirements of formation in complex terrains and environments.

1.2 Objective

This research focuses on UAV distributed formation control and intelligent safety system implementations in narrow airspace terrain environment such as canyon. On-line trajectory generation and real time in air communication

are the main developments of the distributed formation control. Besides, UAV system and area safety implementations are conducted in the study to avoid crash due to system failure during formation flight.

1.3 Contributions

The contributions of this research are listed as follows:

- For distributed multiple UAV formation, we firstly develop a Matlab simulation program to validate the algorithm performances of proportional controller only. Secondly, a proportional-integral controller is designed to eliminate steady state errors in our distributed formation algorithm. Besides, a standalone embedded program is developed in on-board autopilot computer together with the algorithm implementations, safety precautions and etc. In order to link the distributed algorithm program module and be compatible with the system, several configurations have been investigated and developed. The consideration of data link latency has been taken in the program to compensate communication delays due to the procedure of encoding and decoding phases. Simulations and ground tests are carried out to demonstrate that the distributed formation is working properly.
- We develop an on-board safety system to handle different failures during flights for distributed formation flight. Air traffic is a main safety

consideration for outdoor UAV formation. The mission of our UAVs must not influence main civil and military air traffic system. Another important safety concern is UAV's reliability, this includes avionics stability, program robustness, anti-collision reliability and etc. Ground and flight tests have been conducted to verify these safety logics and algorithms.

1.4 Organization

This report is organized as following: Chapter 1 briefs the motivation, objective of the research, and contributions of the thesis. Chapter 2 is the literature review of related research topics on distributed formation and UAV safety system. Chapter 3 discusses the detailed algorithms development and validations followed by Matlab simulation experiment. And Chapter 4 explains the platform design and detailed consideration for the research. Moreover, Chapter 5 contains the information about autopilot program architecture and development. The comprehensive safety system is introduced in Chapter 6. Chapter 7 describes the integration of the platform with the system and experiments of the distributed formation. Eventually, Chapter 8 concludes the thesis and future works.

Chapter 2

Literature Review

Many researches have been studied for multiple UAVs formation control in recent years. The main goal of formation flight is to achieve the formation pattern as and when desired [8]. A nonlinear model predictive control (MPC) framework has been used in designing a collision-free formation flight controller for UAV in [9]. The framework considers input saturation and state constraints, and virtual reference point method is used to determine the formation configuration. The problem of leader-follower formation of quadrotors has been discussed in [10]. The authors propose a distributed control structure to ensure consensus of multiple UAVs and avoidance of collision. In [11], a distributed estimator is implemented to reduce the limitation of virtual structure based formation. Based on information exchanged in local situation, each agent could establish the desired reference. Close target reconnaissance by a formation of three UAVs is discussed in [12]. The

task includes avoiding obstacles, avoiding collisions between each other, approaching closely to a specified target position, and forming an equilateral triangular formation pattern. An acyclic rigid formation moving in a two-dimensional space has been discussed in [13].

Distributed formation is a rich research field including theories of graph and flight control, anti-collision, risk assessment and etc. Thereby, there have been a series of research contributions in distributed formation. In the first decade of twenty-first century, the researches on distributed formation have been mainly focused on mathematical theories [14, 15] and simulations [8, 12, 16–18]. Recently some real distributed flight formation publications [5, 6] have been realized, thanks to the decreasing cost of inertial management units, and size miniaturization of embedded systems [19, 20]. Distributed formation has been extended from military applications to civilian applications.

2.1 UAV Formation Configuration

UAV formation flight can be categorized on configuration basis: centralized and distributed formation.

2.1.1 Centralized Formation

The formation is commanded by a centralized element such as a base station computer, a central robot, etc. The advantage of this configuration method is to unify the control signal to each UAV in such a way that there is much less latency existing due to inter-UAV communications. Based on applications, there are different commands sent to each UAV. In [21], a model predictive control formation flight is proposed via centralized configuration. Leveraging on a motion capturing system, the formation is able to be achieved in an indoor environment. Desired attitude control signals are sent to each UAV by radio communication devices. However, due to high dependency of links between centralized element and UAVs, the formation lacks of reliability when links are not stable. Besides, centralized configuration requires a high performance central unit, which may not be possible in real outdoor situations.

2.1.2 Distributed Formation

Distributed configuration formation [16,22,23] is unit-independent. Namely, there is no centralized unit. Each UAV carries an independent formation controller. There appear several advantages of a distributed formation. Firstly, this configuration is able to be very flexible and more applicable to real-life complex terrains such as narrow canyons, tunnels, forest and etc. Secondly, a distributed formation needs no central unit, this leads to a possibility of

compact UAV formation flight with no base station involved. Thirdly, with the property of being unit-independent, it is less complicated to implement safety mechanisms since each UAV could get its neighbors' information.

2.1.3 Formation Configuration Selection

Comparing centralized and distributed formation configurations, the distributed formation is chosen because of its flexibility and implementation convenience. Moreover, the safety system is easy to be developed and realized due to the acknowledged information between UAVs such as location coordinates, health status and etc.

2.2 UAV Formation Control Schemes

A survey of formation flight by Scharf [24] lists several approaches of formation flying, namely Leader/Follower, Virtual Structure, and Behavioral.

2.2.1 Leader/Follower

Leader/Follower scheme is a hierarchical structure method. UAVs will be assigned with different roles. One UAV in formation is treated as the leader while others are followers. Some will track the information of the leader in the formation pattern such as GPS coordinates, headings, ranges and etc. There are many scenarios of tracking mechanism. Researches [25–27] are

conducted based on application criteria.

2.2.2 Virtual Structure

By means of forming a virtual geometric shape, each UAV is considered as an element of the whole structure, virtual structure control scheme formation is realized without hierarchy structure. Previous researchers focused on satellite constellation formations [28–30] since these kinds of applications are largely dependent on position and neighboring shapes. Research in [29] describes an approach of controlling satellite formation adaptively to achieve distance constrained formation. A constellation template is defined to allow satellites to form the virtual structure with designated positions and heading information. In [31], a general controller application of virtual structure approach is discussed. Following each individual trajectory or maintaining the formation group are not able to change in this research.

2.2.3 Behavioral

Behavioral formation is somewhat like what happens in bionics. UAVs in this formation are required to achieve a common target or goal. Collision avoidance, flocking and obstacle avoidance mainly belong to behavioral formation [32, 33]. Recently in [5] and [6], quad-copter platforms are implemented to verify behavioral formation. This schedule requires the entire formation team to have a consensus target. Therefore, robust communica-

tion links are mandatory to ensure real-time formation transients and failsafe reactions.

2.2.4 Formation Scheme Selection

After reviewing these formation schemes, it can be obtained that there are some paradigms that belong to multiple formation schemes. Differentiations and comparisons are necessary to make a wise scheme selection. Table 2.1 summaries advantages and disadvantages of listed formation schemes.

Table 2.1: Formation Scheme Comparison

	Advantage	Disadvantage
Leader Follower	Easy implementation	not flexible and no failure redundancy
Virtual Structure	Formation flexibility, more applicable to real world case	high communication dependency and requirement
Behavioral	more natural and intelligent way	need to consider formation stability carefully

From Chapter 1, the objective of this thesis aims to achieve distance constrained distributed formation. Consequently, our research belongs to virtual structure scheme since limitation of distance would fix the shape of formation if the final structure is rigid.

2.3 Distributed Multiple UAV Formation

There are many research contributions of distributed formation. In [34], a distance constrained distributed formation algorithm is implemented. To form a rigid formation shape, a proportional controller is introduced into the formation tracking path controller. The stability of the formation error dynamics with velocity inputs is proved. There is a reduction of the formation error as the proportional gain increases. Discussions and implementations are given in Chapter 7 to establish a verification experiment on the theory of this paper. A new technique that applies complex Laplacian is introduced in [35] to solve the problems of formation shapes specified by inter-agent relative positions and increase global stability with distributed control. Based on the target formation shape, they develop a distributed linear control law based on Laplacian.

Researches are also conducted to handle some practical issues. [18] is an application type algorithm. The agents are with limited communication capabilities. [6] has implemented an application using vision methodology. This is the first UAV distributed formation application with a vision capable system in GPS denied environments. Indoor and outdoor flight tests have been illustrated in this paper. In [17], a particular surveillance task is carried out three UAVs with formation control. An equilateral triangle formation pattern is formed to optimize the location estimation accuracy of

targets. They constrain the turning radius of the flight paths with respect to the center of mass of the formation.

In this research, we shall integrate and customize the algorithm in [34] with a proportional controller to generate a robust formation velocity vector command for the UAV. In our system, GPS data are used for localization, therefore, a transformation of coordinate map projection is mandatory. We apply the azimuthal equidistant projection theory in [36]. More details are discussed in Section 3.1.2.

2.4 UAV Safety

Since this research is concerned with outdoor multiple UAVs formation, safety issues become important. Due to the dangers of injuring people and on-board components' failure, several research papers are published focusing on the safety issue of UAV. Article [7] studies the problem of the UAV safety in Singapore, and as the result, Singapore government introduces a law to regulate UAV flights in local airspace. In [37], discussions are provided on accidents caused by human-piloted aircraft. An analysis of risk assessment is studied with the failures of possible factors.

[38] indicates that it may be possible to operate small UAVs with few opera-

tional and size restrictions over the majority of United States. However, the restrictions and limitations for the vehicles must be enforced due to the high risk of ground impactions and midair collisions. Moreover, a detailed analysis of mid-air collision hazards are made so that the possibility of potential accidents could be measured.

2.5 Summary

To sum up this chapter, several current research contributions of distributed UAV formation with distance constrained have been listed and reviewed. Firstly, two formation configurations have been compared. Distributed formation control is chosen on account of its system flexibility. Then, the formation control schemes including leader/follower, virtual structure and behavioral have been discussed. In addition, we have reviewed works accomplished on distributed formation algorithms and implementations. Lastly, UAV safety considerations have been emphasized which will be useful to our safety system designs and implementations.

Chapter 3

Distributed Formation Control Algorithms

3.1 Preliminary

We aim to perform UAV formation by using virtual structure configurations while maintaining the inter-UAV distances. In [34], an investigation of UAVs formation with single integrator dynamics to track a reference velocity vector is made. By introducing a proportional gain, the formation steady-states error can be regulated.

3.1.1 Graph Theory and Rigidity

The concept of graph is important in our UAV formation research. In this thesis, a discussion of graph stability of formation shape is conducted to

ensure that the pattern is safe and unique.

Define an undirected graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ with vertices $\mathcal{V} = \{v_1, \dots, v_n\}$ and edges $\mathcal{E}$. An edge in this graph is denoted as $\{i, j\}$ where i and j are unordered and distinct nodes. Adjacency of i and j happens if $\{i, j\} \in \mathcal{E}$

Distance constrained formation control depends heavily on graph rigidity theory. Some important concepts and definitions of graph rigidity theory, shall be reviewed. Define $x = [x_1^T, \dots, x_n^T]^T \in \mathbb{R}^{2n}$ as a finite points set, where x_i and $x_j \in \mathbb{R}^2$ and $x_i \neq x_j$ for all $i \neq j$. Framework $\mathcal{G}(x)$ is an undirected graph, i in the graph is mapped to the point. The edge vector in graph can be denoted as $e = [e_1^T \dots e_m^T]^T \in \mathbb{R}^{2m}$, where e is the edge vector which is obtained from $e_k = x_j - x_i$. Another important factor is the edge function which is defined as follows:

$$F(x, \mathcal{G}) = [\|e_1\|^2, \dots, \|e_m\|^2]^T \quad (3.1.1)$$

Then, rigidity matrix $R(x)$ in graph $\mathcal{G}(x)$ is given by

$$R(x) = \frac{\partial F(x, \mathcal{G})}{\partial x} \in \mathbb{R}^{m \times 2n} \quad (3.1.2)$$

which is the Jacobian matrix of the edge function F .

If the i -th undirected edge is $\{j, k\}$, then the i -th row of $R(x)$ is defined

as

$$R(x) = \begin{bmatrix} \dots 0 \dots & x_j^T - x_k^T & \dots 0 \dots & x_k^T - x_j^T & \dots 0 \dots \end{bmatrix}$$

A short calculation shows that $R(x)$ can be equivalently written as

$$R(x) = \text{diag}(e_i^T)(E^T \otimes I_2) \quad (3.1.3)$$

where e_i is the edge vector and defined as $e = [e_1^T \ \dots \ e_m^T]^T \in \mathbb{R}^{2m}$, E is the relationship representation matrix between two nodes in the graph framework.

After necessary definitions, graph rigidity theories are described as:

Lemma 3.1.1. *A framework $\mathcal{G}(x)$ is infinitesimally rigid if and only if $\text{rank}(R(x)) = 2n - 3$. [39]*

It is known that MIR framework has $2n - 3$ edges, hence the rows of the rigidity matrix $R(x)$ must be $2n - 3$. In this case, 3 UAVs are formed equilateral triangular shape in distributed formation and hence $2n - 3 = 3$.

$$\begin{bmatrix} \delta_{12} & \delta_{13} & 0 \\ \delta_{21} & 0 & \delta_{23} \\ 0 & \delta_{31} & \delta_{32} \end{bmatrix} = \begin{bmatrix} x_1 - x_2 & x_1 - x_3 & 0 \\ x_2 - x_1 & 0 & x_2 - x_3 \\ 0 & x_3 - x_1 & x_3 - x_2 \end{bmatrix} \quad (3.1.4)$$

Hence it can be observed that the rank of rigidity matrix $R(x)$ is 3 in equation (3.1.4).

Remark 3.1.1. *if x_1 is not equal to x_2 , x_2 is not equal to x_3 , and x_1 is not equal to x_3*

3.1.2 Azimuthal Equidistant Projection

The onboard position outer loop controller uses local tangent plane (LTP) or North East Down (NED) coordinate system [40]. A coordinate transformation from global to local NED reference is necessary. Azimuthal equidistant projection [36] is a map geographic projection method which is adopted in this autopilot program.

Figure 3.1: Global Coordinates Before Azimuthal Earth Projection

Figure 3.2: Projected Coordinates After Azimuthal Earth Projection

As shown in Figure 3.1 and 3.2, azimuthal equidistant projection projects all coordinates on the map with proportionately correct distances to the center reference point. Besides, azimuth (direction) of coordinates on the map, are corrected on from the center reference point. The expression is

shown below:

$$x = \frac{c}{\sin c} \cos \phi \sin (\lambda - \lambda_0) \quad (3.1.5)$$

$$y = \frac{c}{\sin c} [\cos \phi_0 \sin \phi - \sin \phi_0 \cos \phi \cos (\lambda - \lambda_0)] \quad (3.1.6)$$

where ϕ_0 is reference latitude, λ_0 is reference longitude to transfer global position coordinate (ϕ, λ) (*latitude, longitude*) to projected local coordinate (x, y) . c is the angular distance from the center and defined as

$$c = \arccos (\sin \phi_0 \sin \phi + \cos \phi_0 \cos \phi \cos (\lambda - \lambda_0)) \quad (3.1.7)$$

3.2 UAV Modeling

Let's consider n ($n \geq 2$) UAVs in the formation, which are modeled as point masses. Assume they are moving in a pre-defined 2-dimensional space. Hence the expression

$$\dot{x}(t) = u_i(t) \quad i = 1, \dots, n \quad (3.2.1)$$

where $x_i(t) \in \mathbb{R}^2$ is the position of the i -th UAV and $u_i(t)$ denotes the control input signal.

3.3 Positive Potential Function

d_k is defined as k_{th} edge's desired distance between UAV i and j . The set vector $d = [d_1^2 \dots d_m^2] \in \mathbb{R}^2$ is denoted as distance constrained vector.

Then distance error δ is defined as the value differences between desired inter-agent and measurement relative distances, we have

$$\delta_k = \|e_k\|^2 - d_k^2, \quad k \in \{1, \dots, m\} \quad (3.3.1)$$

where e_k is the relative edge vector.

Therefore, positive potential function can be used to monitor formation distance error which is defined as below:

$$\Phi(e) = \frac{1}{2} \sum_{k=1}^m (\|e_k\|^2 - d_k^2)^2 = \frac{1}{2} \sum_{k=1}^m \delta_k^2 \quad (3.3.2)$$

It can be observed that $\Phi(e) = 0$ if and only if $e_k^2 = d_k^2$. This means there is no steady state error in distributed formation control and the result of final shape is perfectly matched.

3.4 Distributed Formation Controller

Taking gradient of potential equation (3.3.2), the control input signal for each UAV in distributed formation, can be derived.

$$u_i = -\nabla_{x_i}(e) = -\sum_{j \sim i} (\|e_k\|^2 - d_k^2) e_k \quad (3.4.1)$$

To show the closed loop dynamics in state space expression, we have

$$\dot{x} = -R(x) R(x)^T x + R(x)^T d \Rightarrow \dot{x} = -R(x)^T (R(x) x - d) \quad (3.4.2)$$

From rigidity matrix definition in equation (3.1.3), it can be observed that the expression $R(x)x - d$ is the measured error distance δ .

$$\delta = R(x)x - d = \text{diag}(e_i^T) (E^T \otimes I_2) x - d \quad (3.4.3)$$

Differentiating equation (3.4.3),

$$\dot{\delta} = 2\text{diag}(e_i^T) \dot{e} = 2\text{diag}(e_i^T) (E^T \otimes I_2) \dot{x} \quad (3.4.4)$$

Combining equation (3.4.4) with (3.1.3) and (3.4.2), the following $\dot{\delta}$ expression can be obtained.

$$\dot{\delta} = f(\delta) = -2R(x)R(x)^T (R(x)x - d) = -2R(x)R(x)^T \delta \quad (3.4.5)$$

Define control signal with introducing stabilizing controller $C(R(x)x - d) = C(\delta)$ and combine with equation (3.2.1)

$$u = -R(x)^T C(R(x)x - d) \quad (3.4.6)$$

3.4.1 Distributed Formation with Proportional Controller

Proportional controller is implemented from equation (3.4.6)

$$u = -R(x)^T (k_p I_n) (R(x)x - d) \quad (3.4.7)$$

where k_p is a scalar constant.

Introduce an external velocity into controller,

$$\dot{x} = u + Bv_{ref} \quad (3.4.8)$$

Substitute equation (3.4.8) into (3.4.7),

$$\dot{x} = -R(x)^T (k_p I_n) (R(x)x - d) + Bv_{ref} \quad (3.4.9)$$

Eventually, by combining equation (3.4.3) and (3.4.9), the proportional controller for distributed formation, can be obtained.

$$\dot{\delta} = f(\delta, v_{ref}) = -2k_p R(x) R(x)^T \delta + 2R(x) Bv_{ref} \quad (3.4.10)$$

In this final proportional controller, k_p is the introduced proportional gain. $\mathfrak{R}(x) = R(x) R(x)^T$ is defined as rigidity matroid. δ is the error. In this thesis, the last term for the final implementation of the autopilot $2R(x) Bv_{ref}$ is set to zero during the flight test due to area limitation. B is to decide which UAV will be set external reference velocity. Therefore, in the final flight experiment, formation will be formed based on distance constrained basis and there is no movement of the whole formation structure.

Remark 3.4.1. *If UAV_i is the reference UAV, we have i_{th} block of B is I_2 , and the remaining blocks are zero.*

The controller structure is as shown in Figure 3.3. With position error δ and reference velocity vector v_{ref} as input, the expansion with variables of proportional gain k_P is shown as “Controller” block in figure, $R(x)$ as rigidity matrix, and $B \in \mathbb{R}^{2n \times 2}$ is to indicate which UAV in the formation may receive external velocity reference. $v_{ref} \in \mathbb{R}^2$ is the reference velocity. In Figure 3.3, d and x are desired and real position for UAV accordingly. UAV’s velocity is defined as $\dot{x}$.

Figure 3.3: Distributed Formation Controller

The algorithm (3.4.10), is modified to fulfill the requirement in this experiment. Since there are three UAVs, the distances between any two UAVs are defined as e_{12} , e_{13} , and e_{23} . Equations (3.4.11):

$$\dot{\delta}_1 = -2k_P \left(\left(\frac{e_{12}^2 - 4}{e_{12}^2} \right) \delta_{12} + \left(\frac{e_{13}^2 - 4}{e_{13}^2} \right) \delta_{13} \right) \quad (3.4.11a)$$

$$\dot{\delta}_2 = -2k_P \left(\left(\frac{e_{12}^2 - 4}{e_{12}^2} \right) \delta_{21} + \left(\frac{e_{13}^2 - 4}{e_{23}^2} \right) \delta_{23} \right) \quad (3.4.11b)$$

$$\dot{\delta}_3 = -2k_P \left(\left(\frac{e_{31}^2 - 4}{e_{31}^2} \right) \delta_{31} + \left(\frac{e_{32}^2 - 4}{e_{32}^2} \right) \delta_{32} \right) \quad (3.4.11c)$$

Remark 3.4.2. The constant 4 in equation (3.4.11a) to (3.4.11c) is the limitation so that distance of each two UAVs is constrained to $\sqrt{4} = 2$ meters when a formation is reached.

Remark 3.4.3. In the algorithm shown in equations (3.4.11), there is an exception when all UAVs are in the same line, the velocity output of each UAV will always point within this line but not tend to form a triangular shape.

Matlab Simulation

The program is developed using Matlab script. The generated graphs are easy to compare and record. An external reference velocity is set as 5m/s onto UAV 1. The main concept and design of this program is based on the algorithm mentioned in equation (3.4.11). There are two types of figure generated.

- Path of UAVs

This figure will generate the trajectory of each UAV. As shown in Figure 3.4, UAV 1, 2, 3, are indicated in color Red, Green, Blue respectively. Given a 5m/s external velocity on UAV 1 and proportional gain is 1, 2, and 3 in this simulation.

Figure 3.4: Distributed Formation with Proportional Controller of $k_P=1$

Figure 3.5: Distributed Formation with Proportional Controller of $k_P=2$

Figure 3.6: Distributed Formation with Proportional Controller of $k_P=3$

In the figures shown above, the paths of three UAVs in formation are listed. However, it is not very obvious to observe the performance differences. Potential Function is used to compare and decide the desirable k_p value for distributed formation.

- Potential Function

By using gradient control law, one can associate potential function graph with equation (3.3.2). Formation error which is the distance differences between two UAVs can be obtained with potential function. It can be observed in Figure 3.7 that there is a steady state error around 2.5 *meters*. This is due to the lack of integral term to compensate external reference velocity effects.

Figure 3.7: Distributed Formation Potential Result with Proportional Controller of $k_P=1$

Figure 3.8: Distributed Formation Potential Result with Proportional Controller of $k_P=2$

Figure 3.9: Distributed Formation Potential Result with Proportional Controller of $k_P=3$

From the results obtained in this matlab simulation, the distributed formation potential error is observed to decrease with increasing proportional gain. And there exist oscillations when $k_P = 3$. Overshoot will appear

if proportional gain k_P is too large. Steady state error is not able to be eliminated.

3.4.2 Distributed Formation with Proportional Integral Controller

To further increase the controller performance in order to eliminate steady state error eventually, an integral term is added into the distributed formation controller.

$$\dot{\delta}(t) = f(\delta(t), v_{ref}) = -2R(x)R(x)^T(k_P\delta(t) + k_i\delta(\tau)) + 2R(x)Bv_{ref} \quad (3.4.12)$$

where $\delta(\tau)$ is the distance error of last controller loop.

Similarly, the algorithm (3.4.12) is modified to fulfill the requirement in this experiment. Since there are three UAVs, the distances between two UAVs are defined as e_{12} , e_{13} , and e_{23} . We have the equations (3.4.13):

$$\dot{\delta}_1(t) = -2k_P \left(\left(\frac{e_{12}^2 - 4}{e_{12}^2} \right) \delta_{12}(t) + \left(\frac{e_{13}^2 - 4}{e_{13}^2} \right) \delta_{13}(t) \right) + (k_i \otimes I) \delta(\tau) + Bv_{ref} \quad (3.4.13a)$$

$$\dot{\delta}_2(t) = -2k_P \left(\left(\frac{e_{12}^2 - 4}{e_{12}^2} \right) \delta_{21}(t) + \left(\frac{e_{23}^2 - 4}{e_{23}^2} \right) \delta_{23}(t) \right) + (k_i \otimes I) \delta(\tau) + Bv_{ref} \quad (3.4.13b)$$

$$\dot{\delta}_3(t) = -2k_P \left(\left(\frac{e_{31}^2 - 4}{e_{31}^2} \right) \delta_{31}(t) + \left(\frac{e_{32}^2 - 4}{e_{32}^2} \right) \delta_{32}(t) \right) + (k_i \otimes I) \delta(\tau) + Bv_{ref} \quad (3.4.13c)$$

In the above, k_i is the integral controller gain, it is a 3×3 matrix and I is the identity matrix with 2×2 dimension. Besides, the last term in PI controller is the external velocity feed forward term with the form of $\begin{bmatrix} vx_{ref} & vy_{ref} \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$.

In order to eventually eliminate the steady state error, an integral term is introduced into formation controller. In the simulation, 5m/s is used as the reference velocity and it will be inputted in UAV 1.

$$\dot{\delta}_1(t) = -2k_P \left(\left(\frac{e_{1i}^2 - 4}{e_{1i}^2} \right) \delta_{1i}(t) + k_i \otimes \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \delta(\tau) + \begin{bmatrix} vx_{ref} & vy_{ref} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \right) \quad (3.4.14)$$

Searching for Appropriate Integral Term

$$k_i = \begin{bmatrix} a1 & a2 & a3 \\ a4 & a5 & a6 \\ a7 & a8 & a9 \end{bmatrix} \quad (3.4.15)$$

Matrix (3.4.15) is the integral term used in this formation controller.

It comes with countless possibilities. Therefore, a searching algorithm is

designed to find the best matrix value combination. Equation (3.4.16) shows the criteria of the satisfied searching result. Program calculates the output of each combination in one loop and get the means μ and the variance σ of three distances by each two UAV combination. If the result meets the criteria defined in Equation (3.4.16), program will take record. Table 3.1 shows the results of different integral term combinations by searching for minimum steady state error. From the table, the third integral term combination appears to be the best result.

$$(\mu - 4 < 0.002) \cap (\sigma < 0.1) \quad (3.4.16)$$

$$k_i = \begin{bmatrix} 0.1 & -0.8 & 0.1 \\ -0.2 & 0.4 & 0.8 \\ 0.5 & 0.5 & 0.05 \end{bmatrix} \quad (3.4.17)$$

as the integral term for the distributed formation controller. Eventually, the final equation is obtained.

$$\dot{\delta}(t) = f(\delta(t), v_{ref}) = -2R(x)R(x)^T(k_P\delta(t) + k_i\delta(\tau)) + 2R(x)Bv_{ref} \quad (3.4.18)$$

After attaining the equation of controller, the code to be filled in matlab is shown below, Kronecker tensor product is used to get the desired matrix dimension.

The main concept and design of this program is based on the algorithm

Table 3.1: Finding Integral Term Result

	1st finding Integral	2nd finding Integral	3rd finding Integral
min dis mean	4.70694245	3.983325017	4.001337965
min dis var	0.717753827	0.112044424	0.081228979
a1	0.1 0.5 0.8	0.05 0.1 0.15	0.1
a2	-0.8 -0.5 0.1 0.5 0.8	-0.82 -0.8 -0.7	-0.8 -0.7 -0.6
a3	-0.5 -0.1 0.1 0.5 0.8	0 -0.1 -0.2	-0.1 0 0.1
a4	-0.5 -0.1 0.1 0.5 0.8	0 -0.1 -0.2	-0.1 -0.2 -0.3
a5	0.1 0.5 0.8	0.4 0.5 0.6	0.3 0.4 0.5
a6	-0.5 -0.1 0.1 0.5 0.8	0.7 0.8 0.82	0.8
a7	-0.5 -0.1 0.1 0.5 0.8	0.4 0.5 0.6	0.3 0.4 0.5
a8	-0.5 -0.1 0.1 0.5 0.8	0.4 0.5 0.6	0.5 0.6 0.7
a9	0.1 0.5 0.8	0.05 0.1 0.15	0.025 0.05 0.075
Time	3.914 * 10e4	1.818 * 10e3	1.997 * 10e2
Result	0.1 -0.8 -0.1	0.1 -0.7 0	0.1 -0.8 -0.1
	-0.1 0.5 0.8	-0.2 0.4 0.8	-0.2 0.4 0.8
	0.5 0.5 0.1	0.4 0.6 0.05	0.5 0.5 0.05

derived above. There are two types of figure to be generated.

- Path of UAVs

This figure will generate the trajectory of each UAV in a preset duration. As shown in Figure 3.10, UAV 1, 2, 3, are indicated in color Red, Green, Blue respectively. Given a 5m/s external velocity on UAV 1, proportional gain is 1 and integral gain is matrix (3.4.17).

Figure 3.10: PI Controller Distributed Formation

- Potential Function

Using gradient control law, the associated potential function graph can be obtained with equation (3.3.2) so that formation error can be achieved. A point to note is that the steady state error is observed to be almost 0. Hence the proportional integral controller eliminates the steady state error eventually.

Figure 3.11: Distributed Formation Potential Result with PI Controller

From the results obtained in this matlab simulation, it is observed that with an appropriate integral term used, the steady state error can be eliminated during distributed formation with external reference velocity.

3.5 Map Projection

The localization sensor is GPS which streams global position data to on-board autopilot in latitude and longitude type. Using the projection method discussed in section (3.1.2), it is required to preset the latitude and longitude of the projection reference. Hence the distance e_{12} of two global coordinates (lat_1, lon_1) , (lat_2, lon_2) is:

$$e_{12} = 2R_{earth} \arctan \left(\sqrt{\frac{\sin\left(\frac{d_{lat}}{2}\right) \sin\left(\frac{d_{lat}}{2}\right) + \sin\left(\frac{d_{lon}}{2}\right) \sin\left(\frac{d_{lon}}{2}\right) \cos(lat_1) \cos(lat_2)}{1 - \sin\left(\frac{d_{lat}}{2}\right) \sin\left(\frac{d_{lat}}{2}\right) + \sin\left(\frac{d_{lon}}{2}\right) \sin\left(\frac{d_{lon}}{2}\right) \cos(lat_1) \cos(lat_2)}} \right) \quad (3.5.1)$$

where d_{lat} and d_{lon} is the latitude and longitude differences between (lat_1, lon_1) and (lat_2, lon_2) .

Remark 3.5.1. *The program initializes the reference point at which UAV is booted up. Therefore, each UAV should have a different set of reference point $(lat_{i,ref}, lon_{i,ref})$ accordingly.*

Therefore the distance between UAV 1 and UAV 2 can be obtained. Besides, in equation (3.4.10), δ is defined as distance error. Consider an example in UAV 1, another two UAVs should be projected into the reference coordinate of UAV 1. Therefore, it is possible to obtain UAV_{*i*} ($i = 2, 3$) coordinates $(x_{i,1}, y_{i,1})$ in the reference axis of UAV 1

$$x_{i,1} = \frac{\arccos(\sin(lat_{i,ref}) \sin(lat_1) + \cos(lat_{i,ref}) \cos(lat_1) \cos(lon_1 - lon_{i,ref}))}{\sin(\arccos(\sin(lat_{i,ref}) \sin(lat_1) + \cos(lat_{i,ref}) \cos(lat_1) \cos(lon_1 - lon_{i,ref})))} (\cos(lat_{i,ref}) \sin(lat_i) - \sin(lat_{i,ref}) \cos(lat_i) \cos(lon_i - lon_{i,ref})) R_{earth} \quad (3.5.2)$$

$$y_{i,1} = \frac{\arccos(\sin(lat_{i,ref}) \sin(lat_1) + \cos(lat_{i,ref}) \cos(lat_1) \cos(lon_1 - lon_{i,ref}))}{\sin(\arccos(\sin(lat_{i,ref}) \sin(lat_1) + \cos(lat_{i,ref}) \cos(lat_1) \cos(lon_1 - lon_{i,ref})))} \cos(lat_i) \sin(lon_i - lon_{i,ref}) R_{earth} \quad (3.5.3)$$

Finally, we can substitute $(x_{i,1}, y_{i,1})$ into δ_{1i}

$$\delta_{1i} = \begin{bmatrix} x_1 - x_i \\ y_1 - y_i \end{bmatrix} \quad (3.5.4)$$

3.6 Summary

In this chapter, the algorithms and logics used in this research, have been discussed. Proportional Controller and Proportional Integral Controller have been studied for distributed formation. A Matlab simulation program has been developed to verify the algorithm performances. Besides, the azimuthal equidistant projection algorithm has been used for local reference position control.

Chapter 4

Development of Distributed Formation Testbed

In order to have precise control of the quadcopters, we need to consider the whole architecture of multicopter autopilot in order to design hardware and avionics system.

4.1 Multicopter Autopilot Architecture

4.1.1 Multicopter Control

To perform a stable multicopter flight, a Pixhawk autopilot [41] is used as the onboard flight controller. It is integrated with three types of Inertial Management Unit (IMU) (Gyroscope, Accelerometer, Magnetometer) to have a better data fusion and estimation. No modification is done on the outer and

inner loop control laws with fine tuned PID parameters.

Figure 4.1 shows system architecture of the autopilot used in distributed formation. With necessary sensor information, state estimation is capable of calculate accurate UAV data such as attitude, velocity, and position. Control system is to generate an appropriate output based on system property to actuator.

Figure 4.1: Multicopter System Architecture

- Outer Loop Controller

This controller has two loops: P loop for position error and PID loop

for velocity error. Output of the controller is the thrust vector that is split into thrust direction. In this experiment, the P loop position controller is skipped and the distributed velocity error is directly linked to the velocity controller input.

- Inner Loop Controller

The controller has two loops: P loop for angular error and PD loop for angular rate error. The desired attitude signal will be inputted to the inner loop controller which will generate scaled pulse width modulation.

4.1.2 Data Fusion

Figure 4.1 shows that data fusion parts also play an important role in the autopilot system.

- Position Estimation

The position estimation module will fuse localization data from several sensors such as: GPS, IMU, Barometer, optical flow etc. and output local estimates for position control. The process is carried out recursively through an inertial filter.

- Correction Signal

The function is shown below. With position and velocity of corresponding variables as output $x[2]$, the correction function will

add in different sensor value e to estimator recursively. Weight coefficient which is the cut-off frequency w will be defined before program starts. This will depend on how important the sensor is.

```
1  void inertial_filter_correct(float e, float dt,
2
3  float x[2], int i, float w)
4
5  {
6
7  if (isfinite(e) && isfinite(w) && isfinite(dt))
8
9  {
10
11  float ewdt = e * w * dt;
12
13  x[i] += ewdt;
14
15  if (i == 0)
16
17  {
18
19  x[1] += w * ewdt;
20
21  }
22
23  }
24
25  }
```

– Prediction Function

With sensor measurement given by accelerometer, prediction of the position and velocity can be done by recursively sum integrating of acceleration into current estimation.

```
void inertial_filter_predict(float dt, float x[2],
float acc)
```

```

2      {
      if (isfinite(dt)) {
4      if (!isfinite(acc)) {
      acc = 0.0f;
6      }
      x[0] += x[1] * dt + acc * dt * dt / 2.0f;
8      x[1] += acc * dt;
      }
10     }

```

- Attitude Estimation

Using Extended Kalman Filter, this program is capable of estimating attitudes based on the input of gyroscope, accelerometer, and magnetometer.

4.2 Quadcopter Frame

Before choosing the frame and thrust components, the weight of the UAV has to be defined. By estimation, a payload of around 1.2 - 1.4 kilograms is reasonable to carry enough items for this experiment. The selection of quadcopter frames in this research is detailed below:

4.2.1 Frame Selection

In this research, UAV needs to operate in outdoor environments. Hence, the size and weight of the frame need to be large enough to compensate wind factor. The average wind in Singapore is breeze-like with a speed of about $2.5m/s$. [42] Flame-wheel frame with arm length of 450mm is large enough to compensate the wind and has relative high payload carrying ability that will be reserved for future modifications. Figure 4.4 shows a normal power distribution board which is used to distribute main power from battery to four Electronics Speed Controllers (ESC) which are used to control motor spinning speed. The conducting connectors, cable skin and soldering material take up a significant weight. While for this UAV, the weight of frame is 282 grams and its power cables and distribution board are integrated into the frame's base board. That will reduce the weight for around 50 - 100 grams [43]. As shown in Figure 4.3, power distribution board of flame wheel not only distributes current from battery, but also supports structure strength due to its stiffness.

Figure 4.2: Flame Wheel 450mm Arm Length

Figure 4.3: Flame Wheel 450mm Central Board

Figure 4.4: Normal Power Distribution Board

4.2.2 Motor and Propeller

In order to have a good payload and maneuvering forces for a quadcopter, there is a need to have a high efficiency and a good balance between weight and performance. As discussed in Section 4.2.1, a weight of 1.2 – 1.4 kilograms is reasonable. After comparing different models of motor and propellers, the respective thrust components are finalized as follows: MT2212 KV1100; Propeller type: T-MOTOR 10 * 33 CF

With a 3-cell lithium polymer battery and 10×33 type carbon fiber propellers, the lifting force can reach 360 grams with half throttle input. Hence, UAV will get $360 \times 4 = 1440$ grams thrust with its hovering throttle. It is capable to lift its own weight of 1440 grams with additional payload. There are many choices of motors and propellers combinations.

Item No.	Volts (V)	Prop	Throttle	Amps (A)	Watts (W)	Thrust (G)	RPM	Efficiency (G/W)	Operating temperature(°C)
MT2212 KV1100	11.1	T-MOTOR 8*2.7CF	50%	2.8	31	240	7550	7.74	41
			65%	3.6	40	285	8000	7.13	
			75%	4	43	300	8500	6.98	
			85%	5.2	58	370	9500	6.38	
			100%	6.2	67	420	10000	6.27	
		T-MOTOR 9*3CF	50%	3.5	39	290	6800	7.44	43
			65%	4.6	50	350	7400	7.00	
			75%	5.8	63	410	8000	6.51	
			85%	7.8	85	530	8600	6.24	
			100%	9.2	99	600	9100	6.06	
		T-MOTOR 10*3.3CF	50%	4	45	360	5700	8.00	44
			65%	5.9	65	480	6400	7.38	
			75%	7.7	82	580	7100	7.07	
			85%	10	108	700	7700	6.48	
			100%	11.9	127	780	8300	6.14	
		T-MOTOR 11*3.7CF	50%	4.3	48	420	5300	8.75	45
			65%	6.9	75	580	6200	7.73	
			75%	8.9	97	690	6800	7.11	
			85%	11.5	124	820	7400	6.61	
			100%	13.7	145	940	7800	6.48	
	14.8	T-MOTOR 8*2.7CF	50%	4	59	340	9400	5.76	43
			65%	5.3	79	420	10200	5.32	
			75%	6.2	91	470	11000	5.16	
			85%	8.1	119	590	11800	4.96	
			100%	9.9	144	700	12600	4.86	
		T-MOTOR 9*3CF	50%	5.1	75	450	8200	6.00	47
			65%	6.9	100	560	9000	5.60	
			75%	131	130	670	9800	5.15	
			85%	11.9	172	810	10500	4.71	
			100%	14.3	202	940	11300	4.65	

Notes: The test condition of temperature is motor surface temperature in 100% throttle while the motor run 10 min.

Figure 4.5: Motor-Propeller Performance Sheet

4.2.3 Battery

Since battery is the highest single weight component in a UAV, there is a need to choose its capacity carefully. Taking into account of current consumption in Figure 4.5, it is noted that the hovering throttle current requires 4 Amps. In order to allow the UAV to operate for 1 hour, the battery capacity requires at least $4 \text{ Amps} \times 1 \text{ hour} = 4 \text{ AmpHours}$. Hence the final selection of this platform is 5.3 AmpHours because of its reasonable weight for this mission.

Figure 4.6: Battery

4.2.4 Electronic Speed Controller (ESC)

Electronic Speed Controller receives control signal from autopilot in Pulse Width Modulation data format and converts into Alternative Current (AC) to drive motor spinning. The main factor that needs to be considered is the current that ESC could carry extremely which is defined as burst current. In Figure 4.5 we can observe that the highest current drawn from our selection is 11.9 Amps. Hence, the maximum current that ESC could handle must be larger than this value. Therefore, the final selection of ESC is 20 Amps.

Figure 4.7: Electronic Speed Controller

4.3 Quadcopter Avionics

This UAV consists of a robust avionics system with Global Positioning System (GPS) sensor, two sets of Inertial Management Unit (IMU), two telemetries, power modules etc. which are taken as data input/output as shown in Figure 4.8. After processing outer and inner loop algorithm within the autopilot, the board will eventually generate Pulse Width Modulation (PWM) signal to drive Electronic Speed Controller (ESC) and control signals will reach motors to trigger UAV's maneuvers.

Figure 4.8: Distributed Avionics System

4.3.1 Autopilot

Autopilot is a mandatory device in any UAV system. It controls motors spinning by forming attitude and position close loop controller. This platform utilizes Pixhawk autopilot, as shown in Figure 4.9. It integrates a lot of

interfaces for different types of sensors that may be used onboard. Our main avionics customization add-on is also attached to autopilot with a secondary telemetry to establish UAV to UAV communication.

Figure 4.9: Pixhawk Autopilot

4.3.2 Power Module

In order to support pixhawk autopilot, a power module to support processing power is required. As shown in Figure 4.10, the module not only supplies power for autopilot board, but also senses voltage and current level from battery in analog signal. Information will be passed through Pixhawk and be shown on ground control station and logging program.

Figure 4.10: Pixhawk Power Module

4.3.3 Inertial Management Unit (IMU)

As mentioned in section 4.3, there are two sets of inertial management units in our system.

- Invensense MPU 6000 3-axis accelerometer/gyroscope

Enabled by default, MPU 6000 is a powerful IMU with accelerometer and gyroscope in local frame sensing. This chip is used as the primary accelerometer and gyroscope.

- ST Micro L3GD20H 16 bit gyroscope

L3GD20H is a gyroscope IMU chip integrated into pixhawk autopilot. It is considered as the secondary gyroscope as a backup IMU if first gyroscope in MPU 6000 fails.

- ST Micro LSM303D 14 bit accelerometer / magnetometer

Secondary accelerometer and magnetometer are occupied in this system. A hot swap algorithm will automatically detect and trigger from primary to secondary IMU to maintain safety and robust flight of UAV.

- 3DR uBlox GPS with Compass Kit

Primary compass is integrated in this 3DR uBlox GPS module in Figure 4.11. Since compass is a global scale sensor, it detects magnetic field in the planet. Although it is easily interfered by other Electronic Magnetic (EM) generator devices such as RF module, high frequency circuit etc, it was chosen as it is able to provide a good magnetic field estimation.

Figure 4.11: Pixhawk uBlox GPS with compass

4.3.4 Global Positioning System (GPS)

GPS module provides position information globally, it sends data to pixhawk autopilot in 5 Hz rate when it locks on to satellites. Some preflight check features will be added to ensure that GPS signal is ready to use. This will be detailed discussed in Chapter 5.

4.3.5 Telemetry Communication

The telemetry system shown in Figure 4.12 is used to communicate between UAVs. It is a customized telemetry system configuration and is different from the original telemetry setup with linkage between UAV and ground control station. Considering peer-to-peer feature for this specific telemetry, UAV 1 and 2 will have two telemetries and only 1 telemetry will be connected to UAV system 3. Detailed communication topology will be explained in Chapter 7.

Figure 4.12: Telemetry System

Since this telemetry module is in peer-to-peer configuration, different channels have been assigned so as to avoid frequency device corruption. As shown in Figure 4.13, this GUI shows some parameters of telemetry such as NET ID, frequency, baud rate, number of hopping channels etc. In this project, each link is assigned with a unique NET ID and channel number with other settings remaining same.

Figure 4.13: Telemetry Setup

4.3.6 Remote Control

As a safety priority in a UAV system, there comes with a remote control device normally. Manual control on the UAV has the highest priority no matter which mode that the UAV will be. In this research, we also take advantage of the remote control functions. Our distributed control mode is assigned in remote control transmitter so that man can easily trigger the mode by the transmitter. Remote control receiver will connect to a pixhawk autopilot with S.BUS signal. The signal is digital data with modulation and it can be combined up to 17 remote control channels.

Figure 4.14: Remote Control Receiver

Figure 4.15: Remote Control Transmitter

4.4 Assembling

In Figure 4.16, it can be observed that the tag on UAV's left behind arm is number 1, and there appear two sets of telemetry systems which indicated

in section 4.3.5. Motors are mounted onto each arm and screwed on tightly, GPS module is attached on one of arm to make sure there is no blocking components for GPS signal link and no interference to magnetometer by reason of other avionics devices. There are in total three UAV prototype platforms built in this research based on the UAVs involved in distributed formation experiment.

Figure 4.16: Assembled Testbed for Distributed Formation

4.5 Summary

In this chapter, the design of the testbed platform has been finalized. Firstly, program modules of inner and outer loop, position and attitude data fusion

have been evaluated for a robust estimation and control. Secondly, quadcopter frame has been selected based on the performance required in this research. Followed by several key components evaluations such as motor and propeller, battery, and electronic speed controller have been conducted for fulfilling distributed formation control. Lastly, avionics system has been discussed and explained in this chapter.

Chapter 5

Autopilot Software Development

5.1 Autopilot Architecture

After researching our autopilot used in this project, we summarize the whole system into a diagram to understand easily as shown in Figure 5.1. This autopilot software is built based on NuttX real time operating system. As we all know, Real Time Operating System (RTOS) is a robust solution for industrial and military use. The whole system is modularized, which means we could insert our coding without interfering original logics and algorithms too much. There are mainly six modules that need us to pay attention.

- Commander

Commander Module is the central component of autopilot system, it

Figure 5.1: Autopilot System Architecture

is like the brain of board. It will control all the priority loops such as, sensors' drivers, navigation logics, flight modes, main system modes, failsafe logics etc. In order to execute the distributed formation algorithm, the flight mode of this system was modified. This will be discussed later.

- Mission

Mission module or navigator module is the path generation control program in autopilot system. Using this module, different paths and way-points could be set for UAV to execute. Position setpoints will be generated and sent to outer loop controller to process position controlling. In this implementation, there is a need to bypass this module and overwrite its output with the distributed formation module.

- Outer Loop

Receiving position setpoint data from path generation module such as Mission Module/ Distributed Formation Module, position control will run in this program. There are two layers of position controller: Position Controller and Velocity Controller. Position Controller is a P controller position loop given by a proportional gain. Additionally, Velocity Controller is a PID controller velocity loop. As discussed in Chapter 3, our distributed formation controller will output velocity setpoints. thereby position loop will be skipped in our implementa-

tion. Moreover, the output of this program is attitude setpoint by transforming using rotation matrix of UAV.

- Inner Loop

Inner loop controller, or another name attitude controller, is the mandatory part in UAV control system. With the information of attitude and its setpoint, controller is capable of calculating final throttle commands. There are two loops in this controller: attitude control which is operating in Proportional P controller, and angular rate control which is using Proportional Derivative (PD) controller. Module is operated in 100 Hz to ensure UAV performance.

- Drivers

All the modules mentioned above are logical processing units. Nevertheless, Driver modules are the interfaces between software and hardware. These modules include GPS, IMU, barometer, voltage and amp meter etc. Limited logics and algorithms are occupied for fusion and data transforming. For distributed formation, we need GPS, IMU and barometer information as our sensor references.

- uORB

uORB is the short form of micro object register broker. This is a unique feature in this system. In fact, this is a data bus for module communications. Messages are created in a specific data and file for-

mat which are shown in Figure 5.2. Module will reserve exact stack size for these messages. In our application, message will be established as shown in Figure 5.3. And there are several types of UAV data needed:

Figure 5.2: Micro Object Register Broker Messages

- UAV Position

Positions including latitude, longitude and altitude will be used in our distributed formation algorithm. And they are cached from UAV fused global position.

- UAV Heading

Heading is the compass reading of each UAV. In our research, heading will not change during flight. Hence, UAV will fly as its initial heading.

– UAV Time

Synchronization is very important in our application because the path is generated real-time. Due to hardware and decoding/encoding latency, we have to add this timing of when UAV sends

its position.

– Other Flags

We reserve other flags for other features and functions such as simulation switch, GPS available flag etc.

```
1 # distribute position in WGS84 coordinates.
2 uint64 utc1      # Timestamp1 (microseconds since system boot)
3 uint64 utc2      # Timestamp2 (microseconds since system boot)
4 uint64 utc3      # Timestamp3 (microseconds since system boot)
5 uint64 utc4      # Timestamp4 (microseconds since system boot)
6
7 float64 lat1     # UAV1 latitude in degrees
8 float64 lon1     # UAV1 longitude in degrees
9 float32 alt1     # UAV1 Altitude in meters (AMSL)
10 float32 head1    # UAV1 heading in degrees
11
12 float64 lat2     # UAV2 latitude in degrees
13 float64 lon2     # UAV2 longitude in degrees
14 float32 alt2     # UAV2 Altitude in meters (AMSL)
15 float32 head2    # UAV2 heading in degrees
16
17 float64 lat3     # UAV3 latitude in degrees
18 float64 lon3     # UAV3 longitude in degrees
19 float32 alt3     # UAV3 Altitude in meters (AMSL)
20 float32 head3    # UAV3 heading in degrees
21
22 float64 lat4     # UAV4 latitude in degrees
23 float64 lon4     # UAV4 longitude in degrees
24 float32 alt4     # UAV4 Altitude in meters (AMSL)
25 float32 head4    # UAV4 heading in degrees
26
27 bool team_gps_loss # UAV team gps flag
28 bool disforSIM_enabled # distribute formation simulation mode flag
29
```

Figure 5.3: Distributed Formation uORB Message

5.2 Distributed Formation Control

5.2.1 System Architecture Modification

The original architecture of the autopilot system as shown in Figure 5.1 needs to be modified to realize distributed formation. Mission module is replaced by Distributed Formation Module. Velocity setpoint will be sent

from distributed formation module and behave as the input of outer loop module for position control.

Figure 5.4: Autopilot System Architecture for Distributed Formation

Three files are established in distributed formation module:

- `distribute_formation_main.cpp`

This C++ source file is the main program of our algorithm and logic operations.

- `distribute_formation_params.c`

This is the parameter management program which stores and manages parameters during algorithm operating.

- module.mk

In our autopilot system, module.mk is a compulsory file for module establishment. It indicates stack size reserved for this module and which files are going to participate in compiling. In this case, distribute_formation_main.cpp and distribute_formation_params.c will be added in this file.

Figure 5.5: Distributed Formation Control Program File List

5.3 Main Function Realization

5.3.1 Coordinate Projection

Before calculating velocity setpoint by the distributed formation algorithm, we have to unify the coordinate reference information. As discussed in Chapter 3, we need to transform the geographic coordinate system into the az-

imuthal equidistant plane for position control in autopilot. Each UAV will have its own origin point when UAV first receives correct GPS signal. Hence, we will not transfer local azimuthal equidistant coordinates but global position for unifying all the reference systems.

There are two steps for this transformation:

- Map Initialization

This will initialize necessary data format preparations for the transformation later.

```
ref->lat_rad = lat_0 * MDEG_TO_RAD;  
2  ref->lon_rad = lon_0 * MDEG_TO_RAD;  
  
ref->sin_lat = sin(ref->lat_rad);  
4  ref->cos_lat = cos(ref->lat_rad);  
  
ref->timestamp = timestamp;  
6  ref->init_done = true;
```

- Map Projection

This is the transformation process from the global reference system to the UAV local reference system. Hence, for different UAVs, the local projected coordinates should not be the same. This is due to the initial position offset when a UAV received first good GPS signal. Code is shown below. We can obtain x and y of a local UAV position.

```

double lat_rad = lat * MDEG.TORAD;
double lon_rad = lon * MDEG.TORAD;
double sin_lat = sin(lat_rad);
double cos_lat = cos(lat_rad);
double cos_d_lon = cos(lon_rad - ref->lon_rad);
double c = acos(ref->sin_lat * sin_lat + ref->cos_lat *
    cos_lat * cos_d_lon);
double k = (fabs(c) < DBLEPSILON) ? 1.0 : (c / sin(c));
*x = k * (ref->cos_lat * sin_lat - ref->sin_lat * cos_lat
    * cos_d_lon) * CONSTANTS.RADIUS_OF_EARTH;
*y = k * cos_lat * sin(lon_rad - ref->lon_rad) *
    CONSTANTS.RADIUS_OF_EARTH;

```

5.3.2 Distributed Formation Controller

Referring to algorithm derived in 3.4.11, we could eventually transform to source code used in autopilot system. The code shown below is an example of UAV 1. Difference between different distributed formation controller is the distance used in the equation. UAV 1 will use distance between 1,2 and 1,3; UAV 2 will use distance between 2,1 and 2,3; UAV 3 will use distance between 3,1 and 3,2. Since we take the square of distance in calculation, therefore, it is not a vector.

```

1  _pos_sp_triplet.current.vx = (-0.2f * ((sqdis12 - 4)/sqdis12 *
    (_local_pos.x - pos2in1[0]))+(sqdis13-4)/sqdis13*( _local_pos
    .x-pos3in1[0])));
3  _pos_sp_triplet.current.vy = (-0.2f * ((sqdis12 - 4)/sqdis12 *
    (_local_pos.y - pos2in1[1]))+(sqdis13-4)/sqdis13*( _local_pos
    .y-pos3in1[1])));

```

5.3.3 Safety and Precaution

It takes plenty of time to make the program robust and stable. There exist several potential errors that make our program not work.

- Neighbor UAV GPS is not valid

There is a case when communication link is publishing GPS data from neighbor UAVs but values are zero. This is because the data bus we used (uORB) will initialize all the memory data to zero when program starts. If specific messages have no update, the value will always be zero. Consequently, we collect experiment data to get the correct GPS value range in our testing environment. Algorithm will validate neighbor GPS information.

```

2  bool valid;
    switch(disid)

```

```

{
4  case 1:
    {
6      valid = (((double_abs(disfor_check.lat2) > 0.8) && (
          double_abs(disfor_check.lon2)) > 103.0) ? true : false);
      *error_id += (valid == true) ? 0: 10;
8      valid = (((double_abs(disfor_check.lat3) > 0.8) && (
          double_abs(disfor_check.lon3)) > 103.0) ? true : false);
      *error_id += (valid == true) ? 0: 1;
10     }
    break;
12  case 2:
    {
14      valid = (((double_abs(disfor_check.lat1) > 0.8) && (
          double_abs(disfor_check.lon1)) > 103.0) ? true : false);
      *error_id += (valid == true) ? 0: 100;
16      valid = (((double_abs(disfor_check.lat3) > 0.8) && (
          double_abs(disfor_check.lon3)) > 103.0) ? true : false);
      *error_id += (valid == true) ? 0: 1;
18     }
    break;
20  case 3:
    {
22      valid = (((double_abs(disfor_check.lat2) > 0.8) && (
          double_abs(disfor_check.lon2)) > 103.0) ? true : false);
      *error_id += (valid == true) ? 0: 10;

```

```

24     valid = (((double_abs(disfor_check.lat1) > 0.8) && (
        double_abs(disfor_check.lon1) > 103.0) ? true : false);
    *error_id += (valid == true) ? 0: 100;
26 }
    break;
28 default:
    {
30     valid = false;
        warnx("Disfor: not support uav id");
32     }
    break;
34 }

36 return valid;

```

- Wrong distributed formation calculation

In spite of the correct data from neighbor UAVs, we still need to pay attention to the accuracy of our algorithms. Solution used in our program is to saturate output velocity setpoint to $1m/s$

```

        _pos_sp_triplet.current.vx = (_pos_sp_triplet.current.vx
            > 1.0f || _pos_sp_triplet.current.vx < -1.0f)
2    ? (_pos_sp_triplet.current.vx / (float_abs(
        _pos_sp_triplet.current.vx))) : _pos_sp_triplet.current.

```

```

        vx;

        _pos_sp_triplet.current.vy = (_pos_sp_triplet.current.vy
        > 1.0f || _pos_sp_triplet.current.vy < -1.0f)
4    ? (_pos_sp_triplet.current.vy / (float_abs(
        _pos_sp_triplet.current.vy))) : _pos_sp_triplet.current.
        vy;

```

5.4 Distributed Formation Simulation

We develop this program for UAV to simulate the result of distributed formation algorithm. As shown in Figure 5.6, the distributed formation simulator will switch signal from position estimator, communication link to itself. Position estimator is a fusion module to provide reliable global position data, meanwhile communication link is a data sharing module with the formation information from neighbor UAVs.

Program will load a `distribute_sim.txt` file in autopilot SD card onboard. It will automatically wrap the information in this file which includes neighbor UAVs' fake information. And this will simulate initialization of real distribute formation startup information. Besides, in order to give UAV a fake response of current UAV movement, we need to simulate the trajectory of the path. Hence we also link the output of distribute formation controller

to this simulator. And the velocity setpoint will be added to current UAV position, thereby we can simulate the movement of the current UAV.

Figure 5.6: Distribute Formation Simulator Diagram

5.5 Signal and Program Interfacing

Interfacing is an important factor in this research. Since the code of this autopilot system is open source, we need to customize a lot to fulfill our requirements for testing.

5.5.1 Flight Mode

The original autopilot source code comes with a structure of flight mode to ensure a safety and reliable system state machine. In Figure 5.7, we can observe the states of our autopilot system. A special mode called offboard mode is used. The purpose of this mode is to establish the control command from other onboard computer. Likewise distributed formation, position or

velocity setpoint in offboard mode are controlled by parameters defined in autopilot. Hence, it is a good choice to modify this flight mode.

Figure 5.7: UAV Flight Mode Diagram

5.5.2 Communication Link

Another key interfacing part is the links between UAVs. To establish these links, we need to modify the communication program in autopilot system. A protocol called MAVLink is used in this research. It is a light weight transmission protocol specially for UAV or small robot. uORB data bus will interface to MAVLink protocol in such way that UAV could receive and send its desired data through uORB data bus directly.

Figure 5.8: MAVLink Communication Interface

5.6 Summary

In conclusion, we discussed the autopilot system architecture in our distributed formation control research. To validate the functionality, we have modified the system architecture and data bus. Besides, the developments of safety and algorithm have been conducted to make our code robust.

Chapter 6

UAV Safety Implementation

As is known, UAV flight needs a pilot to standby just in case when it is out of control. In our research, we deploy three UAVs each with online path generation. Due to the complexity of formation patterns, it is difficult to control all the UAVs with one pilot. Pilot will be confused if two or more UAVs fly together. Our formation configuration needs only one pilot standing by and switching to landing mode or disarm if the UAVs perform unexpectedly. We need to have a robust UAV system and failsafe algorithm to achieve this. This chapter is concerned with algorithm and concept development and the design of a safety system for a UAV. Detailed discussion of safety considerations of distributed platform is conducted in Chapter 7.

The safety system has four main functions: Pre-flight check, In-flight check, Inertial Management Unit Online Swap (IMU online swap), and Real-

time Ground Control Station (GCS) health check. Pre-flight check, In-flight check, and IMU online swap are integrated into the low level autopilot since it is more stable than the upper level autopilot. GCS health check is a graphic user interface (GUI) as shown in Figure 6.9 to display feature for the operator to examine the real-time status of the system on-board. The system was tested in an outdoor UAV centralized formation application.

6.1 Hardware Platform

The hardware platform for safety implementation is shown in Figure 6.1. The dimension of this UAV is 114 cm with 18-inch propellers mounted. Total weight is 4.6 kilograms. The flight speed is up to 12 meters/second.

Figure 6.1: Hardware UAV System

6.2 System Architecture

The avionic system of the UAV uses two autopilot systems that are connected in serial. The upper level autopilot is used for high performance path planning and outer loop position control. It sends attitude setpoints to the low level autopilot to process inner loop attitude controls at 50 Hz. The low level autopilot has a built-in outer loop position control which will only be triggered when the fail-safe system is activated. The whole system has two Global Positioning System (GPS) modules which are connected to each autopilot. There are two wireless data-links, namely, Ground Control Station (GCS) link for main command issuing and receiving and Remote Controller (RC) data link for pilot manual control and safety control.

6.3 Preliminary

With the increasing popularity of UAV, safety of UAV draws more attentions in the world. Several researches have been conducted on different critical hazards of UAV such as midair collision and ground impact [37] and [38].

6.3.1 Fault Tree Analysis

Fault tree analysis (FTA) is a top down, deductive failure analysis in which an undesired state of a system is analyzed using Boolean logic to combine a series of lower-level events. This analysis method is mainly used in the

Figure 6.2: System Architecture

fields of safety and reliability engineering to understand how systems can fail, identify the best ways to reduce risk or determine event rates of a safety accident or a particular system level (functional) failure.

6.3.2 Onboard Autopilot Sensors

It is mandatory to research on the sensors on autopilot before implementing a safety system. Most of the serious problems happened because of sensor failure. The autopilot we are using is Pixhawk Autopilot [41]. This is a high performance autopilot with Cortex M4F 168MHz CPU onboard, there are three Inertial Management Units (IMU) onboard: MPU6000, LSM303D,

Figure 6.3: Fault Tree Analysis

and L3GD20 which are detailed introduced in Chapter 4.

- MPU6000 is a three axis Gyroscope and Accelerometer digital IMU.
- LSM303D is a system-in-package featuring a 3D digital linear acceleration sensor and a 3D digital magnetic sensor.
- L3GD20 is a low-power three-axis angular rate sensor.

From the information above, we can observe that there are two gyroscopes and two accelerators in pixhawk. Hence, a function of swapping IMU is possible. In pixhawk's original firmware, MPU6000 is defined as the primary gyroscope and accelerometer, but the gyroscope in L3GD20 and the accelerometer in LSM303D are not used in firmware although they are

Figure 6.4: Pixhawk

activated when system boots up.

6.4 Pre-flight Check

Before flight, the UAV on-board system will check both the mechanical and avionic system. Firstly, when the system is armed, the four motors will spin at a slow speed to let the pilot confirm their functionality. Secondly, the pre-flight check program in the low-level auto-pilot system will check all the sensors' health situation before flight. [44]

Figure 6.5: Preflight Check Sequence

6.4.1 Sensor Stacks Checked during Pre-flight

The program will start data processing after 60 seconds of autopilot's boot-up. This is to warm up all the on-board sensors including IMU and GPS. There are five kinds of flight measurements examination before each UAV takes off:

1. System Horizon

UAV horizontal inspection before take-off is very important and we need to ensure that on-board autopilot IMU's reading is correct without drifting and incline attitude reading due to calibration loss, Electromagnetic Interference (EMI), and improper UAV setup. In order to filter instant reading errors, the program will capture 60 samples with

a constant time interval, and calculate its mean value recursively. If the mean value is larger than a pre-defined threshold (20 degrees in our case), horizontal inspection is considered as a failure.

2. Accelerometer

Accelerometer readings check is achieved by calculating the variances of three axis (X, Y, and Z) accelerometer readings. Normally accelerometer's default value is rarely interfered by outside electromagnetic wave. However, due to hardware quality and other problems, the accelerator's value may have large variances. Therefore, we need to calculate acceleration variance recursively to conduct the accelerometer check, if any axis variance is more than the threshold, this inspection fails.

3. Magnetometer

Similar to the accelerometer readings, we take note of magnetometer's variance only. If any axis variance is larger than the threshold, failure of inspection is triggered.

4. Global Positioning System (GPS)

GPS system in the autopilot is a bit different from other sensors. The reading is not instantly published when system boots up. The GPS module needs to wait for satellites to lock up (satellite number is more than 6). Normal waiting time is around one minute. Therefore, the

program only starts to collect data when GPS is locked. We calculate variances of horizontal position error (EPH) and vertical position error (EPV) values to judge the quality of GPS readings. EPH is a sigma horizontal error estimation in meters and EPV is a sigma vertical estimation error in meters.

5. Inertial Measurement Unit (IMU) stuck

There is a special case that may cause the above IMU sensor check to malfunction, and this happens when the IMU does not output a reading or is not powered. We recursively check the current IMU readings and other previous four consecutive readings. When UAV is on ground, although the IMU readings are very small, they are not exactly zero. Hence we use experimental data to determine a threshold to whether the IMU fails.

6.4.2 Recursive Means and Variance Algorithm

There are several data requirements for different pre-flight such as means and variance collection. Since there is a need to continuously cache and process the data, a recursive mean and variance algorithm is implemented in the program. We adopt a recursive algorithm in (6.4.1) and (6.4.2). [45,46]

$$\mu_t = \frac{t-1}{t}\mu_{t-1} + \frac{1}{t}x_t \quad (6.4.1)$$

where μ_t is the calculated mean of the input data x_t at time t and t is the count time.

$$\sigma_t^2 = \frac{t-1}{t} \sigma_{t-1}^2 + \frac{1}{t-1} (x_t - \mu_t)^2 \quad (6.4.2)$$

where σ_t^2 is the calculated variance of the input data x_t at t .

Figure 6.6: Fault Tree Analysis Diagram

6.5 In-flight Check

During the in-flight check situation, there are three emergency fail-safe actions involved which are “return home”, “safety landing”, and “flight termi-

nation”.

1. Return Home

When return home is activated, UAV will return to the launch site and each UAV will have a pre-defined unique route to ensure safe returns without collision. As shown in Figure 6.6, return home action which is the least critical case, which is possibly activated by crossing soft-geo-fence, remote radio link loss more than ten seconds, high level autopilot IMU fail are no response. Detailed explanations are shown in Table 6.1

2. Safety Landing

Safety Landing is that UAV will land directly and slowly at current location and cut its throttle when the UAV has landed. This mode has the second highest priority. It will be triggered by the loss of the low level auto-pilot GPS, battery critically low or UAV's in-air attitude angle exceeding $\pm 45^\circ$. No GPS safety landing may cause UAV position drifting since the localization is purely based on on-board IMU and barometer.

3. Flight Termination

Fight termination has the highest priority. It will be triggered when the low level auto-pilot locks up or the hard-geo-fence is breached or several other conditions happen together including one or both conditions in

this mode.

6.5.1 Mode Priority

The priority of these three modes from highest to lowest is Flight Termination - Safety Landing - Return Home. There are cases that several conditions are triggered simultaneously and actions will be taken based on the triggered highest priority mode. As shown in Figure 6.7, some preliminaries are:

- Normal: Health action with no failsafe state happens
- Return Home: Return Home state
- Safety Landing: Safety Landing state
- Flight Termination: Flight Termination state

6.6 Inertial Management Unit Online Swap

IMU health condition directly decides the flight performance. Our autopilot has a set of redundant IMU. If the primary IMU fails and results in no readings or reading stuck, the secondary IMU will stream its data into the attitude control loop.

It is difficult to judge if the IMU readings are correct when no reference data are available. The only way to check the IMU condition is whether sensor reading is stuck. After lots of on-board tests, the variance data between

Table 6.1: In-flight Check Triggered Conditions.

Triggered Condition	Explanation
Soft Geo-fence	when predefined soft geo-fence boundary is violated
RC Loss	when Remote Control Link loses more than 10 seconds
High-Level Autopilot Fault Flag	when high-level autopilot tracking error $>$ predefined value
High-level Autopilot No Update	when high-level autopilot has no heartbeat update to low level autopilot
Low-level Autopilot GPS Loss	when low-level autopilot GPS satellite number < 6 and position estimation error is large
Battery Critical Low	when UAV on-board battery $<$ predefined critical voltage level that may cause UAV to lose power
Attitude Angle $> \pm 45^\circ$	when UAV measured attitude $> \pm 45^\circ$ that may cause UAV to crash
Hard Geo-fence	when predefined hard geo-fence boundary is violated
Low Level Autopilot Lockup	when low level autopilot loses power or system hangs so that motors control signal has no update anymore

Figure 6.7: Mode Priority State Machine.

three consequent readings of different sensors are illustrated in Table 6.2. If reading variance is lower than the threshold value, IMU is considered as no response failure status accordingly.

6.7 GCS Health Monitoring

To monitor the flight status of the UAV in real time, necessary data should be sent back to Ground Control Station(GCS), a rugged and semi-waterproof Panasonic Toughbook. To ensure data is secured, we add in a 128-bit Advanced Encryption Standard(AES) [47] data encryption into the data link. The data set was defined in both the air and ground side communication

Table 6.2: Inertial Management Unit Dead Variance Threshold (sampling of ten readings).

Sensor	Model Name	Threshold
First Gyroscope	MPU6000	1×10^{-10} rad/s
Second Gyroscope	L3GD20	1×10^{-10} rad/s
First Accelerometer	MPU6000	1×10^{-8} rad/s
Second Accelerometer	LSM303D	1×10^{-8} rad/s

module.

1. Main Localization Data

GPS data with projected X, Y, Z coordinates and satellite numbers are shown in each UAV window. UAV's heading based on high level on-board compass is shown on the window as well.

2. Battery Voltage

Battery Voltage is displayed on the screen. If voltage is lower than the preset threshold value, the on-board system will take failsafe action of safety landing.

3. High Level Flight Computer Modes

High Level Flight Computer Modes are Take-off, Hold, Path, Land, Return Home, and Idle. These modules are mainly used for formation path performance. This part will not be expanded.

Figure 6.8: GCS field setup.

4. Low Level Autopilot Mode

This mode includes arm, geo-fence, failsafe, high level faulty flag, battery condition, GPS, and gyro reading flags. These information are integrated and encoded into a 16-bit integer in order to reduce the data payload.

5. Version Control

This includes low level and high level computer firmware git commit number, a tag is attached on the commit number when the repository

is stable. GCS will detect it if firmware is not in clean status, ie. firmware has some changes after the stable version.

6.8 Experimental Results

With the implementation of all the safety features explained in this paper, our system increases the multi-UAV control system's robustness and stability significantly. We eventually have the solution for very critical conditions such as IMU failure, GPS drifting or loss etc. Figure 6.10 and Figure 6.11 show the final results of our performance, we are able to fly up to 16 UAVs formation safely. Figure 6.12 shows the capability of this safety system during formation flight. It may be crashed without adding safety system onboard.

6.9 Summary

In this chapter, a comprehensive safety system designed for distributed UAV formation has been implemented. Detailed analysis of onboard sensor has been conducted. Besides, pre-flight check design has been explained based on sensor property and real flight scenarios. Moreover, in-flight check safety algorithm have been shown in this chapter based on fault tree analysis. There are three safety actions have been used: return home, safety landing, and flight termination. Finally, ground control station program has been

designed for monitoring the health status of each UAV.

Figure 6.9: GCS health monitoring window

Figure 6.10: Formation Performance 8 UAVs

Figure 6.11: Formation Performance 14 UAVs

Date	Safety Action Triggered	Condition	Reasons
15 April 2015	Return Home	High Level Autopilot Faulty Flag	High Level Autopilot Lockup
6 May 2015	Flight Termination	Low Level Autopilot Lockup	Power Interface Module lost power
20 October 2015	Safety Landing	Attitude $> \pm 45^\circ$	Strong Wind
2 November 2015	Return Home	Soft-geo-fence Violence	Wrong Formation Path Design
18 November 2015	Safety Landing	Battery Critical Low	Battery Broken
23 December 2015	Return Home	RC loss	RC antenna improper installation

Figure 6.12: Safety System Testing Result

Chapter 7

Integration Test and Discussion

After the careful study of distributed formation path algorithm and UAV system, we now proceed with system integration and tests.

- Hardware in the Loop Simulation is the experiment operating on-board the autopilot. The performance and correctness of the algorithm will be tested on real avionics system.
- Platform PID Flight Test is an important experiment before conducting formation flights. Each UAV's PID are tuned respectively.
- Safety Precaution Mechanism measures failure possibilities and solutions of long communication latency, exceed velocity commands etc.
- Final Distributed Formation Flight with 3 UAVs is the desired outcome in this research.

7.1 Hardware in the Loop Simulation

For safety reason, it is good to have a hardware in the loop (HIL) test on ground before flight. A program was developed to achieve this HIL test. The challenging part is about transferring the neighbor's information into local and how to make the setup portable and convenient.

Figure 7.1: Distributed Formation Simulation Program Flow

Figure 7.1 shows the state flow of this HIL program. A file named `disfor_sim.txt` is previously stored into autopilot SD card with necessary simulation information. The table shown below is the list of data in this file. UAV 4 initialization data is reserved for future use, limited by

Number of UAVs in Formation parameter.

In Figure 7.2, an example of distributed formation simulation initialization file is shown. This results in a single UAV ground HIL simulation before real flight. This program is capable of providing localizing and controlling response data to distributed formation module operating in autopilot.

Figure 7.2: Distributed Formation Simulation Program Flow

7.2 PID Tuning

There is a need to fine tune the UAV's PID control to make the flight mission smoothly and robustly. Some experimental setups must meet the criteria.

- Test Field

The area of flight must be open and no high surrounded buildings. This will give GPS best signal conditions so that position estimation will be accurate. For the setup in Figure 7.3, the area is relatively open with some low buildings around. It is good enough for UAVs to take off safely.

- UAV Hardware Condition

UAV platform must be in very healthy status. There should be no loose

Table 7.1: Distribute Formation Simulation Initialization File

1	UAV 1 Latitude
2	UAV 1 Longitude
3	UAV 1 Heading
4	UAV 2 Latitude
5	UAV 2 Longitude
6	UAV 2 Heading
7	UAV 3 Latitude
8	UAV 3 Longitude
9	UAV 3 Heading
10	UAV 4 Latitude
11	UAV 4 Longitude
12	UAV 4 Heading
13	Common Altitude
14	Number of UAVs in Formation

screws or hardware joints. Cable connectivities need to be pre-tested on ground.

- UAV Battery

Power system is very important in UAV control as a low voltage may result in poor motors' performance thus it was noted that each UAVs uses a fresh, full charged battery, for test flight.

Figure 7.3: Distributed UAV Flight Area

Figure 7.4: Distributed UAV PID Tuning

7.3 Distributed Formation Flight Experiment

In the real flight experiment of three UAVs, there is a need to consider many cases of possibilities, such as UAVs collision, colliding with obstacles and etc. Hence by leveraging the safety system implementation as discussed in Chapter 6, there are enough mechanisms and logics to handle these fail-safe conditions.

Figure 7.5: Distributed UAV Ground Initialization

Figure 7.5 represents the field setup. Each UAV is initialized and booted up accordingly. The final edge length of the triangular shape is 10 meters. Due to area limitation, the external velocity was disabled for safety reason. And the goal of this experiment is to examine the distributed formation algorithm and platform compatibility.

As shown in Figure 7.6, a simple state machine is designed for the distributed formation test. Changing of states is decided by human input factor. For instance, if there is a desire to have the UAVs to switch to Distributed Formation Mode from Arm state, a specific switch on the remote controller will be assigned. In the figure, the dash lines mean that the connections are

wireless.

Figure 7.6: Distributed Formation State Machine

When UAV is in the Distributed Formation Mode, it will take off to its desired altitude with current position. Then UAV will start to execute path planning in distributed formation program. And the final shape of the flight pattern is an equilateral triangle.

Figure 7.7: Distributed UAV Flight Test

Three UAVs will take off from their landing position and merge flying until distance is satisfied by the algorithm. As shown in Figure 7.7, formation pattern has already been formed.

Figure 7.8, 7.9, 7.10 are the flight results of this flight experiment. The respective global position of the three UAV fused. Figure 7.8 shows the exact flight trajectories of each UAV during distributed formation pattern. It is obvious that three UAVs are merging during formation mode. In Figure 7.9, the flight position data was sorted with time and calculated distances between each two UAVs respectively as shown in Figure 7.9. Consequently a calculation is taken for these distances and one can obtain the potential

function of the whole formation in Figure 7.10. This potential function has been defined in Equation 3.3.2.

From the logged flight results, we can observe that the tracking performances from Figure 7.9 and 7.10 did not reach our expectation. There exists a disturbance about 1 meter. After individually excluding each possible conditions (control accuracy, localization accuracy), we figure out that this disturbance is due to localization precision issue. For instance, during this outdoor flight experiment, we faced a wind speed of 3m/s. GPS signal has its own accuracy tolerance of ± 3 meters. Hence localization accuracy could be the possible reasons for data disturbance during formation.

Figure 7.8: Distributed UAV Flight Test Path Log

Figure 7.9: Distributed UAV Flight Test Distance Log

Figure 7.10: Distributed UAV Formation Error

7.4 Safety Precaution Mechanism

7.4.1 Communication Latency Test

Communication Topology for Three UAVs

As shown in Figure 7.11, there are two communication links on UAV 1 and one each on UAV 2 and UAV 3 accordingly. For the communication between UAV 2 and UAV 3, data will stream to UAV 1 first and forward to the destination UAV. We use Universal Asynchronous Receiver/Transmitter (UART) bridge with the frequency of 10 Hz and packet size of 72 bytes with 57.6 kbps data rate.

Figure 7.11: Distributed Communication Network

In the experiment illustrated in Figure 7.12, each UAV is given a fake

GPS value (count up integer in this case). The data printed in the three windows are corresponding to three UAV autopilot control terminal windows. The data are able to be transmitted to another UAV successfully. It is observed that the functionality of distributed formation is accomplished.

The image displays three terminal windows from a Linux environment, showing the execution of a distributed formation control program. The windows are titled 'ericwelye@ericwelye-Lenovo-IdeaPad-Y470: ~'. The leftmost window shows a series of 'distribute_formation: system 1 data 1.000, 2.000, 3.000' messages. The middle window shows a 'link_receive' message and a 'distribute_formation: system 3 data 7.000, 8.000, 9.000' message. The rightmost window shows a 'distribute_formation: system 1 data 1.000, 2.000, 3.000' message. The data represents GPS coordinates for three different UAVs, demonstrating successful communication and data transmission.

Figure 7.12: Distributed Program Communication Ground Test Three UAVs

Communication Topology for Four UAVs

After the development of three UAV program implemented, an extension for more UAV to join the team, is reserved. Results are as shown in Figure 7.13.

Figure 7.14: Distributed Communication Network Four UAVs

Single Link Latency Test

To monitor the performance of our UAV communication link, a simple mechanism was designed to record the latency in a single communication link. As is shown in Figure 7.15, terminal windows are the output directly from autopilot. Data direction, latency duration, loop frequency and failure count are indicated in the windows. If latency duration is more than 2 seconds, failure count will be incremented. We can observe that in the upper terminal window, failure count is 6. This is because the time when we run the autopilot in upper window is before the lower window autopilot. It takes few seconds to open the lower window program. Hence there is a time difference

to let upper window autopilot to realize that there is no communicational link from the other.

Figure 7.15: Distributed Communication Network Four UAVs

Discussions for Communication Latency

To decrease or even eliminate the communication latency effects, there are several solutions for future works.

1. Optimize Program

Program scheduling inefficiency may be the issue that causes the large communication latency. The distributed formation path generator module runs at 50 Hz but wireless UART bridge runs at only 10 Hz, a study will be conducted on the autopilot module schedule management to optimize the program.

2. GPS Timing

By adding GPS timing into the communication packet of each UAV and assuming the GPS device is working properly on the corresponding UAV, slight communication latency will exist. This is because the time on source UAV_{*i*} may be different from destination UAV_{*j*}. However, the time on UAV_{*j*} is precise and the GPS time difference can be used Δt to adjust position error due to communication latency.

$$\Delta t = \|t_{ij} - t_j\| \quad (7.4.1)$$

where t_{ij} is the UAV_{*s_i*} GPS time that appears on UAV_{*j*}, and t_j is the GPS time of UAV_{*j*}.

7.4.2 Velocity Command

Due to the constrained flight area in this research, it is not possible to perform fast UAV motion with an external reference velocity added. Hence, velocity command given to UAV is normalized.

$$\dot{\delta}_{final} = \frac{\dot{\delta}}{\|\dot{\delta}\|} \quad (7.4.2)$$

7.5 Summary

In this chapter, detailed distributed formation integration and experimental design were explained. Hardware-in-the-loop simulation was conducted to validate the performance of the algorithm. Besides, PID tuning experiment was done in order to have best control accuracy. Moreover, distributed

formation flight test was conducted to verify and final result in real flight environment. In addition, several safety precaution mechanisms in the research were discussed such as communication latency between UAVs, and velocity limitation, etc.

Chapter 8

Conclusion and Recommendations

8.1 Conclusion

This thesis has discussed and presented multiple UAV distance constrained distributed formation designs and implementations. The contributions of the thesis are summarized below.

Firstly, a distributed formation algorithm is developed and implemented. The algorithm is able to perform coordinates initialization and projection onto pre-defined reference points. Besides, the distance constrained distributed formation is implemented into flight-tested. Based on original references in previous work, a proportional controller is designed for our for-

mation flight.

Secondly, to increase performance of this formation, namely eliminate steady state error with external reference velocity, the integral terms are introduced into distributed formation controller and the steady state error are compensated with appropriate parameters used.

Thirdly, a comprehensive safety system is designed for each UAV. Features such as pre-flight inspections, in-flight checks and ground station examinations are explained and discussed in Chapter 6. Fault tree analysis is designed in details so that most possible failure condition occurring during flights, are considered. This system is a novel concept for crowded and dangerous flight environments nowadays.

Last but not least, performance verification experiments such as MATLAB simulations, ground dry run tests and etc. are conducted before the real flights. By minimizing the crash and failure rates due to program bugs and algorithm errors.

8.2 Recommendations

8.2.1 External Reference Velocity Flight Experiment

In spite of proportional-integral controller designed, UAV formation flight does not include external reference velocity tests. Separate mission control commands can be added to the leader UAV so that the formation is movable with a given velocity if there is a large enough testing area.

8.2.2 Increase Number of Formation UAVs

The goal of this thesis is to develop the testbed for outdoor distributed formation. Currently, three UAVs were built for testing and debugging. The shape is always infinitesimally rigid because of the numbers of the UAVs. There exist some uncertainties if more UAVs were introduced into the formation such as formation graph rigidity, etc. We have to discuss and add more intelligent algorithms to deal with complex shape formations.

8.2.3 Risk Assessment Analysis

To decrease the crash rate of UAV operation, a comprehensive and customized risk assessment form will be designed specially for distributed formation UAV applications. Anti-collision and neighbor recognition functions are to be integrated in the future.

8.2.4 Elimination of Communication Latency Effects

Due to communication device processing delay, there exists a significant latency in our experiment shown in Chapter 7. A position predictive algorithm has to be designed in our system to estimate current real positions of neighbor UAV when receiving delayed information from them.

Publication List

- Published papers

1. D. Wei, S. Zhang, L. Xie, X. Dong, and K. Z. Y. Ang, “Comprehensive Safety System Design and Development for Unmanned Aerial Vehicles Formation Flight System,” in *AIAA Infotech@ Aerospace*, p. 0249, 2016

- Papers under review

2. S. Zhang, D. Wei, M. Q. Huynh, J. X. Quek, X. Ma, and L. Xie, “Model Predictive Control Based Dynamic Geofence System for Unmanned Aerial Vehicles,” in *AIAA Infotech@ Aerospace*

Bibliography

- [1] S. Pham, “Missouri Drone Journalism Program to Reconfigure Goals After FAA Letter.” <http://www.missouridronejournalism.com/2013/08/missouri-drone-journalism-program-to-reconfigure-goals-after-faa-letter/>, 2013-08-21.
- [2] “Predator Drone Being used to map Hurricane Damage.” <https://web.archive.org/web/20080915221234/http://www.chron.com/disp/story.mpl/ap/tx/5986181.html>, 2008.
- [3] I. Just Aero Works, “2008 Search and Rescue Missions.” <https://web.archive.org/web/20110713125226/http://www.justaeroworks.com/pages/JAW.08SRM.html>, 2011.
- [4] G. Hoffmann, D. G. Rajnarayan, S. L. Waslander, D. Dostal, J. S. Jang, and C. J. Tomlin, “The Stanford Testbed of Autonomous Rotorcraft for Multi Agent Control (starmac),” in *Digital Avionics Systems Conference, 2004. DASC 04. The 23rd*, vol. 2, pp. 12–E, IEEE, 2004.

- [5] Y. Mulgaonkar, G. Cross, and V. Kumar, “Design of Small, safe and Robust Quadrotor Swarms,” in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pp. 2208–2215, IEEE, 2015.
- [6] T. Nageli, C. Conte, A. Domahidi, M. Morari, and O. Hilliges, “Environment-independent Formation Flight for Micro Aerial Vehicles,” in *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pp. 1141–1146, IEEE, 2014.
- [7] L. M. Kok, “Singapore to Introduce Drone law: 5 Things About These Flying Machines.” <http://www.straitstimes.com/singapore/singapore-to-introduce-drone-law-5-things-about-these-flying-machines>, 2015.
- [8] Y. Zou, P. R. Pagilla, and R. T. Ratliff, “Distributed Formation Flight Control Using Constraint Forces,” *Journal of guidance, control, and dynamics*, vol. 32, no. 1, pp. 112–120, 2009.
- [9] Z. Chao, L. Ming, Z. Shaolei, and Z. Wenguang, “Collision-free UAV Formation Flight Control Based on Nonlinear MPC,” in *Electronics, Communications and Control (ICECC), 2011 International Conference on*, pp. 1951–1956, IEEE, 2011.
- [10] Z. Hou and I. Fantoni, “Distributed Leader-follower Formation Control for Multiple Quadrotors with Weighted Topology,” in *System of Sys-*

- tems Engineering Conference (SoSE), 2015 10th*, pp. 256–261, IEEE, 2015.
- [11] R. Haghighi, D. Wang, and C. Low, “Real-time Distributed Optimal Trajectory Generation for Nonholonomic Vehicles in Formations,” in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pp. 3589–3594, IEEE, 2014.
- [12] I. Shames, B. Fidan, and B. Anderson, “Close Target Reconnaissance Using Autonomous UAV Formations,” in *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*, pp. 1729–1734, IEEE, 2008.
- [13] F. He, Y. Wang, Y. Yao, L. Wang, and W. Chen, “Distributed Formation Control of Mobile Autonomous Agents using Relative Position Measurements,” *Control Theory & Applications, IET*, vol. 7, no. 11, pp. 1540–1552, 2013.
- [14] R. Olfati-Saber and R. M. Murray, “Graph Rigidity and Distributed Formation Stabilization of Multi-vehicle Systems,” in *Decision and Control, 2002, Proceedings of the 41st IEEE Conference on*, vol. 3, pp. 2965–2971, IEEE, 2002.
- [15] X. Chen, A. Serrani, and H. Ozbay, “Control of Leader-follower Formations of Terrestrial UAVs,” in *Decision and Control, 2003. Proceedings. 42nd IEEE Conference on*, vol. 1, pp. 498–503, IEEE, 2003.

- [16] W. Ren and R. Beard, “Decentralized Scheme for Spacecraft Formation Flying via the Virtual Structure Approach,” *Journal of Guidance, Control, and Dynamics*, vol. 27, no. 1, pp. 73–82, 2004.
- [17] D. Van der Walle, B. Fidan, A. Sutton, C. Yu, and B. Anderson, “Non-hierarchical UAV Formation Control for Surveillance Tasks,” in *American Control Conference, 2008*, pp. 777–782, IEEE, 2008.
- [18] R. L. Raffard, C. J. Tomlin, and S. P. Boyd, “Distributed Optimization for Cooperative Agents: Application to Formation Flight,” in *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, vol. 3, pp. 2453–2459, IEEE, 2004.
- [19] P. Kocher, R. Lee, G. McGraw, A. Raghunathan, and S. Moderator-Ravi, “Security as a new Dimension in Embedded System Design,” in *Proceedings of the 41st annual Design Automation Conference*, pp. 753–760, ACM, 2004.
- [20] J. Henkel, W. Wolf, and S. Chakradhar, “On-chip Networks: A Scalable, Communication-centric Embedded System Design Paradigm,” in *VLSI Design, 2004. Proceedings. 17th International Conference on*, pp. 845–851, IEEE, 2004.
- [21] S. Mao, W. K. Tan, and K. H. Low, “Autonomous Formation Flight of Indoor UAVs Based on Model Predictive Control,” in *AIAA Infotech@Aerospace*, p. 0515, 2016.

- [22] F. Borrelli, T. Keviczky, and G. J. Balas, “Collision-free UAV Formation Flight Using Decentralized Optimization and Invariant sets,” in *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, vol. 1, pp. 1099–1104, IEEE, 2004.
- [23] H. Min, F. Sun, and F. Niu, “Decentralized UAV Formation Tracking Flight Control Using Gyroscopic force,” in *Computational Intelligence for Measurement Systems and Applications, 2009. CIMSAS'09. IEEE International Conference on*, pp. 91–96, IEEE, 2009.
- [24] D. P. Scharf, F. Y. Hadaegh, and S. R. Ploen, “A Survey of Spacecraft Formation Flying Guidance and Control. part ii: Control,” in *American Control Conference, 2004. Proceedings of the 2004*, vol. 4, pp. 2976–2985, IEEE, 2004.
- [25] F. Hadaegh, W.-M. Lu, and P. Wang, “Adaptive Control of Formation Flying Spacecraft for Interferometry,” 1998.
- [26] X. Chen and A. Serrani, “Iss-based Robust Leader/follower Trailing Control,” in *Group Coordination and Cooperative Control*, pp. 75–91, Springer, 2006.
- [27] R. Kristiansen, A. Loría, A. Chaillet, and P. Nicklasson, “Output Feedback Control of Relative Translation in a Leader-follower Spacecraft Formation,” in *Group Coordination and Cooperative Control*, pp. 131–151, Springer, 2006.

- [28] R. W. Beard and F. Y. Hadaegh, “Finite Thrust Control for Satellite Formation Flying with State Constraints,” in *American Control Conference, 1999. Proceedings of the 1999*, vol. 6, pp. 4383–4387, IEEE, 1999.
- [29] J. Lawton, R. W. Beard, and F. Y. Hadaegh, “An Adaptive Control Approach to Satellite Formation Flying with Relative Distance Constraints,” 1999.
- [30] N. E. Leonard and E. Fiorelli, “Virtual Leaders, Artificial Potentials and Coordinated Control of Groups,” in *Decision and Control, 2001. Proceedings of the 40th IEEE Conference on*, vol. 3, pp. 2968–2973, IEEE, 2001.
- [31] K.-H. Tan and M. A. Lewis, “Virtual Structures for High-precision Cooperative Mobile Robotic Control,” in *Intelligent Robots and Systems’ 96, IROS 96, Proceedings of the 1996 IEEE/RSJ International Conference on*, vol. 1, pp. 132–139, 1996.
- [32] T. Balch and R. C. Arkin, “Behavior-based Formation Control for Multirobot Teams,” *Robotics and Automation, IEEE Transactions on*, vol. 14, no. 6, pp. 926–939, 1998.
- [33] F. Arrichiello, S. Chiaverini, and T. I. Fossen, “Formation Control of Marine Surface Vessels Using the Null-Space-based Behavioral

- Control,” in *Group Coordination and Cooperative Control*, pp. 1–19, Springer, 2006.
- [34] O. Rozenheck, S. Zhao, and D. Zelazo, “Distance-constrained Formation Tracking Control,”
- [35] Z. Lin, L. Wang, Z. Han, and M. Fu, “Distributed Formation Control of Multi-agent Systems Using Complex Laplacian,” *Automatic Control, IEEE Transactions on*, vol. 59, no. 7, pp. 1765–1777, 2014.
- [36] J. P. Snyder, *Map Projections—A Working Manual*, vol. 1395. US Government Printing Office, 1987.
- [37] R. A. Clothier and R. A. Walker, “Determination and Evaluation of UAV Safety Objectives,” 2006.
- [38] R. E. Weibel and R. J. Hansman, “Safety Considerations for Operation of Unmanned Aerial Vehicles in the National Airspace System,” 2006.
- [39] T.-S. Tay and W. Whiteley, “Generating Isostatic Frameworks,” *Structural Topology* 1985 núm 11, 1985.
- [40] G. Cai, B. M. Chen, and T. H. Lee, *Unmanned Rotorcraft Systems*. Springer Science & Business Media, 2011.
- [41] L. Meier, “Pixhawk Autopilot.” <https://pixhawk.org/modules/pixhawk>, 2015.

- [42] S. Weather, “Climate of singapore.” <http://www.weather.gov.sg/climate-climate-of-singapore/>, 2015.
- [43] D. Innovation, “Flame Wheel ARF Kit.” <http://www.dji.com/product/flame-wheel-arf/spec/>, 2015.
- [44] D. Wei, S. Zhang, L. Xie, X. Dong, and K. Z. Y. Ang, “Comprehensive Safety System Design and Development for Unmanned Aerial Vehicles Formation Flight System,” in *AIAA Infotech@ Aerospace*, p. 0249, 2016.
- [45] T. F. Chan, G. H. Golub, and R. J. LeVeque, “Algorithms for Computing the Sample Variance: Analysis and Recommendations,” *The American Statistician*, vol. 37, no. 3, pp. 242–247, 1983.
- [46] R. F. Ling, “Comparison of Several Algorithms for Computing Sample Means and Variances,” *Journal of the American Statistical Association*, vol. 69, no. 348, pp. 859–866, 1974.
- [47] N.-F. Standard, “Announcing the Advanced Encryption Standard (aes),” *Federal Information Processing Standards Publication*, vol. 197, pp. 1–51, 2001.
- [48] S. Zhang, D. Wei, M. Q. Huynh, J. X. Quek, X. Ma, and L. Xie, “Model Predictive Control Based Dynamic Geofence System for Unmanned Aerial Vehicles,” in *AIAA Infotech@ Aerospace*.