

Doctoral thesis / Dissertation

for the doctoral degree / *zur Erlangung des Doktorgrads*

Doctor rerum naturalium (Dr. rer. nat.)

Distributed Control of Cooperating Mini UAVs

Verteilte Regelung von Kooperierenden Mini UAVs



Submitted by / *Vorgelegt von*

Qasim Ali

from / *aus*

Rawalpindi, Pakistan

Würzburg, 2016

Submitted on / *Eingereicht am*:

Stamp / *Stempel* Graduate School

Members of thesis committee / *Mitglieder des Promotionskomitees*

Chairperson / *Vorsitz*: Prof. Dr. Andreas Nüchter

1. Reviewer and Examiner / *1. Gutachter und Prüfer*: Prof. Dr. Sergio Montenegro

2. Reviewer and Examiner / *2. Gutachter und Prüfer*: Prof. Dr. Klaus Schilling

3. Examiner / *3. Prüfer*: Prof. Dr. Herbert Werner

Additional Examiners / *Weitere Prüfer*:

.....

Day of thesis defense / *Tag des Promotionskolloquiums*:

Acknowledgements

I would like to convey my deepest gratitude to my doktorvater, Prof. Dr.-Ing. Sergio Montenegro, for his valuable insights and guidance to complete this project in an efficient and successful manner. He is a great mentor as he continually and convincingly conveyed a spirit of adventure in research. Without his guidance, encouragement and persistent help, this thesis would not have been possible. I would like to thank my thesis committee members, Prof. Dr. rer. nat. Klaus Schilling and Prof. Dr. Herbert Werner from Technical University Hamburg-Harburg (TUHH), who guided me whenever I needed their direction. PhD workshops conducted at TUHH by Prof. Werner accelerated the learning process in the domain of formation flight control. Oberseminars administered by Prof. Schilling at Institute of Robotics and Telematics, University of Würzburg also provided a good platform for continuous learning and sharing new ideas. A bundle of thanks goes to Dr. Nils Gageik for his guidance in the domain of quadcopter hardware and software. Valuable suggestions by Erik Dilger in the domain of embedded system programming are acknowledged. I would also like to thank Muhammad Faisal for a pleasant company. Supportive spirit and the warmth from all the colleagues of my Institute (Institute of Aerospace Information Technology) is going to last in my memories. This is a testimony to the supportive environment that Prof. Montenegro created at the Institute, which I cherished during my stay at the Institute.

Many thanks go to Higher Education Commission (HEC) of Pakistan for providing me the scholarship and German Academic Exchange Service (DAAD) for their administrative support for PhD studies in Germany. Financial support from Prof. Montenegro to present my papers in international conferences is also gratefully acknowledged. Particularly I would like to thank German Research Foundation (DFG) and University of Würzburg for financially supporting the journal publications through the funding programme Open Access Publishing. I would like to pay special thanks in general to the administration of Graduate School of Science and Technology (GSST), University of Würzburg and in particular to Dr. Stefan Schroeder-Koehne for arranging valuable workshops of transferable skills, and other social gatherings. I would also like to thank Dr. Zeeshan Ahmed (then at Würzburg University) for giving me the inspiration to publish my results, and suggesting stylistic corrections for some of my papers. Last

but not the least, I cannot say enough about my wife and children for their continuous support and help to complete my work quite smoothly. Their patience is equally acknowledged.

Summary

Mini Unmanned Aerial Vehicles (MUAVs) are becoming popular research platform and drawing considerable attention, particularly during the last decade due to their affordability and multi-dimensional applications in almost every walk of life. MUAVs have obvious advantages over manned platforms including their much lower manufacturing and operational costs, risk avoidance for human pilots, flying safely low and slow, and realization of operations that are beyond inherent human limitations. The advancement in Micro Electro-Mechanical System (MEMS) technology, Avionics and miniaturization of sensors also played a significant role in the evolution of MUAVs. These vehicles range from simple toys found at electronic supermarkets for entertainment purpose to highly sophisticated commercial platforms performing novel assignments like offshore wind power station inspection and 3D modelling of buildings etc. MUAVs are also more environment friendly as they cause less air pollution and noise. Unmanned is therefore unmatched. Recent research focuses on use of multiple inexpensive vehicles flying together, while maintaining required relative separations, to carry out the tasks efficiently compared to a single exorbitant vehicle. Redundancy also does away the risk of loss of a single whole-mission dependent vehicle. Some of the valuable applications in the domain of cooperative control include joint load transportation, search and rescue, mobile communication relays, pesticide spraying and weather monitoring etc. Though realization of multi-UAV coupled flight is complex, however obvious advantages justify the laborious work involved.

Distributed control of cooperating units is a multi-disciplinary topic, and in order to realize it one requires to work in diversified domains. These domains include MUAV hardware and software, inter-communication for necessary information sharing, flight dynamics, control engineering particularly distributed / cooperative control techniques, graph theory for communication topology modeling, and sensors technology like Differential GPS (DGPS) etc. For a fleet of agents flying in close vicinity, accurate position determination is mandatory to avoid collisions and to meet the requirements for most of the missions like geo-referencing. For such scenarios, DGPS is a potential candidate. A part of research has therefore been dedicated to the development of DGPS code.

One of the modules of this research was hardware implementation. A simple test setup has been developed at Institute of Aerospace Information Technology, University of

Würzburg to realize basic functionalities for formation flight of quadcopters. The testing environment may be utilized not only for testing and validating the algorithms for formation flying capability in real environment but also for education purpose. An already existing test bench for single quadcopter was extended with necessary inter-communication and distributed control mechanism to test the algorithms for formation flights in three degrees of freedom (roll/pitch/yaw). This study encompasses the domains of communication, control engineering and embedded systems programming. Bluetooth protocol has been used for inter-communication between two quadcopters. A simple technique of Proportional-Integral-Derivative (PID) control in combination with Kalman filter has been exploited. MATLAB Instrument Control Toolbox has been used for data display, analysis and plotting. Plots can be drawn in real-time and received data can also be stored in the form of files for later use and analysis. The test setup has been developed at considerably low cost while giving due consideration to simplicity. Proposed setup is quite flexible that can be modified as per changing requirements.

For distributed control scheme, a centralized heterogeneous formation flight position control technique has been formulated using explicit model following Linear Quadratic Regulator Proportional Integral (LQR PI) controller. Leader quadcopter is a stable reference model with desired dynamics whose output is perfectly tracked by the two wingmen quadcopters. Leader itself is controlled through pole placement control method with desired stability characteristics while the two followers are controlled through a robust and adaptive LQR PI control method. For this study, a full-state vector of quadcopter is considered, while tracking only the performance output. Selected 3D formation geometry and static stability is maintained under a number of possible perturbations. In case of communication loss between leader and any of the followers, the other follower provides the data, received from the leader, to the affected follower. Stability of closed-loop system has been analyzed using singular values. Proposed approach for tightly coupled formation flight of MUAVs has been validated with the help of extensive simulations using MATLAB/Simulink that provided promising results. Tracking performance has been demonstrated for time-varying commands as well. Proposed architecture is scalable and can be expanded easily. This approach is appropriate for the scenarios where tightly coupled formation flight is desired like cooperative grasping, joint load transportation etc.

An innovative framework has been developed for teamwork of two quadcopter fleets.

The mission has been set, as an example, to extinguish the fire over an area. Each formation has its specified formation geometry and assigned task. Position control for quadcopters in one of the formations has been implemented through LQR PI control scheme based on explicit model following. Quadcopters in other formation are controlled through LQR PI servo-mechanism control scheme. The two control schemes are compared in terms of their performance and control effort. Both formations are commanded by respective ground stations through virtual leaders. The ground stations share the commanded altitude information to ensure safe mutual separation between the formations. Quadcopters are able to track desired trajectories as well as to hover at desired points for selected time duration. In case of communication loss between ground station and any of the quadcopter, the neighboring quadcopter provides the command data, received from the ground station, to the affected unit. Proposed framework has been validated through extensive simulations using MATLAB/Simulink that provided favorable results. Cluster reconfiguration of agents is also demonstrated in our work, where formation geometry may be switched to any arbitrary shape during flight. For the stated applications, consensus algorithms are not desired as we require the quadcopter fleets to track the trajectories of our interest, rather than decided by the agents themselves.

A number of practical problems involving networks may be appropriately represented by the graphs that facilitate problem formulation and analysis process. Communication topology for networks involving a large number of units, like swarm of aerial vehicles, may be conveniently examined using the notion of graph theory. To facilitate the formulation of such problems, an appropriate mathematical solution is to represent the graph with the help of Laplacian matrix. Eigenvalues of Laplacian matrix have been given appropriate consideration in our study to give an insight into the graph / subgraph properties. Some have been exploited to generalize the well-known Euler's Formula in order to make it applicable for graphs as well as subgraphs. A modified Euler's formula is also presented. Utilization of graph theory in distributed / cooperative control schemes has also been demonstrated through simulations.

Cooperative control schemes based on consensus algorithm have been demonstrated for position control of quadcopters in a fleet where no explicit leader exists. Consensus algorithms in combination with different control schemes have been employed which add towards autonomy of quadcopters. The control schemes utilized for this purpose include

LQR PI control based on model following and LQR PI servomechanism. The schemes have been studied under different communication topologies, including fully connected undirected graphs, directed graphs and cycle topology. Information flow among the agents in a cluster has been modeled through Laplacian matrix. Effects of input biases on consensus values have also been studied. Quadcopters are able to track the trajectories and reach the destination points agreed upon through mutual consensus. Proposed schemes under different communication topologies have been validated through extensive simulations in Matlab/Simulink environment. The results authenticate the efficacy of presented schemes with added advantage of simplicity in its implementation. The proposed scheme is scalable for large group of MUAVs.

For formation flying, position accuracy requirements are quite stringent. GPS signals alone do not provide position accuracy enough to meet the requirement; a technique for accurate position determination is therefore necessarily required, for example DGPS. A number of public codes exist for GPS positioning and baseline determination in off-line mode. However, no software code exists for DGPS that exploits correction factors at base stations, without relying on double difference information. In order to accomplish it, a methodology is introduced in MATLAB environment for DGPS using *C/A* pseudoranges on single frequency *L1* only to make it feasible for low-cost GPS receivers. Our base station is located at accurately surveyed reference point. Pseudoranges and geometric ranges are compared at base station to compute the correction factors. These correction factors are then handed over to rover for all valid satellites observed during an epoch. The rover takes it into account for its own true position determination for corresponding epoch. In order to validate the proposed algorithm, our rover is also placed at a pre-determined location. The proposed code is an appropriate and simple to use tool for post-processing of GPS raw data for accurate position determination of a rover, e.g. a UAV during post-mission analysis.

Kurzfassung

Mini Unmanned Aerial Vehicles (MUAVs) werden immer beliebtere Forschungsplattformen. Vor allem in den letzten Jahren ziehen sie aufgrund ihrer Erschwinglichkeit und ihrer Flexibilität, die es erlaubt sie in fast allen Lebensbereichen einzusetzen, beträchtliche Aufmerksamkeit auf sich. MUAVs haben offensichtliche Vorteile gegenüber bemannten Plattformen einschließlich ihrer viel geringeren Herstellungs- und Betriebskosten, Risikovermeidung für den menschlichen Piloten, der Möglichkeit sicher niedrig und langsam fliegen zu können, und Realisierung von Operationen, die über die inhärenten Grenzen des menschlichen Körpers hinausgehen. Der Fortschritt in der Micro Electro-Mechanical System (MEMS) Technologie, Avionik und Miniaturisierung von Sensoren spielte auch eine bedeutende Rolle bei der Entwicklung der MUAVs. Diese Fluggeräte reichen von einfachem Spielzeug aus dem Elektrofachhandel bis zu hoch entwickelten, kommerziellen Plattformen, die die Durchführung neuer Aufgaben wie Offshore-Windkraftwerk Inspektionen, 3D-Modellierung von Gebäuden usw. erlauben. MUAVs sind auch umweltfreundlich, da sie weniger Luftverschmutzung und Lärm verursachen. Unbemannt ist daher unübertroffen. Aktuelle Forschung konzentriert sich auf die Möglichkeit mehrere kostengünstige Fluggeräte zusammen fliegen zu lassen, während die erforderliche relative räumliche Trennungen beibehalten wird. Dies ermöglicht es effizient Aufgaben zu erfüllen im Vergleich zu einem einzigen sehr teuren Fluggerät. Durch die Redundanz entfällt auch das Risiko des Scheiterns der Mission durch den Verlust eines einzigen Fluggeräts. Wertvolle Aufgaben, die kooperative Fluggeräte ausführen können, sind beispielsweise gemeinsame Lasttransporte, Such- und Rettungsmissionen, mobile Kommunikationsrelais, Sprühen von Pestiziden und Wetterbeobachtung. Obwohl die Realisierung von Flügen mit mehreren, gekoppelten UAVs komplex ist, rechtfertigen dennoch offensichtliche Vorteile diese mühsame und aufwändige Entwicklungsarbeit.

Verteilte Steuerung von kooperierenden Einheiten ist ein multidisziplinäres Thema, das es erfordert in diversifizierten Bereichen zu arbeiten. Dazu gehören MUAV Hardware und Software, Kommunikationstechniken für den notwendigen Informationsaustausch, Flugdynamik, Regelungstechnik, insbesondere für verteilte / kooperative Steuerungstechniken, Graphentheorie für Kommunikationstopologie Modellierung und Sensoren-Technologie wie Differential GPS (DGPS). Für eine Flotte von Agenten, die in unmittelbarer Nähe fliegen, ist eine genaue Positionsbestimmung zwingend nötig um Kollisionen zu vermeiden und die Anforderungen für die meisten Missionen wie Georeferenzierung

zu erfüllen. Für solche Szenarien ist DGPS ein potenzieller Kandidat. Ein Teil der Forschung konzentriert sich daher auf die Entwicklung von DGPS Code.

Eines der Module dieser Forschung war Hardware-Implementierung. Ein einfacher Test-Aufbau zur Realisierung von Basisfunktionalitäten für Formationsflug von Quadrocoptern wurde am Lehrstuhl für Informationstechnik in der Luft- und Raumfahrt der Universität Würzburg entwickelt. Diese Testumgebung kann nicht nur zur Prüfung und Validierung von Algorithmen für Formationsflug in realer Umgebung genutzt werden, sondern dient auch zur Ausbildung von Studenten. Ein bereits vorhandener Prüfstand für einzelne Quadrocopter wurde mit den notwendigen Kommunikation und verteilten Steuerung erweitert, um Algorithmen für Formationsflüge in drei Freiheitsgraden (Roll / Nick / Gier) zu testen. Diese Studie umfasst die Bereiche der Kommunikation, Steuerungstechnik und Embedded-System-Programmierung. Das Bluetooth-Protokoll wurde für die gegenseitige Kommunikation zwischen zwei Quadrocoptern verwendet. Eine einfache Technik der Proportional-Integral-Differential (PID) Steuerung in Kombination mit Kalman-Filter wurde genutzt. Die MATLAB Instrument Control Toolbox wurde für die Datenanzeige, die Analyse und das Plotten verwendet. Plots können in Echtzeit gezeichnet werden und empfangene Daten können auch in Form von Dateien zur späteren Verwendung und Analyse gespeichert werden. Das System wurde preisgünstig, unter Berücksichtigung eines einfachen Aufbaus, entwickelt. Der vorgeschlagene Aufbau ist sehr flexibel und kann einfach an veränderte Anforderungen angepasst werden.

Als verteiltes Steuerungsschema wurde ein zentralisierter, heterogener Formationsflug Positionsregler formuliert, der einen explicit model following Linear Quadratic Regulator Proportional Integral (LQR PI) Regler verwendet. Der Anführer Quadrocopter ist ein stabiles Referenzmodell mit der gewünschten Dynamik, deren Ausgang vollkommen von den beiden Wingmen Quadrocopter verfolgt wird. Der Anführer selbst wird durch Pole Placement Steuerverfahren mit den gewünschten Stabilitätseigenschaften gesteuert, während die beiden Anhänger durch robuste und adaptive LQR PI Steuerverfahren geregelt werden. Für diese Studie wird ein Vollzustandsvektor der Quadrocopter betrachtet während nur die resultierende Leistung verfolgt wird. Die ausgewählte 3D Formationsgeometrie und die statische Stabilität bleibt unter einer Vielzahl von möglichen Störungen erhalten. Bei Kommunikationsverlust zwischen Anführer und einem der Anhänger, leitet der andere Anhänger die Daten, die er vom Anführer erhalten hat,

an den betroffenen Anhänger weiter. Die Stabilität des Regelsystems wurde unter Verwendung von Singulärwerten analysiert. Der vorgeschlagene Ansatz für eng gekoppelten Formationsflug von MUAVs wurde mit Hilfe von umfangreichen Simulationen unter MATLAB / Simulink validiert und ergab viel versprechende Ergebnisse. Auch die Tracking-Leistung wurde für zeitlich veränderliche Befehle gezeigt. Die vorgeschlagene Architektur ist skalierbar und kann problemlos erweitert werden. Dieser Ansatz ist für die Szenarien geeignet, die eng gekoppelte Formationsflug benötigen, wie kooperatives Greifen oder gemeinsame Lasttransporte.

Ein innovatives Framework für die Teamarbeit von zwei Quadrocopter Flotten wurde entwickelt. Als Beispielmision wurde ein Szenario gewählt, bei dem ein Feuer auf einer größeren Fläche gelöscht werden muss. Jede Formation hat ihre angegebene Formationsgeometrie und eine zugewiesene Aufgabe. Die Lageregelung für die Quadrocopter in einer der Formationen wurde durch ein LQR PI-Regelschema, das auf explicit model following basiert, umgesetzt. Die Quadrocopter in anderen Formation werden durch ein LQR PI Servomechanismus Regelsystem gesteuert. Die beiden Steuersysteme werden in Bezug auf ihre Leistung und ihren Steuerungsaufwand verglichen. Beide Formationen werden durch entsprechende Bodenstationen durch virtuelle Anführer kommandiert. Die Bodenstationen tauschen die befohlene Höheninformation aus, um gegenseitig eine sichere Trennung zwischen den Formationen zu gewährleisten. Die Quadrocopter können kommandierte Solltrajektorien folgen und über erwünschten Punkten für eine vorgegebene Zeit schweben. Bei Kommunikationsverlust zwischen Bodenstation und einem der Quadcopter leitet der benachbarte Quadrocopter die Befehlsdaten, die er von der Bodenstation erhalten hat, an die betroffene Einheit weiter. Das vorgeschlagene Framework wurde durch umfangreiche Simulationen mit Hilfe von MATLAB / Simulink validiert und liefert sehr brauchbare Ergebnisse. Cluster-Rekonfiguration von Agenten wird in unserer Arbeit ebenfalls gezeigt. Dies erlaubt es die Formationsgeometrie während des Fluges auf eine beliebige neue Form umzuschalten. Für die genannten Anwendungen sind Konsens Algorithmen nicht erwünscht, da wir von den Quadrocopter Flotten fordern, dass sie dem von uns gewählten Weg folgen, und nicht ihren Weg selbst wählen.

Eine Reihe der praktischen Probleme von Kommunikationsnetzen kann in geeigneter Weise durch Graphen dargestellt werden. Dies erleichtert die Problemformulierung und den Analyseprozess. Kommunikationstopologien für Netzwerke mit einer großen Anzahl

von Einheiten, wie zum Beispiel Schwärme von Luftfahrzeugen, können durch einen graphentheoretischen Ansatz untersucht werden. Um die Bildung solcher Probleme zu erleichtern, wird der Graph mit Hilfe der Laplace-Matrix dargestellt. Eigenwerte der Laplace-Matrix wurden in unserer Studie angemessene Berücksichtigung gegeben einen Einblick in die Graphen / Subgraphen Eigenschaften zu verleihen. Der gleiche wurden genutzt um die bekannte Euler Formel zu verallgemeinern und somit auf Graphen und Subgraphen anwendbar zu machen. Eine modifizierte Euler-Formel wird ebenfalls vorgestellt. Die Verwendung der Graphentheorie in verteilten / kooperativen Regelsystemen wird auch durch Simulationen gezeigt.

Kooperative Kontrolschemas, die auf auf Konsens-Algorithmen beruhenden, wurden für die Lageregelung von Quadrocopter-Flotten, in denen kein expliziter Anführer existiert, verwendet. Konsens-Algorithmen wurden in Kombination mit verschiedenen Steuersystemen verwendet, was zur Autonomie von Quadrocoptern beiträgt. Die Steuersysteme, die für diesen Zweck verwendet werden, umfassen LQR PI-Regelung basierend auf model following und LQR PI Servo-Mechanismus. Die Regelungen wurden unter verschiedenen Kommunikationstopologien untersucht, darunter voll verbundene ungerichtete Graphen, gerichteten Graphen und Zyklus-Topologie. Der Informationsfluss unter den Agenten in einem Cluster wurde durch Laplace-Matrix modelliert. Die Auswirkungen von Eingang Verzerrungen auf Konsens Werte wurden ebenfalls untersucht. Quadrocopter können durch gegenseitigen Konsens Flugbahnen verfolgen und die Zielpunkte erreichen. Die vorgeschlagenen Regelungssysteme wurden unter verschiedenen Kommunikationstopologien in Matlab / Simulink-Umgebung durch umfangreiche Simulationen validiert. Die Ergebnisse bescheinigen die Wirksamkeit der präsentierten Schemata mit dem zusätzlichen Vorteil der Einfachheit der Umsetzung. Das vorgeschlagene Regelungssystem ist skalierbar für große Gruppen von MUAVs.

Für Formationsflug sind die Anforderungen an die Positionsgenauigkeit sehr hoch. GPS-Signale allein bieten keine ausreichend hohe Positionsgenauigkeit um die Anforderung zu erfüllen; eine Technik für die genauere Positionsbestimmung ist daher erforderlich, beispielsweise DGPS. Es existiert eine Anzahl von öffentlichen Codes für die GPS-Positionsbestimmung und Baseline-Bestimmung im Offline-Modus. Es existiert jedoch keine Software für DGPS, die Korrekturfaktoren der Basisstationen nutzt, ohne auf Doppel Differenz Informationen zu vertrauen. Um dies zu erreichen, wurde eine Methodik in MATLAB-Umgebung für DGPS mit C/A Pseudoranges nur auf einzelne Frequenz

L1 eingeführt es machbar für Empfänger kostengünstig GPS zu nutzen. Unsere Basisstation wird an einem genau vermessen Referenzpunkt aufgestellt. Pseudorange und geometrische Abstände werden an der Basisstation verglichen, um die Korrekturfaktoren zu berechnen. Diese Korrekturfaktoren, für aller gültigen Satelliten während einer Epoche, werden dann an einen Rover übergeben. Das Rover berücksichtigt innerhalb der entsprechenden Epoche diese für seine eigene wahre Positionsbestimmung. Zur Validierung der vorgeschlagenen Algorithmen wird unsere Rover ebenfalls an einer vorbestimmten Stelle platziert. Der vorgeschlagene Code ist ein geeignetes und einfaches Werkzeug für die Nachbearbeitung von GPS-Rohdaten für eine genaue Positionsbestimmung eines Rover, z.B. eines UAV während der Post-Missionsanalyse.

List of Publications

Journal Papers

1. **Qasim Ali**, and Sergio Montenegro, "Decentralized control for scalable quadcopter formations," *International Journal of Aerospace Engineering*, (2016). (Accepted)
2. **Qasim Ali**, and Sergio Montenegro, "Explicit model following distributed control scheme for formation flying of mini UAVs," *IEEE Access Journal*, vol. 4, pp. 397-406, (2016).
3. **Qasim Ali**, and Sergio Montenegro. "A Matlab implementation of Differential GPS for low-cost GPS receivers." *TransNav: International Journal on Marine Navigation and Safety of Sea Transportation*, vol. 8. no. 3, pp. 343 - 350 (2014).
4. **Qasim Ali**, Nils Gageik, and Sergio Montenegro, "A review on distributed control of cooperating mini UAVs," *International Journal of Artificial Intelligence and Applications*, vol. 5, no. 4, pp. 1-13, (2014).

Conference Papers

1. **Qasim Ali**, and Sergio Montenegro, "Role of graphs for multi-agent systems and generalization of Euler's formula," in *Proceedings of 8th IEEE International Conference on Intelligent Systems*, Sofia, Bulgaria, September 2016. (Accepted)
2. **Qasim Ali**, and Sergio Montenegro, "A simple approach to quadcopter formation flying test setup for education and development," in *Proceedings of 9th International Technology, Education and Development Conference*, pp. 2776-2784, Madrid, Spain, March 2015.

The publications mentioned above have partly been used in this dissertation. The following table itemizes to what extent the different sections of the publications have been reused at which positions in this work. The sources of adapted figures and tables are additionally indicated at the end of the corresponding figure/table captions.

Publication	Usage	Dissertation
Ref. [1] p. 1 pp. 2-3 pp. 4-6 p. 7 pp. 8-10	Text partly reproduced and extended Text reproduced and extended, figures adapted Text reproduced with minor modifications, figure adapted Text partly reproduced and extended Text reproduced with minor modifications	p. 1 pp. 2-5 pp. 9-11, p.13 & p.15 p. 7 & p. 15 pp. 11-14, p. 21 & pp. 120-121
Ref. [2] p. 1 p. 2 p. 3 p. 4 p. 5 pp. 6-8	Text reproduced with minor modifications Text reproduced, modified and extended Text partly reproduced and extended, only figure 3 adapted Text reproduced with minor modifications, figures adapted Text reproduced with minor modifications Text reproduced and extended, figures adapted	p. 23 p. 14, p. 24 p. 24-25 pp. 26-27, p. 29, pp. 30-31 p. 32, pp. 34-37
Ref. [3] pp. 1-2 pp. 3-4 p. 5 pp. 6-7 pp. 8-9	Text partly reproduced, modified and extended Text reproduced with minor modifications, figures adapted Text reproduced with minor modifications figure adapted with modifications Text reproduced, modified and extended, table modified and extended Text reproduced, figures adapted except fig. 9	p. 15, pp. 38-39 pp. 39-44 pp. 44-46, p. 47 pp. 47-52 pp. 53-55
Ref. [4]	Text reproduced with minor modifications, Figure 6, 10, 11, 12 and 13 adapted with slight modification Figure 1, 3, 4, 7, adapted Figure 14 adapted with slight modification	pp. 56-69, p. 71 p. 69
Ref. [5]	Text reproduced with minor modifications Tables I adapted with slight modification Tables II and III adapted Figures 3, 4 and 5 adapted	pp. 72-84 p. 118
Ref. [6] p. 1 pp. 2-5 pp. 6-8	Text partly reproduced, modified and extended Text reproduced, modified and extended, table and figure adapted Text reproduced, modified and extended, Table adapted	pp. 100-101 pp. 102-111 pp. 111-116, pp. 119-120

For Ref. [1], Ref. [4] and Ref. [6], the copyrights of the paper are retained by the authors under the terms of Creative Commons (CC) Attribution license. This license authorizes reproduction, provided that the original work is properly cited.

For Ref. [2], the publisher IATED permits the authors to re-use all or portions of the work in other works, as per the Copyright Transfer Agreement, Part 3, Para "Retained Rights".

For Ref. [3], the IEEE does not require individuals working on a thesis to obtain a formal reuse license as per IEEE Open Access Publishing Agreement and the email reply received from IEEE on 15.01.2016.

For Ref. [5], the IEEE authorizes the authors to reproduce the work, provided that the source and the IEEE copyright notice are indicated. Para Retained Rights/Terms and Conditions of IEEE copyright and consent form refers in this regard.

Contents

Acknowledgements	i
Summary	iii
Kurzfassung	vii
List of Publications	xii
List of Figures	xx
List of Tables	xxiii
List of Abbreviations and Acronyms	xxiv
List of Symbols	xxvii
List of Constants	xxx
1 Introduction	1
1.1 Preface	1
1.1.1 Classification of Small UAVs	2
1.1.2 Constellation Architecture	3
1.1.2.1 Leader-Follower	4
1.1.2.2 Virtual Structure	4
1.1.2.3 Behavioural Approach	5

1.2	Architecture of Control Techniques for MAS	5
1.2.1	Synopsis of Control Theory	5
1.2.2	Architectures of Control Techniques for MAS	6
1.2.3	Distributed Systems	7
1.3	MUAV Model Identification	7
1.4	Sensors for MUAV Formation Flight	8
1.4.1	Obstacle Detection and Collision Avoidance	9
1.4.2	Attitude and Heading Determination	9
1.4.3	GNSS Receiver	9
1.4.4	PMD Camera	10
1.5	Information Exchange Among the Agents	11
1.5.1	Communication Requirements	11
1.6	Related Hardware and Software Tools	12
1.6.1	Hardware	12
1.6.2	Software Tools	12
1.7	State of the Art	13
1.7.1	Formation Flight Test Setups	14
1.7.2	Control Techniques	15
1.7.2.1	Consensus Algorithms for Cooperative Control	16
1.7.3	Graph Theory	17
1.7.4	Differential GPS	18
1.8	Statement of Contributions	19
1.9	Summary	21
1.10	Thesis Outline	21
2	Quadcopter Formation Flying Test Environment	23
2.1	Introduction	23
2.2	Formation Flying Test Environment	24
2.2.1	Hardware	24
2.2.1.1	Engine	26
2.2.1.2	Inertial Measurement Units (IMUs)	26
2.2.1.3	Two Wire Interface (TWI) Cable	28
2.2.1.4	Quadcopter and IMU Orientation	28
2.2.1.5	IMU Calibration on Test Bench	29
2.2.2	Software	29
2.2.2.1	AVR32 Studio	29
2.2.2.2	Terminal Program HTerm	30
2.2.2.3	MATLAB Instrument Control Toolbox	30
2.3	Inter-Communication	31
2.3.1	Information to be Shared between MUAVs	31
2.3.2	Inter-Communication between MUAVs	31
2.3.2.1	Bluetooth Communication	32

2.3.2.2	IMU Data Transmission	32
2.3.2.3	GPS Data Transmission	33
2.4	Control Technique	34
2.5	Results	36
2.6	Summary	37
3	Formation Flying of Quadcopters through Explicit Model Following Distributed Control Scheme	38
3.1	Introduction	38
3.2	Problem Formulation	39
3.2.1	Quadcopter Dynamic Model	39
3.2.2	Formation Dynamic Model	42
3.3	Control Strategy	45
3.4	Simulation Results and Discussion	48
3.5	Stability Analysis	54
3.6	Summary	54
4	Decentralized Control Scheme for Quadcopter Formations	56
4.1	Introduction	56
4.2	Architecture	57
4.3	Problem Formulation	59
4.3.1	Quadcopter Dynamic Model	60
4.3.2	Formation Dynamic Model	61
4.4	Control Strategies	63
4.4.1	Comparison of Control Schemes	65
4.5	Simulation Results and Analysis	66
4.5.1	Trajectory Tracking Performance	66
4.6	Cluster Reconfiguration of Agents	69
4.7	Summary	70
5	Graph Theory for Distributed Control	72
5.1	Introduction	72
5.2	Notations and Preliminary Background	73
5.2.1	Algebraic Connectivity and Its Utility	76
5.3	Graph Properties and Propositions	76
5.4	Generalization of Euler's Formula	80
5.4.1	Generalization of Euler's Formula for Planar Graphs	80
5.4.2	Euler's Modified Formula	80
5.4.3	Generalization of Euler's Formula for 3D Graphs	82
5.5	Graph Theory in Distributed/Cooperative Control	82
5.6	Summary	84
6	Consensus based Cooperative Control Schemes for Quadcopters Fleet	85

6.1	Introduction	85
6.1.1	Cooperative Control	86
6.1.2	Discrimination of Cooperative Control	86
6.2	Problem Formation	87
6.2.1	Consensus Dynamics	87
6.2.2	Input Bias	88
6.3	Simulation Results and Analysis	89
6.3.1	Consensus Algorithm with LQR PI Servomechanism (Two Agents)	90
6.3.2	Consensus Algorithm with LQR PI Controller based on Model Following (Two Agents)	90
6.3.3	Consensus Algorithm with LQR PI Control Scheme with Undi- rected Topology	93
6.3.4	Consensus Algorithm with LQR PI Control Scheme with Directed Topology	96
6.3.5	Weighted-Average Consensus Algorithm with LQR PI Control Scheme with Undirected Topology	96
6.3.6	Switching Consensus Values	98
6.4	Summary	98
7	Differential GPS Implementation	100
7.1	Introduction	100
7.1.1	Merits and Demerits of GPS	100
7.1.2	GPS Signals	101
7.1.3	GPS Navigation Solution	101
7.1.4	Why DGPS is Required?	102
7.1.5	Characteristic of Present Study	102
7.2	Differential GPS Techniques	103
7.2.1	GPS Error Sources	103
7.2.2	Correction Methodologies	104
7.2.3	DGPS Implementation Modes	105
7.2.4	DGPS Accuracy	105
7.3	RINEX Files	106
7.3.1	RINEX Files Classification	106
7.3.2	Conversion of GPS Binary Data to RINEX Format	107
7.4	DGPS Software and Simulation Results	107
7.4.1	Mathematical Model for Correction Factor (CF)	110
7.4.2	Simulation Results and Discussion	111
7.4.3	Reasons for not Using Double Differences	113
7.4.4	Reasons for not Using Dual Frequency	115
7.4.5	Technical Constraints	115
7.5	Summary	116

8 Conclusion	117
8.1 Future Works	119
8.2 Areas Requiring Attention	120
A Matrices Used for Simulations	122

List of Figures

1.1	Quadcopter developed at University of Würzburg. Figure adapted from Ref. [1].	2
1.2	Cooperative emergency quadcopter. Figure adapted from Ref. [1].	2
1.3	Interception of an intruder through a group of quadcopters.	3
1.4	Birds in leader-follower formation. Figure adapted from Ref. [1].	4
1.5	A typical output feedback system.	6
1.6	A state feedback system using an observer.	6
1.7	Model identification of quadcopter using Matlab.	8
1.8	Front and side view of a PMD camera. Figure adapted from Ref. [1].	11
1.9	Modules for distributed control of cooperating agents.	22
2.1	Quadcopter test bench mechanism.	25
2.2	Bluetooth module BTM-222. Figure adapted from Ref. [2].	25
2.3	RN41-I/RM Bluetooth module.	25
2.4	AVR32 test board (left) and AVR32 small board (right). Figure adapted from Ref. [2].	26
2.5	IMU3000 Combo. Figure adapted from Ref. [2].	27
2.6	Test environment for different modules.	27
2.7	IMU LSM303DLM (in ready to use configuration).	27
2.8	LSM303DLM connected to AVR32 board during testing.	28
2.9	Testing for IMU data transmission between AVR32 small boards.	33
2.10	Testing for GPS data transmission.	34
2.11	Layout for complete test setup. Figure adapted from Ref. [2].	34
2.12	Block diagram for PID control scheme.	35
2.13	Block diagram showing PID control in leader-follower architecture. Figure adapted from Ref. [2].	36
2.14	Attitude information of both quadcopters. Figure adapted from Ref. [2] and slightly improved.	37
3.1	Communication topology of quadcopters in formation. Figure adapted from Ref. [3]. © (2016) IEEE.	40
3.2	Quadcopter motion dynamics. Figure adapted from Ref. [3]. © (2016) IEEE.	40

3.3	LQR PI control scheme for leader-follower architecture. Figure partly adapted from Ref. [3]. © (2016) IEEE.	47
3.4	Sub-block level diagram of LQR PI control scheme.	47
3.5	Three-axes position of whole formation under disturbances.	50
3.6	Trajectory for leader and followers under disturbances.	50
3.7	Three axes position of units under switching formation geometry.	51
3.8	Inter-unit separations in the formation.	51
3.9	Response of formation under a sinusoidal command.	52
3.10	Trajectory of formation for an extended distance.	52
3.11	Control input for throttle and moments to regain equilibrium. Figure adapted from Ref. [3]. © (2016) IEEE.	53
3.12	Control voltages of four motors of quadcopter. Figure adapted from Ref. [3]. © (2016) IEEE.	54
3.13	Sigma plot for leader-follower scheme in open-loop and closed-loop. Figure adapted from Ref. [3]. © (2016) IEEE.	55
4.1	Architecture for two formations. Figure adapted from Ref. [4].	58
4.2	The mission scenario for fire extinguishment by quadcopter formations.	59
4.3	Communication topology for formation 1 and 2. Figure adapted from Ref. [4].	60
4.4	LQR PI control scheme with reference model.	64
4.5	LQR PI servomechanism control scheme. Figure adapted from Ref. [4].	65
4.6	Comparison of control effort for two control schemes.	66
4.7	Comparison of tracking performance for two control schemes.	66
4.8	Quadcopters joining to make desired formation geometry. Figure partly adapted from Ref. [4].	67
4.9	Trajectory tracking by two formations. Figure partly adapted from Ref. [4].	68
4.10	Sinusoidal motion of formation 2. Figure partly adapted from Ref. [4].	68
4.11	Formation 2 in hovering mode. Figure partly adapted from Ref. [4].	69
4.12	Same trajectory tracking by two formations. Figure adapted from Ref. [4].	69
4.13	Cluster reconfiguration of agents in 3D view.	70
4.14	Cluster reconfiguration of agents on time scale.	71
5.1	An undirected graph with four vertices.	74
5.2	A directed graph with four vertices.	75
5.3	Block diagram for consensus algorithm by two clusters of quadcopters. Figure adapted from Ref. [5]. © (2016) IEEE.	83
5.4	Communication topology for cluster 1 and 2. Figure adapted from Ref. [5]. © (2016) IEEE.	83
5.5	Consensus for hovering heights by two clusters. Figure adapted from Ref. [5]. © (2016) IEEE.	84

6.1	Block diagram for consensus algorithm with input bias.	89
6.2	Cooperative control scheme with consensus algorithm.	90
6.3	Consensus for two agents with input bias on X-axis (3D view).	91
6.4	Consensus for two agents with input bias on X-axis.	91
6.5	Consensus for two agents without input bias.	92
6.6	Consensus for two agents with input bias on both agents.	92
6.7	Consensus for two agents with input bias on one agent only.	93
6.8	Consensus for three agents to make a triangle.	93
6.9	Consensus for three agents with input bias.	94
6.10	Consensus for three agents in X- and Y-axis.	95
6.11	Consensus for three agents in X- and Y-axis without input bias.	95
6.12	Consensus for three agents in X- and Y-axis for directed topology of S. No. 3 in Table 6.2.	97
6.13	Consensus for three agents in X- and Y-axis for directed topology of S. No. 4 in Table 6.2.	97
6.14	Weighted-average consensus for undirected topology of S. No. 1 in Table 6.2.	98
6.15	Switching consensus for undirected topology of S. No. 1 in Table 6.2.	99
7.1	A quadcopter receiving signals from four GPS satellites.	101
7.2	Pictorial view of DGPS technique.	103
7.3	Flow chart for DGPS implementation. Figure adapted from Ref. [6].	109
7.4	Position error with and without DGPS in three-axes.	112
7.5	Radial position error with and without DGPS.	112
7.6	Relative position of rover for one epoch.	113
7.7	DOP values at rover.	113
7.8	Visible satellites and PDOP for 30° elevation mask.	114
7.9	Visible satellites and PDOP for 5° elevation mask.	114

List of Tables

3.1	Metadata for Simulation. Table adapted and extended from Ref. [3]. © (2016) IEEE.	49
5.1	Sample Graphs and Corresponding Metadata. Table adapted with small modifications from Ref. [5]. © (2016) IEEE.	78
5.2	Sample Subgraphs and Corresponding Metadata. Table adapted from Ref. [5]. © (2016) IEEE.	79
5.3	Merging of Subgraphs. Table adapted from Ref. [5]. © (2016) IEEE. . .	81
6.1	Communication Topologies and Related Metadata for Two Agents. . . .	91
6.2	Communication Topologies and Related Metadata for Three Agents. . . .	94
7.1	Classification of RINEX Files. Table adapted from Ref. [6].	107
7.2	Statistical Results. Table adapted from Ref. [6].	111

List of Abbreviations and Acronyms

ARE	Algebraic Riccati Equation
C/A	Course Acquisition
CF	Correction Factor
CGT	Command Generator Tracker
CPDGPS	Carrier Phase Differential Global Positioning System
DAAD	Deutscher Akademischer Austausch Dienst (Germany)
DFG	Deutsche Forschungsgemeinschaft (Germany)
DGNSS	Differential Global Navigation Satellite System
DGPS	Differential Global Positioning System
DLR	Deutsches Zentrum fuer Luft-und Raumfahrt (Germany)
DMPC	Distributed (Decentralized) Model Predictive Control
DOP	Dilution Of Precision
ECEF	Earth Centred Earth Fixed
EGNOS	European Geostationary Navigation Overlay Service
ETH	Eidgenossische Technische Hochschule (Zuerich)
FPGA	Field-Programmable Gate Array
GCC	GNU C Compiler
GDB	GNU Debugger
GDOP	Geometric Dilution Of Precision
GEO	Geostationary Earth Orbit
GLONASS	GLObal NAVigation Satellite System
GNC	Guidance Navigation and Control
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
GRASP	General Robotics, Automation, Sensing and Perception

HEC	Higher Education Commission (Pakistan)
ICS	Institute of Control Systems (TUHH)
ICT	Instrument Control Toolbox
IDE	Integrated Development Environment
IGS	International GNSS Service
IMU	Inertial Measurement Unit
IR	Infra Red
JTAG	Joint Test Action Group
LQR	Linear Quadratic Regulator
LQR PI	Linear Quadratic Regulator Proportional Integral
LTI	Linear Time Invariant
MAS	Multi Agent System
MEMS	Micro Electro Mechanical Systems
MIMO	Multi-Input Multi-Output
MIT	Massachusetts Institute of Technology (USA)
MPC	Model Predictive Control
MSAS	Multi-functional Satellite Augmentation System
MUAV	Mini Unmanned Aerial Vehicle
NAVSTAR	NAVigation Satellite Timing And Ranging
NMPC	Non-linear Model Predictive Control
OFDM	Orthogonal Frequency Division Multiplexing
PC	Professional Computer
PD	Proportional Differential
PDOP	Position Dilution Of Precision
PID	Proportional Integral Differential
PMD	Photonic Mixer Device
PPP	Precise Point Positioning
PPS	Precise Positioning Service
PRN	Pseudo-Random Noise
PWM	Pulse Width Modulation
RHC	Receding Horizon Control
RINEX	Receiver INdependent EXchange (files)
RODOS	Real-time On-board Dependable Operating System
RTCM	Radio Technical Commission for Maritime-Service

RTK	Real Time Kinematics
RTKLIB	Real Time Kinematics LIBrary
RTOS	Real Time Operating System
RTW	Real Time Workshop (MATLAB)
SAPOS	SATelliten POSitionierungs dienst
SBAS	Satellite Based Augmentation System
SNR	Signal to Noise Ratio
SPP	Serial Port Profile
SPS	Standard Positioning Service
SV	Satellite Vehicle
TDOP	Time Dilution Of Precision
TUHH	Technical University Hamburg-Harburg (Germany)
TWI	Two-Wire Interface
UAV	Unmanned Aerial Vehicle
USART	Universal Synchronous/ Asynchronous Receiver/Transmitter
UTC	Universal Time Coordinated
VL	Virtual Leader
VTOL	Vertical Take-Off and Landing
WAAS	Wide Area Augmentation System

List of Symbols

A	State matrix (for state space systems) Adjacency matrix (for graph theory)
a	Semi-major axis of ellipsoid
B	Input matrix (or Control matrix)
b	Semi-minor axis of ellipsoid
C	Output matrix
$C1$	C/A code pseudoranges on L1 frequency
c	Speed of light in vacuum
D	Feed-through matrix (for state space systems) Degree matrix (for graph theory)
$D1$	Doppler on L1 frequency
dT	Other GPS measurement biases (expressed as lump sum)
dt_r	Receiver clock offset
dt^s	Satellite clock offset
E	Edge of a graph
$ E $	Number of edges (of a graph)
e_y	Integrated tracking error
$ F $	Number of faces (of a polyhedron)
f	Frequency
f_i	Force produced due to rotation of rotor i
f^R	Received frequency
f^T	Transmitted satellite signal frequency
Inc	Incidence matrix
J	Cost function
j	Unit vector pointing along the line of sight from receiver to the satellite

K	Gain matrix
K_D	Derivative gain (a tuning parameter for PID controller)
K_I	Integral gain (a tuning parameter for PID controller)
K_P	Proportional gain (a tuning parameter for PID controller)
k	Consensus value
$ \mathbf{L} $	Number of loops (of a graph)
$L1$	Primary carrier frequency
$L2$	Secondary carrier frequency
m	Mass of quadcopter
\mathbf{P}	Controllability matrix
r	Geometric range
	Algebraic multiplicity
$r(t)$	Reference command as a function of time
$S1$	Signal to noise ratio on L1 frequency
T	Tropospheric delay
u	Total thrust generated by four rotors of quadcopter
\mathbf{V}	Vertex of a graph
$ \mathbf{V} $	Number of vertices (of a graph)
x_f	States of follower quadcopter
x_l	States of leader quadcopter
x_q	States of quadcopter being controlled from ground station
y_f	Performance output vector of follower quadcopter
y_l	Performance output vector of leader quadcopter
y_r	Performance output vector of reference model
Δf	Doppler offset
λ	Longitude
	Eigenvalue
\mathcal{L}	Laplacian matrix
m_1^L	Algebraic multiplicity of zero eigenvalues of a Laplacian matrix
ρ	Pseudorange
ϕ	Latitude
ψ	Yaw angle (around the z-axis)
θ	Pitch angle (around the y-axis)
ϕ	Roll angle (around the x-axis)

Υ	Weighting factor
τ_ψ	Control input for yawing moment
τ_θ	Control input for pitching moment
τ_ϕ	Control input for rolling moment
ζ	State vector
\mathbf{U}	Control vector
ω_i	Angular speed of motor i
δy	Tracking error

List of Constants

<i>c</i>	2.99792458×10^8 m/sec (Speed of light)
<i>g</i>	9.8 <i>m/sec</i> ² (Gravitational constant)
<i>L1</i>	1575.42 MHz (GPS L band primary carrier frequency)
<i>L2</i>	1227.60 MHz (GPS L band secondary carrier frequency)

To my dear wife . . .

Chapter 1

Introduction

1.1 Preface

In this chapter, an overview of the main aspects in the domain of distributed control of cooperating MUAVs is provided to facilitate the potential users in this fascinating field. It also gives details on state of the art in MUAV technologies e.g. Photonic Mixer Devices (PMD) camera, distributed control methods and on-going work and challenges, which is the motivation for many researchers all over the world to work in this field. MUAVs can be regarded as flying robots that may operate in three-dimensional space, like quadcopter shown in Figure 1.1. Quadcopter is also named as quadrocopter and quadrotor etc. The biggest advantages of MUAVs include their much lower cost compared with manned vehicles, risk avoidance for human pilots and their remote sensing capabilities. Single MUAV may be used for forest fire monitoring, oil pipeline inspection and flood damage assessment etc. However, some other interesting applications are envisaged that may not be performed efficiently by a single MUAV and necessitate the use of multiple units. Such valuable applications include traffic monitoring, aerial mapping, search and rescue, mobile communication relays, pesticide spraying and weather monitoring etc. These platforms are becoming more and more multifaceted as the sensors are miniaturized and on-board computing power is enhanced. Although these vehicles have several advantages compared with manned platforms, however control requirements are comparatively more stringent and therefore generally require more sophisticated control techniques. Cooperative control for MUAVs poses strict performance requirements like task coupling, robustness, optimality and scalability [7]. Highly non-linear flight dynamics of MUAVs make it demanding to determine real-time relative position and attitude vis-a-vis data latency and communication data packets loss.

Cooperation behaviour of units implies some degree of coupling among the units. Generally speaking, greater the degree of coupling, more challenging it is to formulate effective cooperative solutions [8]. This becomes more obvious taking into consideration that a



FIGURE 1.1: Quadcopter developed at University of Würzburg.
Figure adapted from Ref. [1].

swarm of MUAVs need to cooperate in a way that it can rescue a person from a burning house as depicted in Figure 1.2. Another interesting application is envisaged where a group of quadcopters may be exploited to intercept an intruder through a net. An exemplary scenario to this effect is shown in Figure 1.3.

1.1.1 Classification of Small UAVs

Small UAVs are divided into two major categories; mini UAV and micro UAV. Mini UAV wing span is between 0.5 – 2 meters, while that of micro UAV is between 0.1 – 0.5



FIGURE 1.2: Cooperative emergency quadcopter.
Figure adapted from Ref. [1].

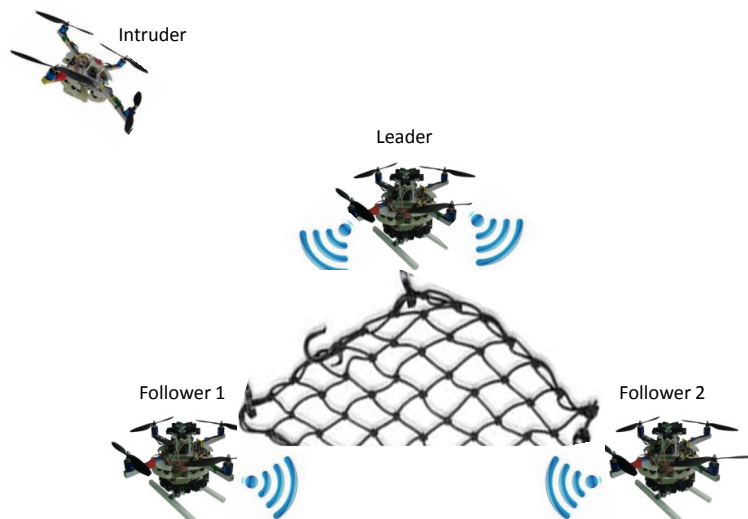


FIGURE 1.3: Interception of an intruder through a group of quadcopters.

meters [9]. MUAVs are classified into fixed wing and rotary wing aircraft (rotorcraft), each type having its own merits and demerits. Rotorcraft include Twinrotor, Trirotor, Quadcopter, Hexacopter and Octocopter etc. Rotorcraft differ from fixed wing planes in many perspectives. Fixed wing aircraft have control surfaces like aileron, rudder and elevator etc. Trimming surfaces are also generally available on primary control surfaces. Quadcopters do not have control surfaces and trimmers, rather control dynamics are to be influenced by the differential speeds of rotors which poses a challenge from control perspective. A fixed wing plane generally does not fall down like a stone and has the margin to regain stability in case of some failure, however rotorcraft may fall like a stone. All rotorcraft have Vertical Take Off and Landing (VTOL) capability.

Mini rotorcraft are drawing the attention due to their agility, ability to hover over desired points and no requirement for a take-off/ landing strip. These vehicles also have the capability to operate in limited workspace. Due to off-center location of propellers, fast rotational dynamics are possible that may be exploited. Their drawbacks include less speed and endurance. Merits of fixed wing and VTOL are combined in the form of Tail-Sitters and Quad Tilt-Rotor convertible MUAV [10].

Quadcopter is the most popular rotorcraft due to simplicity in its design and ease of construction. It has simple structure but requires complex control system due to non-linear dynamics, hence drawing attention of researchers around the globe. Quadcopter is the focus of our thesis.

1.1.2 Constellation Architecture

Selection of suitable constellation architecture is important for the formation control problem. Centralized strategies are usually easier to design than decentralized strategies



FIGURE 1.4: Birds in leader-follower formation.
Figure adapted from Ref. [1].

[8]. However decentralized algorithms reduce the communication between units and improve the robustness. Mainly three types of architecture have been formulated in the literature namely, leader-follower, virtual structure, and behavioural approach [11].

1.1.2.1 Leader-Follower

Most of the Multi-Agent System (MAS) control researches have been carried out with leader-follower scheme. This is a concept taken from nature and an example can be seen in the formation flight of birds, as shown in Figure 1.4. With this approach, some vehicles are designated as leaders while others are treated as followers. For small formations, there may be only one leader. The states of the leader constitute the coordination variable, since the actions of the other vehicles in the formation are completely specified once the leader states are known [12]. This architecture is easier to understand and implement. However, this approach lacks robustness with respect to leaders failure. Though virtual leader strategy is also proposed to improve its robustness.

1.1.2.2 Virtual Structure

In this scheme, the entire formation is treated as a single virtual rigid body structure. Rather than following a path, each vehicle follows a moving point, which allows the virtual structure to potentially be attached to another vehicle. The guidance of a group is easier than other approaches since all agents in the formation are treated as a single object. But the formation can only perform synchronized manoeuvres, and it is difficult to consider obstacle avoidance.

1.1.2.3 Behavioural Approach

For this approach, several desired behaviours are prescribed for each vehicle, including formation keeping, goal seeking, and collision/ obstacle avoidance. The control action of each vehicle is a weighted average of the control for each behaviour. It is suitable for uncertain environments, however lacks a rigorous theoretic analysis. There are other approaches, e.g. [13], where a consensus based decentralized approach has been used for controller design. In order to provide redundancy, no explicit leader exists for such cases.

This chapter is structured as follows; Section 1.2 covers the control techniques and relevant material. Section 1.3 gives basic details for MUAV model identification. Section 1.4 informs about the sensors used/ to be used for MUAV cooperative flight. Information exchange among the units is covered in Section 1.5. Related information about hardware and software tools may be seen in Section 1.6. Section 1.7 describes the state of the art.

1.2 Architecture of Control Techniques for MAS

1.2.1 Synopsis of Control Theory

Control theory deals with how to influence the behaviour of a dynamical system with the help of a designed controller. Behavior is generally modified based on feedback and/or system dynamical model. Feed-forward control approach is also adopted when exact dynamical model is known. However due to inherent limitations of feed-forward control schemes, we will restrict our focus in this thesis to feedback control schemes. Feedback helps to get rid of disturbances. Performance of a controller is generally measured in terms of following:

1. Stability
2. Set-point tracking
3. Robustness to change in parameters
4. Transient response
5. Steady state error, and
6. Disturbance rejection

A typical output feedback system is shown in Figure 1.5. Here $r(t)$ refers to reference commands, $u(t)$ is control vector, $y(t)$ is the system output vector and $e(t)$ is the error vector between reference values and output vector. The states of a system are generally

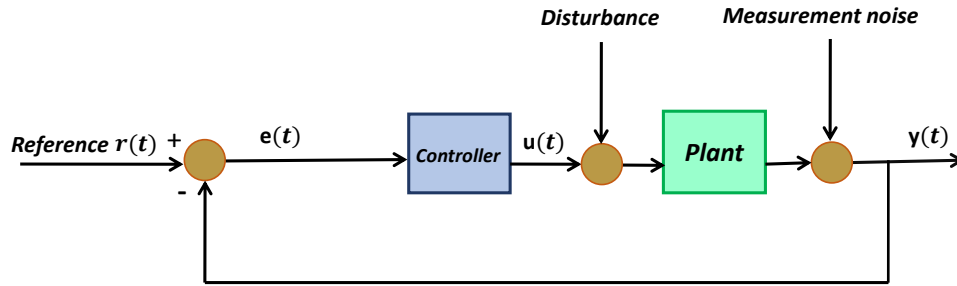


FIGURE 1.5: A typical output feedback system.

not available for measurement, however these may be estimated using an *Observer* or a *Kalman filter*. Observers and Kalman filter are dynamical systems that estimate the full state from measurements of the system outputs [14]. A state feedback control using an observer is shown in Figure 1.6. Here control signals $u(t)$ and outputs $y(t)$ are exploited to estimate the states $\hat{x}(t)$ and outputs $\hat{y}(t)$.

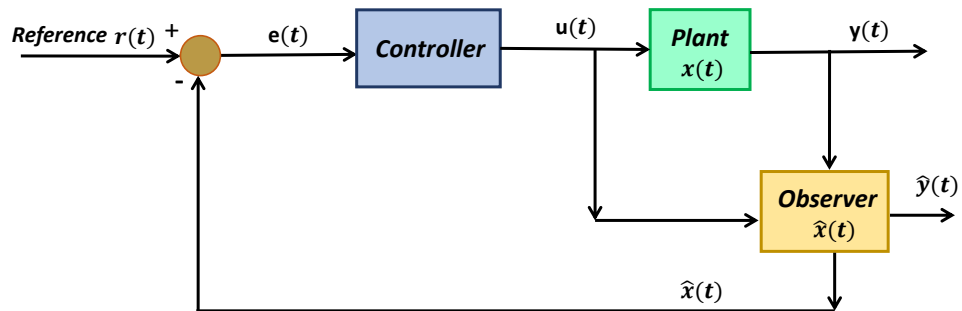


FIGURE 1.6: A state feedback system using an observer.

1.2.2 Architectures of Control Techniques for MAS

Here we describe the differences among three basic architectures for MAS, namely distributed control, decentralized control and cooperative control. For **Distributed Control**, each agent is equipped with a local controller C_i which receives information not only on the state of agent i but also on the state of a subset of other agents in the formation. While in **Decentralized Control**, each agent is controlled by a local controller C_i which accesses the state of agent i with no information exchange among the other agents, as defined in [15]. Yet another domain is that of **Cooperative Control**, where multiple dynamic entities share information or tasks to accomplish a common objective, that is greater than the purpose of each individual. In this case, autonomous agents are self-executing and not relying on external input to initiate behaviors.

1.2.3 Distributed Systems

A modern trend in transportation field is to distribute the electronic systems. Networking has thus become an essential part of system design for connecting independent control units. Communication over the network is vastly growing in modern electronic systems with the advent of smart sensors and actuators. This trend will increase the number of distributed systems with distributed hardware, control and data. It would lead to distribution of functions, control algorithms and data to improve hardware usage. This distribution however poses new problems which are not encountered in monolithic systems, for e.g., data inconsistency due to transmission delays and communication data packets loss causing inconsistencies in identical functions. Safety related systems are to be taken care of for such scenarios. Design of distributed systems should be able to avoid these so called negative effects of distribution [16].

1.3 MUAV Model Identification

Model identification refers to the determination of an accurate mathematical model of a system that describes the system dynamics. A model defines how the inputs are related with the outputs, and helps to predict the behavior of the system under different conditions. The common forms of this model are *differential equations*, *transfer function* and *state space model*. Correlations among inputs, states and outputs are more obvious in case of state space models. Model identification of a MUAV is considered to be an important milestone when high accuracy is aimed and sophisticated control techniques are planned to be applied.

As quadcopter is a highly non-linear system with fast variation of parameters (system attributes), it is demanding to get an accurate mathematical model to describe its dynamics. Situation becomes even more cumbersome when its configuration also keeps on changing due to different sensors installed. Performance requirements are even higher for some applications like formation flight of quadcopters in close vicinity.

Not all control techniques require the system to be modelled. Adaptive neural network controller does not require an accurate mathematical model and is suitable for multi-variable flight control. The PID and neural network autopilots are non-model based, however the optimality and robustness of the controller cannot be guaranteed. As it is hard to get an accurate non-linear model, a linear model can also be used to approximate the MUAV dynamics [11]. Most of the systems by nature are nonlinear, however these systems may be approximated to linear systems as these are easy to analyze. System can be linearized in a particular region. Linearized model of an aerial vehicle is a viable solution for ordinary flight conditions.

From safety point of view, most of the proposed algorithms require to be simulated before actual flights may be undertaken. It also helps to authenticate the efficacy of

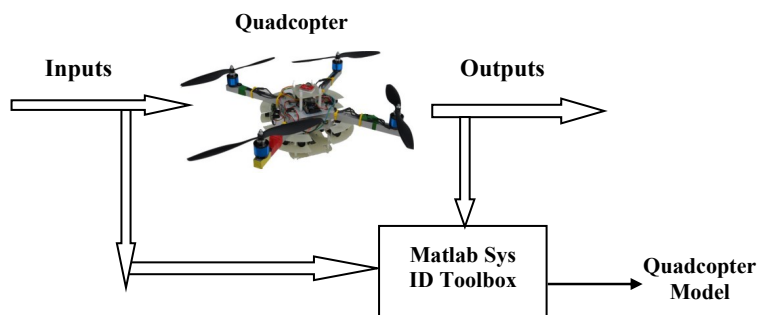


FIGURE 1.7: Model identification of quadcopter using Matlab.

the algorithm. However for simulation purpose, a mathematical model of the system under consideration is mandatory. Health of simulation results directly corresponds to accuracy of the modelling. The obligation for model identification is application dependent. From MUAV point of view, its model can also be identified using MATLAB System Identification Toolbox [17]. Main concept for model identification using this toolbox is presented in Figure 1.7. System is characterized with its response to input signals. The input and output values of MUAV are to be measured in a time-tagged fashion and logged in a file for subsequent feeding to the toolbox. Some methods may also require the reference signals to be logged. An example of model identification may be seen in [18] that describes the results of modelling, parameter identification and control of the rotational axes of a quadcopter. In this study, a Grey Box based iterative parameter identification approach was exploited.

It is worth-mentioning that accurate model of a quadcopter may be identified through flight experiments and noting the time tagged input and output data for further processing. Though real-flight experiments require more time and effort compared with those carried out on a test bench. Results may be compromised to some extent when process is realized on a test-rig. Degree of trade-off depends upon the off-set of quadcopter center of mass from true position, induced friction, deflection of mounting rod and non-linearities of the test-rig. As open-loop identification is not feasible due to highly unstable and non-linear behavior of quadcopter, so generally closed-loop identification is attempted. Selection of appropriate control input signals is important for the system to disclose all of its modes. Two types of excitation signals are generally used for model identification, namely Pseudo-Random Binary Sequence (PRBS) signals and Partial Response Maximum Likelihood (PRML) signals. Applied signals are required to be uncorrelated so that effect of different inputs could be decoupled.

1.4 Sensors for MUAV Formation Flight

We discuss some of the sensors vis-a-vis their purpose. Special emphasis is given to Global Navigation Satellite System (GNSS) receiver and PMD camera. Low-cost MEMS

sensors do not provide refined output, so we may need to use Kalman filter to get the *true* signal.

1.4.1 Obstacle Detection and Collision Avoidance

For MUAVs operating in close vicinity, it is vital for safe operation to avoid mutual collision. One option to tackle this situation may be to install multiple active sensors, like laser scanner or ultrasonic sensors [19], covering full 360° view. An alternate may be to equip the MUAVs with one GPS receiver each exploiting Differential GPS (DGPS) techniques. However for obstacles (like a wall), GPS is not helpful and we are bound to employ multiple sensors on all sides of a quadcopter. Ultrasonic and infra-red sensors or conventional cameras may be the good candidate for this purpose because of their low cost and power consumption. Small radar may also be a good sensor for obstacle detection [20]. Modern technologies like PMD camera and computer vision require on-board high computational power, but can provide better accuracy, resolution and coverage than ultrasonic or infra-red sensors.

1.4.2 Attitude and Heading Determination

Inertial Measurement Unit (IMU) is a traditional sensor to measure the attitude and heading, that is the orientation. However alternate methods also exist exploiting the devices which are not meant for measuring the attitude like converting motor power consumption to Euler angles [21]. Infra-Red (IR) sensors are investigated to be used not only for height above the ground [22] but also for absolute attitude determination [23]. One idea of infra-red attitude sensor is to measure the heat difference between two sensors on one axis to determine the angles of the UAV because the earth emits more IR than the sky. However accuracy directly depends on the baseline length. Higher the baseline length, better the accuracy. Attitude can also be determined using multiple GPS antennas [24] or Signal-to-Noise Ratio (SNR) measurements [25]. Though accuracy is dependent on the distance between the sensors. Computer Vision based approaches are also possible. However IMUs are still the most relevant orientation sensor for systems which do not rely on external reference systems like optical tracking. A common approach is to use multiple sensors and then apply data fusion and some filtering technique like Kalman filter or Particle filter to find the *true* value. Although knowledge of correct attitude and heading is the basis for accurate navigation, but this information is even more critical for cooperating units operating in close locality.

1.4.3 GNSS Receiver

GNSS observables, the pseudo-ranges and carrier phase, are used to determine the position of the receiver. GPS carrier phase can facilitate quite accurate relative navigation

[26]. It can provide accuracy to cm level, however it is demanding to solve integer ambiguity in real time [27]. Information provided by GPS receiver can be directly used to measure the ground velocity. GPS measurements can be used in real-time as well as in off-line mode. Examples include navigation and geo-referencing for aerial photography respectively. For post-mission analysis, GPS observables are stored in the form of Receiver INdependent EXchange (RINEX) files [28] and then processed later. Geo-referencing can be verified to some degree using the Google Earth [29].

Differential Global Navigation Satellite System (DGNSS) has been widely used in many applications requiring high accuracy [30]. It is a technique to improve the accuracy of a rover coordinates applying some correction methodology to remove GPS errors. DGPS services are either to be hired through some agency providing correction signals or an indigenous reference station is required to be set-up, where correction factors are computed and then sent to MUAV. Prior knowledge of reference station exact coordinates is mandatory that restricts the use of indigenous reference station in the field. However exact position can be determined using some emerging technologies like Precise Point Positioning (PPP) that does away the need of a base station and is able to provide position solutions at centimeter to decimeter level. It needs to be considered that it requires long initialization time, that is a drawback for real-time applications [31]. Though PPP can serve the purpose to find the exact coordinates of reference station in the field.

A quick and easy solution to this problem may be the use of Satellite Based Augmentation System (SBAS) services being provided in different regions of the world. It is the cheapest and quickest method (even in challenging environments) to improve the accuracy as these signals are transmitted on $L1$ frequency and no decoder is required, that is otherwise required in case of RTCM-104 format DGPS. Feature of $L1$ frequency make the signals usable for low-cost GPS receivers too. Performance of European Geostationary Navigation Overlay Service (EGNOS), the European SBAS, in terms of accuracy is 3m lateral and 4m vertical for open service, while its availability is 99 percent [32]. There are many low-cost GNSS receiver providers like u-blox that provides quite precise GNSS receivers e.g. LEA-M8S and NEO-7P (for PPP) being the latest products [33]. DGPS coupled with IMU (for integrated navigation) has been extensively used uniting the merits of both systems [34]. For relative navigation, one is more interested in relative position than absolute position. So in spite of GPS error sources, it can be a good candidate for relative positioning applications [35]. More details on DGPS may be seen in Chapter 7.

1.4.4 PMD Camera

PMD camera is another emerging technology that is drawing attention during the last few years. It is being used for the domains that have strict time compliance requirements, for example the deployment of an air-bag of a car in case of an accident [36]. PMD camera has the potential to be an excellent sensor for cooperative flight of multiple units. An envisaged application may be where a MUAV exploits a PMD camera for



FIGURE 1.8: Front and side view of a PMD camera.
Figure adapted from Ref. [1].

obstacle detection and/or relative position and attitude determination. The PMD Nano from pmdtechnologies (Siegen, Germany) is an optimal PMD camera with regards to price, size and weight, shown in Figure 1.8, that can be used for MUAV. It is capable to measure 3D distances with a resolution of 1mm within a range of about 3m. Use of this emerging technology for MUAVs can introduce new horizons.

1.5 Information Exchange Among the Agents

A basic question to deal with cooperative flight is that what is the minimum information required to be shared among the units to effect cooperation and to coordinate actions? A flying object has 12 states in general; the position coordinates (x, y, z) ; velocity components (u, v, w) along the three-axes; roll, pitch and yaw angles (ϕ, θ, ψ) ; and the angular rates (p, q, r) measured along the three-axes [14]. Relative position, velocity and attitude are considered as the minimum variables to be determined for cooperative control. Redundant information sharing is to be avoided in order to reduce the burden on communication.

1.5.1 Communication Requirements

Formation flight control techniques generally require transmission of states of the leader to the follower, that involves a suitable communication data link. The transmissions can be divided into three categories; vehicle telemetry data, commands for the vehicle, and coordination information between vehicles [8]. For applications like live video streaming to the ground station and carrier phase double-difference GPS data in real time, we need to have wide bandwidth communication requirements. For MUAVs, mostly Wi-Fi, bluetooth [37] and XBee links have been used, each having their own merits and demerits in terms of range, data rate and power requirements. Communication packet loss may occur thereby producing errors in cooperative control of MUAVs and compromising

the communication integrity. An Orthogonal Frequency Division Multiplexing (OFDM) technique with adaptive resource allocation was used at Bleking Institute of Technology, Sweden for small UAV communication with the ground station and between UAVs [38]. Reference [39] discusses the robust stability of multiple cooperative units under time-varying communication topologies and communication delays.

1.6 Related Hardware and Software Tools

1.6.1 Hardware

Some of the open source quadcopter projects may be listed as ArduCopter, AeroQuad and MikroKopter etc., which may be customized as per the mission requirements. Institute of Aerospace Information Technology, University of Würzburg has constructed its own quadcopter frame which is also being used for educational purpose to verify the control algorithms [40]. Another example is Institute of Aircraft Design, University of Stuttgart, which is also designing and manufacturing its own UAVs. For computational purposes, mainly on-board computers, FPGAs, Gumstix and micro controllers have been used. An on-board processor may be required to process the data at a faster rate especially for those units operating in close vicinity.

1.6.2 Software Tools

MATLAB/ Simulink environment has been mostly used for simulation of MUAV models. Same may be hooked up with C/C++ programming languages which are predominantly used for real time systems. MATLAB Real Time Workshop (RTW) can be used for automatic generation of C code directly from Simulink models. Similarly MATLAB Coder toolbox can also be used for generation of standalone C and C++ codes from MATLAB code. Some research groups are using Real time Operating System (RTOS) for quadcopters, for example μ C/OS-IITM [41], Tiptoe [42] and FreeRTOS [43]. Real-time On-board Dependable Operating System (RODOS) [44], developed by Deutsches Zentrum fuer Luft- und Raumfahrt (DLR) Germany and Institute of Information Technology, Würzburg University, has been successfully applied on operational satellites and quadcopters. It is written in C++ and emphasis has been placed on simplicity. It has the potential to be utilized for the constellation of MUAVs. Selection of appropriate software depends on the application requirement.

RTKLIB [45] software is one of the most popular in the domain of open-source GNSS precise positioning and has been used extensively for low-cost GNSS receivers [46]. It is a compact and portable program library written in C to provide a standard platform for Real Time Kinematics (RTK) applications. It is capable not only for single point positioning but also for DGPS, PPP and relative navigation etc. The software

has good capabilities for real-time as well as post-mission analysis. It is a precise positioning technology, with which users can obtain cm-level accuracy of the position in real-time by processing carrier-phase measurements of GPS signals [47]. Accuracy of cooperative behaviour is based on the control technique selected and the sensors used. Implementation of effective distributed control technique is also dependent on integrity of communication data.

1.7 State of the Art

A synopsis is provided on state of the art in MUAV technologies, distributed control methods and on-going work and challenges, which is the motivation for many researchers all over the world to work in this fascinating field. Quadcopter concept is as old as the history of aviation. The Breguet-Richet quadrotor helicopter Gyroplane, developed in 1907, is considered to be first quadcopter that lifted into air [48]. The idea of formation flight of multi-units was inspired by the nature e.g. birds and bees. Though benefits of formation flight in terms of fuel saving have also been investigated in [49]. Much work has been done at Department of Aeronautics and Astronautics, Massachusetts Institute of Technology (MIT) in the domain of formation flight of micro UAVs in real time. For relative navigation, mainly Carrier Phase Differential GPS (CPDGPS) has been used exploiting pseudolite transmitters to provide accuracy of the order of few centimeters [50]. A mid-air rendezvous of two UAVs was also realized at MIT [51]. A control scheme was designed for autonomous aerial refueling in Simulink environment [52]. A quadcopter with a cable-suspended load is considered in [53].

Load transportation with multiple quadcopters is useful when the load is heavier compared with the maximum thrust of a single quadcopter, or when additional redundancy is required for safety. However this is challenging since dynamically coupled quadcopters need to cooperate safely to transport load. GRASP laboratory at University of Pennsylvania has demonstrated many technological innovations with a cluster of autonomous micro UAVs flying inside the constrained environment [54] and performing stunning actions like cooperative grasping and transportation [55] using decentralized PID control laws. Institute of Aerospace Information Technology, Würzburg University is working in the domain of quadcopters autonomy for indoor applications with the objective to assist the fire fighters against the burning structures. It is fully autonomous in the sense that it does not rely on external signals like GPS and optical tracking system etc [37].

Today small quadcopters are able to perform the wonderful tasks such as basic construction while assembling the parts. MUAVS can perform aggressive manoeuvres requiring very high precision. Quadcopters can even play the musical instruments in a coordinated fashion. They can grasp and transport the load using multiple units operating in harmony [55]. We can watch the stunning feats of cooperative micro UAVs flying very accurately and even capable to play ping-pong [56]. An indoor formation flight of 20 micro quadcopters with coordinated actions has already been materialized. ETH

Zuerich exhibits state of the art coordination of MUAVs like cooperative construction inverted pendulum balancing and ball juggling with a net utilizing multiple MUAVs. The Institute has also designed a Distributed Flight Array, a flying platform consisting of multiple autonomous single propeller vehicles which are able to drive, dock with their peers, and fly in a coordinated fashion. However all these systems rely on a highly accurate external positioning system like optical tracking.

Traditional quadcopters have fixed-pitch propellers that can generate the thrust only in one direction. Variable pitch quadcopters have been designed which are capable of accelerating and decelerating at a very fast rate. These are flight demonstrated for inverted flight and aggressive manoeuvres [57]. Such platforms have higher control bandwidth. A group of MUAVs is also drawing a lot of attention for 3D aerial maps generation [58] as well as weather monitoring and communication relays [59].

Further description of this section is divided into subsections covering respective domains, as in following paragraphs.

1.7.1 Formation Flight Test Setups

A number of test setups have been developed for testing and validation of formation flights and control techniques for coordinated missions. Some noteworthy contributions in this domain are mentioned here. A formation flying test-bed was developed at Deutsches Zentrum fuer Luft- und Raumfahrt (DLR) Germany to support the design, implementation, testing and validation of real-time embedded GPS based Guidance Navigation and Control (GNC) systems [60]. A testing platform was designed at Aerospace Controls Laboratory, MIT to evaluate and compare different control algorithms for coordinated missions [61]. A micro UAV test bed at GRASP laboratory, University of Pennsylvania helped to support research on coordinated flight of micro UAVs [54]. A multi-vehicle platform was designed and developed at Stanford University for experimentation and validation of multi-agent control algorithms, using both centralized and decentralized approaches [62]. A UAV test bed was developed jointly by faculty members and the students at Brigham Young University for cooperative control experiments [63]. It provided opportunity for the students to have an exciting multi-disciplinary experience. A multi-UAV experimental test bed was designed at Utah State University with detailed presentation of algorithms on centralized formation controller [64]. The California Institute of Technology introduced a platform for testing decentralized control methodologies for multiple vehicle coordination and formation stabilization [65]. This test-bed consisted of eight mobile vehicles, an over-head vision system providing GPS-like position information and wireless Ethernet for communications. These all test beds are quite sophisticated in nature. University of Würzburg has constructed its own quadcopter frame and associated test bench which is also being used for educational purpose to verify the control algorithms [40]. This test setup is indigenously developed at Institute of Aerospace Information Technology and offers all basic functionalities.

1.7.2 Control Techniques

Quadcopter is dynamically unstable system that has to be stabilized by an appropriate controller. In order to control the quadcopter, a number of control techniques have been used ranging from the basic controller of PID and LQR [66] to much sophisticated techniques like Non-linear Model Predictive Control (NMPC), Decentralized Model Predictive Control (DMPC) [67], Back-Stepping and Fuzzy Logic controls etc. The PID controllers have limitations in optimality and robustness. Besides, it is also difficult to tune the parameters under some circumstances [23]. Disturbance rejection and tracking control with negligible steady-state error may also be achieved with a simple Integral Control. Low-cost common-off-the-shelf GPS receivers in combination with the MEMS-based IMUs have been sufficient to maintain the formation of the micro aerial vehicles for mobile communication relay [59] as the units are largely apart. However control requirements are quite strict when the units are operating in close vicinity.

A number of control techniques have been successfully employed for formation flight of aerial vehicles. Some of the contributions are mentioned here. Institute of Control Systems (ICS) at Technical University of Hamburg-Harburg (TUHH) exploited H_∞ control for formation flight simulation of multiple quadcopters. Constraints on H_∞ norm guarantee closed-loop stability for a given range of uncertain parameters. The weights W_s and W_{k_s} are the tuning parameters for H_∞ controller, and it typically requires some iterations to obtain weights which will yield a good controller. The H_∞ norm is the peak gain of system across all frequencies and all input directions. In reference [68], a formation controller has been designed to minimize H_∞ performance measure while guaranteeing robust stability. For this purpose, H_∞/l_1 control technique was proposed for formation flight simulation of multiple quadcopters. The proposed method is also efficient as the synthesis procedure is based only on a single agent model, instead of a model of the whole formation with a possibly large number of agents. Reference [39] examines the robust stability of multiple cooperative units under time-varying communication topologies and communication delays.

Coordination and trajectory tracking control design for a leader/follower structure of multiple mini rotorcrafts was simulated in [69] using nonlinear coordinated control design with state feedback. The tracking control law was combined with an eigenstructure assignment and optimization technique in [70] to compute the feedback and feedforward gain matrices. The scheme was applied for pitch pointing control. An important approach to control design is *model following* where it is desired for the quadcopter to perform like an ideal model with desired flying qualities. Pitch pointing flight control laws have been designed in [71] by using the model following control scheme utilizing an eigenstructure assignment and Command Generator Tracker (CGT). In [72], CGT based direct model reference adaptive controller has been exploited to eliminate the adverse effects of bounded uncertainties for Mars atmospheric entry guidance. A leader-follower formation strategy was realized in [73] utilizing a robust tracking control approach; and a Kalman filter based formation command generator was executed on the follower to keep in formation.

A cluster of UAVs has been used as a phased array antenna in [15] to show the feasibility of a distributed control strategy. Here each quadcopter, with a 2D model, has a local controller that is based on the information of its own states as well as states of a subset of other vehicles in the formation. A simple formation geometry is assumed in this paper, where the three units remain in a straight line maintaining a distance of $1m$ from each other. In Reference [74], each quadcopter plans its trajectory based on the information of neighboring quadcopter including its planned trajectory and an estimate of its state. Formation is described by the shape vectors and quadcopters can safely change the shape of formation. Reference [75] controls a formation of three quadcopters using LQR as local controller and PI as formation controller. However PI controller parameters are chosen experimentally, and the control laws adopted are sensitive to parameter changes. Fixed height of quadcopters is assumed in this paper.

A distributed control scheme is suggested in [76], where units are not coupled rather making independent motions cooperatively. The scheme for leader-follower was extended to an arbitrary number of units in [77], and the network dynamics and manageability of swarms was analyzed based on open loop H_2 norm of the network. Due focus has been given in [78] to the networks that are leader symmetric, restricted to the case when there is only one leader present.

The control architecture may be considered as decentralized where each agent is controlled by a local controller C_i which accesses the state of agent i with no information exchange among the other vehicles [15]. Decentralized control has been exploited in other works, as in [79]. Choice of appropriate control technique depends on the application. For a control system designing, general design goals are zero steady-state error, fast rise-time with little overshoot and good input-disturbance rejection. There is generally a trade-off between the intricacy of control technique employed and the level of accuracy achieved there off.

1.7.2.1 Consensus Algorithms for Cooperative Control

No explicit leader exists in case of control schemes based on consensus algorithms [13], that gives rise to redundancy. A number of cooperative control schemes have been suggested in literature. Some remarkable contributions are mentioned here. A lot of work has been done at TUHH in the domain of cooperative control of MAS, for example [80]. In this paper, a distributed robust control scheme for formation flight of autonomous quadcopters was presented where two existing approaches, namely cooperative and consensus-based formation control, are combined using a mixed H_∞/l_1 design approach. It showed to provide robustness against switching communication topologies and time-varying delays. Reference [81] proposes a decentralized hybrid MPC for autonomous navigation of a formation of quadcopters under obstacle and collision avoidance constraints. A hierarchical control structure has been used in this study. Reference [82] gives a theorem that reveals that the stability of a formation of n identical vehicles can be verified by stability analysis of a single vehicle with the same dynamics and an

output that is scaled by the eigenvalues of the normalized Laplacian matrix of the network. A consensus controller has been designed in [83] while choosing a common gain matrix and the connection weight to achieve hovering synchronization (in 1DOF) for a fleet of quadcopters.

Formation control problem has been formulated in [84] and [85] based on consensus approach. In [84], only one agent receives information from virtual leader. In [85] a time-varying formation is considered, that may be useful for scenarios like rotation formation. Only one control input has been used in this study. Fixed height of quadcopters is assumed in this study.

1.7.3 Graph Theory

Graph theory was first introduced by the famous mathematician Leonhard Euler in the 18th century during an endeavor to solve the historical problem of *Seven Bridges of Königsberg* [86]. Since then graph theory has evolved and today we find its uses in many domains and disciplines, for example to model the information exchange among units of a MAS and to represent the computation flow and data organization in computer networks.

Graph theory has strong links with distributed / cooperative control schemes to efficiently handle communication topology affairs. It has been exploited by a number of researchers in the domain of MAS, formation flight and network control. Examples include the use of directed graphs for formation control [87] and [80], and UAV swarm modelling where the leader-follower scheme of aerial vehicles was extended to an arbitrary number of units [77]. The problem formulation is accommodated with the conversion of a graph into the Laplacian matrix. This matrix gives an insight into the communication topology, properties of the underlying graph, and graph connectivity through its eigenvalues. Problem formulation based on graph theory helps to handle communication topology and formation control matters for a large number of units. Graph theory has been used by Murray and Olfati-Saber [88] and Phillip R. Chandler [89] to control cooperating units.

Laplacian matrix eigenvalues were identified as an important object of study in [90]. Eigenvalues of Laplacian matrix were exploited in [91] to determine the effects of communication topology on formation stability, and Laplacian matrix itself was used to represent the sensing of relative position of units in a formation. A class of graphs called weakly connected graphs were studied in [76] to represent the formation of multiple vehicles that are weakly connected.

Algebraic graph theory is useful in modeling the communication network and relating its topology to formation stability [91]. Graph theory has been exploited in [77], where a UAV swarm has been modeled as a two-component hierarchical system comprising of network dynamics and UAV dynamics. Here a leader-follower consensus model has

been adopted. It is noteworthy that in this work, only undirected graphs are considered and stability is not considered as an issue due to undirected graphs. Alternatives to Laplacian-based consensus algorithms in computer science are Gossip-based algorithms such as the push-sum protocol [82].

There are some interesting properties related to the second smallest eigenvalue of Laplacian matrix, known as **Algebraic Connectivity**. Work has been done to increase the algebraic connectivity in [92]. Reference [93] suggests to choose a set of edges effectively in order to maximize the algebraic connectivity for a given number of vertices. Algebraic connectivity is also regarded as a measure of stability and robustness of the networked dynamic systems [92]. It is also used in analyzing the synchronization of coupled dynamical systems [94], and is a measure of performance and speed of convergence of consensus algorithms [82].

1.7.4 Differential GPS

For post-mission analysis, GPS observables are stored in the form of RINEX files [28] and then processed later. DGNSS has been widely used in many applications requiring high accuracy [30]. It is a technique to improve the accuracy of a rover coordinates applying some correction methodology in order to remove GPS errors. Differential GPS techniques are well documented and a number of codes exist for its realization in off-line mode. A well-known open source software is *rtklib* developed in *C* language [95]. A remarkable contribution is the set of MATLAB codes developed by Kai Borre [96]. It is quite comprehensive suite to visualize a number of GPS working principles. This set of codes is tailored to the RINEX files provided with the suite. Another key development is *goGPSProject* [97]. It is basically a software library designed to improve the positioning accuracy. Another contribution is by Wen Zheng who post processed GPS raw data from RINEX files with MATLAB codes for single point positioning [98] and base line estimation using dual frequency receiver [28]. Algorithms for position determination and relative positioning stated in GPS Theory and Practice by B. Hofmann-Wellenhof [99], and Interface Control Document IS-GPS-200D [100] formed the basis for almost all of these codes. A MATLAB code for single point positioning developed by Michael Gaeb [101] is well structured.

More innovative applications are foreseen with advancement in sensor technology, autonomous control techniques, on-board processors and software. Quadcopters have tremendous potential for their growth keeping in view their future applications. Miniaturization of these vehicles will still open new vistas for many other useful and interesting applications.

1.8 Statement of Contributions

The contributions for this thesis are stated in the following paragraphs.

Chapter 1. In this chapter we review the main aspects in the domain of distributed control of cooperating mini UAVs. For this purpose, an extensive literature survey has been carried out for all related spheres. Different constellation architectures, with their inherent merits and demerits, are discussed. Some prime sensors vis-à-vis their current and prospective roles in MUAV cooperative flight are narrated. A number of control techniques employed successfully for MUAVs have been given due coverage. Significance of MUAV model identification is perceived. Communication requirements among units in a formation are discussed. Hardware and software tools used for MUAVs are described. State of the art has been identified and areas requiring attention of related scientific community have been determined.

This work was published as a review paper in [1].

Chapter 2. An existing test bench setup for single quadcopter has been extended to two stations with necessary inter-communication in order to realize basic functionalities for formation flight of quadcopters. Synchronized motion of quadcopters in 3DOF (roll/pitch/yaw) has been demonstrated in real-time. MATLAB Instrument Control Toolbox has been exploited for real-time data display, plotting and analysis for both stations. Data integrity has been ensured with the introduction of a checksum function.

This work was published in [2].

Chapter 3. Explicit model following design based on LQR PI control scheme has been tailored for tightly coupled heterogeneous formation flight of quadcopters in leader-follower constellation. This scheme offers the advantage that performance criteria are clearly described for the followers to make them behave like the model with desired dynamics. Additional advantage of simplicity in implementation is also noteworthy. Implementation of this scheme for centralized heterogeneous leader follower architecture is not seen in the literature. Compared with most of the earlier works, a full-state of quadcopter is considered while tracking only the outputs of interest (performance output) in the presence of perturbations. Trajectory tracking performance under different types of commanded values and rejection for different types of disturbances have been demonstrated. The stability of the closed-loop system has been analyzed using singular values.

This work appears in [3].

Chapter 4. A simple framework has been presented to control a network of two quadcopter formations from ground stations exploiting the notion of virtual leader, which enables the scheme to be robust against any node failure. We have endeavored to tailor two control schemes namely Linear Quadratic Regulator Proportional Integral (LQR PI) based on explicit model following and LQR PI servomechanism to control

two quadcopter formations. For the former type, the reference model resides within the quadcopter to explicitly define the performance criteria to make them behave like the model with desired dynamics. Four control inputs to each quadcopter are used, and the controller designing has been described in much details. These control schemes are compared in terms of convergence to desired tracking values and the control effort. Cluster reconfiguration of agents is also demonstrated. Both formations track the trajectories with varying heights. Implementation of this simple, adaptive and robust scheme to control quadcopter formations from ground stations is not seen in the literature.

This work has been accepted for publication in [4].

Chapter 5. Eigenvalues of Laplacian matrix have been exploited to give an insight into the subgraph properties. Euler's formula has been generalized while relating it to eigenvalues of underlying Laplacian matrix that makes it applicable for graphs as well as subgraphs. A modified version of Euler's formula is also presented which relates it to sum of eigenvalues and trace of underlying Laplacian matrix. Effects of addition and removal of communication links for a given number of agents are examined. Effects of addition and removal of agents, and portioning a graph into subgraphs (multiple formations of vehicles) are also investigated.

Algebraic multiplicity of zero eigenvalues of Laplacian matrix has been related with number of loops for a directed graph. A loop indicates redundant information flow from/to multiple units. It also gives an indication for connectedness of a graph. Product (or trace) of all nonzero eigenvalues of a Laplacian matrix has been considered as a measure for nodes connectivity of a given graph or subgraph. The product (or trace) may also be considered as a measure of feasibility of information exchange. Eigenvalues of Laplacian matrix of an undirected graph has been related with number of communication paths. Number of zeros in a Laplacian matrix have been related with number of edges to make the graph fully connected. Role of graph theory for cooperative control of two clusters of quadcopters is demonstrated with the help of simulation that gives an insight to some of the presented notions.

This work will appear in [5].

Chapter 6. Cooperative control of agents has been realized making use of consensus algorithms under different communication topologies. Effect of input bias on average consensus algorithm has been studied. Different schemes have been simulated under different communication topologies. Switching consensus algorithm and weighting consensus algorithms are also studied and simulated.

Chapter 7. An existing software in MATLAB for single point positioning has been extended for DGPS positioning. This software uses correction factors determined at base station, without relying on double-difference information. Atmospheric errors have been catered through correction factors and hence no atmospheric model has been used for proposed DGPS algorithm. This code provides solution for rover coordinates with and without DGPS for all the epochs, and number of epochs for processing can be

selected. Total number of visible and valid GPS satellites can be viewed for all the epochs. DOP values for Geometric Dilution Of Precision (GDOP), Position Dilution Of Precision (PDOP) and Time Dilution Of Precision (TDOP) have been calculated at rover. Effect of number of visible valid satellites on PDOP values has been studied. Coordinates determined with the code have been compared with the accurately known coordinates. Statistical results are computed for minimum offset, maximum offset and mean positional error. Results for position error in three axes Earth Centered Earth Fixed (ECEF) as well as radial difference have been plotted.

This work was published in [6].

All the works mentioned in this section have been carried out by the author of this thesis. The contributions by other persons for quadcopter test bench development are acknowledged in [2]. All the papers were compiled as a first author.

1.9 Summary

MUAVs have already replaced manned aircraft in many fields and are even capable to perform novel assignments which cannot be performed by manned platforms. Utilization of MUAVs is expected to rise steadily, in particular for remote sensing missions with the exploitation of emerging technologies. Their utilization is foreseen ranging from emergency situations handling, for e.g. fire fighting, volcanoes monitoring etc., to routine tasks like postal and parcel services, film shooting, patrolling and wild life survey etc. Incorporation of a robotic arm to a MUAV will enhance their applications to further extent. Unmanned is therefore unmatched. There is still a lot to explore for this fascinating field. It definitely has the potential to serve the humanity in an even better and advanced fashion.

1.10 Thesis Outline

Distributed control of cooperating agents is a multi-disciplinary topic and in order to realize it, one requires to work in different domains. These domains include related hardware and software tools for flying quadcopter, distributed control schemes and accurate position determination for the agents. The key modules with their inter-connections are shown in Figure 1.9 for one quadcopter.

Further description of this thesis is organized into six chapters. Chapter 2 covers the hardware and software aspects of quadcopter formation flying test setup in realtime. Chapters 3, 4 and 6 are focussed towards simulation aspects of different control schemes for formation flying of quadcopter fleets. Chapter 5 is dedicated to the graph theory due to its influential role in cooperative control schemes. As accurate position determination

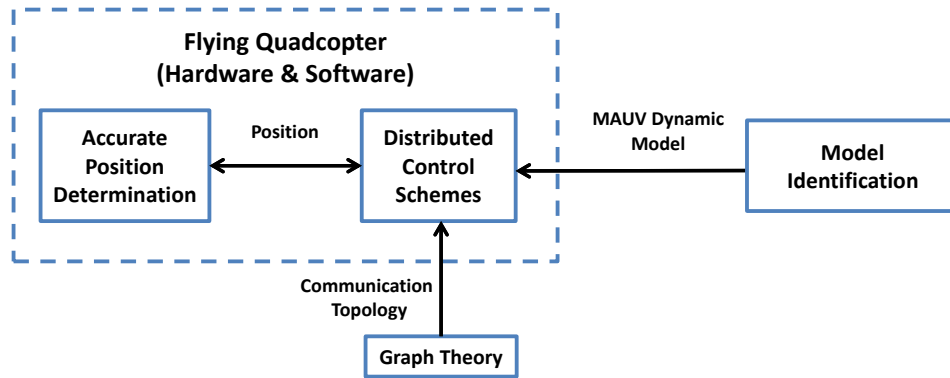


FIGURE 1.9: Modules for distributed control of cooperating agents.

of agents is mandatory to operate cooperatively and to avoid collisions, chapter 7 is therefore assigned for DGPS. It is a potential and well-tested candidate for outdoor operations to meet stringent position accuracy requirements.

Chapter 2

Quadcopter Formation Flying Test Environment

2.1 Introduction

With the advancement in MEMS technology, avionics equipment and miniaturization of sensors; MUAVs have drawn considerable attention, particularly during the last decade. Due to their affordability for common and versatile applications [1], their popularity is growing exponentially. In order to increase efficiency and redundancy, researchers all over the world are now concentrating towards multiple MUAVs flying together and performing cooperative tasks. In recent years, VTOL vehicles received more attention due to a number of advantages over fixed wing vehicles which are commensurate with their usage. VTOL vehicles are able to fly inside enclosed spaces and buildings without compromising on safety requirements. Though realization of multi-UAV coupled flight is complex, however obvious advantages justify the laborious work involved.

Cooperative flight is a multi-disciplinary topic involving aerospace engineering, avionics, control engineering, mechanical engineering, communication systems and information technology; therefore addressed by researchers from diversified background. In recent years major contribution and innovation came from information technology involving embedded systems programming. Institute of Aerospace Information Technology, Würzburg University is distinctive in the perspective that it focuses on information technology only in the domain of aerospace and is the only institute of its kind in Germany. It is building its own quadcopters frame and programming algorithms. The institute quadcopter, shown in Figure 1.1 in Chapter 1, spans 78 cm including rotor blades and weighs about 1.25 kg. It is able to lift approx. 1 kg of payload and has endurance of about 10 – 15 minutes (depending on rotors speed) with 3 cell LiPo batteries (3Ah) and hence exhibits more flexibility to perform a number of tasks. The Institute had developed a test bench for quadcopter [40] that is being used not only for education purpose but also for further development. The setup was fascinating for

a large number of students who eagerly did their Bachelor and Master theses making use of this test bench. For our present study, same test bench was extended to two platforms (one leader and other follower) with necessary inter-communication to test the algorithms for formation flights. MathWorks MATLAB Instrument Control Toolbox (ICT) was exploited to plot and analyse the data that is being received in real time through Bluetooth.

Our proposed test setup is indigenously developed at the Institute and offers all basic functionalities. Main contribution of this chapter is the realization of quadcopter formation flight test-bed in 3 Degrees Of Freedom (3DOF) in its simplest form and at considerably low-cost. This chapter describes in details the communication setup between quadcopters; and plotting and analysing the results using MATLAB ICT while transmitting attitude information of leader and follower to a desktop PC using serial interface. Total cost for the complete setup is about 800 Euros (excluding PC). It includes two quadcopter frames, four controllers and motors on each platform, two IMUs, one AVR32 test board, one AVR32 on-board version, two Bluetooth modules, one USART-USB converter board, two rods and ancillary equipment to support quadcopters. Details of the hardware are provided in further sections of this chapter. Main motivation for this work was the development of a basic formation flying test bed with simple control techniques that can later be extended and upgraded for more sophisticated control techniques. It may be noted that in this chapter we have demonstrated real time attitude tracking only, because quadcopters are fixed on rods and may only exhibit motions in attitude. However for a free flying quadcopter, a position controller is rather required. Control schemes for positions tracking of quadcopters are simulated in subsequent chapters.

Further description of this chapter is organized into four sections. Section 2.2 introduces our test bench system architecture with related hardware components and software. Quadcopter inter-unit communication setup details are described in Section 2.3. Control technique employed has been described in Section 2.4. Section 2.5 narrates the experimental results and effectiveness of proposed approach. Summary of the chapter is provided in Section 2.6.

2.2 Formation Flying Test Environment

2.2.1 Hardware

The quadcopter on test bench, developed at Würzburg University [40], is free to move in roll, pitch and yaw, as shown in Figure 2.1. With the arranged hardware mechanism, it is possible to gain height up to a certain level; thus it offers four degrees of freedom. Quadcopter is fixed in x-y plane so it is quite safe to evaluate different algorithms.



FIGURE 2.1: Quadcopter test bench mechanism.

For our present study, this setup was extended to two stations having mutual communication through Bluetooth protocol that is a wireless version of Universal Synchronous/Asynchronous Receiver/Transmitter (USART) protocol. Bluetooth BTM-222 modules, shown in Figure 2.2, were exploited for this study. It is a class-I device from *Rayson* with a nominal range of 100m and operates in 2.4 GHz frequency range. Leader quadcopter bluetooth was configured as *slave* and follower quadcopter module as *master*, as information flow was from leader to follower. Baud rate was set as 57600 for communication. Additionally RN41-I/RM from *Microchip Technology Inc.*, shown in Figure 2.3 was also tested for this purpose.



FIGURE 2.2: Bluetooth module BTM-222.

Figure adapted from Ref. [2].

Already existing test bench [40] is utilizing AVR32 micro controllers from Atmel [102] which are available in two versions; one for test purpose and the other for on-board installation, shown in Figure 2.4. The board has one Two-Wire Interface (TWI) and



FIGURE 2.3: RN41-I/RM Bluetooth module.

four USART interfaces. Test-version micro controller also has a small LCD screen for display, several push-buttons and LEDs which can be programmed as required.

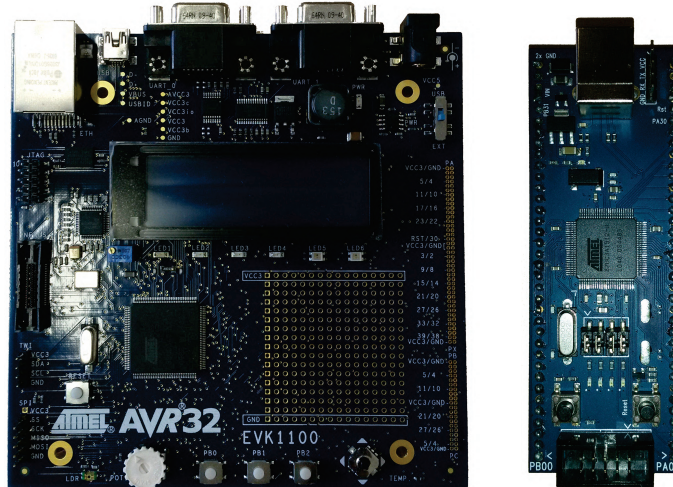


FIGURE 2.4: AVR32 test board (left) and AVR32 small board (right).
Figure adapted from Ref. [2].

2.2.1.1 Engine

Quadcopter is powered by four motors mounted on the tip of each shaft. Motor power can be selected between 0255, where 0 and 255 correspond to minimum and maximum power settings respectively. For ground testing, it is recommended to set the power to 100. Alternatively, motor speed can be controlled through the DC voltage, with 12V DC being the maximum. Four brushless controllers drive the brushless motors.

2.2.1.2 Inertial Measurement Units (IMUs)

An IMU can sense the current acceleration with the help of one or more accelerometers, and can sense the angular velocity and orientation with the help of one or more gyroscopes. Angular velocity may be integrated to compute roll, pitch and yaw angles. Similarly acceleration information may be manipulated to derive the linear velocity and position information. We used IMU3000 Combo from Sparkfun, shown in Figure 2.5, to measure the attitude information at a rate of 100 Hz on each station. IMU3000 incorporates ADXL345 accelerometer and ITG3200 gyroscope. An IMU3000 Combo and BTM-222 Bluetooth modules connected to AVR32 test board are shown in Figure 2.6 during testing phase. Further details for Bluetooth modules may be seen in section 2.3.2.

In addition to IMU3000, the LSM303DLM IMU from Polulu [103] was used to measure roll, pitch and yaw information. It also encompasses a magnetometer to measure yaw

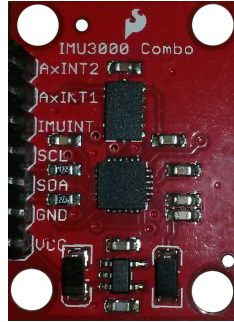


FIGURE 2.5: IMU3000 Combo.
Figure adapted from Ref. [2].

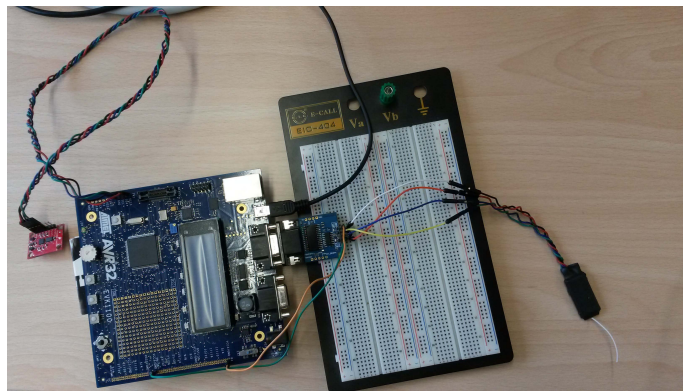


FIGURE 2.6: Test environment for different modules.

information. It is a compact board and provides six independent readings. Pictorial view of LSM303DLM (sealed for ready to install on quadcopter) is shown in Figure 2.7. The device is connected to AVR32 board through I^2C bus. Accelerometer and magnetometer have their own 7-bit slave addresses on I^2C bus. A LSM303DLM module connected to AVR32 board during testing is shown in Fig. 2.8.

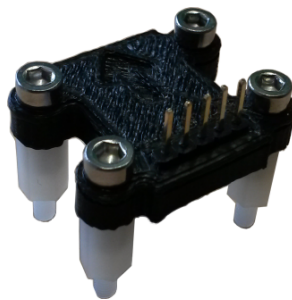


FIGURE 2.7: IMU LSM303DLM (in ready to use configuration).

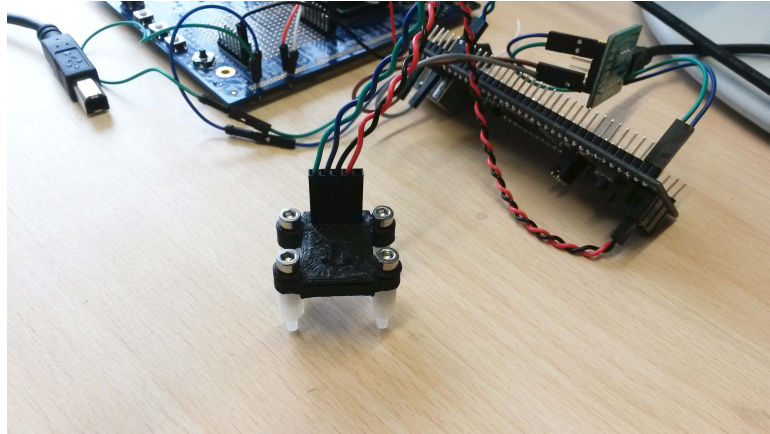


FIGURE 2.8: LSM303DLM connected to AVR32 board during testing.

2.2.1.3 Two Wire Interface (TWI) Cable

The I^2C bus was invented by Philips Semiconductors which is a multi-master, multi-slave, single ended, serial computer bus used for attaching low-speed peripherals to computer motherboards and embedded systems. The TWI is used by the manufacturers (like Atmel) while implementing proprietary features, in addition to I^2C features. It is half duplex serial data cable using serial data along with serial clock. AVR32 small board has one TWI port available corresponding to PINs PA29 (for serial data) and PA30 (for serial clock). TWI may be considered as a superset of I^2C bus.

2.2.1.4 Quadcopter and IMU Orientation

A quadcopter does not have an explicit nose, contrary to the fixed wing aircraft. One of the shafts is therefore designated as the nose to determine the orientation of quadcopter. Same also needs to be considered while installing and orientating IMU on quadcopter. Sign convention for Euler angles depends on how we define the reference system. Following sign conventions are generally assumed:

Bank right	Positive roll angle
Bank left	Negative roll angle
Nose up	Positive pitch angle
Nose down	Negative pitch angle
Nose right	Positive yaw angle
Nose left	Negative yaw angle

Sign convention was considered while fixing the IMU to the quadcopter on test bench. It was manually rotated in different directions while transmitting the quadcopter roll, pitch and yaw angles information to the terminal program *HTerm* for display on the computer screen.

2.2.1.5 IMU Calibration on Test Bench

When IMU is being calibrated, its orientation at that instant is assumed as $(0, 0, 0)$ that corresponds to zero roll, pitch and yaw angles. In other words, IMU calibration means to assign reference values of $(0, 0, 0)$ to the Euler angles. IMU can also be calibrated on test bench. For this purpose, when program on AVR board is initialized, the orientation of IMU at that point in time is taken as $(0, 0, 0)$ for all three Euler angles. It is quite important that IMU is held in truly horizontal position as wrong calibration may lead to erroneous operation of quadcopter attitude and heading control.

2.2.2 Software

Following three software have been used for this study;

1. AVR32 Studio to program the AVR32 micro controllers,
2. Terminal program HTerm for real-time display of attitude information of both quadcopters, and
3. MATLAB R2014a with Instrument Control Toolbox ver. 3.5 for real-time plotting and storing the attitude information from both quadcopters received through Bluetooth serial interface.

2.2.2.1 AVR32 Studio

Quadcopter software was developed in *C* language by the Institute itself with main contribution by Dr. Nils Gageik. Software is of modular fashion that can easily absorb the modifications and improvements as well as new functions. It makes use of AVR32 Studio that is based on Eclipse environment. For this study, AVR32 Studio was used to program the attitude information exchange between the two quadcopters through Bluetooth, and to send the formatted information through Bluetooth/ serial interface to HTerm and MATLAB ICT for display and plotting purpose. Received information was also used at follower quadcopter as reference value for PID control. AVR32 Studio is an Integrated Development Environment (IDE) that supports all Atmel 32-bit AVR applications. It is a C/C++ editor with syntax highlighting, navigation and code completion. Additional features include project file management, and target configuration and management. The software suite is built on Eclipse for easy integration with third-party plugins for increased functionality. AVR32 Studio also supports development and debugging of standalone (without an operating system) applications [104]. The IDE integrates with the AVR32 GNU toolchain. The GNU C Compiler (GCC) compiles C/C++ programs, while the GNU Debugger (GDB) debugs the target application. External debugger JTAGICE mkII can also be used that is Atmels on-chip debugging tool

for the AVR microcontroller family. It supports debugging with AVR's traditional Joint Test Action Group (JTAG) interface [102]. The JTAGICE mkII allows access to all the powerful features of the AVR microcontroller.

2.2.2.2 Terminal Program HTerm

The HTerm is a terminal program running on desktop computer that was used for;

1. Configuration of Bluetooth BTM-222 modules, and
2. Communication with AVR32 board at a baud rate of 57600 to display the attitude information of both quadcopters.

Communication port can be selected and different parameters like baud rate, data bits, parity bit and stop bit etc can be specified. Information can be exchanged with microcontroller through USART or Bluetooth interface. Output can also be saved in the form of a text file that can be post-processed for analysis purpose.

2.2.2.3 MATLAB Instrument Control Toolbox

Quadcopter operation is controlled through embedded programming in C language. The MATLAB ICT has been used to display and plot the attitude information of both quadcopters through USART-USB interface. Plots can be drawn in real-time and received data can also be logged in the form of files for later use and analysis. This toolbox supports direct communication with serial port interface including Bluetooth protocol to read and write text data (ASCII coded) and binary data [14]. ICT supports Serial Port Profile (SPP) of Bluetooth. SPP Bluetooth devices can be identified and a two-way connection can be established. Remote name or remote ID can be used to communicate with a device. The toolbox can also be used in Simulink environment to fetch the data through serial COM Port. Attitude information of leader is transmitted to the follower after every 10ms. However for display and plotting purpose, data rate can also be controlled through embedded programming at follower. There are three ways to receive data from AVR32 microcontroller to MATLAB ICT through Bluetooth;

1. Writing a MATLAB script (.m file),
2. Using *APPS* tab in MATLAB R2014a and then using Bluetooth node under *Test and Measurement Tool*, and

3. Through a simple Simulink model using blocks of Instrument Control Toolbox.

Method 2 has the limitation that it receives only one chunk of information till a terminator is reached, and is therefore not suitable for continuous data collection. Bluetooth data can be read in two modes; *Continuous* or *Manual*. In manual mode, attitude information coming from quadcopter is not automatically stored in the input buffer. We used continuous mode to put the data in input buffer and then reading it and converting ASCII to double format before plotting it. Initially only leader attitude information was sent to MATLAB through Bluetooth. Later, we organized in such a way that leader sent its attitude information to follower and then we received the attitude information of both quadcopters to MATLAB through USART-USB converter. MATLAB script is written to establish connection between AVR32 micro controller and MATLAB ICT through serial interface for reading and plotting the attitude information. We organized the attitude data in the following order while sending it from micro controller of follower quadcopter to PC;

pitch angle (L), roll angle (L), checksum (L), pitch angle (F), roll angle (F)

where L represents the leader quadcopter and F denotes the follower quadcopter. The MATLAB code has been tailored keeping in view the received information order.

2.3 Inter-Communication

2.3.1 Information to be Shared between MUAVs

A flying vehicle has 12 states in general, namely the position coordinates (x,y,z) ; velocity components (u,v,w) along the three-axes; roll, pitch and yaw angles (ϕ, θ, ψ) and the angular rates (p,q,r) measured along the three-axes [14]. Relative position, velocity and attitude are considered as the minimum variables to be determined for cooperative control. As quadcopter position is fixed on a mounting rod, so we transmit leader attitude information (as reference values) to follower using BTM-222 modules. Attitude information sent from one AVR board to the other is ASCII coded.

2.3.2 Inter-Communication between MUAVs

In order to communicate among multiple MUAVs, we need to have a reliable communication setup. Communication media including Bluetooth, WiFi and XBee were explored for this study keeping in view the bandwidth, cost and power consumption. Bluetooth was selected due to its ease of use, low-cost and low power consumption. Bluetooth protocol is nothing but a wireless version of UART protocol. Though bandwidth is low, however for our present study this is sufficient to meet the purpose.

2.3.2.1 Bluetooth Communication

Bluetooth protocol has been used in two scenarios;

1. Inter-communication between quadcopters, and
2. Communication between quadcopter and desktop PC.

For inter-quadcopter communication, BTM-222 Bluetooth modules (introduced in subsection 2.2.1) have been used. These modules are required to be configured in order to meet the requirements. Different configurations for stop bits, parity bits, baud rate, device name, auto connect setting, PIN code, role setting as a master or slave etc. may be made. Default settings for this module are baud rate as 19200, data bits as 8 and no parity and stop bit. Before installing IMUs and BTM-222 bluetooth modules to quadcopter, the scheme for inter-communication between AVR32 boards was verified.

For second scenario (communication between quadcopter and desktop PC), Logilink USB bluetooth V4.0 Dongle has been used that is also a class 1 device. Logilink dongle is automatically configured, while BTM-222 module is to be configured by the operator before using it. A USB Bluetooth adaptor is plugged into the computer to establish communication between micro controller and MATLAB ICT. The laptop computers already have a built-in adaptor. When a bluetooth adaptor is plugged into a PC, two virtual serial ports are generated, that can be seen in the port field, one for incoming data and the other for outgoing data. In case of incoming COM port, device initiates the connection and in case of outgoing COM port, computer initiates the connection. ICT can identify bluetooth devices within range when queried. Logilink bluetooth dongle was used to communicate with bluetooth of leader quadcopter to acquire the attitude information.

2.3.2.2 IMU Data Transmission

Current pose of the leader is required to be transmitted to the follower in real-time for synchronized motion of two units in attitude. To measure the attitude values, Euler angles or quaternion values may be used. Quaternion values provide the information about rotating objects in space, and can be derived through some manipulations of IMU measured outputs. For this testing, IMU-3000 and two small AVR boards were used. IMU-3000 was connected to PIN PA29 and PA30 (corresponding to TWI-SDA and TWI-SCL) of board 1. BTM-222 was connected to USART0 for transmission of quaternion values derived from IMU outputs. Another BTM-222 module was connected to USART0 of board 2 to receive this information. In order to use the quaternion values for computation purpose, a ring buffer was used at the receiving board. Received quaternion information was placed in the ring buffer and then read from it in order to manipulate this information for further data processing. Received information was

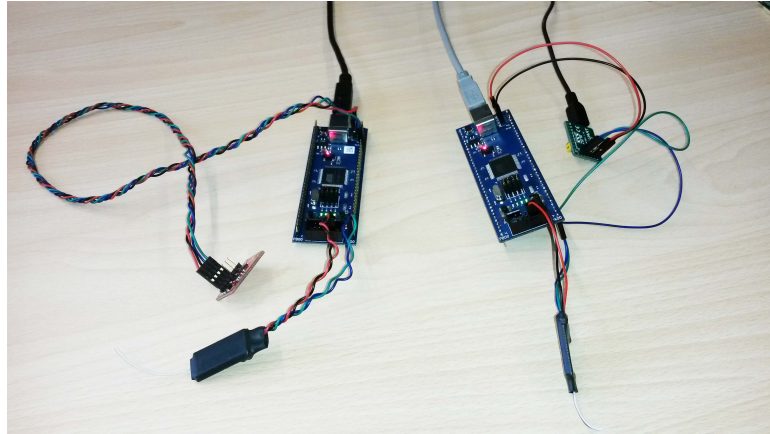


FIGURE 2.9: Testing for IMU data transmission between AVR32 small boards.

further fetched to PC for display purpose, through USB UART board connected to USART1 of board2. A test environment to this effect is shown in Figure 2.9. Using the same scheme, we can transmit different types of information from one micro controller to the other using bluetooth and then manipulate this information at the receiving end. This testing verified the efficacy of communication methodology to be used for inter-quadcopter communication.

2.3.2.3 GPS Data Transmission

We used two AVR32 micro controllers for transmission of GPS data from one board to the other through bluetooth modules. For testing purpose, NAVILOCK GPS receiver model NL-507ETTL was used. Baud rate was set to 9600, the default baud rate for said GPS receiver. Same program was loaded on both the boards. BTM222 module connected to AVR board 1 was configured as slave while the other with AVR board 2 was set as master. GPS receiver was connected to pins PA00 and PA01 corresponding to USART0-RXD and USART0-TXD of AVR32 board 1. Testing was aimed to transmit GPS processed information, in *NMEA 0183* format, to AVR32 board 2. BTM-222 module was connected to pins PA05 and PA06 corresponding to USART1-RXD and USART1-TXD of board 1. Other BTM-222 module was connected to USART0 of board 2. Received GPS data was further fetched to the computer through USART1 and displayed on the terminal program *HTerm*. Connection arrangement during testing is shown in Figure 2.10. It is also possible that GPS receiver communicates with AVR board at 9600 baud rate, and AVR board communicates with other board through bluetooth at a different rate. Same was tested and found working in order. GPS data was displayed in the form of *NMEA 0183* sentences, thereby displaying UTC time, GPS position (latitude, longitude and altitude), total number of satellites in view, azimuth/elevation/SNR of visible satellites, GPS satellites used for position solution, Horizontal Dilution Of Precision (HDOP) values, Vertical Dilution Of Precision (VDOP)

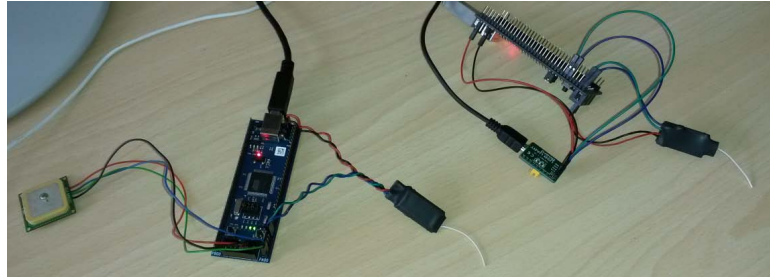
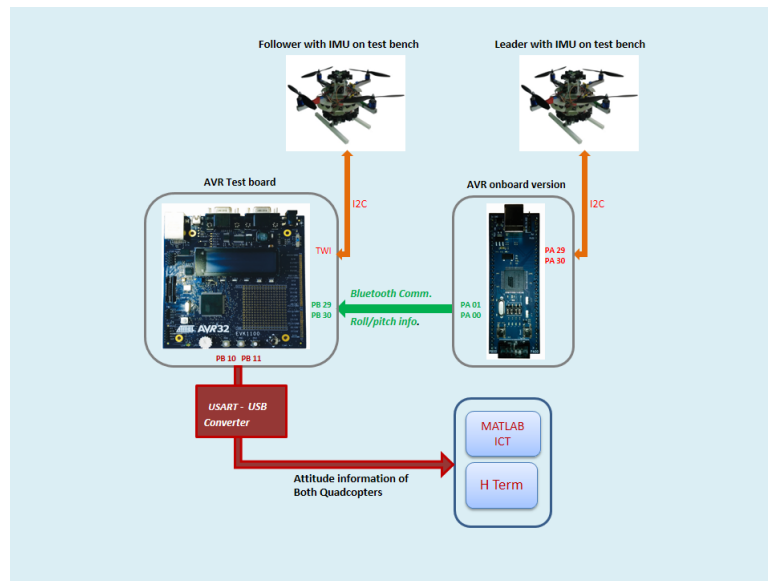


FIGURE 2.10: Testing for GPS data transmission.

FIGURE 2.11: Layout for complete test setup.
Figure adapted from Ref. [2].

values, Position Dilution Of Precision (PDOP) values, ground speed and course over ground. Overall layout for complete test setup is shown in Figure 2.11.

2.4 Control Technique

Before applying sophisticated control techniques, it was planned to employ a comparatively simple approach of PID controller. It is the most widely used controller due to its ease of application. Performance may be improved while using it in combination with Kalman filter. Low-cost MEMS sensors do not provide refined output, so we need to use Kalman filter to get the *true* signal. Here Kalman filter reduces the measurement noise to extract the true signal to be used for feedback purpose. PID controller is principally a linear combination of Proportional, Integral and Differential of error to generate the control signal for the system under consideration. Here error is the difference between

reference value and filtered output of the system. PID controller application necessitates to adjust in an iterative manner the three parameters K_P , K_I and K_D , referred to as tuning parameters. K_P is the proportional gain, K_I is the integral gain and K_D is the derivative gain. System performance depends on these three parameters. If the tuning parameters are not chosen correctly, system will not achieve the commanded value, and can even be unstable. Note that all three tuning parameters may not be required depending on the application. Some parameters may be set as zero, and resultant controller is known as PI, PD or Integral controller etc. It is noteworthy that integral action is necessary to reach the reference value. Mathematically PID control law can be expressed as follows [105]:

$$u(t) = K_p e(t) + K_I \int_0^t e(\tau) d\tau + K_D \frac{de(t)}{dt}$$

or alternatively [106];

$$u(t) = K_p [e(t) + \frac{1}{T_I} \int_0^t e(\tau) d\tau + T_D \frac{de(t)}{dt}]$$

Here $e(t) = r(t) - y(t)$ is the error signal, K_p is scaling factor, T_I is integral time constant and T_D is derivative time constant. PID control can usually perform in a variety of settings without the need for a precise model of the underlying plant that is a big advantage when system identification is either difficult or imprecise [107]. However this control technique is not suitable for sophisticated applications where high accuracy is required; as PID controller suffers from limitation in optimality and robustness. It implies that there may be more control effort exerted than optimum values. It is also demanding to tune the PID parameters under some conditions. A typical PID control scheme is shown in Figure 2.12

PID controller and Kalman filter were used in [108] to design UAV formation flight control system for speed and pitch angle, and simulation results demonstrated feasibility

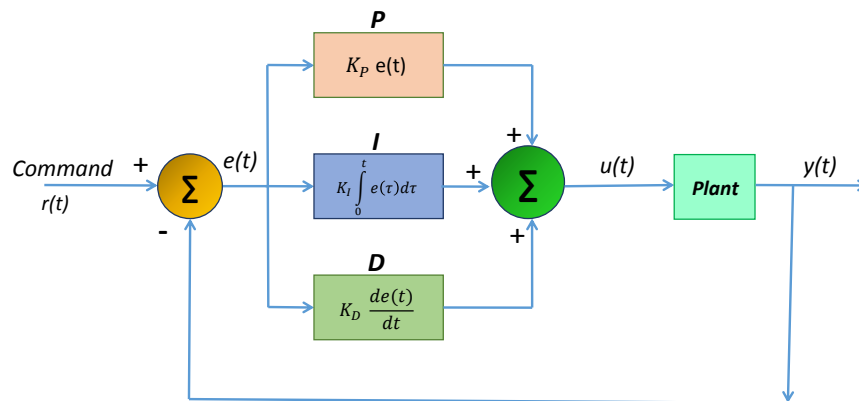


FIGURE 2.12: Block diagram for PID control scheme.

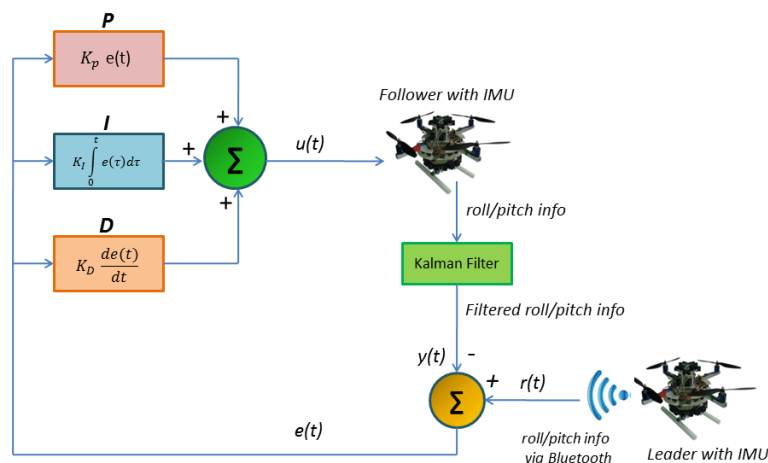


FIGURE 2.13: Block diagram showing PID control in leader-follower architecture.
Figure adapted from Ref. [2].

of the method. The study concluded in [108] demonstrated that Kalman filter and PID control performs adequately for short transition, and anti-disturbance. Results showed good stability, precision and control. For our present study, we have employed the technique on real hardware for attitude information. It fulfils the requirement of real-time and accurate control.

Filtered attitude information of leader quadcopter was communicated to follower quadcopter, with an interval of 10ms, where it was treated as reference values to generate the error signal for implementation of PID control. Follower reacted to the received information to exhibit the same attitude. Proposed approach is shown in Figure 2.13.

This scheme can later be extended with inclusion of 3D position for formation flying in real time while incorporating suitable sensors. This is the simplest approach that can be used for formation flight of multiple quadcopters. Students can implement other control algorithms as well that can be tested and subsequently employed for formation flying in real-time.

2.5 Results

Attitude information of leader quadcopter was sent to the follower quadcopter that followed the same attitude values making use of PID control combined with Kalman filter. Attitude information of both quadcopters was directed to MATLAB ICT for data plotting and analysis purpose. Main concern was communication data packets loss and communication / computation latency. Data integrity was addressed with the introduction of a checksum function. Results to this effect for roll and pitch information of both quadcopters are shown in Figure 2.14 for 1 minute that indicates that the follower quadcopter followed to leader attitude in real-time without causing much time

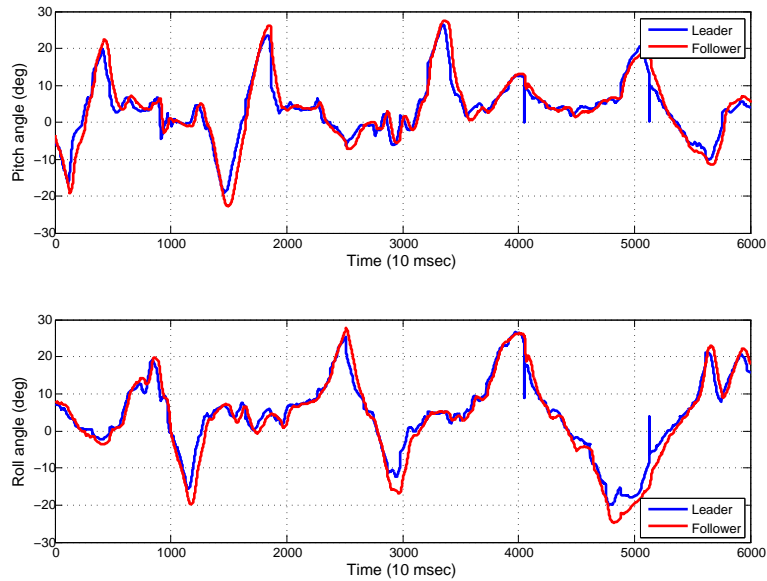


FIGURE 2.14: Attitude information of both quadcopters.
Figure adapted from Ref. [2] and slightly improved.

delay. It also shows the effectiveness of PID control (in combination with Kalman filter) that is considered to be a simple control technique, though it served the purpose. The scheme has been extensively tested and refined during the course of development. As a simple checksum function has been implemented, so we can see a jump phenomenon (for two readings out of six thousand readings) in sensor measurements for the leader. It suggests that data integrity mechanism may be further improved. The test setup is basically a foundation for implementation of sophisticated techniques in the domain of distributed control. It is planned to be used as an educational tool and for testing and validating the algorithms for cooperative control in real-time environment.

2.6 Summary

An elementary real-time formation flying test set-up in 3DOF has been conceived at Institute of Aerospace Information Technology, University of Würzburg. Proposed approach appears to be useful for further research projects as well as for education purpose. It served as a basis for realization of synchronized attitude of two quadcopters in real-time. Same setup can be extended to multiple quadcopters making synchronized motions. The test bench is flexible and may be modified as per the requirements. As it is possible for quadcopter on rod to gain height up to a certain level, so with the addition of a height sensor the setup may even be extended for fourth degree of freedom. Existing ground station for controlling one quadcopter may also be extended for two quadcopters. Transmitted data integrity algorithm may also be improved.

Chapter 3

Formation Flying of Quadcopters through Explicit Model Following Distributed Control Scheme

3.1 Introduction

Today MUAVs are a popular research platform serving the humanity in a number of ways like forest fire monitoring, spraying the insecticides and flood damage assessment etc. Some other interesting applications are envisaged that may not be performed effectively by a single MUAV and necessitate the use of multiple units. Such valuable applications include cooperative transportation, weather monitoring, search and rescue, communication relays and air refueling etc. A group of low-cost UAVs designed for cooperative missions provides redundancy and more effectiveness compared with a single high-tech and expensive UAV. However control requirements are generally more stringent and performance criteria are higher for such applications. More innovative applications are foreseen with advancement in autonomous control techniques.

In order to ensure stability of the formation, robust controllers are mostly proposed to provide insensitivity to possible uncertainties in the motion of other vehicles and communication delays. Adaptive control schemes are used to improve performance and reliability of aerial platforms. These techniques are useful to handle modeling uncertainties and time varying dynamics. As an aerial vehicle is required to operate under different environmental conditions and performance requirements are also more strict in case of formation flight, a robust as well as adaptive control scheme is highly desired.

Although a number of sophisticated adaptive and robust control schemes have been suggested for formation flight of aerial vehicles, however the algorithms involved are complex from implementation point of view. There is generally a trade-off between the

intricacy of control technique employed and the level of accuracy achieved there off. Proposed scheme for tightly coupled formation flight is simple to implement and exhibits excellent trajectory tracking performance with disturbance rejection. Also compared with most of the earlier contributions, a full-state vector of quadcopter is considered in this study while tracking only the outputs of interest (performance output) in the presence of perturbations and communication delays. Tracking performance is demonstrated through simulations for a wide range of commands including step, ramp and other time varying commands.

3.2 Problem Formulation

We have chosen leader-follower constellation architecture in a V-shaped formation as it is more intuitive and inspired by the nature. With this scheme, trajectory of the leader defines the trajectory of the formation. It is desired for the leader to track a commanded signal that may be constant or time-varying. Controller on-board the leader is designed to keep the output values close to the commanded values. For the formation, the states of the leader constitute the coordination variable, since the actions of the other vehicles in the formation are completely specified once the leader states are known [12]. Each follower is controlled to maintain its user-defined position in the formation using information of its own position and the leader position. Information of leader may be received either via inter-vehicle communication or estimated using the sensors, for example radar, laser scanner and PMD camera etc., onboard the follower. We here assume that information of leader is always available to the followers. It may be through a communication link, or sensors onboard the followers, or received via the other follower. The control problem for followers is to maintain the user-defined distances.

Communication topology for leader and two followers is shown in Figure 3.1. In case of communication loss between leader and any of the followers, the other follower quadcopter provides the leader states to effected follower quadcopter in order to keep the formation intact.

3.2.1 Quadcopter Dynamic Model

Quadcopter motion dynamics are described in a number of publications, for e.g. [109]. However these are repeated here for completeness. Quadcopter has four rotors numbered 1 to 4, as shown in Figure 3.2, representing front, left, rear and right rotors respectively. Front and rear rotors rotate anti-clockwise while left and right rotors rotate clockwise. Thus two pairs of rotors mutually cancel the gyroscopic effects and aerodynamic torques. Unlike a fixed-wing aircraft with conventional control surfaces, a quadcopter is controlled through differential speed of rotors. In hovering mode, speeds of all four rotors is same. Vertical motion is controlled through aggregate thrust generated by all rotors. Pitch angle (around y-axis) is generated through differential speeds of front and rear rotors

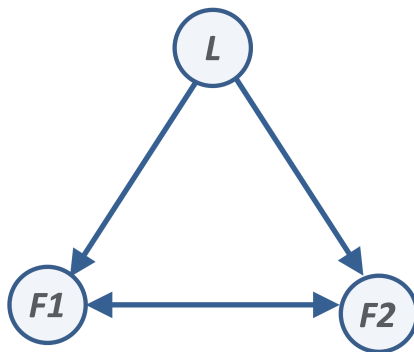


FIGURE 3.1: Communication topology of quadcopters in formation.
Figure adapted from Ref. [3]. © (2016) IEEE.

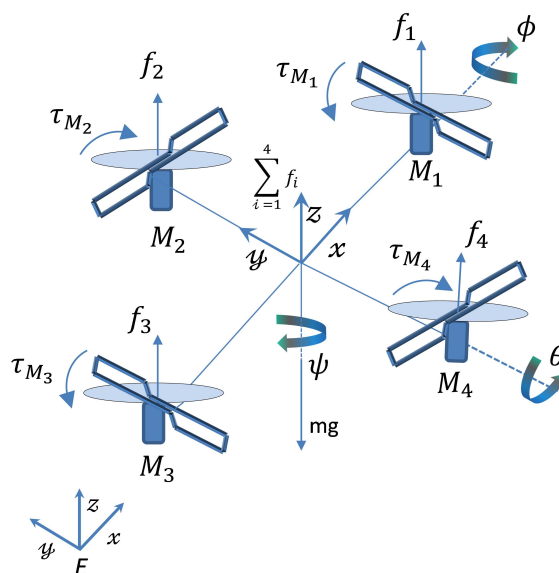


FIGURE 3.2: Quadcopter motion dynamics.
Figure adapted from Ref. [3]. © (2016) IEEE.

causing the forward/backward motion in x -direction. Similarly for lateral motion in y -direction, speeds of left and right rotors is varied differentially thereby generating the roll angle. For yawing motion, speeds of pair of rotors 1 and 3 is varied compared with 2 and 4 in such a manner that total thrust remains same in order to maintain the altitude. Torque so generated causes the yaw motion. Thus all rotors play their role for yaw motion.

Quadcopter dynamics, as described above, are modeled as per the following equations [110]:

$$\begin{aligned}
u &= f_1 + f_2 + f_3 + f_4 \\
f_i &= k_i \omega_i^2, i = 1 \dots 4 \\
m\ddot{x} &= -u \sin \theta \\
m\ddot{y} &= u \cos \theta \sin \phi \\
m\ddot{z} &= u \cos \theta \cos \phi - mg \\
\ddot{\psi} &= \tau_\psi \\
\ddot{\theta} &= \tau_\theta \\
\ddot{\phi} &= \tau_\phi
\end{aligned} \tag{3.1}$$

Here u shows the total thrust generated by four rotors, f_i is the force produced due to the rotation of rotor i , $k_i > 0$ is a constant, ω_i is the angular speed of motor i ($M_i, i = 1 \dots 4$), m is mass of quadcopter and g is gravitational constant. τ_ψ , τ_θ and τ_ϕ represent the control inputs for yawing, pitching and rolling moments respectively. Quadcopter system under consideration has four inputs, twelve states and six outputs. Our state vector comprises $[x \dot{x} y \dot{y} z \dot{z} \psi \dot{\psi} \theta \dot{\theta} \phi \dot{\phi}]^T$. It represent the 3D position of center of mass of quadcopter in x, y and z-direction relative to the earth-fixed frame E [110], and Euler angles namely the yaw angle ψ around the z-axis, the pitch angle θ around the y-axis, and the roll angle ϕ around the x-axis respectively with their time derivatives denoted with dot ($\dot{\cdot}$) overhead. Outputs vector is $[x y z \psi \theta \phi]^T$, and performance output vector is $[x y z]^T$. We now define the following state equations in order to linearize the system 3.1 as given in [111]:

$$\begin{aligned}
\zeta &= [x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9 x_{10} x_{11} x_{12}]^T \\
&= [x \dot{x} y \dot{y} z \dot{z} \psi \dot{\psi} \theta \dot{\theta} \phi \dot{\phi}]^T
\end{aligned}$$

and our control vector is:

$$\mathbb{U} = [u_1 u_2 u_3 u_4]^T = [u - mg \tau_\psi \tau_\theta \tau_\phi]^T$$

Dynamic equations may be written as:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \\ \dot{x}_5 \\ \dot{x}_6 \\ \dot{x}_7 \\ \dot{x}_8 \\ \dot{x}_9 \\ \dot{x}_{10} \\ \dot{x}_{11} \\ \dot{x}_{12} \end{bmatrix} = \begin{bmatrix} x_2 \\ -u_1 \sin x_9 / m - g \sin x_9 \\ x_4 \\ u_1 \cos x_9 \sin x_{11} / m + g \cos x_9 \sin x_{11} \\ x_6 \\ u_1 \cos x_9 \cos x_{11} / m + g \cos x_9 \cos x_{11} - g \\ x_8 \\ u_2 \\ x_{10} \\ u_3 \\ x_{12} \\ u_4 \end{bmatrix}$$

or shortly:

$$\dot{\zeta} = f(\zeta, \mathbb{U})$$

Now let $\zeta = 0$ be an equilibrium point with $f(0,0)=0$. The linearization by Taylor series about the origin gives the following linear system

$$\dot{\zeta} = A\zeta + B\mathbb{U} \quad (3.2)$$

where A and B represent the state matrix and input matrix respectively. For further details, please refer to [111].

Here longitudinal and lateral dynamics are decoupled after linearization. Instead of focusing on either lateral or longitudinal dynamics, we consider them simultaneously. Leader dynamic model is taken as a reference model thereby defining the ideal output response to be followed by two wingmen. We further assume a *heterogeneous formation*, where the leader hardware (and hence mass) is different from the followers to simulate a more realistic scenario. This assumption is based on the fact that in reality a leader is likely to be more equipped than followers in terms of sensors and hence will be more massive. Weight of *micro* UAVs is between 0.1 – 0.5 kg [9]. For our present study, we have assumed that the mass of leader is 2kg and that of followers is 0.64kg each. Therefore a new acronym of MUAV has been introduced in this thesis to refer to *mini* UAVs.

3.2.2 Formation Dynamic Model

Desired dynamics of the leader, referred to as reference model, are defined as an LTI system in the form:

$$\begin{aligned}\dot{x}_l &= A_l x_l + B_l r(t) \\ y_l &= C_l x_l + D_l r(t)\end{aligned}\tag{3.3}$$

Suffix l represents the leader and f will represent follower. For the system (3.3), $x_l \in \mathcal{R}^{n_l}$ are states of the leader quadcopter, $r(t)$ is the external bounded command value, and $y_l \in \mathcal{R}^{p_l}$ is the performance output vector of leader. Leader state space matrices (A_l, B_l, C_l, D_l) have their conventional meaning. For the leader to exhibit stable dynamics, the state matrix A_l of the leader needs to be Hurwitz i.e. all its eigenvalues have strictly negative real part. Follower quadcopter dynamics are described as follows:

$$\begin{aligned}\dot{x}_f &= A_f x_f + B_f \mathcal{U} \\ y_f &= C_f x_f + D_f \mathcal{U}\end{aligned}\tag{3.4}$$

Here $x_f \in \mathcal{R}^{n_f}$ are states of the follower quadcopter. $\mathcal{U} \in \mathcal{R}^{m_f}$ is the control vector. $y_f \in \mathcal{R}^{p_f}$ is the performance output vector of follower with $p_f \leq m_f$ [112]. p_f and m_f refer to number of outputs y_f and number of control inputs \mathcal{U} respectively. The matrices (A_f, B_f, C_f, D_f) are of corresponding dimensions. Quadcopter is an under-actuated system (four inputs and 6DOF motion) and LQR PI approach is not applicable for under-actuated systems [14] to track all outputs. However this control scheme may be applied with suitable formulation of problem while tracking only outputs of interest (performance output). With this scheme, maximum number of trackable outputs may not exceed number of control inputs (four in case of quadcopter). It is assumed that the matrix pair (A_f, B_f) is stabilizable and state vectors x_l and x_f are available as these are to be used for formulation of control signal \mathcal{U} for follower.

Before implementing the controller, we first do the controllability check for the followers using the controllability matrix $P = [B_f \ A_f B_f \ A_f^2 B_f \ \dots \ A_f^{n_f-1} B_f]$ and find it to be full ranked. All of the closed-loop poles may be assigned to desired locations through pole placement method. Eigenvalues of the leader state matrix A may be effected through the gain matrix of pole placement method. Signs of eigenvalues of state matrix determine stability and their absolute values inform about how quickly the steady state may be reached. Though the leader in our study is controlled through pole placement method, however same may be controlled though some other scheme as well, for example LQR. Leader receives a known bounded command $r(t)$ that may or may not vary with time. It is emphasized that the proposed control scheme is meant for the followers to maintain the desired separations from the leader. For *centralized* leader-follower architecture, we are primarily interested to determine control input signal \mathcal{U} for the follower such that the desired 3D formation geometry, given by the relative distance vector $r = [r_x \ r_y \ r_z]^T$, is maintained. For this an LQR controller with PI feedback connection [112] is implemented for the follower. We assume that all the states of quadcopters are available through suitable sensors. Implementation of this scheme for *centralized heterogeneous leader follower architecture* is not seen in the literature, as

per the knowledge of author of this thesis. The control signal is computed with LQR scheme while suitably augmenting the states of follower with that of leader. Design objectives in LQR scheme are defined through Q and R matrices. Q matrix, a positive semi-definite matrix, shows the weightage (or importance) to states. R being a positive definite matrix indicates weightage of control efforts corresponding to control inputs. The controller can be tuned by changing the elements in the Q and R matrices to achieve a desirable response. An efficient LQR control scheme is based on finding the right weighting factors. Q and R matrices used for this study are given in Appendix A.

Model following LQR PI control scheme given in [112] has been exploited for our present study and tailored for leader-follower tightly coupled formation flight. Open-loop dynamics of follower and leader may be formulated as:

$$\begin{bmatrix} \dot{x}_f \\ \dot{x}_l \end{bmatrix} = \begin{bmatrix} A_f & O_{n_f \times n_l} \\ O_{n_l \times n_f} & A_l \end{bmatrix} \begin{bmatrix} x_f \\ x_l \end{bmatrix} + \begin{bmatrix} B_f \\ O_{n_l \times m_f} \end{bmatrix} \mathcal{U} + \begin{bmatrix} O_{n_f \times m_f} \\ B_l \end{bmatrix} r(t) \quad (3.5)$$

Tracking error represented as δy may be written as:

$$\delta y = y_f - y_l = C_f x_f + D_f \mathcal{U} - C_l x_l - D_l r(t) \quad (3.6)$$

Writing equation (3.6) in matrix form:

$$\delta y = [C_f \quad -C_l] \begin{bmatrix} x_f \\ x_l \end{bmatrix} + D_f \mathcal{U} + (-D_l) r(t) \quad (3.7)$$

Now the open-loop dynamics given by equations (3.5) and (3.7) may be written concisely as:

$$\begin{aligned} \dot{x} &= Ax + B \mathcal{U} + B_r r(t) \\ \delta y &= Cx + D \mathcal{U} + D_r r(t) \end{aligned} \quad (3.8)$$

We aim to find such a \mathcal{U} that the tracking error (or system output) asymptotically tends to zero in the presence of any known, bounded and possibly time varying command $r(t)$. Formulation of above problem suggest that the tracking problem has been converted into a regulation problem [113], referred to as CGT. Now an *Integral Control* is applied to track a step input command with zero errors. For this scenario a practical tracker like Command Generator Tracker (CGT) is chosen. Integrated tracking error e_y may be written as:

$$\dot{e}_y = \delta y \quad (3.9)$$

Introducing equation (3.9) into system dynamics (3.8), it gives:

$$\begin{bmatrix} \dot{e}_y \\ \dot{x} \end{bmatrix} = \begin{bmatrix} O_{p_f \times p_f} & C \\ O_{(n_f + n_l) \times p_f} & A \end{bmatrix} \begin{bmatrix} e_y \\ x \end{bmatrix} + \begin{bmatrix} D \\ B \end{bmatrix} \mathcal{U} + \begin{bmatrix} D_r \\ B_r \end{bmatrix} r(t) \quad (3.10)$$

Tracking error δy may be defined as:

$$\delta y = [O_{p_f \times p_f} \quad C] \begin{bmatrix} e_y \\ x \end{bmatrix} + D \mathcal{U} + D_r r(t) \quad (3.11)$$

Now equations (3.10) and (3.11) may be written as:

$$\begin{aligned} \dot{\tilde{x}} &= \tilde{A} \tilde{x} + \tilde{B} \mathcal{U} + \tilde{B}_r r(t) \\ \delta y &= \tilde{C} \tilde{x} + \tilde{D} \mathcal{U} + \tilde{D}_r r(t) \end{aligned} \quad (3.12)$$

Care is to be exercised while formulating the \tilde{A} matrix for proper dimensions. Also note that for above system (3.12), total number of states are equal to the sum of vector length of $\delta y + x_f + x_l$.

3.3 Control Strategy

We now assume that external command is step-input command with zero errors and then differentiate state dynamics equation of system (3.12) w.r.t. time and introduce new variables:

$$\wp = [\dot{e}_y \quad \dot{x}]^T, \text{ and } v = \dot{\mathcal{U}} \quad (3.13)$$

we may now write:

$$\dot{\wp} = \tilde{A} \wp + \tilde{B} v \quad (3.14)$$

Open-loop dynamics of system (3.14) may be controlled through LQR control scheme. LQR is a very attractive control approach as it is capable to handle multiple actuators and complex system dynamics. Furthermore, it offers very large stability margins to errors in the loop gain. However, LQR assumes access to the states. In order to minimize the LQR cost, control input v is used. Cost function J may be defined as:

$$J = \int_0^{\infty} (\wp^T Q \wp + v^T R v) dt \quad (3.15)$$

Q and R are LQR weight matrices defined as before. Now solving the Algebraic Riccati Equation (ARE)

$$\tilde{A}^T P + P \tilde{A} + Q - P \tilde{B} R^{-1} \tilde{B}^T P = 0 \quad (3.16)$$

gives the solution $P = P^T > 0$, that is used to compute the control signal \mathcal{U} . LQR gain matrix K is given as under:

$$K = R^{-1} \tilde{B}^T P \quad (3.17)$$

LQR control signal may be written as:

$$v = \dot{\mathcal{U}} = -K \varphi \quad (3.18)$$

LQR PI controller for a MIMO system corresponds to a gain matrix with order equal to the number of elements in $\mathcal{U} \times (\delta y + x_f + x_l)$. Our gain matrix is of the following form:

$$K = [K_I^e \quad K_p^{x_f} \quad K_p^{x_l}] \quad (3.19)$$

Here subscript indicates type of gain (proportional or Integral) and superscript shows the variable to whom this gain matrix is applied. Now

$$\dot{\mathcal{U}} = - [K_I^e \quad K_p^{x_f} \quad K_p^{x_l}] \begin{bmatrix} \dot{e}_y \\ \dot{x}_f \\ \dot{x}_l \end{bmatrix} \quad (3.20)$$

Equation 3.20 may be written as:

$$\dot{\mathcal{U}} = - [K_I^e \quad K_p^{x_f} \quad K_p^{x_l}] \begin{bmatrix} \dot{e}_y \\ \dot{x}_f \\ \dot{x}_l \end{bmatrix} \quad (3.21)$$

Integrating equation (3.21) and ignoring constants of integration, LQR PI control solution for the follower is given as:

$$\mathcal{U} = K_I^e \frac{(y_l - y_f)}{s} - K_p^{x_f} x_f - K_p^{x_l} x_l \quad (3.22)$$

We note that structure of LQR PI gain matrix comprises three parts; an output feedback $K_p^{x_f}$, a feed-forward compensator $K_p^{x_l}$ and an additional feed-forward filter K_I^e in the error channel that guarantees perfect tracking. Although our derivation is based on

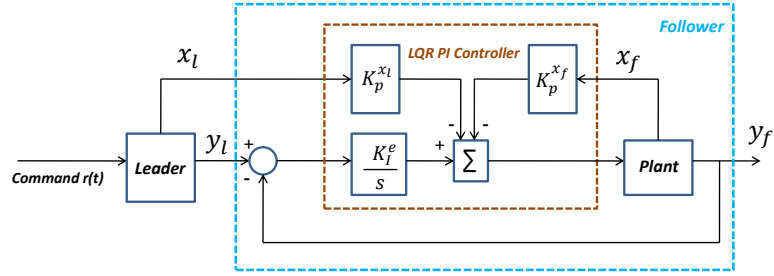


FIGURE 3.3: LQR PI control scheme for leader-follower architecture. Figure partly adapted from Ref. [3]. © (2016) IEEE.

step reference command, the resulting control system gives good time response for any arbitrary reference command signal $r(t)$ [14], as will be demonstrated in simulation section. Figure 3.3 shows the interconnection diagram for LQR PI control scheme implemented for leader-follower architecture. Detailed sub-block level diagram is shown as Figure 3.4. Dynamics of leader and follower plant are defined by (3.3) and (3.4) respectively.

It is established in [114] that for perfect tracking it is necessary to have as many control inputs in vector $\mathcal{U}(t)$ as there are in command signals $r(t)$ for tracking. However we slightly modify it to state that number of control inputs should be equal to or more than the command signals, as demonstrated in this study while using three command signals (for 3D position control) and four control signals. We now give some propositions / definitions to determine the stability of a leader-follower architecture based on LQR PI control scheme.

Proposition 1. All signals for a leader-follower formation, based on LQR PI control scheme, are bounded if $\|G_L\|_\infty \leq \Gamma$ and $\|K\|_\infty \leq \delta$ where G_L is the leader state space model, K is LQR PI gain matrix, and Γ and δ are bounded numbers. Infinity norm of gain matrix K gives an indication of maximum control effort.

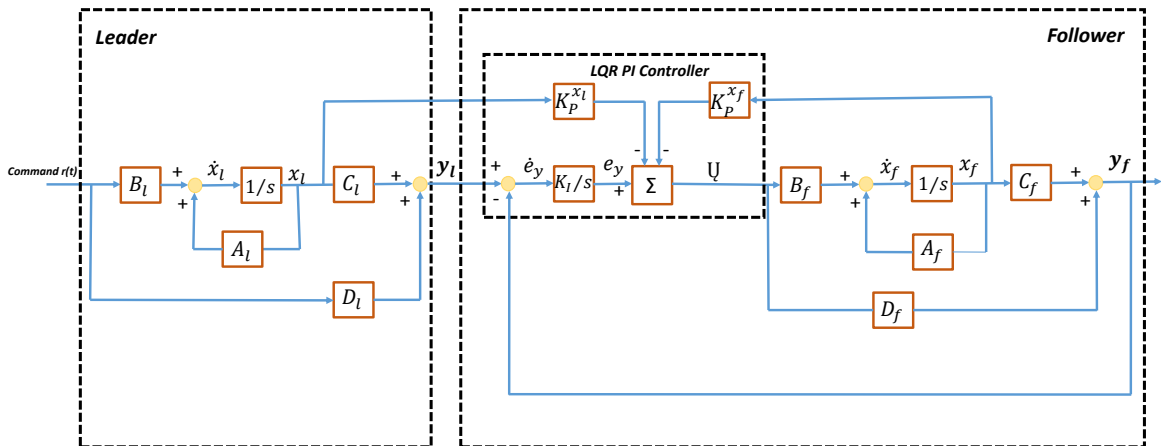


FIGURE 3.4: Sub-block level diagram of LQR PI control scheme.

Proposition 2. A leader-follower formation based on LQR PI control scheme is stable if $\|\tilde{G}_{cl}\|_{\infty} \leq 1$ where \tilde{G}_{cl} represents the state space system comprising leader-follower, controlled through LQR PI control scheme, and is given by the matrices $(\tilde{A}_{cl}, \tilde{B}, \tilde{C}, \tilde{D})$ where $\tilde{A}_{cl} = \tilde{A} - \tilde{B} * K$.

Definition 1. A leader-follower formation based on LQR PI control scheme is stable if all poles of the matrix \tilde{A}_{cl} are strictly in open left-half plane.

3.4 Simulation Results and Discussion

From safety point of view, most of the proposed algorithms require to be simulated before actual flights may be undertaken. It also helps to authenticate the efficacy of the algorithm. Above defined model of leader-follower scheme was implemented in *MATLAB/Simulink*. For this study, possible perturbations on outputs of leader, on control inputs and outputs of followers have been considered and implemented. Wind gust was simulated as a step function on quadcopters outputs. Communication delays between all three agents were also implemented in our simulation model as is the case in real scenarios. System was also tested for ramp and sinusoidal command values. This command generator is capable of handling a wide range of motion trajectories, including position unit step commands, unit ramp commands, oscillatory commands and more [115]. Table 3.1 shows some of the metadata under which the system was simulated while taking a combination of different entries from the table. Simulation results are depicted in Figure 3.5 and Figure 3.6 that show the positions of all three units in three-axes and the 3D view of the whole formation respectively in presence of perturbations. Initial and final positions of units are also shown in Figure 3.6 to indicate the direction of movement. From the plots, it is evident that output of leader is tracked quite smoothly by the followers while maintaining the desired separations. Another remarkable feature of this scheme is that when some disturbance is encountered on the output of leader (e.g. a wind gust) and there is some deviations from the intended trajectory, the followers also make the similar movement in order to maintain the formation geometry. Simulations in Figure 3.5 and Figure 3.6 under different perturbations at different time instants show excellent tracking performance for commanded values in the form of step input (x-axis and z-axis) and ramp command with a slope of 1.5 (y-axis). Tracking performance was also observed for sinusoidal command and good performance was observed, as demonstrated in ensuing paragraphs.

For a formation having a single leader, the equilibrium point is the desired relative position of the vehicles [91] that is shown to be maintained under a number of perturbations as depicted in Figure 3.5 and Figure 3.6. Results may also be interpreted as follower quadcopters exhibiting the static stability. This is verified by observing their immediate response following a disturbance from a trimmed flight condition [14], where follower quadcopters return to their equilibrium points.

TABLE 3.1: Metadata for Simulation.
Table adapted and extended from Ref. [3]. © (2016) IEEE.

	Leader	Follower 1	Follower 2	Activation Time (s)
Initial Position	[10,10,10]	[0,2,3]	[4,1,2]	[0,0,0]
Commanded Position 1	[40,y_des ^a ,40]	x = - 10 ^c y = - 15 ^c z = - 5 ^c	x = - 15 ^c y = - 10 ^c z = - 10 ^c	[0,0,0]
Commanded Position 2	[100,y_des ^b ,100]	x = - 20 ^c y = - 30 ^c z = - 10 ^c	x = - 30 ^c y = - 20 ^c z = - 20 ^c	[0,0,0]
New Formation Geometry	[40,y_des ^a ,40]	x = - 20 ^c y = - 30 ^c z = - 10 ^c	x = - 30 ^c y = - 20 ^c z = - 20 ^c	[40,30,30]
Input Disturbance	Nil	[1,1,1,1]	[2,2,2,2]	[20,20,20,20] & [50,50,50,50]
Output Disturbance (wind gust 1)	[0,0,0]	[5,10,12]	[10,5,15]	[50,60,70] & [40,50,60]
Output Disturbance (wind gust 2)	[0,0,0]	[20,5,10]	[10,5,10]	[50,40,40] & [30,50,50]
Comm. Delay	[1,1] ^d	[1,1] ^e	[2,2] ^f	-
Comm. Loss	-	-	-	35 ^g

^ay_des is a ramp command with slope 1.5 for changing position in y-direction.

^by_des is a ramp command with slope 1 for changing position in y-direction.

^cSeparation between leader and follower.

^dFor states & outputs between leader and follower 1.

^eFor states & outputs between leader and follower 2.

^fFor states & outputs between follower1 and follower 2.

^gCommunication loss between leader and follower 1.

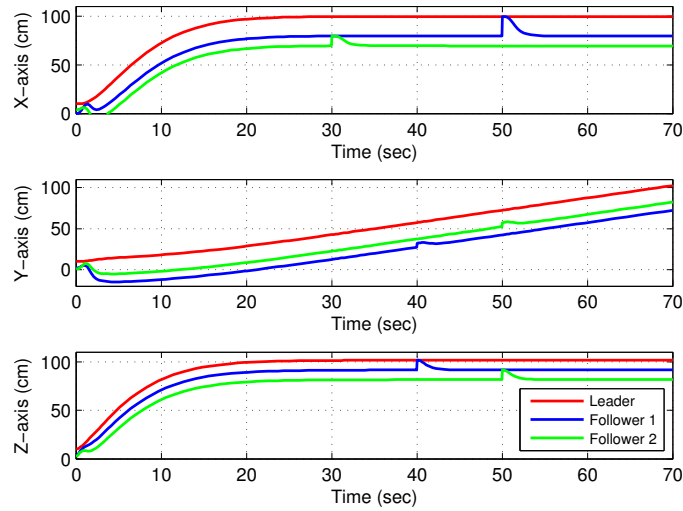


FIGURE 3.5: Three-axes position of whole formation under disturbances.

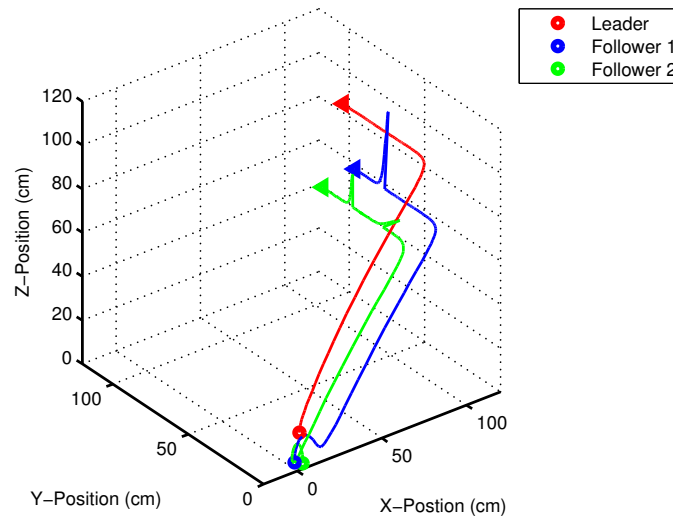


FIGURE 3.6: Trajectory for leader and followers under disturbances.

Contrary to the Proportional Differential (PD) approach used in [111], no oscillation behavior is observed in steady state with the use of LQR PI control scheme that shows its strength. One PID controller gives one control signal at a time. However for controller like LQR, all the states are taken care of simultaneously with the use of controller in the form of matrix K that facilitates to provide multiple appropriate control signals at the same time. Solving matrix equations allows all the control gains to be computed simultaneously so that all the loops are closed at the same time [14].

Some missions may require to change the formation geometry during flight to cope with changing situations and environment, like collision avoidance. In order to simulate such scenario, we changed the formation geometry, at $t = 40$ sec for x-axis and at $t = 30$ sec for y-axis and z-axis, while doubling the separations between leader and followers. Corresponding simulation is shown in Figure 3.7 that reveals excellent performance to

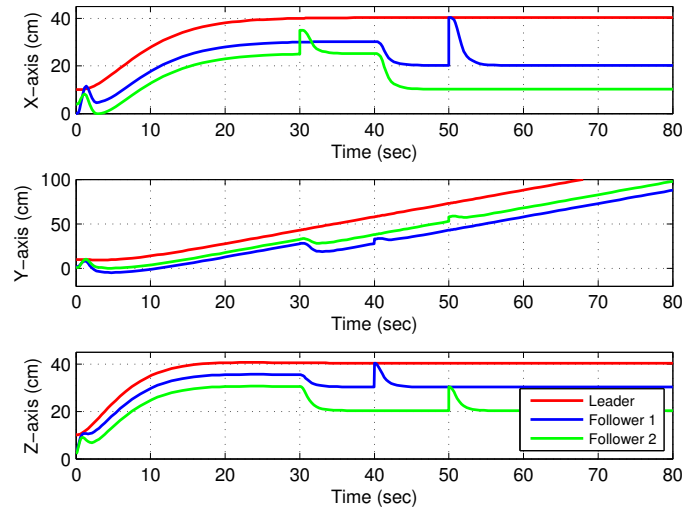


FIGURE 3.7: Three axes position of units under switching formation geometry.

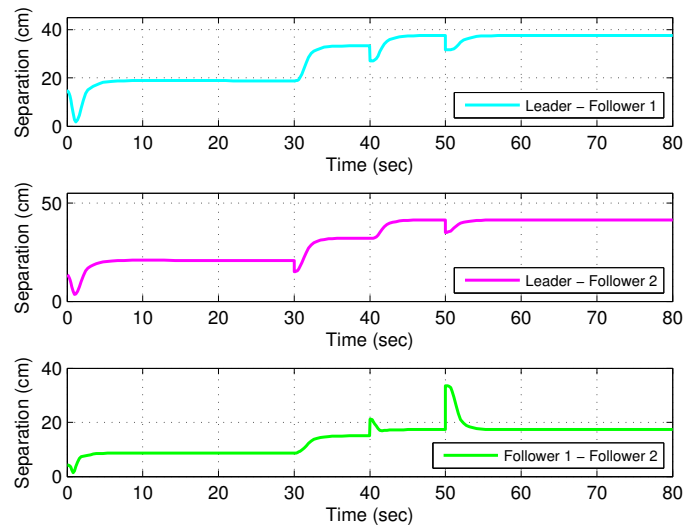


FIGURE 3.8: Inter-unit separations in the formation.

maintain new desired formation geometry even under perturbations at different time instants. For this particular maneuver and formation geometry, separations are increased among all the units so collisions are not imagined. However for a different scenario, a suitable collision avoidance scheme is required to be adopted.

We now incorporate a combination of possible perturbations together with their numerical values depicted in Table 3.1 and plot the relative distances between all the three units while also changing the formation geometry at $t = 40$ sec for x-axis and at $t = 30$ sec for y-axis and z-axis. Corresponding plot is shown in Figure 3.8 which demonstrates that required 3D formation geometry is maintained under a number of disturbances.

We now demonstrate the response of whole formation for a sinusoidal command to the leader on x-axis with a frequency of 0.2 rad/sec and a bias of 35cm. Plot to this effect,

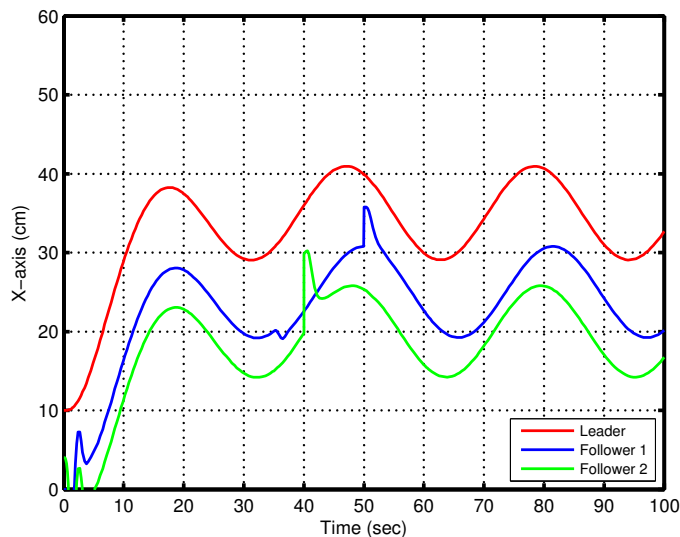


FIGURE 3.9: Response of formation under a sinusoidal command.

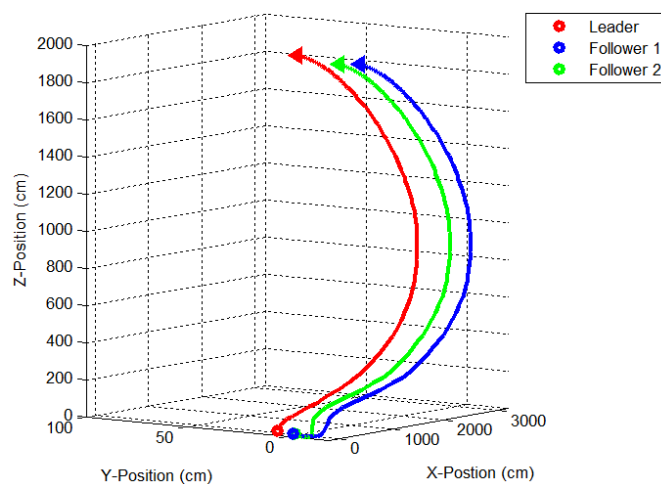


FIGURE 3.10: Trajectory of formation for an extended distance.

in the presence of disturbances and communication route switching (at $t = 35\text{sec}$), is shown in Figure 3.9. However for such commands, desired tracking (of same frequency) is achieved if rate of change of command signal is within the closed-loop system bandwidth [112]. Though we get the sinusoidal output but with a different amplitude and phase determined by the magnitude of the system transfer function. Desired amplitude may be achieved using a suitable pre-compensator. Thus the control scheme is capable of variable set-point tracking.

Quadcopters trajectories in all the foregone plots are shown for about one meter only to clearly show the effects of perturbations and their rejection. We now show the formation trajectory under time-varying command for a distance of 30m , 1m and 18m in x -, y - and z -direction respectively in Figure 3.10.

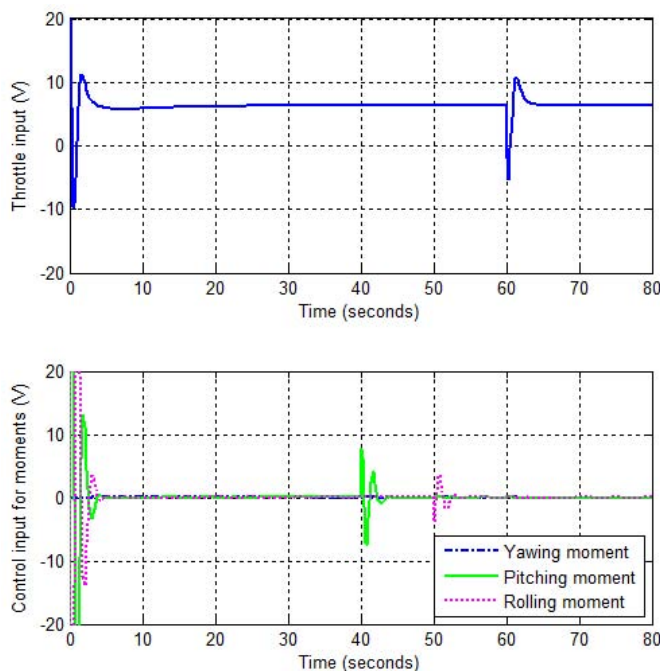


FIGURE 3.11: Control input for throttle and moments to regain equilibrium.
Figure adapted from Ref. [3]. © (2016) IEEE.

In order to further verify the health of simulation results, we now introduce a perturbation of magnitude [10,5,15] on follower 2 on x-, y- and z- axis individually at $t = 40, 50$ and 60 sec respectively. Required throttle input and control inputs for the yawing, pitching and rolling moments to bring the quadcopter back to desired equilibrium condition are shown in Figure 3.11 that gives the expected results. This plot is just to realize how the control inputs are exercised to cater for the perturbations on the desired position.

Pulse Width Modulation (PWM) output values of four motors speed are as following, as defined in [111]:

$$\begin{aligned}
 PWM_{M1} &= u + \tau_{\theta} + \tau_{\psi} \\
 PWM_{M2} &= u - \tau_{\phi} - \tau_{\psi} \\
 PWM_{M3} &= u - \tau_{\theta} + \tau_{\psi} \\
 PWM_{M4} &= u + \tau_{\phi} - \tau_{\psi}
 \end{aligned}$$

Control voltages of four motors (under the same perturbation and time instants) to bring the quadcopter back to its desired position are plotted in Figure 3.12 that are commensurate with the expected results.

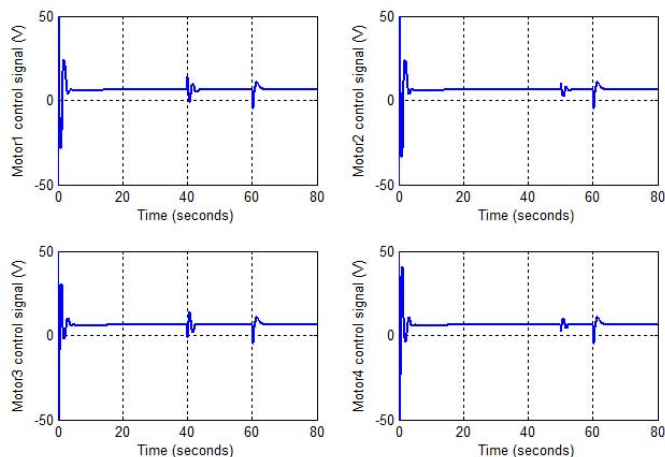


FIGURE 3.12: Control voltages of four motors of quadcopter.
Figure adapted from Ref. [3]. © (2016) IEEE.

3.5 Stability Analysis

For a MIMO system, individual gain and phase margins between different pairs of inputs and outputs mean little from the point of view of overall robustness [14]. This is due to the coupling that generally exists between different inputs and outputs of a MIMO system. Singular values are useful for robustness analysis of a MIMO system. Singular values can provide a better indication of the overall response, stability, and conditioning of a MIMO system than a channel-by-channel Bode plot. Sigma plot also gives indication of cross-over frequency. At this frequency we get the same output frequency as commanded value (though output amplitude will be different determined by transfer function of the system). In MATLAB, we can get it through the command `getGainCrossover(sys, gain)`. H_∞ norm of a MIMO system is the largest singular value across frequencies and hence is an indicator of stability of a MIMO system. Sigma plot of \tilde{G} and \tilde{G}_{cl} representing leader-follower system in open-loop and closed-loop respectively are shown in Figure 3.13 that demonstrates that LQR PI control scheme has efficiently controlled the system. The maximum singular value for the closed-loop system is below an upper limit (zero dB) that guarantees stability despite parameter variations in the linearized model. At low frequencies also, singular values of closed-loop system are below zero dB, unlike open-loop system where these are much higher than the threshold value.

3.6 Summary

For centralized formation flying we are interested for followers to track time varying output of leader in order to smoothly maintain relative 3D distances. For presented algorithm, extensive simulations were realized under a number and types of disturbances

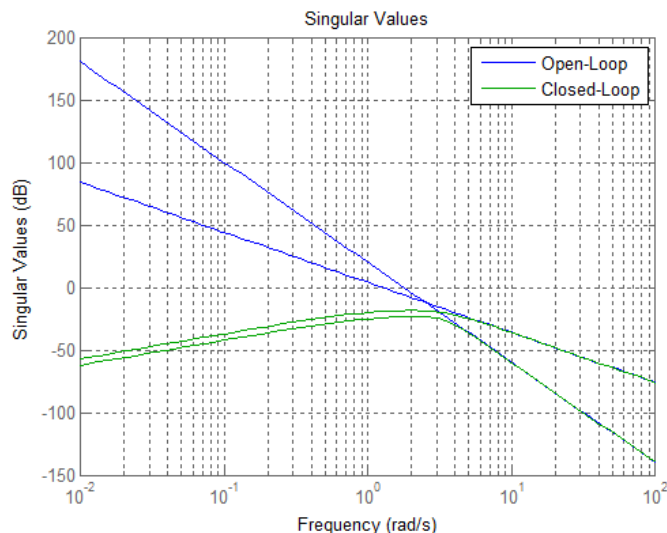


FIGURE 3.13: Sigma plot for leader-follower scheme in open-loop and closed-loop. Figure adapted from Ref. [3]. © (2016) IEEE.

including input disturbance on control values, output disturbances (e.g. a wind gust) and communication delays that revealed the follower systems to be quite robust while maintaining the desired formation geometry. This technique has promising results in terms of stability and leader output tracking even in the presence of significant perturbations and under arbitrary switching formation geometries during flight. The approach is appropriate for the scenarios where tightly coupled formation flight is desired, for example cooperative grasping and joint load transportation etc. Proposed approach for followers in the formation, is simple from implementation point of view. However it necessitates to know the exact dynamic model of aerial vehicle. It also assumes that states of leader and follower are available. In case of non-availability of states, a suitable *observer* may be designed for estimation of states. Although this scheme is suitable for small formations, however appropriate arrangements may be made to extend it to medium sized formations.

As it is quite laborious to model all the dynamics of rotorcraft flying in close formation, adaptive and robust control techniques, like LQR PI, may be explored to their full potential to cater for such scenarios. For future work, presented scheme may be implemented in real-time using the formation flying test setup developed at Institute of Aerospace Information Technology, Würzburg University [2]. A suitable collision avoidance mechanism may also be implemented for safe operations.

Chapter 4

Decentralized Control Scheme for Quadcopter Formations

4.1 Introduction

The idea of formation flight was inspired by nature like bees and birds. Such behavior exhibited by the living species is characterized by self-propelled individuals following some basic rules for their collective motion and parallel actions. Our control strategy is also somewhat inspired by this natural behavior. For MUAV swarm behavior, job of a control engineer is to implement self-defined rules through appropriate controller design for desired collective actions keeping in view the MUAV dynamics. There may be different constellation architectures for a formation depending upon the mission requirements. Performance requirements are high when flying in close proximity to each other. Simple mathematical models to mimic the natural swarm behavior are generally based on the rules such as,

1. All agents move in the same direction,
2. Agents remain reasonably close to their neighbors, and
3. Agents avoid collisions with their neighbors.

Some of the agents in a swarm may have superior sensing, computation or communication abilities, called as leaders [78]. However for such configurations, loss of the leader may result in collapse of the mission. Therefore for our study, we have chosen a virtual leader strategy that promises mission accomplishment. In virtual leader approach, the vehicles in the formation jointly synthesize a single, possibly fictitious, leader vehicle whose trajectory acts as a leader for the group [91]. Note that a single leader may not efficiently control the large formations or swarms; hence multiple leaders with networked

systems may be necessarily required, like we have exploited two virtual leaders for the proposed networked system.

The work in this chapter assumes the potential applications of photogrammetry and fire extinguishment through quadcopter fleets. Here consensus algorithms are not desired as we require the quadcopter fleets to track the trajectories of our interest, rather than decided by the agents themselves. In this chapter, every agent in a formation receives information from virtual leader that adds to redundancy. Four control inputs to each quadcopter are used, and the controller designing has been described in much details. The control scheme is robust and adaptive, and tracking performance validates the approach. Both formations track the trajectories with varying heights.

For the study in this chapter, reference model resides within the quadcopters to explicitly define the performance criteria for the quadcopters to make them behave like the model with desired dynamics. Here we have exploited a distributed control strategy where each quadcopter's controller is based on its own states and the commanded values for position. In this chapter, we have presented a simple framework to control a network of two quadcopter formations from ground stations exploiting the notion of virtual leader, so that loss of a leader does not affect the mission accomplishment. We have endeavored to tailor two control schemes namely LQR PI based on explicit model following and LQR PI servomechanism to control two quadcopter formations. Here we consider a full-state vector of quadcopters while tracking only the outputs of interest (performance output). Implementation of this simple, adaptive and robust scheme to control two quadcopter formations from ground stations is not seen in the literature, as per the knowledge of the author of this thesis. Performance of two control schemes has also been compared. LQR PI control scheme based on model following has also the potential for use even for some critical flight phases, like automatic flare control for smooth touch down, where the model dictates the desired trajectory.

Further description of this chapter is organized into four sections. Architecture of suggested scheme is proposed in section 4.2. The problem formulation is given in section 4.3. Control scheme adopted for this study is described in Section 4.4. Simulation results and analysis are reported in Section 4.5. Summary of the chapter is given in Section 4.7.

4.2 Architecture

An innovative framework has been developed for teamwork of two quadcopter formations; each having its specified formation geometry, assigned task and matching control scheme. The complete framework to control cluster of quadcopter formations for mission accomplishment is presented in this section. The objective, as an example, has been set to extinguish the fire over an area using two quadcopter formations with the defined roles of spraying the fluid and sending live data to the ground station respectively. Main constituent parts include:

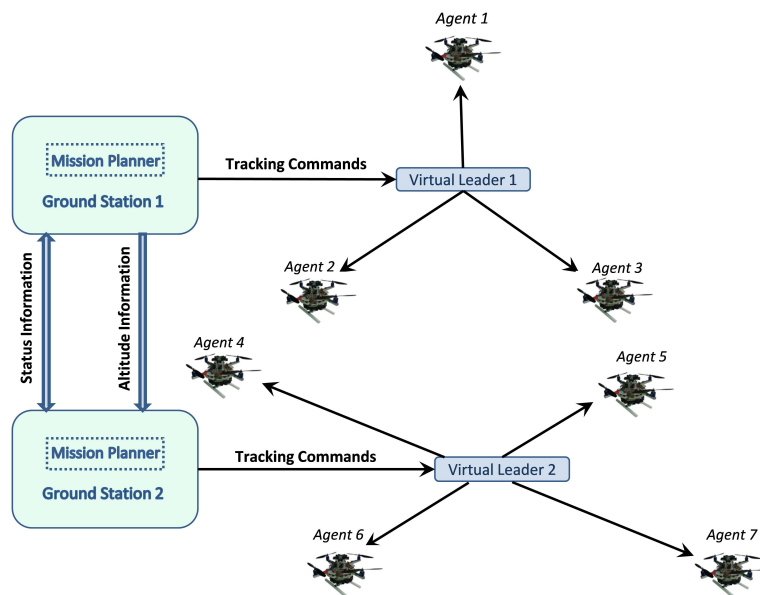


FIGURE 4.1: Architecture for two formations.
Figure adapted from Ref. [4].

1. Ground Stations,
2. Mission Planners,
3. Virtual Leaders (or virtual reference points), and
4. Quadcopters with allied equipment (fire extinguisher or camera).

The architecture with inter-connections is shown in Figure 4.1. We consider two formations comprising of three and four quadcopters with triangle and parallelogram geometric shapes respectively. Parallelogram shape is chosen to cover maximum area underneath. Though any arbitrary shape is possible with this scheme. The arrangement may be considered as cluster of formations. From now on, we will call triangle formation as formation 1 and parallelogram formation as formation 2. Agents in formation 1 are numbered as 1 – 3 and those in formation 2 are labeled as 4 – 7 as shown in Figures 4.1, and 4.3. Formation 1 is meant for aerial stereo-photography or videography that sends live data to the ground station 1. Though two quadcopters can meet the purpose, the third unit is added in the formation 1 for redundancy purpose. The four quadcopters in formation 2 are equipped with fire extinguishers to spray the fluid on area under fire. This formation is demonstrated, in section 4.5, to make sinusoidal motions in X-Y plane as well for spraying the fluid over maximum area. Each formation is homogeneous as the agents are structurally identical within a formation. A pictorial view for the described scenario is shown in Figure 4.2.

Each ground station encompasses a Mission Planner, whose role is to define the mission navigation legs in order to scan the area of interest. Tracking commands comprise



FIGURE 4.2: The mission scenario for fire extinguishment by quadcopter formations.

3D position. Reference signals are injected from ground station to the quadcopters in respective formation. These agents in the formation may be considered to follow a virtual leader. Agents are required to maintain customized separations from the respective virtual leader. Minimum distance between agents is to be considered to avoid collision and aerodynamic interactions. Two ground stations share the commanded altitude information to ensure safe height separation between the formations. These stations also act as standby to each other in case of malfunction of either unit, and therefore continuously exchange the status information. Thus it may be considered as one system of two sub-formations. The whole scheme has been simulated using the MATLAB/Simulink. Simulation results are presented in section 4.5.

4.3 Problem Formulation

For this work, we have assumed directed information flow to the agents. Communication topology of both formations is shown in Figure 4.3 where every node (or vertex) represents a quadcopter and every edge shows a communication link. Every agent receives command values from ground station to follow a Virtual Leader (VL). In addition, one of the agents is redundantly providing the data, received from the ground station, to the neighboring agent to handle the individual communication link failure. However at one instant in time, only the signal from either virtual leader or the neighboring unit is utilized in the control scheme.

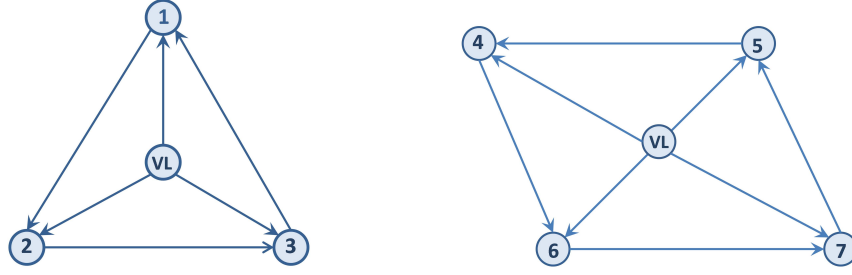


FIGURE 4.3: Communication topology for formation 1 and 2.
Figure adapted from Ref. [4].

Neighborhood of the agents is defined in terms of metric distance or topological distance. Neighborhood based on metric rules may be defined as:

Definition: A node j is a neighbor of node i if l_2 norm of displacement vector d between the two nodes is less than δ . Mathematically:

$$\|d\|_2 < \delta \quad (4.1)$$

where δ is the range of communication media. On the other hand, topological distance model relies on surrounding agents which are immediate neighbors irrespective of their distances. Interactions are based on topological rules rather than metric rules. For this study, we have used the notion of neighborhood based on metric rules.

4.3.1 Quadcopter Dynamic Model

A quadcopter has two pairs of rotors in cross-configuration that can rotate at different angular velocities to achieve rotational and translational motions. Detailed motion dynamics of quadcopter are already described in Section 3.2.1 of Chapter 3, and its motion dynamics are illustrated with the help of Figure 3.2. Quadcopter dynamics are modeled as per the following equations [110]:

$$\begin{aligned} u &= f_1 + f_2 + f_3 + f_4 \\ f_i &= k_i \omega_i^2, i = 1 \dots 4 \\ m\ddot{x} &= -u \sin \theta \\ m\ddot{y} &= u \cos \theta \sin \phi \\ m\ddot{z} &= u \cos \theta \cos \phi - mg \\ \ddot{\psi} &= \tau_\psi \\ \ddot{\theta} &= \tau_\theta \\ \ddot{\phi} &= \tau_\phi \end{aligned} \quad (4.2)$$

Here u shows the total thrust generated by four rotors, f_i is the force produced due to the rotation of rotor i , $k_i > 0$ is a constant, ω_i is the angular speed of motor i , m is mass of quadcopter and g is gravitational constant. τ_ψ , τ_θ and τ_ϕ represent the control inputs for yawing, pitching and rolling moments respectively. Above system 4.2 was linearized and corresponding Linear Time-Invariant (LTI) model of quadcopter given in [111] was used in this work. Details for linearization and trim points may be viewed in section 3.2.1, and are not included here for brevity. Quadcopter system under consideration has four inputs, twelve states and six outputs. Our state vector comprises $[x \dot{x} y \dot{y} z \dot{z} \psi \dot{\psi} \theta \dot{\theta} \phi \dot{\phi}]^T$. It represents the 3D position of center of mass of quadcopter in x, y and z-direction relative to the earth-fixed frame E , and Euler angles named as the yaw angle ψ around the z-axis, the pitch angle θ around the y-axis, and the roll angle ϕ around the x-axis respectively with their time derivatives denoted with dot ($\dot{\cdot}$) overhead. Output vector is $[x y z \psi \theta \phi]^T$, and performance output vector is $[x y z]^T$. Our control vector is:

$$\mathbb{U} = [u_1 \ u_2 \ u_3 \ u_4]^T = [u - mg \ \tau_\psi \ \tau_\theta \ \tau_\phi]^T$$

that corresponds to throttle input, yawing, pitching and rolling moments respectively. We assume that mass of quadcopters (including peripherals) in formation 1 is 2 kg and that of each unit in formation 2 is 3 kg.

4.3.2 Formation Dynamic Model

Desired dynamics of the reference model, defined as an LTI system, are as follows:

$$\begin{aligned} \dot{x}_r &= A_r x_r + B_r r(t) \\ y_r &= C_r x_r + D_r r(t) \end{aligned} \quad (4.3)$$

Suffix r represents the reference model and q will represent the follower quadcopter plant. For the system 4.3, $x_r \in \mathcal{R}^{n_r}$ are states of the reference model, $r(t)$ is the external bounded command value from ground station. For scheme implementation point of view, it may be considered as coming from the virtual leader. $y_r \in \mathcal{R}^{p_r}$ is the performance output vector of reference model. Reference model state space matrices (A_r, B_r, C_r, D_r) are of corresponding dimensions. For the reference model to exhibit stable dynamics, its state matrix A_r needs to be Hurwitz i.e. all its eigenvalues have strictly negative real part. Quadcopter plant dynamics are described as follows:

$$\begin{aligned} \dot{x}_q &= A_q x_q + B_q \mathbb{U} \\ y_q &= C_q x_q + D_q \mathbb{U} \end{aligned} \quad (4.4)$$

Here $x_q \in \mathcal{R}^{n_q}$ are states of the quadcopter, $\mathcal{U} \in \mathcal{R}^{m_q}$ is the control vector, $y_q \in \mathcal{R}^{p_q}$ is the performance output vector of quadcopter with $p_q \leq m_q$ [112]. Here p_q and m_q indicate the number of outputs y_q and the number of control inputs \mathcal{U} respectively. The matrices (A_q, B_q, C_q, D_q) are of corresponding dimensions. With this scheme, maximum number of trackable outputs may not exceed number of control inputs (four in case of quadcopter). It is assumed that the matrix pair (A_q, B_q) is stabilizable and state vectors x_r and x_q are available as these are to be used for formulation of control signal \mathcal{U} for quadcopter.

Before proceeding to the controller design, we first do the controllability check for the quadcopters using the controllability matrix $P = [B_q \ A_q B_q \ A_q^2 B_q \ \dots \ A_q^{n_q-1} B_q]$ and find it to be full ranked. A reference model resides within each agent in formation 2 to define the ideal output response to be followed by that agent. All the closed-loop poles of reference model may be assigned to desired locations through pole placement method. Though reference model dynamics are manipulated here through pole placement method, however same may be realized through some other control scheme as well, for e.g. LQR. Reference model in formation 2 receives a known bounded command $r(t)$ from respective ground station that may or may not vary with time. We are mainly interested to determine the control input for the quadcopter such that the desired 3D formation geometry, given by the relative distance vector $r = [r_x \ r_y \ r_z]^T$, is maintained. For this an LQR controller with PI feedback connection [112] is implemented for the quadcopter. Design objectives in LQR scheme are defined through Q and R matrices. Q matrix, a positive semi-definite matrix, shows the weightage (or importance) to states. R matrix, a positive definite matrix, indicates weightage of control efforts corresponding to control inputs. The controller can be tuned by changing the elements in the Q and R matrices to achieve a desirable response. An efficient LQR control scheme is based on finding the right weighting factors. Model following LQR PI control scheme given in [112] has been exploited for our present study, and tailored to control quadcopter formations from ground stations through virtual leaders. Open-loop dynamics of a quadcopter and reference model, from 4.4 and 4.3, may be formulated as:

$$\begin{bmatrix} \dot{x}_q \\ \dot{x}_r \end{bmatrix} = \begin{bmatrix} A_q & O_{n_q \times n_r} \\ O_{n_r \times n_q} & A_r \end{bmatrix} \begin{bmatrix} x_q \\ x_r \end{bmatrix} + \begin{bmatrix} B_q \\ O_{n_r \times m_q} \end{bmatrix} \mathcal{U} + \begin{bmatrix} O_{n_q \times m_q} \\ B_r \end{bmatrix} r(t) \quad (4.5)$$

Equation 4.5 may be written more concisely as:

$$\dot{x} = A x + B \mathcal{U} + B_R r(t) \quad (4.6)$$

Now tracking error δy , from 4.3 and 4.4, may be written as:

$$\delta y = y_q - y_r = [C_q \ -C_r] \begin{bmatrix} x_q \\ x_r \end{bmatrix} + D_q \mathcal{U} + (-D_r)r(t) \quad (4.7)$$

or

$$\delta y = C x + D \mathcal{U} + D_R r(t) \quad (4.8)$$

We aim to find such a \mathcal{U} that the tracking error asymptotically tends to zero in the presence of any known, bounded, and possibly time varying command $r(t)$. In order to track a step input command, an integral control is applied. The integrated tracking error may be represented as:

$$\dot{e}_y = \delta y \quad (4.9)$$

Equations 4.6, 4.8 and 4.9 may be combined as:

$$\begin{bmatrix} \dot{e}_y \\ \dot{x} \end{bmatrix} = \begin{bmatrix} O_{p_q \times p_q} & C \\ O_{(n_q+n_r) \times p_q} & A \end{bmatrix} \begin{bmatrix} e_y \\ x \end{bmatrix} + \begin{bmatrix} D \\ B \end{bmatrix} \mathcal{U} + \begin{bmatrix} D_R \\ B_R \end{bmatrix} r(t) \quad (4.10)$$

Concisely written as:

$$\dot{\tilde{x}} = \tilde{A} \tilde{x} + \tilde{B} \mathcal{U} + \tilde{B}_R r(t) \quad (4.11)$$

4.4 Control Strategies

It is assumed that external command $r(t)$ is a step-input command with zero errors. We now differentiate equation 4.11 w.r.t. time and introduce new variables $\varphi = [\dot{e}_y \ \dot{x}]^T$ and $v = \dot{\mathcal{U}}$, that yields:

$$\dot{\varphi} = \tilde{A} \varphi + \tilde{B} v \quad (4.12)$$

We here assume that agents complete state knowledge is available for implementation of LQR PI controller, which we use to control the open-loop dynamics of 4.12. We use the control input v to minimize the LQR cost. Cost function J may be defined as:

$$J = \int_0^{\infty} (\varphi^T Q \varphi + v^T R v) dt \quad (4.13)$$

Here Q and R are LQR weight matrices. Now we solve the Algebraic Riccati Equation:

$$\tilde{A}^T P + P \tilde{A} + Q - P \tilde{B} R^{-1} \tilde{B}^T P = 0 \quad (4.14)$$

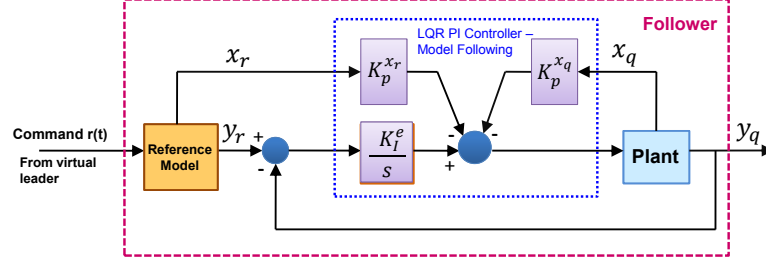


FIGURE 4.4: LQR PI control scheme with reference model.

and get the solution $P = P^T > 0$, that is used to compute the LQR gain matrix K given as:

$$K = R^{-1} \tilde{B}^T P \quad (4.15)$$

The gain matrix K is of the following form:

$$K = [K_I^e \ K_p^{x_q} \ K_p^{x_r}] \quad (4.16)$$

Here subscript indicates the type of gain (proportional or integral) and superscript shows the variable to whom this gain matrix is applied. Now

$$\dot{U} = - [K_I^e \ K_p^{x_q} \ K_p^{x_r}] \begin{bmatrix} \dot{e}_y \\ \dot{x}_q \\ \dot{x}_r \end{bmatrix} \quad (4.17)$$

Integrating equation 4.17 and ignoring constants of integration, LQR PI control solution for the quadcopter in formation 2 is given as:

$$U = K_I^e \frac{(y_r - y_q)}{s} - K_p^{x_q} x_q - K_p^{x_r} x_r \quad (4.18)$$

LQR PI control scheme guarantees perfect tracking [14] and [3]. Although our derivation is based on step reference command, the resulting control system gives good time response for any arbitrary reference command signal $r(t)$ [14], as will be demonstrated in simulation section. This command generator is capable of handling a wide range of motion trajectories, including position unit step commands, unit ramp commands, oscillatory commands and more [115]. Figure 4.4 shows the interconnection diagram for LQR PI control scheme based on model following architecture. Here the reference model acts as a command pre-filter and the quadcopter smoothly tracks the output of reference model. Dynamics of reference model and follower quadcopter are defined by 4.3 and 4.4 respectively.

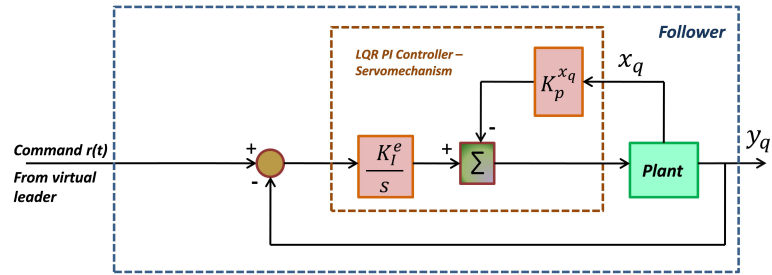


FIGURE 4.5: LQR PI servomechanism control scheme.
Figure adapted from Ref. [4].

For some scenarios, we may wish for not to smooth out the command $r(t)$ through a reference model, rather to follow the command directly. For this, a modified version of the above scheme may be exploited, known as *LQR PI servomechanism*. This may be achieved by setting A_r , B_r , C_r as null matrices and D_r as identity matrix. Then equation 4.3 yields $y_r = r(t)$. Block diagram showing interconnection for this control scheme is given in Figure 4.5. We need to consider that a system based on LQR PI servomechanism scheme is controllable as long as the following rank condition is true [112]:

$$\text{rank} \begin{bmatrix} A_q & B_q \\ C_q & D_q \end{bmatrix} = n_q + m_q$$

We tested the rank condition for our case and it came out to be 16, the full ranked. However rank condition also implies that maximum number of regulated outputs may not exceed the number of control inputs to the system (four in the case of quadcopter).

4.4.1 Comparison of Control Schemes

Two types of command tracking schemes are implemented; LQR PI control scheme based on reference model following to track the commands quite smoothly, and the LQR PI servo-mechanism for rapid command tracking. We have exploited both the schemes for two formations to meet the respective mission requirements. Formation 1 is managed through LQR PI servomechanism control scheme, which demonstrates quick response at the cost of high control effort. This forces the agents to follow the desired trajectory vigorously as per the requirements of the mission e.g. photogrammetry. However for Formation 2 we are rather interested in smooth movements so we implement LQR PI control scheme based on reference model following. Control efforts for both the control schemes are simulated in MATLAB environment for comparison, and the plot is shown in Figure 4.6. Tracking performance for two control schemes is demonstrated in Figure 4.7. For the sake of comparison and clarity in plots, only two agents, one from each formation, are considered here which receive same tracking commands.

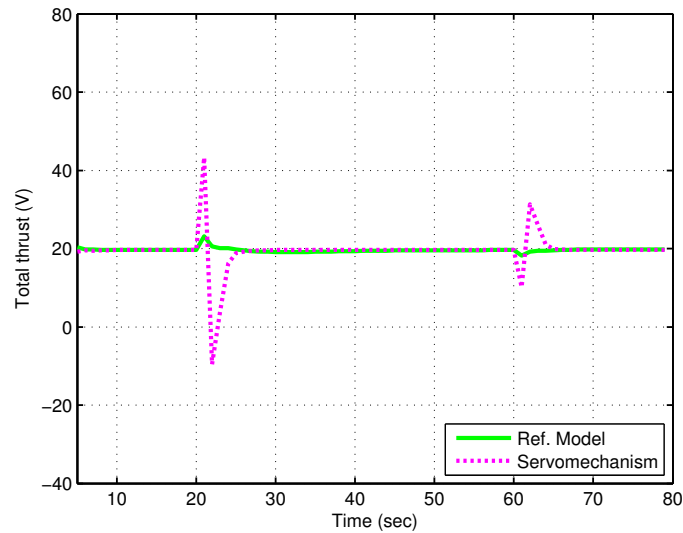


FIGURE 4.6: Comparison of control effort for two control schemes.

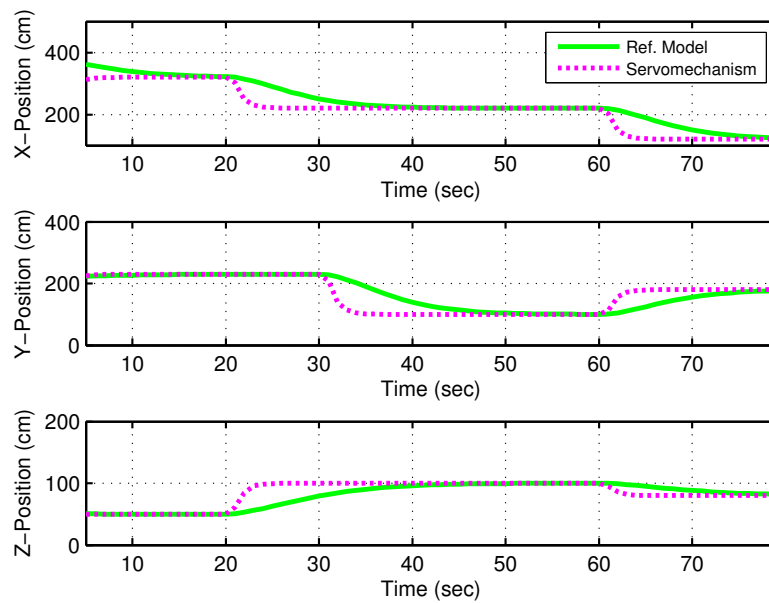


FIGURE 4.7: Comparison of tracking performance for two control schemes.

4.5 Simulation Results and Analysis

4.5.1 Trajectory Tracking Performance

The two types of control schemes defined above have been implemented in MATLAB/Simulink. The whole mission of fire extinguishment is divided into two phases. In first phase the agents join to form the required formation geometry (e.g. triangle or parallelogram); while in second phase the agents follow the trajectory commands from the ground station through Mission Planner.

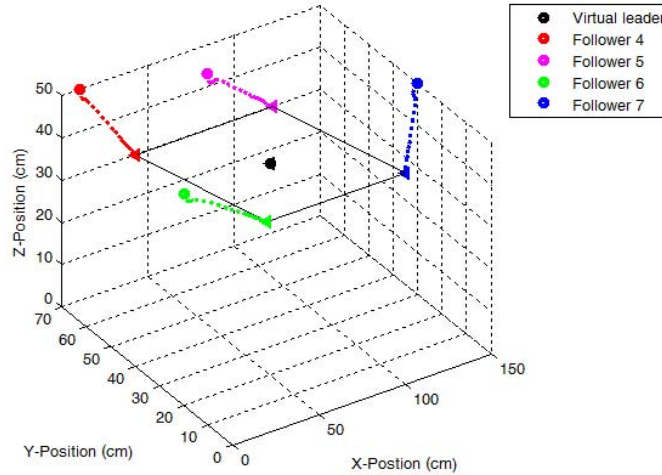


FIGURE 4.8: Quadcopters joining to make desired formation geometry.
Figure partly adapted from Ref. [4].

Commanded relative separation vectors for agents, defined as $r = [r_x \ r_y \ r_z]$ in formation 1 are $[-10, 0, 0]$, $[20, -20, 0]$ and $[20, 20, 0]$; while those in formation 2 are $[-50, 20, 0]$, $[30, 20, 0]$, $[-30, -20, 0]$ and $[50, -20, 0]$ w.r.t. their respective virtual leaders. In the first phase, agents depart from their initial arbitrary points in order to form the desired formation geometry. Simulation results to this effect for formation 2 are shown in Figure 4.8.

Tracking reference values are introduced from network operators to the quadcopters. Simulation results for the two formations in 3D view are shown in Figure 4.9. Initial and final positions of agents and corresponding virtual leaders (or references) are also shown to indicate the direction of motion. From the plots, it is evident that output of reference models is tracked quite smoothly by the agents in formation 2 while maintaining the desired mutual separations. The agents in formation 1 vigorously follow the tracking commands.

Tracking performance is now demonstrated for sinusoidal command in order to cover maximum area underneath the agents carrying the fire extinguishers. Results to this effect are shown in Figure 4.10. However for such commands, desired tracking (of same frequency) is achieved if rate of change of command signal is within the closed-loop system bandwidth [112]. Though we get the sinusoidal output but with a different amplitude and phase determined by the magnitude of the system transfer function. Desired amplitude may be achieved using a suitable pre-compensator. Thus the control scheme is capable of variable set-point tracking as well. LQR PI control scheme based on model following is quite robust against different types of possible perturbations as shown in chapter 3. For the sake of brevity and clarity of plots, disturbances are not studied in this chapter.

We now make the quadcopters to hover at desired location for specified time duration. Figure 4.11 shows the formation 2 in hovering mode on a time scale, where the

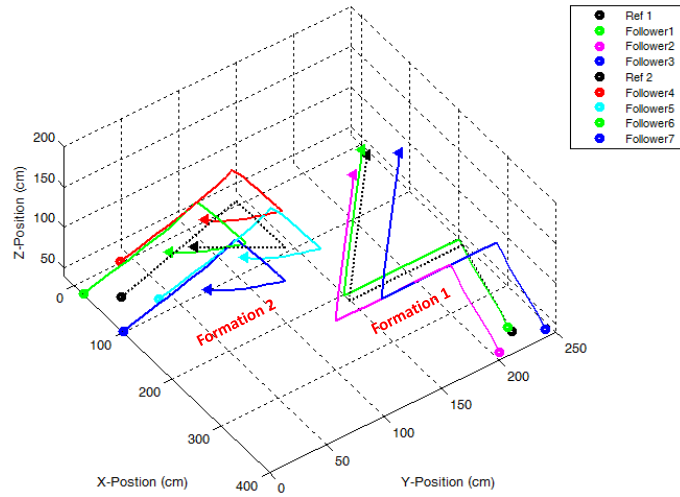


FIGURE 4.9: Trajectory tracking by two formations.
Figure partly adapted from Ref. [4].

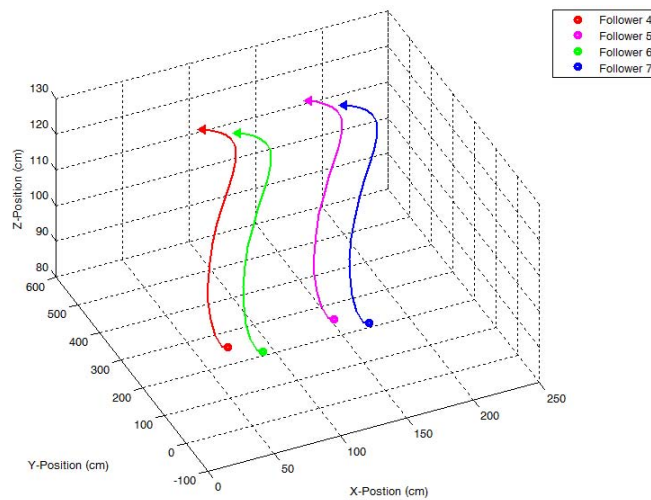


FIGURE 4.10: Sinusoidal motion of formation 2.
Figure partly adapted from Ref. [4].

agents maintain their horizontal position. Note that hovering may be considered as an equilibrium point.

For further data comparison, two formations are commanded to track the same trajectory while maintaining their respective formation geometries and adhering to their respective control schemes. Simulation result for y-position of both formation agents is plotted in Figure 4.12 that emphasizes the scalable property of the formation control system.

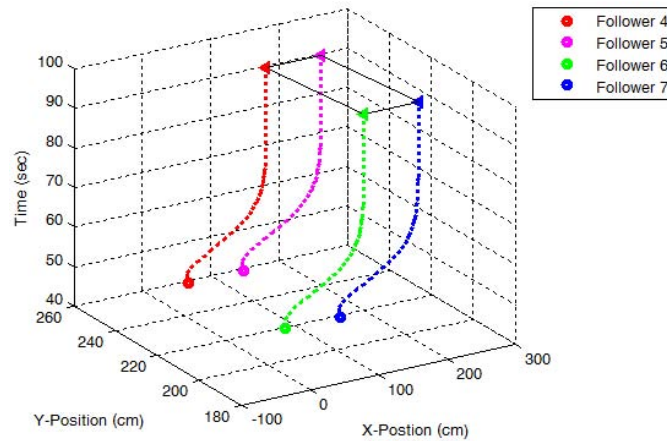


FIGURE 4.11: Formation 2 in hovering mode.
Figure partly adapted from Ref. [4].

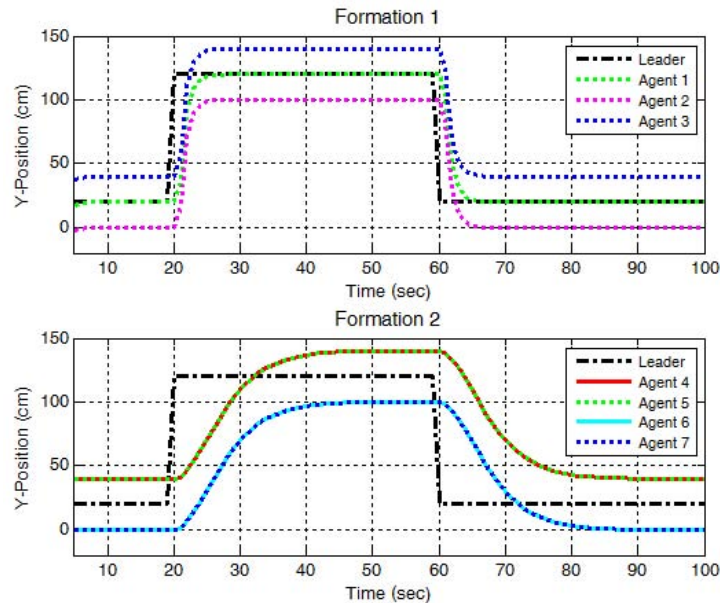


FIGURE 4.12: Same trajectory tracking by two formations.
Figure adapted from Ref. [4].

4.6 Cluster Reconfiguration of Agents

We now describe the framework to demonstrate the cluster reconfiguration of quadcopters in formation 2. Initial formation geometry of a parallelogram is switched to a straight line to meet the changing requirements of the mission. For this reconfiguration, following four steps are followed:

1. First of all, units go to *hovering mode*
2. All units flying at same altitude change their heights by a safe separation

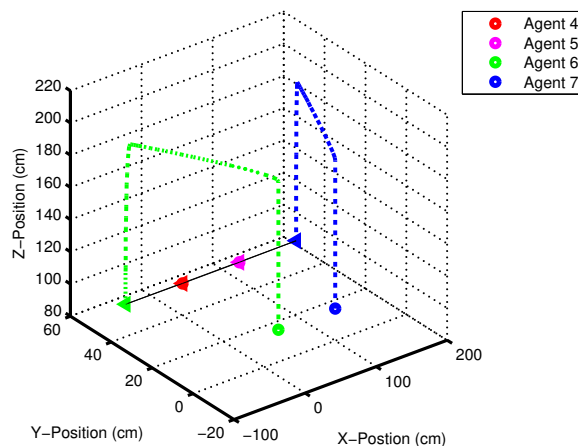


FIGURE 4.13: Cluster reconfiguration of agents in 3D view.

3. Units fly to their desired positions in horizontal (x-y) plane, and
4. Units regain their initial altitudes (maintaining their new x and y positions)

This is the simplest approach that also ensures collision avoidance during cluster reconfiguration. However this technique is realizable when we have ample of space available. Minimum and maximum height separation limitations may be imposed in order to avoid collision and to remain in communication zone of other vehicles respectively. Simulation to this effect in 3D view is shown in Figure 4.13, where the agents 6 and 7 move to their new desired positions. In order to visualize this scenario in time domain, Figure 4.14 is provided.

4.7 Summary

Position controllers for two formations have been implemented using the control schemes LQR PI based on model following and LQR PI servomechanism. These control schemes have been compared in terms of convergence to desired tracking values and the control effort. LQR PI model following controller behaves well in terms of transient response and exhibits no overshoot, while LQR PI servomechanism reacts faster to the commands but at the cost of high control effort. With these two control schemes, quadcopters in the formations are able to track the desired trajectory while maintaining defined relative 3D separations. Extensive simulations proved efficacy of the proposed schemes while giving promising results. Proposed architecture is scalable and can be expanded easily. Notion of virtual leader enables the scheme to be robust against any node failure. Load of information (command values) from ground station to the agents is quite low, as only position information is transmitted from ground station to the quadcopters. Working range of communication media needs to be considered that determines the maximum distance between ground station and respective quadcopter formation.

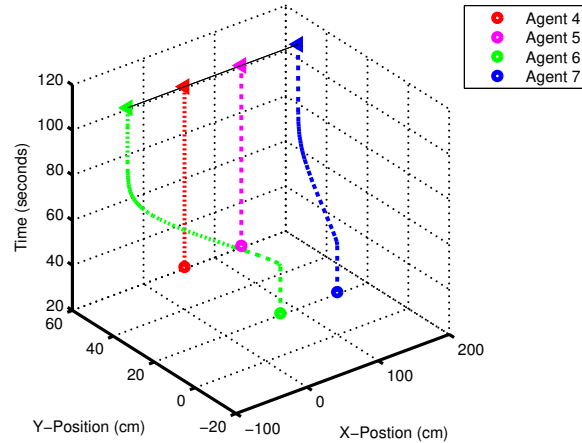


FIGURE 4.14: Cluster reconfiguration of agents on time scale.

The two presented control approaches for formation flights are simple from implementation point of view. However it necessitates knowing the exact dynamic model and the states of MUAVs. As it is laborious to model all the dynamics of rotorcraft flying in close formation; adaptive and robust control techniques, like LQR PI, may be explored to their full potential. Advanced designs may be realized while introducing the network dynamics that is a new area of research.

Cluster reconfiguration of agents to change the formation geometry to other desired shape during flight has also been undertaken in an efficient and simple way using the proposed scheme. A suitable collision avoidance mechanism may be introduced for safe operations. For future work, presented scheme may be implemented in real-time using the formation flying test setup developed at Institute of Aerospace Information Technology, University of Würzburg.

Chapter 5

Graph Theory for Distributed Control

5.1 Introduction

Our analysis framework in this Chapter is based on tools from matrix theory and graph theory. The graphs meant for inter-vehicle communications have been given due consideration. A physical interpretation of presented results may be the information flow within a formation or swarm of aerial vehicles that plays an important role towards stability. For large formations or swarms, a single leader may not efficiently control the whole formation and hence multiple leaders may be required. Handling of communication topology for large formations / swarms of aerial vehicles is facilitated with the use of graph theory.

Communication links among the units may be directional or bidirectional, depending upon the course of information flow. Directed and undirected graphs correspond to directional and bidirectional communication links respectively. Addition and removal of communication links among the units affects the Laplacian matrix, which will also be considered in this chapter. Subgraphs are also the focus of our study, where these have been related to multiple formations or clusters of units represented by a single Laplacian matrix. Some results for the properties of Laplacian matrix and associated eigenvalues are also discussed.

One of the contributions of this chapter is the generalization of Euler's formula that holds true for the planar graphs as well as subgraphs. An endeavor has been made to relate the Euler's formula to the eigenvalues of underlying Laplacian matrix. Here we focus on directed and undirected graphs and related Laplacian matrices. However, normalized Laplacian matrix is not the focus of our present study.

Further description of this chapter is organized as follows. Section 5.2 describes preliminary background on graph theory. Section 5.3 provides definitions and propositions

in the domain of graph theory and generalization of Euler's formula. Summary of the chapter is given in section 5.6.

5.2 Notations and Preliminary Background

A graph G with a set of vertices V and edges E is generally represented as $G = (V, E)$. The number of vertices (or nodes) and edges will be represented as $|V|$ and $|E|$ respectively. Well-Connectedness of a graph may be defined as the number of edges in a graph. Fully connected graph, also called a complete graph, is one where all possible connections between the units exist. Degree of a graph means maximum number of edges for any of its vertices [82]. Graphs are broadly classified as directed graphs and undirected graphs.

If all vertices of a graph are connected then there is only one eigenvalue zero (simple eigenvalue) of related Laplacian matrix. However, when a graph is partitioned into multiple subgraphs then algebraic multiplicity of zero eigenvalue represents the number of subgraphs. Individual connected components of a graph are also called as subgraphs. For a node to be a part of a graph or subgraph, it must have at least one edge connected to it. In physical terms, a unit must communicate with at least one other unit to be part of a graph or subgraph.

A **loop** may be defined as a closed path that does not enclose another closed path. For communication among a large number of units (e.g. a swarm), a loop indicates redundant information flow from / to multiple units. It also gives an indication for connectedness of a graph. Note that a loop here does not mean a self-loop that corresponds to connection of a vertex with itself. A loop is also referred to as a *cycle* in graph theory. *Tree* in graph theory means no loops.

Now we define some matrices in the domain of graph theory. **Degree matrix D** is a $n \times n$ diagonal matrix that gives the information about the number of edges (degree) of each vertex. Degree of a vertex is given by the number of its neighbors. If elements of a degree matrix are represented as $D[i, j]$ where i and j represent the vertices of a matrix then:

$$D[i, j] := \begin{cases} \text{deg}(v_i) & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

Here $\text{deg}(v_i)$ denotes degree of vertex i . An example of a degree matrix is given below that corresponds to an arbitrary graph of Figure 5.1.

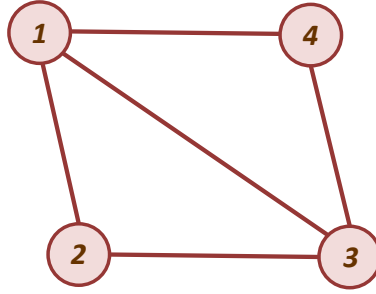


FIGURE 5.1: An undirected graph with four vertices.

$$D = \begin{bmatrix} 3 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 2 \end{bmatrix}$$

Adjacency matrix A is a square $n \times n$ matrix that gives information about adjacency of vertices in an undirected graph. Adjacency matrix is defined as:

$$A[i,j] := \begin{cases} 1 & \text{if an edge exists between vertex } i \text{ and } j \\ 0 & \text{if no edge exists between vertex } i \text{ and } j \end{cases}$$

An example of adjacency matrix, corresponding to the graph in Figure 5.1, is as under:

$$A = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$$

Laplacian matrix \mathcal{L} is a square $n \times n$ matrix that may be defined, using the notion of degree matrix D and adjacency matrix A , as following:

$$\mathcal{L} = D - A$$

Laplacian matrix may also be directly defined as follows:

$$\mathcal{L}[i,j] := \begin{cases} \text{deg}(v_i) & \text{if vertex } i = j \\ -1 & \text{if } i \neq j \text{ and } i \text{ and } j \text{ are adjacent} \\ 0 & \text{otherwise} \end{cases}$$

$\text{deg}(v_i)$ is defined as before. Laplacian matrix for the graph in Figure 5.1 is given below:

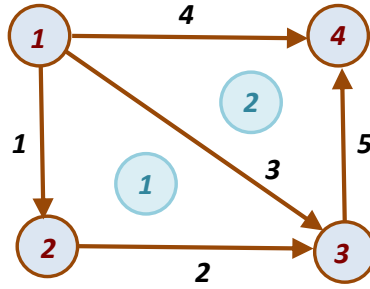


FIGURE 5.2: A directed graph with four vertices.

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

Laplacian matrix \mathcal{L} being a symmetric and positive semi-definite matrix has special properties. Its lowest eigenvalue is always zero and all other eigenvalues are positive and real. \mathcal{L} has a right eigenvector of $\mathbf{1}$ associated with the zero eigenvalue because of the identity $\mathcal{L}\mathbf{1} = 0$, where $\mathbf{1}$ is a vector of all ones. This implies that the row sum of all elements in a \mathcal{L} matrix is zero. Laplacian matrix is only diagonalizable for undirected topologies. Laplacian matrix (for undirected graph) is always singular, as zero is always its eigenvalue. Laplacian matrix may indicate the type of control scheme being used; e.g. Laplacian matrix is an Identity matrix for decentralized control [15].

Incidence matrix Inc is designed for directed graphs which is useful for scenarios like current flow in electric circuits, and information flow among different units in a MAS etc. The number of vertices and edges of a graph correspond to number of columns and rows of an incidence matrix respectively. An arbitrary directed graph is shown in Figure 5.2. Corresponding Incidence matrix is as under:

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

Note that edges are also required to be numbered (in addition to vertices) for writing an incidence matrix. Also note that loop corresponds to linearly dependent rows. For example, row 3 is a sum of rows 1 and 2; similarly row 4 is a sum of rows 3 and 5 of incidence matrix. Two loops are shown in aqua color in Figure 5.2. Unlike adjacency matrix, degree matrix and Laplacian matrix; an incidence matrix is not symmetric. Incidence matrix and Graph Laplacian matrix are interlinked as $\mathcal{L} = Inc * Inc^T$ [93], where the superscript T shows matrix transpose.

Some of the agents in a swarm may have superior sensing, computation or communication abilities, called as leaders [78]. Information flow is usually from leader to the followers. For Figure 5.2, we may consider vertex 1 as a leader (that only provides information to other units) and the remaining vertices as the followers. Every vehicle in a formation only needs to know the states of its neighboring units, not all in the formation. Neighborhood of the units is defined in terms of metric distance or topological distance, as described in section 4.3 of Chapter 4.

5.2.1 Algebraic Connectivity and Its Utility

Spectrum of a Laplacian matrix \mathcal{L} is ordered as $0 = \lambda_1 \leq \lambda_2 \leq \lambda_3 \leq \dots \leq \lambda_n(G)$. Here, the zero eigenvalue is known as the trivial eigenvalue of \mathcal{L} . The second smallest eigenvalue of Laplacian matrix is called the *algebraic connectivity* of graph G , and the corresponding normalized eigenvector is called the *Fiedler Vector* [93]. The second smallest eigenvalue $\lambda_2 > 0$ is a necessary and sufficient condition for connectivity of graph G . If it is also zero then it implies that there exist two subgraphs. Algebraic multiplicity of zero eigenvalue represents the number of subgraphs.

Algebraic connectivity is considered to be a measure of how well connected a graph is [93]. This value monotonically increases when we increase the number of edges in the graph [116]. It suggests that algebraic connectivity either maintains its value or it increases but never decreases with the increase in number of edges of the graph. Note that algebraic connectivity is not an absolute measure rather a relative measure of graph connectivity. It implies that the second smallest eigenvalue may not always give an absolute measure of connectedness of a graph, as two graphs having different number of edges may have the same algebraic connectivity. Please see rows 4 – 7 of Table 5.1. Here note that $\prod^* \lambda$ represents product of non-zero eigenvalues in all the tables in this chapter. Reference [93] states that if the second eigenvalue has an algebraic multiplicity r , then we need to add r number of edges before the second smallest eigenvalue increases from its current value. In other words, the second smallest eigenvalue along with its algebraic multiplicity gives complete information about the connectedness of a graph. The number of vertices in a graph corresponds to an upper limit on algebraic connectivity. The maximum attainable value of algebraic connectivity may not exceed the number of vertices in the graph [116].

5.3 Graph Properties and Propositions

Tables 5.1 and 5.2 show some arbitrary graphs and corresponding Laplacian matrices and eigenvalues as a ready reference to provide an insight into forthcoming discussion in this section. Table 5.1 also indicates that it does not matter how we number the vertices, eigenvalues of resultant Laplacian matrix remain unchanged. For an Incidence matrix Inc , the following relations hold [117]:

$$\dim N(Inc^T) = |E| - |V| + 1 \quad (5.1)$$

or,

$$\dim N(Inc^T) = |L| \quad (5.2)$$

Here $\dim N$ represents dimensions of null space, and $|L|$ the number of loops. Equations 5.1 and 5.2 are only valid for directed graphs.

Proposition 1: Let m_1^L represent the algebraic multiplicity of zero eigenvalues of associated Laplacian matrix, then following relation holds for a directed graph:

$$|L| = m_1^L \quad (5.3)$$

Though a relation between product of nonzero eigenvalues of Laplacian matrix and number of spanning trees of a connected graph already exists in the form of *matrix tree theorem* [118], we exploit this measure as follows:

Definition 2: Product (or trace) of all nonzero eigenvalues of a Laplacian matrix \mathcal{L} is a measure for nodes connectivity of a given graph (or subgraph), as it always increases with the increase in number of edges. The product (or trace) may also be considered as a measure of feasibility of information exchange.

Definition 3: For a Laplacian matrix of an undirected graph, sum of its eigenvalues is equal to number of communication paths. Note that if agent 1 sends information to agent 2 and vice versa, these are taken as two communication paths. Mathematically it may be written as:

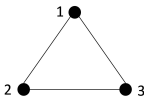
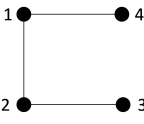
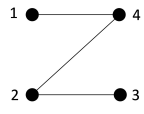
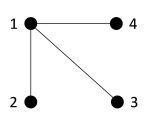
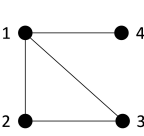
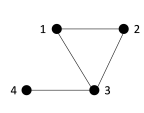
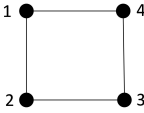
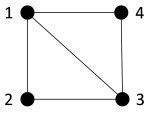
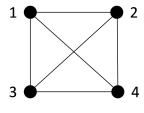
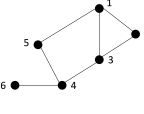
$$\frac{\sum \lambda}{2} = |E| \quad (5.4)$$

$\sum \lambda$ denotes the sum of eigenvalues of Laplacian matrix. Equation 5.4 is applicable to graphs as well as subgraphs. As trace of a matrix (whether diagonalized or otherwise) is equal to sum of its eigenvalues, the trace of a Laplacian matrix shows total number of connections. Same is also evident from Table 5.1 – 5.3.

Definition 4: If a graph is fully connected (all possible connections in the graph exist), then following holds true:

- All elements in the Laplacian matrix are nonzero.
- All nonzero eigenvalues of Laplacian matrix are same.

TABLE 5.1: Sample Graphs and Corresponding Metadata.
Table adapted with small modifications from Ref. [5]. © (2016) IEEE.

Topology	Laplacian Matrix	Spectrum
	$\begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix}$	$\lambda = 0, 3, 3$ $\Sigma\lambda = 6$ $\prod^* \lambda = 9$
	$\begin{bmatrix} 2 & -1 & 0 & -1 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 1 & 0 \\ -1 & 0 & 0 & 1 \end{bmatrix}$	$\lambda = 0, 0.59,$ $2, 3.41$ $\Sigma\lambda = 6$ $\prod^* \lambda = 4$
	$\begin{bmatrix} 1 & 0 & 0 & -1 \\ 0 & 2 & -1 & -1 \\ 0 & -1 & 1 & 0 \\ -1 & -1 & 0 & 2 \end{bmatrix}$	$\lambda = 0, 0.59,$ $2, 3.41$ $\Sigma\lambda = 6$ $\prod^* \lambda = 4$
	$\begin{bmatrix} 3 & -1 & -1 & -1 \\ -1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ -1 & 0 & 0 & 1 \end{bmatrix}$	$\lambda = 0, 1, 1, 4$ $\Sigma\lambda = 6$ $\prod^* \lambda = 4$
	$\begin{bmatrix} 3 & -1 & -1 & -1 \\ -1 & 2 & -1 & 0 \\ -1 & -1 & 2 & 0 \\ -1 & 0 & 0 & 1 \end{bmatrix}$	$\lambda = 0, 1, 3, 4$ $\Sigma\lambda = 8$ $\prod^* \lambda = 12$
	$\begin{bmatrix} 2 & -1 & -1 & 0 \\ -1 & 2 & -1 & 0 \\ -1 & -1 & 3 & -1 \\ 0 & 0 & -1 & 1 \end{bmatrix}$	$\lambda = 0, 1, 3, 4$ $\Sigma\lambda = 8$ $\prod^* \lambda = 12$
	$\begin{bmatrix} 2 & -1 & 0 & -1 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ -1 & 0 & -1 & 2 \end{bmatrix}$	$\lambda = 0, 2, 2, 4$ $\Sigma\lambda = 8$ $\prod^* \lambda = 16$
	$\begin{bmatrix} 3 & -1 & -1 & -1 \\ -1 & 2 & -1 & 0 \\ -1 & -1 & 3 & -1 \\ -1 & 0 & -1 & 2 \end{bmatrix}$	$\lambda = 0, 2, 4, 4$ $\Sigma\lambda = 10$ $\prod^* \lambda = 32$
	$\begin{bmatrix} 3 & -1 & -1 & -1 \\ -1 & 3 & -1 & -1 \\ -1 & -1 & 3 & -1 \\ -1 & -1 & -1 & 3 \end{bmatrix}$	$\lambda = 0, 4, 4, 4$ $\Sigma\lambda = 12$ $\prod^* \lambda = 64$
	$\begin{bmatrix} 3 & -1 & -1 & 0 & -1 & 0 \\ -1 & 2 & -1 & 0 & 0 & 0 \\ -1 & -1 & 3 & -1 & 0 & 0 \\ 0 & 0 & -1 & 3 & -1 & -1 \\ -1 & 0 & 0 & -1 & 2 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 \end{bmatrix}$	$\lambda = 0, 0.72, 1.68,$ $3, 3.70, 4.89$ $\Sigma\lambda = 14$ $\prod^* \lambda = 66$

- Algebraic multiplicity of nonzero eigenvalues indicates the number of edges that each vertex has with other vertices.
- Numeric value of nonzero eigenvalues shows the total number of vertices in the graph.

Corollary 1: Half of the number of zeros in Laplacian matrix inform about the number of edges that need to be added to make the graph fully connected.

Definition 4 and Corollary 1 are useful for small graphs. For large formations, fully connected graphs may not be desirable. Table 5.3 shows two subgraphs and their merging into a single graph in two different ways. Corresponding changes in Laplacian matrices and eigenvalues are also depicted in the Table.

TABLE 5.2: Sample Subgraphs and Corresponding Metadata.
Table adapted from Ref. [5]. © (2016) IEEE.

Topology	Laplacian Matrix	Spectrum
	$\begin{bmatrix} 1 & -1 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & -1 & 1 \end{bmatrix}$	$\lambda = 0, 0, 2, 2$ $\Sigma\lambda = 4$ $\prod^* \lambda = 4$
	$\begin{bmatrix} 3 & -1 & -1 & -1 & 0 & 0 & 0 \\ -1 & 2 & 0 & -1 & 0 & 0 & 0 \\ -1 & 0 & 2 & -1 & 0 & 0 & 0 \\ -1 & -1 & -1 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & -1 & -1 \\ 0 & 0 & 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & 0 & -1 & -1 & 2 \end{bmatrix}$	$\lambda = 0, 0, 2$ $3, 3, 4, 4$ $\Sigma\lambda = 16$ $\prod^* \lambda = 288$
	$\begin{bmatrix} 2 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & -1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 2 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & -1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 2 \end{bmatrix}$	$\lambda = 0, 0, 0, 3$ $3, 3, 3, 3, 3$ $\Sigma\lambda = 18$ $\prod^* \lambda = 729$

5.4 Generalization of Euler's Formula

5.4.1 Generalization of Euler's Formula for Planar Graphs

The relation between Euler's formula and eigenvalues of underlying Laplacian matrix has been determined. Euler's formula for a planar graph is as follows [117]:

$$|V| - |E| + |L| = 1 \quad (5.5)$$

This formula is applicable to the connected graphs (directed and undirected). For undirected graphs, algebraic multiplicity of zero eigenvalue of corresponding Laplacian matrix is one.

Proposition 2: For graphs as well as subgraphs (when represented as a single Laplacian matrix), Euler's formula may be generalized as following:

$$|V| - |E| + |L| = 1 * m_1^L \quad (5.6)$$

where m_1^L is already defined in Proposition 1. Equation 5.6 is a generalization of Euler's formula, which is applicable not only for a connected graph but also multiple subgraphs; whereas relation 5.5 is only applicable for one connected graph. Remember that for relation 5.6 to hold good, only one Laplacian matrix is to be exploited for representation of multiple subgraphs. This is useful when, for example, multiple formations of aerial vehicles are to be controlled through a single ground station or some other platform through distributed / cooperative control schemes. Equation 5.6 may be readily verified in the light of data given in all the tables. Table 5.3 is also of interest, where two subgraphs merge to one graph and algebraic multiplicity and other parameters also change accordingly.

As we know that dimensions of null space correspond to the algebraic multiplicity of eigenvalue zero, so from equation 5.6 it follows that:

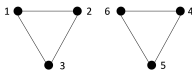
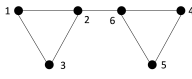
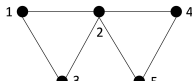
$$|V| - |E| + |L| = \dim N(L) \quad (5.7)$$

5.4.2 Euler's Modified Formula

Keeping in view previous discussion, Euler's formula may be written in modified form as:

$$|V| - \frac{\sum \lambda}{2} + |L| = 1 \quad (5.8)$$

TABLE 5.3: Merging of Subgraphs.
Table adapted from Ref. [5]. © (2016) IEEE.

Topology	Laplacian Matrix	Spectrum
	$\begin{bmatrix} 2 & -1 & -1 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 & 0 \\ -1 & -1 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & -1 & -1 \\ 0 & 0 & 0 & -1 & 2 & -1 \\ 0 & 0 & 0 & -1 & -1 & 2 \end{bmatrix}$	$\lambda = 0, 0, 3,$ $3, 3, 3$ $\Sigma\lambda = 12$ $\prod^* \lambda = 81$
	$\begin{bmatrix} 2 & -1 & -1 & 0 & 0 & 0 \\ -1 & 3 & -1 & 0 & 0 & -1 \\ -1 & -1 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & -1 & -1 \\ 0 & 0 & 0 & -1 & 2 & -1 \\ 0 & -1 & 0 & -1 & -1 & 3 \end{bmatrix}$	$\lambda = 0, 0.44, 3,$ $3, 3, 4.56$ $\Sigma\lambda = 14$ $\prod^* \lambda = 54$
	$\begin{bmatrix} 2 & -1 & -1 & 0 & 0 \\ -1 & 4 & -1 & -1 & -1 \\ -1 & -1 & 2 & 0 & 0 \\ 0 & -1 & 0 & 2 & -1 \\ 0 & -1 & 0 & -1 & 2 \end{bmatrix}$	$\lambda = 0, 1, 3,$ $3, 5$ $\Sigma\lambda = 12$ $\prod^* \lambda = 45$

or,

$$\Sigma\lambda = 2 * (|V| + |L| - 1) \quad (5.9)$$

Equation 5.9 is applicable for a connected graph. It may be generalized as:

$$\Sigma\lambda = 2 * (|V| + |L| - m_1^L) \quad (5.10)$$

Equation 5.10 is applicable for multiple subgraphs as well as for a single connected graph. As sum of eigenvalues is the same as trace of a square matrix, so equation 5.9 may be formulated as:

$$Trace = 2 * (|V| + |L| - 1) \quad (5.11)$$

5.4.3 Generalization of Euler's Formula for 3D Graphs

For three dimensional connected graphs, Euler's formula is as follows:

$$|V| - |E| + |F| = 2 \quad (5.12)$$

Here $|F|$ represents the number of faces. Now generalizing it for multiple subgraphs:

$$|V| - |E| + |F| = 2 * m_1^L \quad (5.13)$$

This formula is applicable for multiple geometrical shapes when they are represented as a single Laplacian matrix. Now equation 5.12 may be reformulated using equation 5.4 as follows:

$$\Sigma\lambda = 2 * (|V| + |F| - 2) \quad (5.14)$$

that is applicable for single graph only. It may be generalized to make it applicable for multiple connected parts as well as for one single graph, as follows:

$$\Sigma\lambda = 2 * (|V| + |F| - 2 * m_1^L) \quad (5.15)$$

5.5 Graph Theory in Distributed/Cooperative Control

This section aims at describing the use of one Laplacian matrix for communication topology of multiple clusters of agents. Fig. 5.3 shows the block diagram of the scenario where two clusters of quadcopters are exploiting consensus algorithm for cooperative control. Input bias vector is introduced from the ground station (or some operator) to realize customized separations among the agents. Here the communication topology of both the clusters is represented by a single Laplacian matrix as under:

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

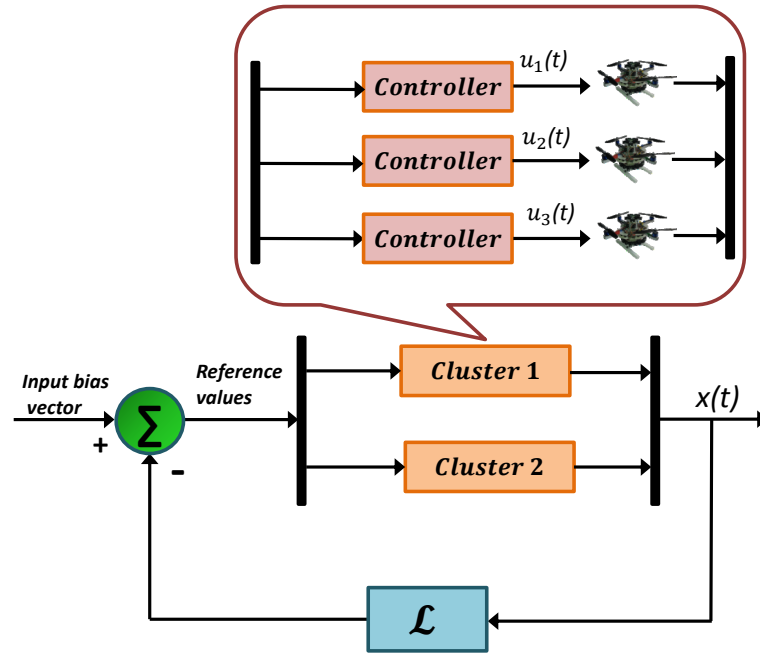


FIGURE 5.3: Block diagram for consensus algorithm by two clusters of quadcopters. Figure adapted from Ref. [5]. © (2016) IEEE.

Laplacian matrix reveals that cluster 1 exhibits undirected topology while cluster 2 has directed topology. Communication topology of each cluster is shown in Fig. 5.4. Each cluster is sharing the information within its own agents only. For the sake of brevity, we here assume that consensus is sought only for one observable element of states. In order to make the quadcopters hover at same altitude in respective clusters, estimated altitude information is utilized for consensus. Input bias for altitude will be zero for quadcopters in a cluster to hover at same height. Simulation to this effect is shown in Fig. 5.5. Consensus value for cluster 1 converges to the average value of initial height of quadcopters. Communication topology of cluster 2 is directed and consensus value is not the average value of initial heights. For further details on consensus algorithms, please see Chapter 6. Readers interested in details for input bias are referred to [82].



FIGURE 5.4: Communication topology for cluster 1 and 2. Figure adapted from Ref. [5]. © (2016) IEEE.

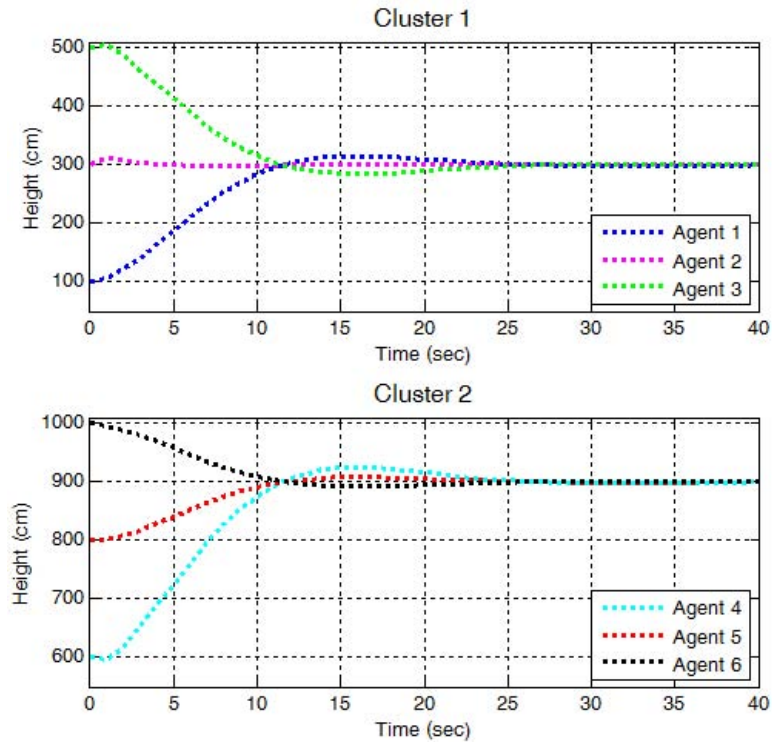


FIGURE 5.5: Consensus for hovering heights by two clusters.
Figure adapted from Ref. [5]. © (2016) IEEE.

5.6 Summary

Euler's formula has been generalized and modified to make it applicable to multiple subgraphs while exploiting the eigenvalues and trace of underlying Laplacian matrix. A few propositions and definitions are suggested in this Chapter. Proposed equations provide insights into the graph properties and connectivity of multiple subgraphs. It may be realized that proposed relations are based on empirical evidences, hence their proofs are not provided. However sufficient examples in the tables have been provided to follow and validate the presented notions. Same may also be verified by considering any other arbitrary graph / subgraph of own choice. Applications of presented results may be found in swarm of aerial vehicles, formation flight, information flow in a MAS and Kirchhoff's current law etc. These are envisaged to be helpful for analysis of the network of networks. Network dynamics and control for large number of agents may be conveniently investigated using the notion of graph theory.

Chapter 6

Consensus based Cooperative Control Schemes for Quadcopters Fleet

6.1 Introduction

Most of the swarms exhibited by the living species do not involve an explicit leader, like in case of ants the queen does not give the orders to other ants for what to do. Such frameworks relate to decentralized algorithms, and a specific formation shape or geometry is generally not maintained. Swarm behavior exhibited by the living species does not involve any central coordination, rather self-propelled individuals follow some basic rules for their collective motions and parallel behaviors. Man-made swarm behavior are influenced by this natural phenomena, for example swarm of aerial vehicles. There is no central controller directing the whole system and no member has a global view. For such artificial swarms, job of a control engineer is to implement self-defined rules through appropriate controller design keeping in view the vehicle dynamics, in order to achieve desired collective behavior. Further sophistication may be incorporated with the introduction of network dynamics that is a new area of research. Modeling and simulating the swarm behavior help to understand this phenomenon in a better fashion.

For trivial applications of small aerial formations, traditional PID feedback control is most popular due to its ease of implementation and low computational burden. However for cooperative control, additional tools like consensus algorithms and graph theory are to be involved. Consensus algorithms have proved their performance for self-organizing networked systems [82]. Graph theory helps to meet the challenges that may be encountered for MAS control, e.g. to handle communication topology that may also change due to a number of reasons. Graph theory is even helpful to analyze the network of networks and system of systems, as demonstrated in chapter 5. For large fleets or swarms, a single leader may not efficiently control the whole fleet and hence multiple leaders may

be necessarily required. Handling of communication topology for large fleets of aerial vehicles is facilitated with the use of graph theory.

6.1.1 Cooperative Control

A cooperative system could always be modeled as a single entity. The level of cooperation may be indicated by the amount of information exchanged between the units. Cooperative systems may also consist of heterogeneous units. The decision-making processes (control) are typically thought to be distributed or decentralized to some degree. Examples of cooperative control include collision avoidance by a fleet of quadcopters, hovering at the same altitude by the units in a fleet and anti-drone scheme etc. In the latter example, a fleet of quadcopters equipped with a net engages an intruder quadcopter. A scenario to this effect is shown in Figure 1.3 in Chapter 1. One of the distinct features of cooperative control is that this technique may be applied to the vehicles operating in a non-formation style which is mostly the case in real world scenarios. A number of existing and proposed applications for distributed control of cooperating mini UAVs may be seen in [1].

6.1.2 Discrimination of Cooperative Control

In conventional control approaches, one of the agent is generally assigned the role of a leader and the other agents are designated as followers. Reference (or command) values are injected to the agents to be controlled, and the objective is set to force the agents to follow these commands. Control values are computed keeping into account the state/output values and the command values. However for cooperative control, reference values are also determined by the agents based on the underlying communication topology and state/output values from other agents in a dynamic environment. Cooperative control may be considered to involve decision making process also, and hence makes the agents autonomous to a great extent. Agents are expected to decide about the command values themselves, e.g. through consensus algorithms. Concisely speaking, cooperative control may be characterized as a collection of interconnected decision making systems with limited processing capabilities, locally sensed information and limited inter-unit communications, all seeking to achieve a collective (global) objective. No explicit leader exists in case of control schemes based on consensus algorithms [13], that gives rise to redundancy.

A fleet of quadcopters has been considered in this chapter that exhibits cooperative control scheme through consensus algorithms. The control schemes of LQR PI servomechanism and LQR PI based on model following in combination with different consensus algorithms have been studied and the results compared. Consensus algorithms under different communication topologies and control schemes have been simulated. Reference

values for the trajectories are determined by the agents themselves, and no external command is dictated to the fleet of agents. Here we have considered dynamically decoupled systems which are coupled to each other through a common objective. The emphasis has been placed on simplicity without compromising on performance.

Further description of this chapter is as follows. Section 6.2 describes the problem formulation. Simulation results and analysis are shown in section 6.3. Summary of the chapter is given in section 6.4.

6.2 Problem Formation

Quadcopter four rotor inputs, collective and differential, specify the forces along the body-fixed x, y and z-axis, and the three moments in the body-fixed frame. Quadcopter motion dynamics and its dynamic model are given in details in subsection 3.2.1 of chapter 3.

Analysis framework in this chapter is based on tools from matrix theory, algebraic graph theory and control theory. A *graph* G is formally represented as $G = (V, E)$. Here V represents the set of vertices (or nodes) i.e. $V = \{1, 2, 3, \dots, n\}$ and E denotes the set of edges as $E \subseteq V \times V$. Neighbors j of agent i may be defines as $N_i = \{j \in V \wedge (i, j) \in E\}$. The neighbors are the agents which communicate with each other.

The units in MAS are social and communicate with each other. Communication topology may be modeled using the notion of Laplacian matrix which is considered as an interconnection operator. Eigenvalues of this matrix provide an insight into the stability and connectedness of agents in a formation. Laplacian matrix, for directed as well as undirected graphs, may be defined through

$$\mathcal{L} = D - A$$

where D is a diagonal matrix with $D_{i,i}$ equal to the out degree of vertex i and A is a matrix with $A_{i,j}$ equal to the number of edges from vertex i to j (including loops). A Laplacian matrix may be normalized as well. For our present study, we will only focus on Laplacian matrices without normalization.

6.2.1 Consensus Dynamics

Let $x_i(t)$ denote the state of agent i at time t for which agreement is required for all other agents in a fleet. We denote initial values of observable elements of states at time $t = 0$ as $x_i(0) = z_i$. Consensus dynamics over continuous time t for the graph $G = (V, E)$ are defined as:

$$\dot{x}_i(t) = \sum \{ x_j(t) - x_i(t) \} \quad (6.1)$$

Here all $\{V_i, V_j\} \in E$ and agents j are the neighbour of agent i . Collective consensus dynamics may be written in concise form as:

$$\dot{x}(t) = -\mathcal{L}(G) x(t) \quad (6.2)$$

where $\mathcal{L}(G)$ is graph Laplacian matrix, as defined before. Consensus is achieved by a team of vehicles if, for all $x_i(0)$ and all $i, j = 1, 2, \dots, n$, $|x_i(t) - x_j(t)| \rightarrow 0$, as $t \rightarrow \infty$, with n the total number of agents. It implies that state value will asymptotically converge to an agreement in equilibrium state i.e.

$$x_1(t) = x_2(t) = \dots = x_n(t) = k \quad (6.3)$$

where k is a constant number. The consensus value k is given as:

$$k = \frac{1}{n} \sum_{i=1}^n z_i \quad (6.4)$$

Equation (6.4) is nothing but the average of the initial values of states, hence commonly known as *Average Consensus*. Thus the collective decision of all agents is average of the initial states of agents. Consensus algorithm or *Consensus Manager* decides about the consensus values which are fed to the local controllers of agents as command values. Consensus control method is used to synchronize the controller actions. A number of applications for average consensus may be seen in networks and distributed systems.

It is to be noted that for *Average consensus*, states of all the agents are given equal weightage (or importance). However there may be some scenarios when states of some of the agents are to be given more weightage, for example due to having more sophisticated sensors. For such cases, *Weighted-Average consensus* as suggested in [82] may be exploited. Also note that consensus may not be desired for all the states, rather for few states of interest, e.g. we may only be interested in consensus for altitude in case of hovering of a fleet of quadcopters. Consensus for all the states means that agents will converge to a common point that may cause collision and hence not desired. In order to deal with such scenarios, the notion of Input bias is used as defined subsequently.

6.2.2 Input Bias

For many practical scenarios, we may be interested in consensus of agents states separated by a desired value. For example for a formation flight we desire the vehicles to

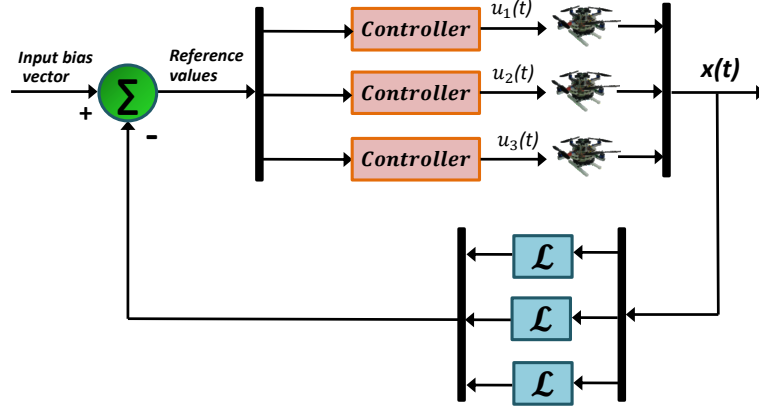


FIGURE 6.1: Block diagram for consensus algorithm with input bias.

be separated by a reasonable value in $x - y$ plane to avoid collision. To handle it with consensus algorithm, we need to introduce an input bias for reference value to the controller of agents. This may be represented with the help of a block diagram, as shown in Figure 6.1. For such cases, equation 6.1 may be written as [82]:

$$\dot{x}_i(t) = \Sigma\{x_j(t) - x_i(t)\} + b_i \quad (6.5)$$

The objective of agent i may be achieved with the introduction of this input bias b_i . Equation (6.5) may also be written as:

$$\dot{x}_i(t) = \Sigma\{x_j(t) - x_i(t) - r_{ij}\} \quad (6.6)$$

where $\Sigma r_{ij} = b_i$. It may be possible that some of the states of agents are controlled from ground station with injection of input bias vector and consensus may be sought for some other states of interest at the agent level. For example position of agents in $x - y$ plane may be controlled through ground station and consensus may be reached by the agents for the altitude only. Formation flight may be realized using the consensus based controllers with input bias. The whole scheme of cooperative control adopted in this chapter may be represented with the help of block diagram shown as Figure 6.2.

6.3 Simulation Results and Analysis

Different architectures are proposed and simulated in this section, thereby showing the viability of the schemes. Versatility of architectures is based on different control schemes, communication topologies and consensus algorithms. Performance of different cooperative control schemes under a number of scenarios has been assessed through extensive simulations. We first validate the algorithm with two agents only, and then

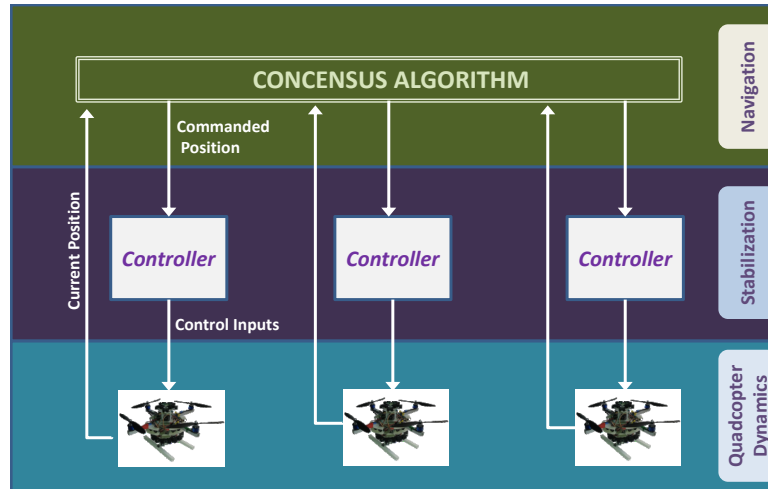


FIGURE 6.2: Cooperative control scheme with consensus algorithm.

add third agent to the cluster. Details for each scenario are provided in the following paragraphs.



6.3.1 Consensus Algorithm with LQR PI Servomechanism (Two Agents)

An *undirected graph* is assumed here, as shown at S. No. 1 of Table 6.1. Underlying Laplacian matrix and other details are also depicted in Table 6.1. Algebraic connectivity as well as largest eigenvalue is 2. More details on algebraic connectivity may be seen in chapter 5. Average consensus algorithm has been used, while introducing an input bias of 40cm in X-axis on agent 1. It is understandable that input bias is also averaged out when average consensus algorithm is used, i.e. an input bias of 40cm for either agent results in a mutual separation of 20cm, as demonstrated in 3D view of Figure 6.3. To appreciate it on single axis (X-axis), Figure 6.4 is presented. Transient response was not smooth, as expected from LQR PI servomechanism.

6.3.2 Consensus Algorithm with LQR PI Controller based on Model Following (Two Agents)

Average consensus algorithm with undirected topology was assumed here. Relevant details for this scenario are provided at S. No. 2 of Table 6.1. First of all, we simulate average consensus algorithm (for X-axis only) without input bias. Average consensus is a function of initial values of states. Initial positions of agents, in X-axis, were 200cm and 300cm respectively, so agents made a consensus to converge on 250cm as shown in Figure 6.5. With the application of LQR PI controller based on model following, smooth transient response was observed.

TABLE 6.1: Communication Topologies and Related Metadata for Two Agents.

S. No.	Topology	Control Scheme	Laplacian Matrix	Spectrum	Input Bias (cm) on X-axis	Convergence Value
1		LQR PI Servomechanism	$\mathcal{L} = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$	$\lambda = 0, 2$	+ 40 on agent 1	20cm mutual separation & 40cm total drift of agents from average value of 250cm
2		LQR PI model following	$\mathcal{L} = \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$	$\lambda = 0, 2$	0	Average value of initial positions
					-20 on agent 1 +20 on agent 2	-10 cm from average value (Agent 1) +10 cm from average value (Agent 2)
					-40 on agent 1	-30 cm from average value (Agent 1) -10 cm from average value (Agent 2)

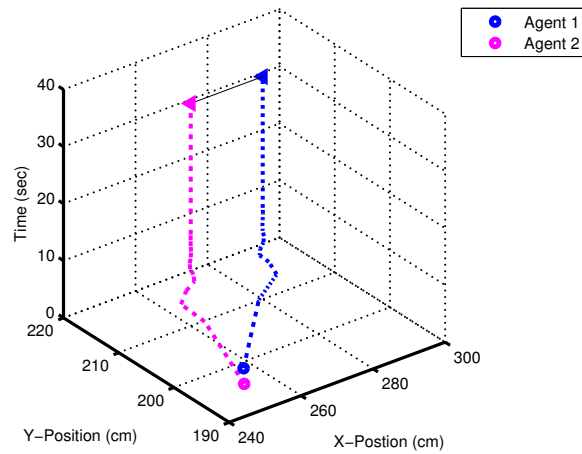


FIGURE 6.3: Consensus for two agents with input bias on X-axis (3D view).

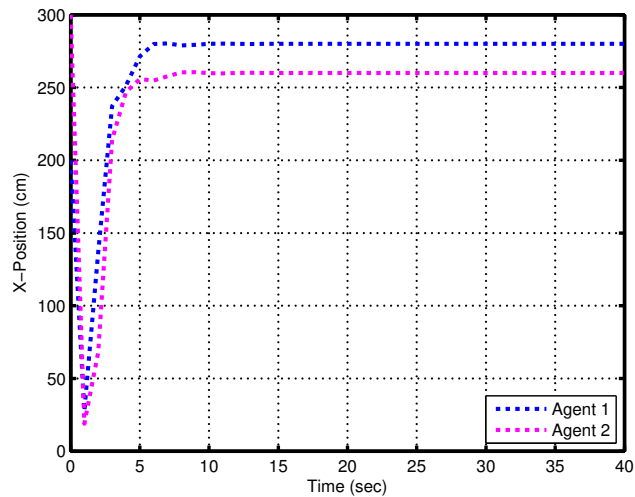


FIGURE 6.4: Consensus for two agents with input bias on X-axis.

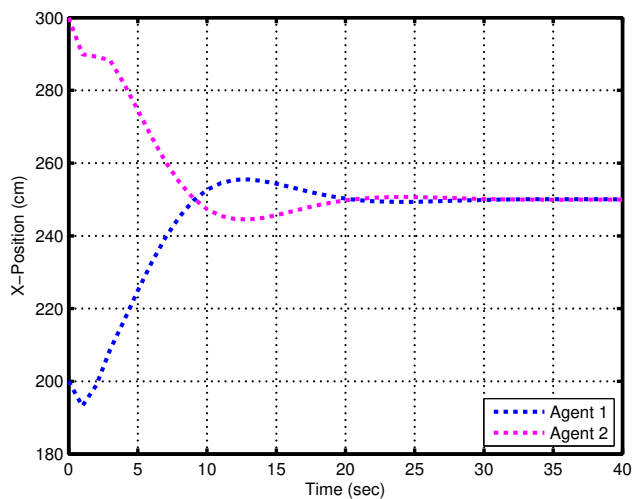


FIGURE 6.5: Consensus for two agents without input bias.

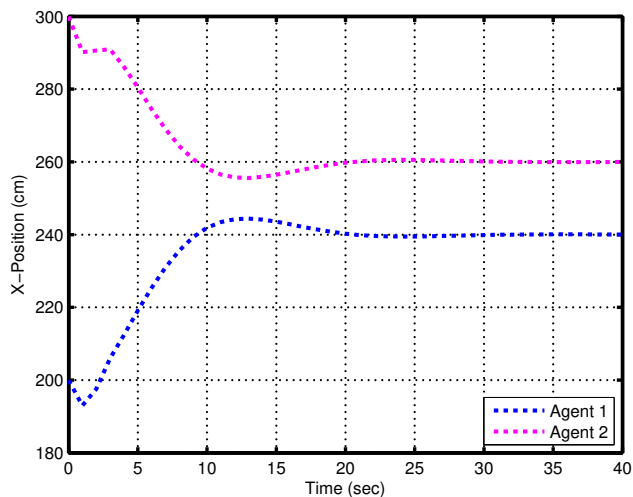


FIGURE 6.6: Consensus for two agents with input bias on both agents.

Now we study the effects when we introduce an input bias of -20cm and 20cm on agents 1 and 2 respectively. Each agent converged to -10cm and $+10\text{cm}$ away from the average value of 250cm . Simulation to this effect is shown in Figure 6.6.

Now we study an interesting scenario where an input bias of -40cm is applied only on agent 1. Simulated result is depicted in Figure 6.7. We note that agent 1 and agent 2 converged to -30cm and -10cm away from average of their initial values respectively. However their mutual separation is 20cm .

Now we extend the scenario to three agents and study the effects under different communication topologies and control schemes. Results are depicted in ensuing paragraphs.

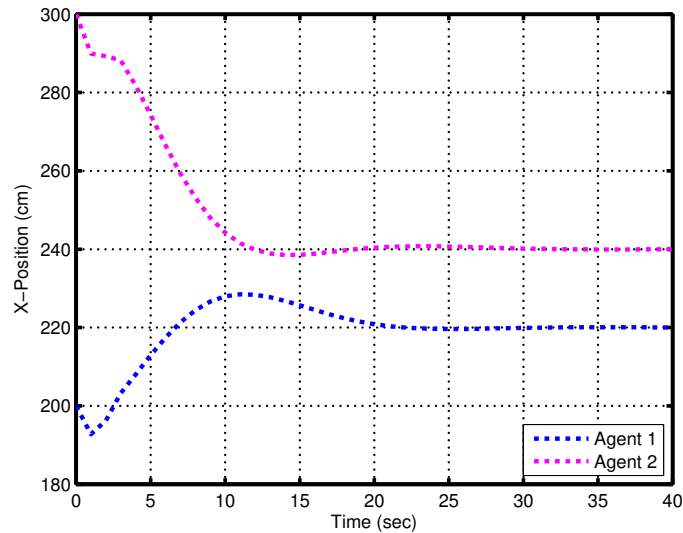


FIGURE 6.7: Consensus for two agents with input bias on one agent only.

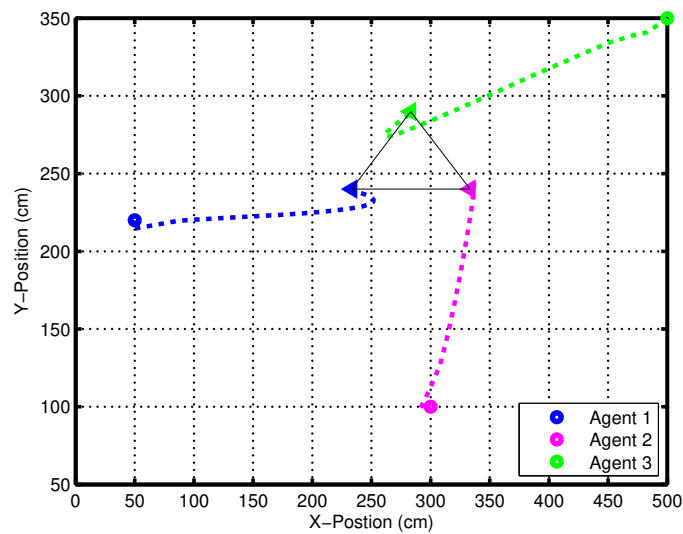
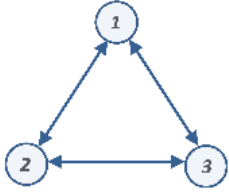
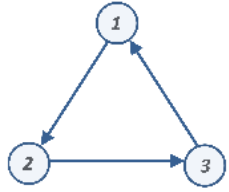
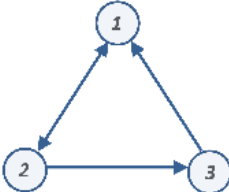
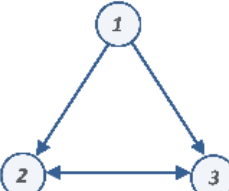


FIGURE 6.8: Consensus for three agents to make a triangle.

6.3.3 Consensus Algorithm with LQR PI Control Scheme with Undirected Topology

Average consensus algorithm with undirected graph is assumed here. Initial positions of agents 1, 2 and 3 are $[50, 220, 50]$, $[300, 100, 50]$ and $[500, 350, 50]$ respectively. Input biases of $[-100, 0]$, $[100, 0]$ and $[0, 100]$ were imposed on agent 1, 2 and 3 respectively in X-Y axes. Corresponding Laplacian matrix is given at S. No. 1 of Table 6.2. Plots to this effect are shown in Figures 6.8, 6.9 and 6.10.

TABLE 6.2: Communication Topologies and Related Metadata for Three Agents.

S. No.	Topology	Laplacian Matrix	Input bias on X- and Y-axis	Convergence Value
1		$\mathcal{L} = \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix}$	$[-100, 0]$ agent 1 $[100, 0]$ agent 2 $[0, 100]$ agent 3	Adjusted as per input biases and initial positions
			Zero	Average value of initial positions
2		$\mathcal{L} = \begin{bmatrix} 1 & -1 & 0 \\ 0 & 1 & -1 \\ -1 & 0 & 1 \end{bmatrix}$	Zero	Average value of initial positions
3		$\mathcal{L} = \begin{bmatrix} 1 & -1 & 0 \\ -1 & 2 & -1 \\ -1 & 0 & 1 \end{bmatrix}$	Zero	Drifted 50cm from average value towards initial position of agent 1
4		$\mathcal{L} = \begin{bmatrix} 2 & -1 & -1 \\ 0 & 1 & -1 \\ 0 & -1 & 1 \end{bmatrix}$	Zero	Drifted 100cm from average value away from initial position of agent 1

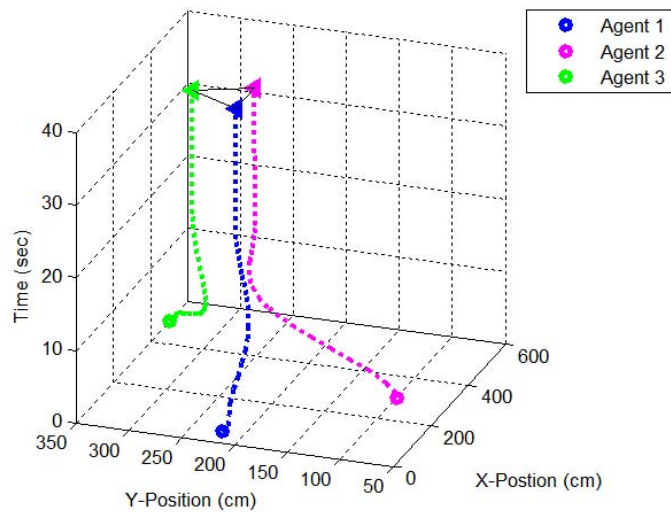


FIGURE 6.9: Consensus for three agents with input bias.

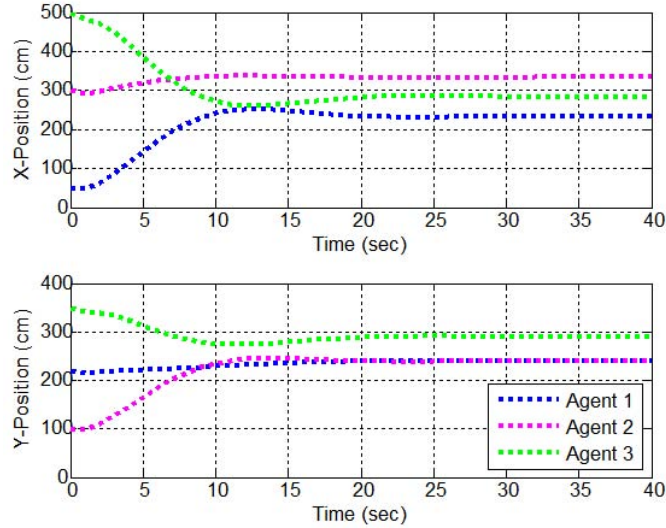


FIGURE 6.10: Consensus for three agents in X- and Y-axis.

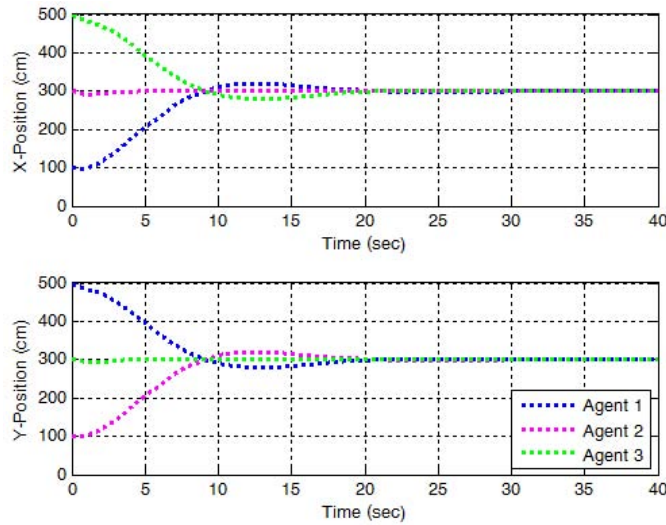


FIGURE 6.11: Consensus for three agents in X- and Y-axis without input bias.

In order to appreciate the difference for consensus values under different types of communication topologies, we remove the input bias and simulate it again for X- and Y-axis. Initial positions of agents 1, 2 and 3 are set as $[100, 500, 50]$, $[300, 100, 50]$ and $[500, 300, 50]$ respectively. Plot is shown in Figure 6.11. Based on the results, we can deduce the following definition:

Definition 1. If a graph is fully connected, it always gives rise to *average consensus*.

Row sum of a Laplacian matrix is zero, so the first eigenvalue is always zero. Laplacian matrix of an undirected graph is symmetric. However for a directed graph, Laplacian matrix is non-symmetric, as seen in Table 6.2 also.

6.3.4 Consensus Algorithm with LQR PI Control Scheme with Directed Topology

Now we consider the effects when undirected topology is changed to different types of directed topologies. Results for each type of directed graph are studied individually.

- (a) **Cycle Topology.** Graph to this effect is shown in at S. No. 2 of Table 6.2. Corresponding Laplacian matrix is also given. Simulation under cycle topology revealed that a plot identical to that for an undirected topology is generated. In both cases, consensus values converged to the average of initial values of states. As simulation plot is same, so this is not reproduced here.
- (b) **Directed Graph (S. No. 3 in Table 6.2).** We now changed the topology where the agent 2 is also transmitting the information to agent 1, as shown at S. No. 3 of Table 6.2. Corresponding Laplacian matrix is also given. It is observed that with this topology the consensus values are not the average of initial values, rather average consensus value is attracted towards the initial position of agent who is receiving additional information (high number of incoming edges) compared to other agents. For the topology under consideration, agent 1 is receiving information from two agents while the other two agents are receiving information from one agent only. Consensus value is drifted by $50cm$ from the average value towards initial position of agent1, as shown in Figure 6.12.
- (c) **Directed Graph(S. No. 4 in Table 6.2).** Now we consider the scenario as depicted at S. No. 4 of Table 6.2. Here agent 1 is providing information to other two agents, but not receiving information from any of the other agents. We observe that the consensus value is attracted by agent 2 and 3. For this topology, consensus value is $100cm$ away from average value of initial positions of agents. Simulation result is shown in Figure 6.13.

Definition 2. Undirected graph and cycle topology give same consensus values (average of initial values). In other words, average consensus values are achieved if communication topology pertains to undirected graph or cycle topology graph.

6.3.5 Weighted-Average Consensus Algorithm with LQR PI Control Scheme with Undirected Topology

The Laplacian for the continuous-time consensus may be chosen as per the application of interest. If average-consensus is desired to be reached, only $\mathcal{L} = D - A$ may be used [82]. However if want to assign weights to the states of a certain agent (e.g. an agent with a more precise sensor), then we may exploit weighted-average consensus with

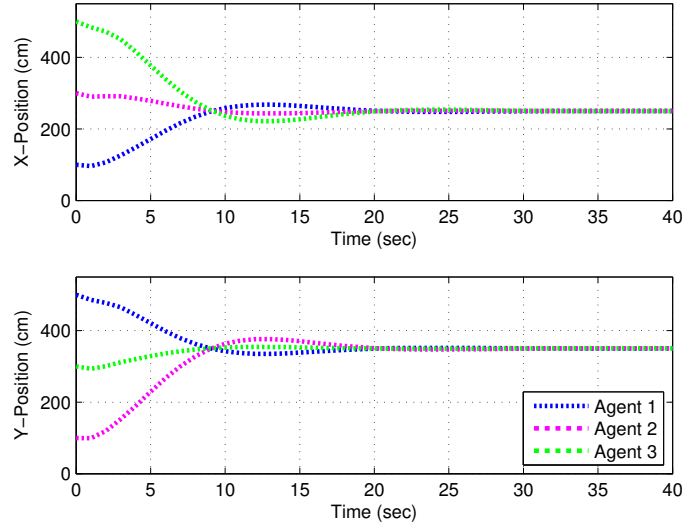


FIGURE 6.12: Consensus for three agents in X- and Y-axis for directed topology of S. No. 3 in Table 6.2.

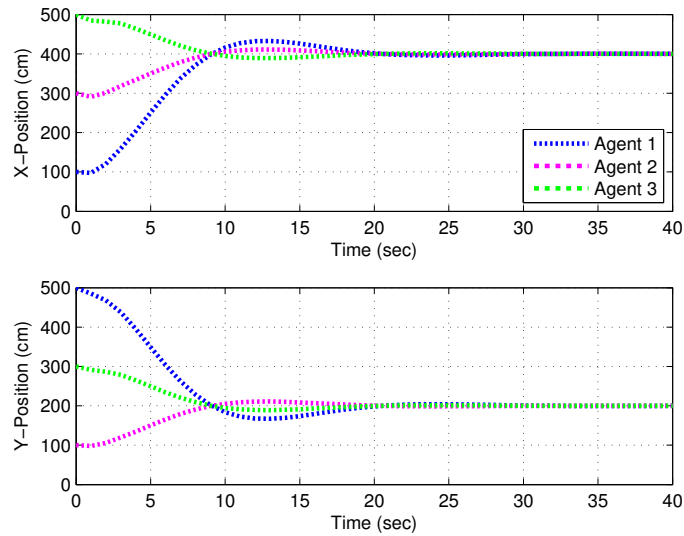


FIGURE 6.13: Consensus for three agents in X- and Y-axis for directed topology of S. No. 4 in Table 6.2.

desired weighting factors $\Upsilon = (\Upsilon_1, \Upsilon_2, \dots, \Upsilon_n)$. For such scenarios, following relation holds:

$$K\dot{x}(t) = -\mathcal{L}(G)x(t) \quad (6.7)$$

Where $K = \text{diag}(\Upsilon_1, \Upsilon_2, \dots, \Upsilon_n)$. Equation 6.7 may be written as:

$$\dot{x}(t) = -K^{-1}\mathcal{L}(G)x(t) \quad (6.8)$$

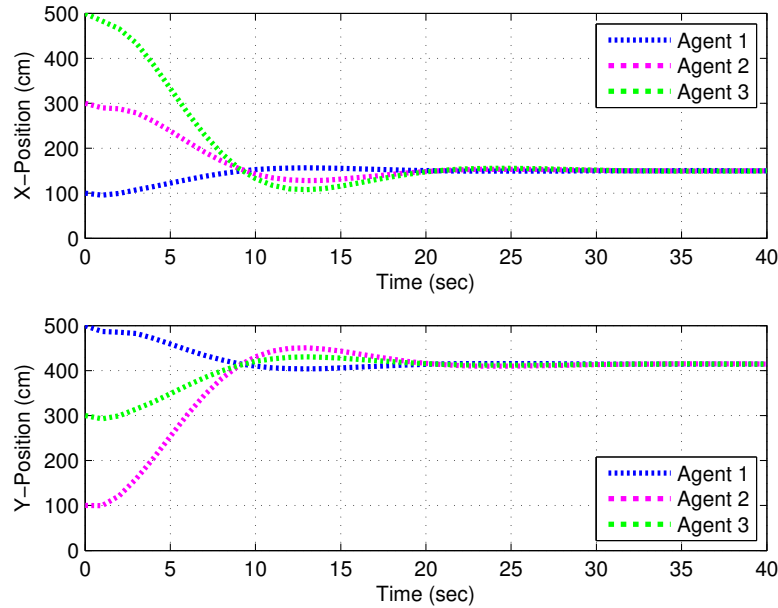


FIGURE 6.14: Weighted-average consensus for undirected topology of S. No. 1 in Table 6.2.

Weighted-average consensus algorithm was simulated for three agents with $K = \text{diag}(10, 1, 1)$ and $K = \text{diag}(5, 1, 1)$ for consensus on X- and Y- position respectively. We here assign high weightage to the states of agent 1 assuming that it has more reliable measurements compared to those of agent 2 and 3. Simulation results is plotted in Figure 6.14. We note from the plot that consensus values are attracted towards initial values of agent 1 corresponding to the weighting factor of 10 and 5 in X- and Y- position respectively. Observing the plots of directed graphs and weighted-average consensus, it may be concluded that consensus for directed graphs is a special case of weighted-average consensus.

6.3.6 Switching Consensus Values

Different missions may require to switch consensus values. Such a scenario is simulated in Figure 6.15 where three agents first converge to a common value, and then they get apart to make a mutual separation.

6.4 Summary

Consensus equilibrium is a function of only the initial information states of those vehicles that have a directed path to all of the other vehicles [119]. Same is evident in the simulations demonstrated in this study. The presented scheme is scalable and can be extended to include large number of quadcopters. However there is a trade-off between

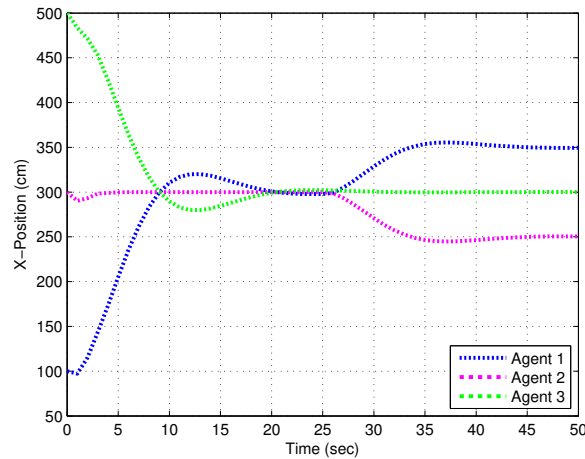


FIGURE 6.15: Switching consensus for undirected topology of S. No. 1 in Table 6.2.

having a large number of agents and robustness to time-delay. Therefore construction of engineering networks with large number of units is not a good idea for reaching consensus [82].

In this chapter, consensus algorithms in combination with different control schemes, including LQR PI servomechanism and LQR PI based on model following, have been simulated and evaluated. Extensive simulations under different conditions validated the efficacy of proposed schemes. The schemes are simple to implement for cooperative control of agents. Depending on the application, consensus algorithm may be tailored accordingly. The work presented here assumes that states of interest for a quadcopter are available. In case of non-availability of states, a suitable *Observer* may be designed to estimate the states.

Chapter 7

Differential GPS Implementation

7.1 Introduction

The term Global Navigation Satellite System (GNSS) refers to satellite based global positioning system that is used to provide autonomous geo-spatial positioning [120]. The system allows tiny electronic receivers to determine their absolute position (longitude, latitude and altitude) anywhere on the globe. NAVSTAR GPS is the first GNSS that was developed to provide precise location, based on data transmitted from a constellation of more than 30 satellites [99]. Users can determine their coordinates by receiving range information (and other observables, if required) from at least four GPS satellites. A pictorial view to this effect is shown in Figure 7.1. A GPS receiver is used not only for position determination but also for other purposes like navigation, attitude determination and relative positioning of vehicles [121] etc. Other similar systems are Russian GLObal NAVigation Satellite System (GLONASS), European Union Galileo and Chinese Beidou [120]. Galileo is still in deployment phase.

7.1.1 Merits and Demerits of GPS

A GNSS can generally be used under all weather conditions [122]. Its merits include accuracy, robustness, flexibility, low cost (*L1* receivers only), reduced size and power consumption. It also provides highly accurate timing for on-board time synchronization. One of the biggest advantages for GPS based formation flight is that it is not limited to short baseline, as is the case for laser scanner and vision based approaches. GPS receiver acts as a primary or backup sensor for most of the GNC applications. The drawbacks of GPS are dependency on external signals, and requirement of minimum of four satellites for 3D position determination. Quality of position solution is dependent on satellites geometry. GPS signals are also vulnerable to jamming. Accuracy of GPS based navigation suffers considerably from Ionospheric effects.

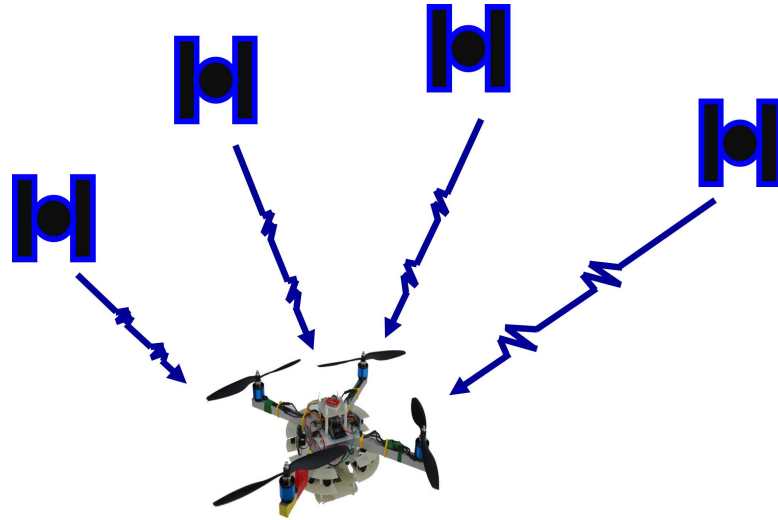


FIGURE 7.1: A quadcopter receiving signals from four GPS satellites.

7.1.2 GPS Signals

GPS signals contain a pseudo-random code, broadcast ephemeris and almanac data [99]. The pseudo-random code identifies that which satellite is transmitting. Ephemeris is description of the satellite orbits and clock correction parameters which vary over the time. This data is transmitted by each satellite and contains information like status of the satellite (healthy or unhealthy), current time/date, Ionospheric correction parameters, Keplerian orbital parameters and satellite clock corrections etc. Each satellite broadcasts only its own ephemeris data. This data is used to compute the coordinates of GPS satellites, which are subsequently used to determine the GPS receiver coordinates. The ephemeris may be broadcast (projected ahead into time) or precise (post-fitted). The orbital positions sent in the navigation message are based on the predicted position of the satellite, and are updated every two hours by the ground control [123]. As GPS satellites travel at a speed of approx. 4km/sec [124] so they travel almost 29,000 km between orbit updates. In order of ascending accuracy, the ultra-rapid orbits are available after approx. 6 hours, the rapid orbits are available after 13 hours and the final post-fit precise orbits are available after about 10 days on International GNSS Service (IGS) website [125]. The almanac data are a reduced-precision subset of the clock and ephemeris parameters; and are updated by the Control Station at least once every 6 days [100].

7.1.3 GPS Navigation Solution

The GPS navigation solution determines the 3D coordinates and clock offset of a GPS receiver using the pseudorange measurements of at least four GPS satellites. The equations linking the pseudoranges and the receiver coordinates are nonlinear. Direct solution of these nonlinear equations is also possible [126]. The widely used alternative is

to linearize the pseudorange equations and to use the tool of linear algebra for position determination.

7.1.4 Why DGPS is Required?

GPS observables include four fundamental quantities including time, pseudorange, carrier phase and Doppler. Using these observables, a typical GPS receiver is able to determine the position and velocity. Further processing of data gives heading, attitude information and relative position. However these observables are corrupted by the biases and noises thereby leading to positioning inaccuracy. In order to remove these errors, Differential GPS (DGPS) technique was developed. It improves the accuracy of coordinates of a GPS receiver, installed on a rover, applying some correction methodology. The technique is of utmost importance for many applications, like photogrammetry, requiring information of true coordinates of a camera carrying vehicle. In order to accurately geo-reference the data, it is important to know the exact position and attitude of the vehicle when a measurement or a picture was taken [127]. Depending on the requirements of a particular application, the position of the vehicle often needs to be known with a precision down to the decimeter-level or even centimeter-level. Such demands can be met by applying the techniques like DGPS.

7.1.5 Characteristic of Present Study

Although some public MATLAB codes are already available for positioning using RINEX files, as mentioned in Section 1.7 of Chapter 1. However, DGPS implementation in offline mode using correction factors for determination of rover true coordinates without using double differences is not seen in the literature, as per the knowledge of author of this thesis. In this study, we have developed a simple to use DGPS algorithm using correction factors computed at base station in post-processing mode exploiting Course Acquisition (C/A) code on single frequency (L1 signals) to make it practicable for low-cost GPS receivers.

Further description of this chapter is organized as follows. In Section 7.2, we provide a synopsis of GPS errors and discuss different DGPS techniques to remove these errors. In Section 7.3, a brief introduction to RINEX format files is provided. Section 7.4 describes in details all aspects of software code for DGPS and mathematical model for correction factors. Simulation results of data processing with and without DGPS implementation are also shown. Summary of the chapter is give in Section 7.5.

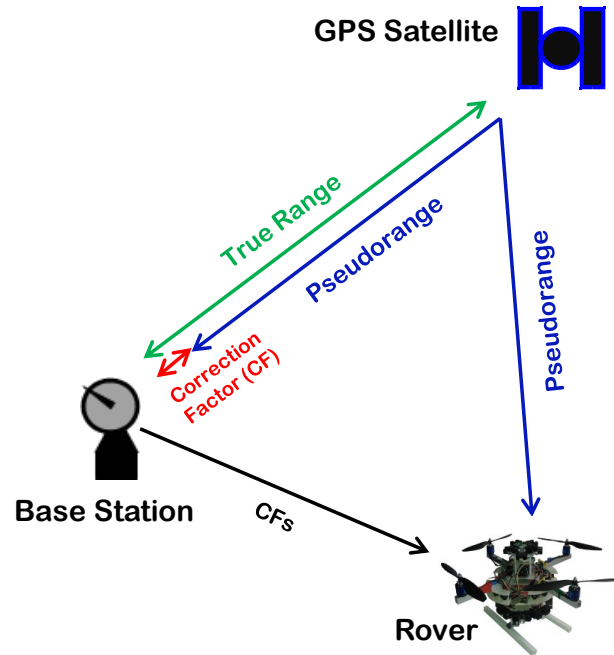


FIGURE 7.2: Pictorial view of DGPS technique.

7.2 Differential GPS Techniques

GPS measurements consist of biases and noises that affect the positioning accuracy. In order to improve the GPS positioning solution, it entails to know these errors and the methods to remove them. DGPS is a technique that improves the solution accuracy while removing these errors. It was developed to meet the needs of positioning and distance measuring applications that required higher accuracies than stand-alone Standard Positioning Service (SPS). DGPS can be considered as a calibration method where the calibration standard is established at the base station. Figure 7.2 illustrates the principal for DGPS based on pseudoranges correction. The correction factors are computed at base station in real-time and transmitted to rovers, which take it into account to correct pseudoranges and hence find the 'true' ranges. Based on received information, rovers are able to determine their 'true' coordinates.

7.2.1 GPS Error Sources

Major sources of errors that affect the accuracy of stand-alone GPS receiver include ephemeris errors, atmospheric errors (including Ionospheric and Tropospheric propagation delays), satellite and receiver clock errors, Dilution of Precision (DOP), receiver noise and multipath errors. Ephemeris errors are largely diminished by differential corrections. Ionospheric errors can be remedied either with dual frequency (L1/L2), or

with proper mathematical modelling like Klobuchar model. Double difference information also caters for these errors. Troposphere affects the two frequencies equally, however their effects can be fixed by Hopfield model. When base station and rover are close enough, satellite signals pass through almost same atmospheric conditions, so ionosphere and troposphere errors are almost identical and can be effectively cancelled with DGPS technique. Satellite clock errors are due to asynchronization between satellite clock and receiver clock. These include Satellite Vehicle (SV) clock offset, clock drift and clock drift rate. It can be corrected using the polynomial coefficients af_0 , af_1 and af_2 transmitted in navigation message. This error can be effectively compensated as per the algorithm defined in [100]. Receiver clock drift is usually treated as an extra parameter and corrected in the standard solution. Furthermore, it does not significantly add to differential errors.

How precisely a GPS receiver can measure the pseudorange and carrier phase largely depends on how much noise accompanies the signals in the receivers tracking loops. This noise either comes from the receiver electronics itself or is picked up by the receivers antenna [128]. Multipath error is site dependent and varies significantly as the site conditions change. This type of error cannot be modelled. Multipath and receiver noise errors cannot be corrected by DGPS and hence lead to residual errors in DGPS methods.

7.2.2 Correction Methodologies

Some of the correction methodologies include Position Differential, Pseudo Range Differential, Carrier Phase Differential, Precise Point Positioning (PPP) and Satellite Based Augmentation System (SBAS) etc. Each technique has its own merits and demerits. Most widely used method is carrier phase Double Difference to achieve high accuracy (centimeter level), but the solution suffers from integer ambiguity and cycle slips. Whenever a cycle slip occurs, it must be corrected for, and the integer ambiguity must be re-calculated.

We have implemented pseudorange correction method in offline mode for our present study. With this technique, a reference station is set up to track all satellites in view, ensuring that it will see at least the four satellites that the roving receiver is using to compute positions. This station with exactly known position measures the signal travel time to all visible GPS satellites and uses these values to calculate pseudoranges. These measured values will typically include errors. Since the real position of the reference station is known, the actual distance (nominal value) to each GPS satellite can be calculated. The difference between geometric and measured ranges can be calculated by a subtraction called as *correction factor*. These correction factors are different for all GPS satellites and are even different for same satellite at different epochs. *Epoch* is a moment when a measurement is taken by a GPS receiver. These correction factors are sent to rover receiver for all the epochs using suitable media, in case of real-time applications. Rover uses these factors to correct its pseudorange measurements which are subsequently used to determine its accurate position. The corrected pseudorange

at the moving receiver is corrupted by only two errors, the multipath error and receiver noise.

7.2.3 DGPS Implementation Modes

DGPS data processing may be realized in following two ways:-

- (a) **Real-Time Processing** For navigation applications, pseudorange corrections are needed in real-time that can be transmitted to the users via a communication link in Radio Technical Commission for Maritime Service (RTCM) SC-104 standard format. This is an encrypted format and is used for DGPS applications. This is the most common technique where a large number of users may be served in real-time. Although precision level of real time applications is comparatively low, however the technique is quite useful to confirm that a test is progressing properly and also because many applications require real time processing. However data latency issues are to be taken into consideration for such applications. SBAS systems like WAAS, EGNOS and MSAS are all real time DGPS applications.
- (b) **Post Mission Processing** For offline processing, GPS raw observations (pseudoranges, carrier phase, Doppler and Signal to Noise Ratio (SNR)) are stored by a rover receiver and then later processed in combination with raw observations of base station receiver stored for the same time period. Most of the low-cost GPS receivers are also able to provide the raw data. The advantage of the post-mission solution over the real-time one is being more accurate and reliable, because the user can detect data errors and analyse the residuals. Also for some applications, like photogrammetry, the cost and effort to maintain a real time data link may be unnecessary. On the other hand the main disadvantage of the post-mission solution is that the results are not available immediately.

7.2.4 DGPS Accuracy

Two levels of accuracy are achievable with DGPS, meter-level and centimetre level. Meter-level accuracy relies on C/A code data while centimeter-level relies on carrier phase data. Many applications of DGPS use C/A code pseudorange as the only observable, with achieved accuracies of 1 – 5 m in real-time [99]. DGPS not only increases the GPS positioning accuracy, but also enhances GPS integrity by compensating for anomalies in the satellite ranging signals and navigation data message. If intermediate level accuracy is required, the SBAS services may be exploited. A big advantage to use these services is that the signals are transmitted on L1 frequency and no decoder is required which make this service usable for low-cost GPS receivers. Performance of EGNOS, the European SBAS, for open service in terms of accuracy is 3m lateral and 4m vertical, while its availability is 99 percent. Typical applications of DGPS include relative navigation and determination of rover accurate coordinates, baseline and attitude.

7.3 RINEX Files

Although receivers calculate positions in real time; in many cases it is suitable to store GPS observables for later use. RINEX is a standard format that allows the management of the observables generated by a receiver, as well as their off-line processing by a number of applications. It is a set of standard definitions to promote the free exchange of GNSS data and to facilitate the use of data from any GNSS receiver with any post processing software package. RINEX allows the user to post-process the received data in order to produce a more accurate solution, usually with other data unknown to the original receiver such as better models of the atmospheric conditions at the time of measurement. Rover data may be used in combination with other data stored at a base station. RINEX format is designed to evolve over time, adapting to new types of measurements and new satellite navigation systems. It enables storage of raw measurements for all GNSS along with data from SBAS simultaneously.

7.3.1 RINEX Files Classification

RINEX files are classified into six categories as depicted in Table 7.1. For this study, only first two types of files have been utilized. In fact, at least these two files are required to completely define the data in RINEX format. Each observation file and each meteorological data file contains the data for one site and one session. Each file type consists of a header section and a data section. The header section contains global information for the entire file and is placed at beginning of the file. The header section contains header labels in columns 61 – 80 for each line contained in the header section. These labels are mandatory and must appear exactly as specified in [129] and [130]. There is no limitation for maximum length of observation records.

We briefly describe the two types of files used for our study. RINEX *observation file* (data section) typically includes number/ type of observations, epoch time when the measurement was taken, number of visible satellites, visible satellites ID commonly referred as Pseudo-Random Noise (PRN) code, and numerical value of observables. For DGPS applications, we need to record observation file simultaneously at base station and at rover site. RINEX *navigation message file* contains the broadcast ephemeris data. This data is useful for a number of functions like computation of satellite clock error and satellites coordinates etc. If data from more than one receiver have to be exchanged, it would not be economical to include the identical satellite messages collected by the different receivers several times. Therefore the navigation message file from one receiver may be exchanged or a composite navigation message file may be created containing non-redundant information from several receivers in order to contain complete information in one file. For DGPS with short baseline (till 10 km), one navigation file collected either at base station or at rover may serve the purpose. Every GNSS has its own navigation message data.

TABLE 7.1: Classification of RINEX Files.
Table adapted from Ref. [6].

S.No	File Category	Description
1	Observation data file	Contains measurement data like GPS time, pseudoranges, carrier phase, Doppler and SNR etc.
2	Navigation message file	Contains GPS broadcast ephemeris data and Ionospheric parameters
3	Meteorological data file	Contain meteorological data (ambient pressure, temperature and humidity etc.) for post processing with high accuracy
4	GLONASS navigation message file	Contains GLONASS satellites ephemeris data
5	GEO navigation message file	For WASS/EGNOS geostationary satellites
6	Satellite and receiver clock data file	Contains clock data

7.3.2 Conversion of GPS Binary Data to RINEX Format

GPS binary data may be converted to RINEX format using suitable software like *teqc* or *rtklib*. Many proprietary formats (like .ubx files of u-blox GPS receivers) may be converted to RINEX format using such software. However care should be exercised to ensure that proper settings have been made on the receiver to output both observation as well as navigation data. User can choose the RINEX version, as required, provided the software supports the chosen version.

7.4 DGPS Software and Simulation Results

DGPS data has been post processed using MathWorks MATLAB that facilitates matrices handling/ manipulation, as required for this study. DGPS simulations have been performed with observations collected at base station and a rover. Differential correction factors computed at base station have been exploited for determination of rover accurate coordinates.

Base station and rover coordinates have been calculated with iterative least square method using pseudoranges from at least four SVs. If more than four satellites are visible (as mostly is the case for airborne vehicles or for GPS receivers in open area), it is recommended not to utilize the data from the near-horizon satellites. As signals from these satellites travel comparatively longer distances through atmosphere, so are more prone to atmospheric effects. It is recommended to set the elevation mask to at least 10° to eliminate the most noisy data (but not more than 15° so that usable data

is not lost). Spherical coordinates of SVs are therefore computed for determination of corresponding elevation angle. For our present study, we have set elevation mask to 10° . As total transmitted power from a satellite is less than 50 watts, so GPS signals are relatively weak. Comparison of SNR between satellites can show the source of the cleanest data. It is important to use only that data for computation that does not fall below acceptable SNR (commonly set as 20 – 30 dB). For our study, this threshold is set to 20 dB. Base station and rover do not see the same set of satellites for all the epochs. Most of the time, a new satellite appears (or disappears) at different epochs at base station and rover. So we make a further criterion of common satellites to improve the position accuracy at the rover side.

For our study, TRIMBLE NETR5 GPS receiver installed at the German State Survey SATelliten POSitionierungs dienst (SAPOS), Stuttgart was treated as base station. Its GPS receiver antenna coordinates are exactly known. While data from Trimble NETR8 GPS receiver, positioned at Institute of Navigation, University of Stuttgart was treated as rover data. Its coordinates are also accurately known. With this arrangement, we can compare the rover coordinates (determined by using correction factors) with those already known accurately, thereby verifying the efficacy of developed algorithm. Rover coordinates have been computed with and without applying correction factors for comparison purpose. A self-explanatory flow chart is shown in Figure 7.3.

A MATLAB script using C/A pseudoranges on L1 frequency for positioning and velocity computation of GPS receiver was developed by Dipl.-Ing. Michael Gaeb [101]. This code was extended for DGPS implementation in our study. The original code is written while taking into consideration RINEX format 2.11 as input files containing GPS observables in the sequence C1, L1, D1, S1, where,

C1: C/A code pseudoranges on L1 frequency (m)

L1: Carrier phase on L1 frequency (cycle)

D1: Doppler on L1 frequency (Hz)

S1: Signal to noise ratio on L1 frequency (dBHz)

Main features of the code for DGPS implementation are as following:

1. RINEX observation files (recorded at base station and rover) and navigation message file (recorded at either location) can be used.
2. Correction factors are computed at base station and sent to rover for its accurate position determination.
3. No atmospheric model has been used. Atmospheric errors have been catered through correction factors.
4. The code provides solution for rover coordinates with and without DGPS for all the epochs.

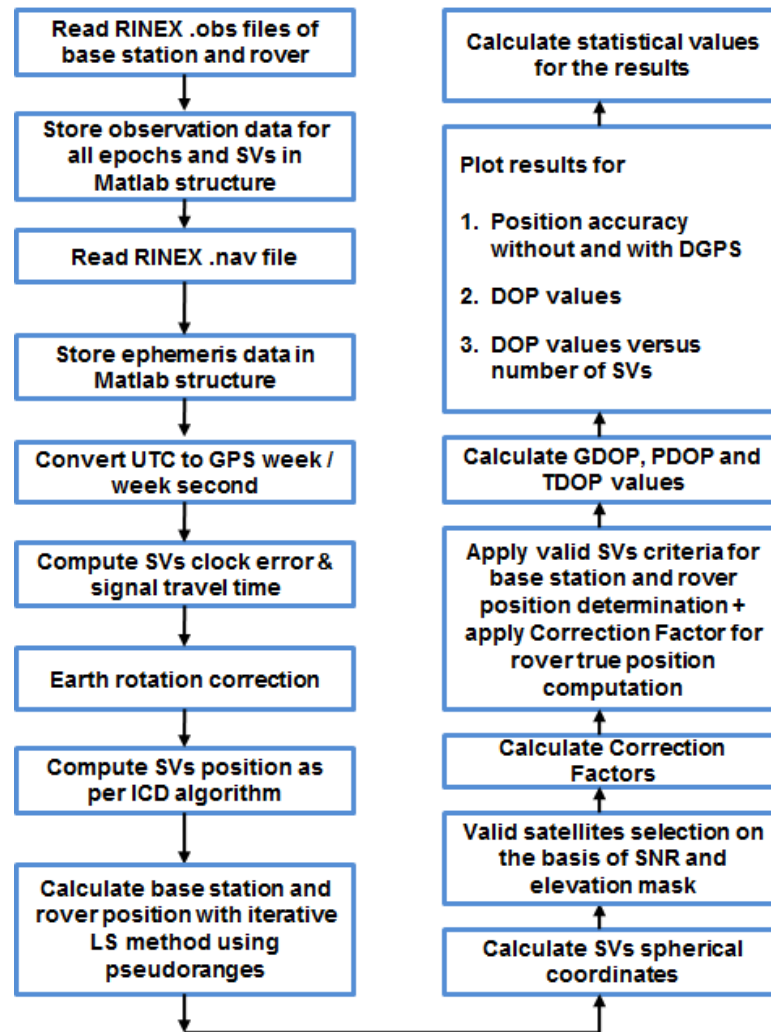


FIGURE 7.3: Flow chart for DGPS implementation.
Figure adapted from Ref. [6].

5. Number of epochs for processing can be selected.
6. Total number of visible satellites and valid satellites can be viewed for all the epochs. Corresponding PRNs are also observable.
7. DOP values for GDOP, PDOP and TDOP have been calculated at rover. Effect of number of visible valid satellites on PDOP values has been studied.
8. Code has been validated while placing a rover at known location. Coordinates determined with the code have been compared with the accurately known coordinates.
9. Statistical results are computed for minimum offset, maximum offset and mean positional error.

10. Results for position error in three axes Earth Centered Earth Fixed (ECEF) as well as radial difference have been plotted.

7.4.1 Mathematical Model for Correction Factor (CF)

We first define the notion of pseudorange and geometric range. *Pseudorange* is an indicative of travel time of satellite signals. It is a noisy estimate of range, hence named as pseudorange. Pseudorange ρ is defined as the distance from the receiver antenna to the satellite antenna including receiver and satellite clock offsets (and other biases such as atmospheric errors). Mathematically, it can be expressed as:

$$\rho = r + c.(dt_r - dt^s + dT) \quad (7.1)$$

where r is the geometric range between satellite and receiver, dt_r is the receiver clock offset, dt^s is the satellite clock offset, c is the speed of propagation and dT corresponds to other biases. Pseudorange reflects the actual behaviour of the receiver and satellite clocks. It can be measured via code and/or carrier phase, and is stored in units of meters.

Geometric range is the true distance between two points. If coordinates of satellite and receiver are known, geometric range r between these two points can be obtained using the following formula:

$$r = \sqrt{(X^s - X_r)^2 + (Y^s - Y_r)^2 + (Z^s - Z_r)^2} \quad (7.2)$$

where (X^s, Y^s, Z^s) are satellite coordinates and (X_r, Y_r, Z_r) are receiver coordinates in ECEF coordinate system. The *CF* at base station for any satellite in view at epoch t , is computed as following [131]:

$$CF = r - \rho + (dt_r - dt^s + T) * c \quad (7.3)$$

here,

r = Geometric range between satellite and base station receiver (determined as per eq. 7.2)

ρ = Pseudorange (GPS observable measured by receiver)

dt_r = Receiver clock offset (estimated through GPS navigation solution)

dt^s = Satellite clock offset (part of ephemeris transmitted by GPS satellite)

T = Tropospheric delay (not used in present algorithm)

c = Speed of propagation (a constant)

These CFs are then handed over to rover for all valid satellites observed during each epoch. The rover takes it into account while adding it to its observed pseudorange for correction. Its position is then computed based on the corrected pseudoranges. Here, no atmospheric model has been used for computation of base station and rover receiver coordinates. As atmospheric conditions at base station and rover site are almost identical for short base line, hence their effects are mutually cancelled out. In other words, atmospheric effects are included in correction factors. Corrected position at rover site is free from atmospheric effects.

7.4.2 Simulation Results and Discussion

Correction factors computed at base station are handed over to rover to compute its own exact coordinates for respective epochs. For simulation purpose, the computation is repeated for 800 continuous epochs (13 minutes and 20 seconds). For the sake of comparison, rover coordinates are determined with and without utilizing the correction factors. Results are plotted in Figures 7.4 and 7.5. Here x-axis (zero vertical position) corresponds to true coordinates of rover. Mean positional error from true position without and with DGPS is 18.73m and 0.78m respectively. Significant improvement in position accuracy is observed.

To appreciate the results in 3D view, corresponding plot for one epoch is shown in Figure 7.6. It shows the relative positions of true coordinates, coordinates determined without DGPS and the coordinates determined with DGPS. Statistical results for the simulation are placed in Table 7.2.

TABLE 7.2: Statistical Results.

Table adapted from Ref. [6].

	X (cm)	Y (cm)	Z (cm)	Norm (cm)
Minimum Error	0.01	0.1	0.1	5.3
Maximum Error	214.4	93.4	193.0	258.9
Mean Error	55.1	23.4	36.6	77.8

Now we calculate GDOP, PDOP and TDOP values at rover for 800 epochs with an elevation mask of 10° using our code. The results are shown in Figure 7.7. In order to examine the effect of number of SVs on PDOP values, we compute PDOP values under two different elevation mask values so that different numbers of SVs become visible.

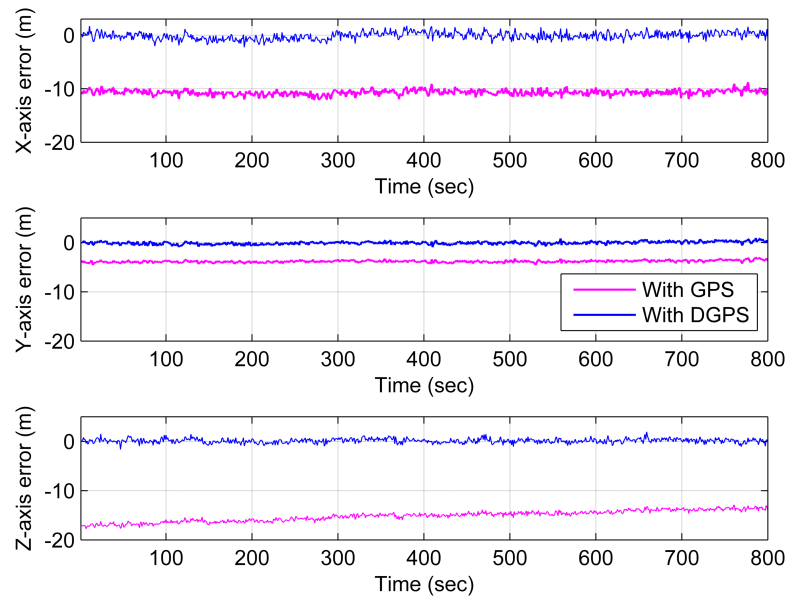


FIGURE 7.4: Position error with and without DGPS in three-axes.

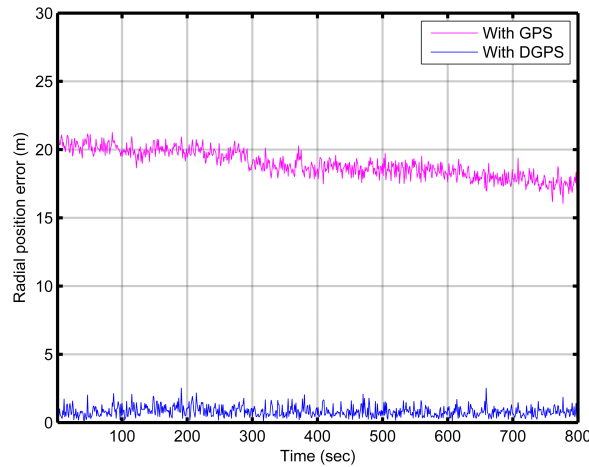


FIGURE 7.5: Radial position error with and without DGPS.

First, we set elevation mask to 30° and then to 5° . Corresponding plots are shown in Figure 7.8 and Figure 7.9 respectively. We notice that number of visible satellites increase with decrease in elevation mask, and correspondingly PDOP values decrease and hence the smaller the solution error. DOP values increase with increases in the elevation mask angle. DOP value multiplied by measurement and other input errors, provides the position error, some component of position error, or time error [132]. This means that when the DOP value doubles, the positional error increases by a factor of two. Because various DOPs are only functions of receiver and satellite coordinates, they may be predicted ahead of time for any given set of satellites in view from a specified location using a satellite almanac [133].

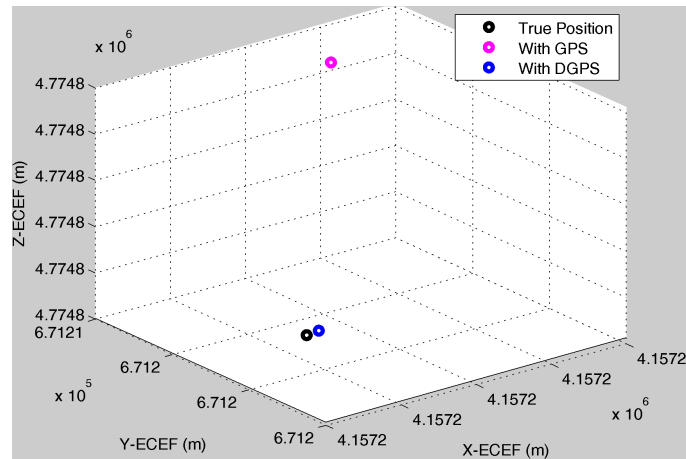


FIGURE 7.6: Relative position of rover for one epoch.

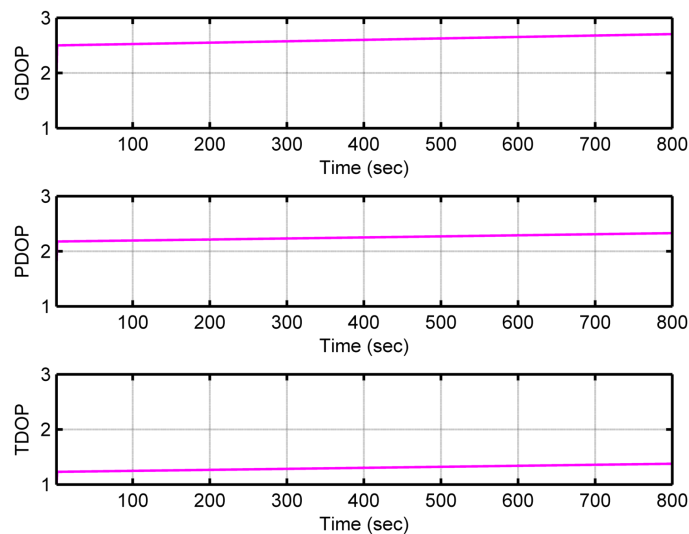
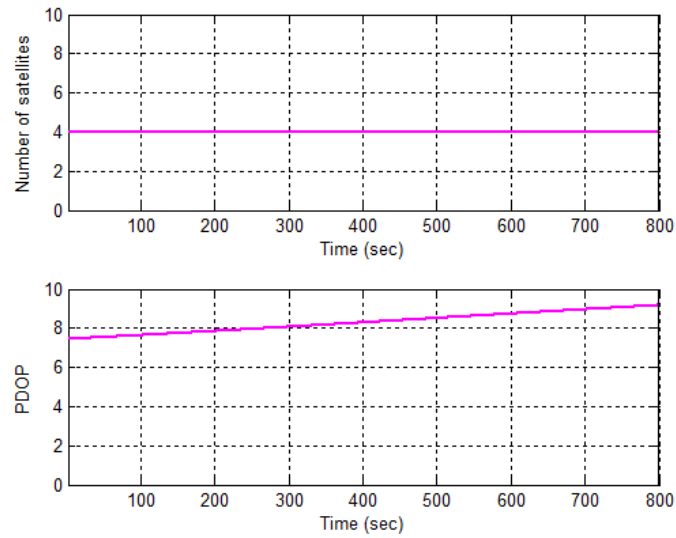
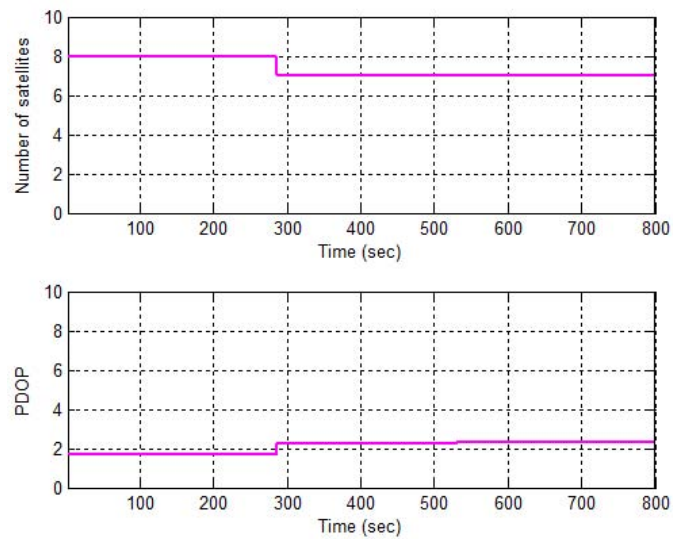


FIGURE 7.7: DOP values at rover.

7.4.3 Reasons for not Using Double Differences

For GNSS real time and post mission applications, the receiver data from the base station may be combined with the data from the other receiver to form double-differenced observations, which is used for baseline vector determination. We have not used double differences for this study for the following reasons:

1. For double differences, we need to transmit the complete raw data from rover to the base station that is voluminous information consuming larger bandwidth. After computing the double differences at base station, we get the baseline information. With this information, we can obtain the rover true information that can be transmitted to the rover. This strategy has two considerations; first it requires more bandwidth and second it necessitates two way transmission of information.

FIGURE 7.8: Visible satellites and PDOP for 30° elevation mask.FIGURE 7.9: Visible satellites and PDOP for 5° elevation mask.

2. Errors due to receiver noise and multipath are amplified up to a factor of two, in the worst case [134].
3. Notations also become cluttered in case of double difference.

It may be mentioned that double difference technique is suitable where rover true position information is required at base station, while DGPS technique with correction factors is appropriate in scenarios where rover true position information is required at the rover.

7.4.4 Reasons for not Using Dual Frequency

High end professional receivers exploit both $L1$ and $L2$ frequency while low cost GPS receivers rely on single frequency $L1$ only. Encrypted P(Y) code is transmitted on both $L1$ and $L2$, however access to P-code is provided in Precise Position Service (PPS) that is designed primarily for authorized users [135]. Without decryption keys, it is still possible to use a codeless technique to compare the P(Y) codes on $L1$ and $L2$ to gain much of the same error information. However, this technique is slow, so it is currently available only on specialized surveying equipment. Also it is lot more expensive to build, mostly because there is no high-volume consumer market for these chips [136]. This technique is appropriate for geodetic applications but not for navigation purposes. As ionospheric effects can be cancelled in a better fashion with dual frequency receivers, so their performance is better than single frequency receivers. However following considerations are made for use of dual frequency receivers on small platforms like MUAVs:

1. P-code on $L2$ frequency is encrypted and reserved for authorized users.
2. Dual frequency receivers are quite expensive, at least of the order of 40 than single frequency receivers.
3. Dual frequency receivers necessitate dual band antenna.
4. For real time DGPS applications using double differences, dual frequency raw data would require more bandwidth compared with single frequency raw data.

Most of the other codes deal with dual frequency receivers, while this code is meant for single frequency receiver ($L1$ only), that makes this study useful for DGPS implementation with low cost and tiny GPS receivers like u-blox and Skytraq etc. However feasibility of using $L2$ frequency for better performance may be explored while exploiting new $L2C$ signals (free from encryption) being transmitted by GPS satellites block IIR-M.

7.4.5 Technical Constraints

This software may be used in situations where true coordinates of rover are of interest for post-mission analysis, e.g. geo-referencing for aerial photography. GPS binary (or proprietary format) data recorded at base station and rover may be later converted to RINEX format and used in conjunction with this code, provided the following conditions are met:

1. RINEX observation files are version 2.11.
2. GPS observables are in the sequence of $C1 L1 D1 S1$ in RINEX observation files.

3. First epoch is identical for RINEX observation files generated from the data recorded at two sites.

For points (1) and (2), measures can be taken to conform to RINEX version and order of observables while converting GPS binary (or proprietary format) data to RINEX format. Open source software rtklib version 2.4.2 allows to convert GPS binary data directly to RINEX version 2.11. Also RINEX files of other version (2.10/3.00 etc.) may be converted to desired format (RINEX 2.11) using this software [95]. So RINEX 2.11 is not really a limitation of the code. For point (3), appropriate changes may be made to RINEX observation files if first epoch is not same in both the files. Start Time and End Time can also be selected using rtklib to include only the time of interest in resultant RINEX files.

7.5 Summary

In this study, DGPS positioning for rover is accomplished in offline mode using RINEX files. Correction factors have been exploited instead of computing double differences. All computations have been performed in MATLAB using C/A code and L1 frequency only to make the code compatible with low-cost GPS receivers. Almost all error factors which are common to base station and rover (in case of short baseline) are cancelled out. The residual errors are due to multi-path reflections and receiver noise. It is an easy to use tool and is quite flexible for processing of RINEX files. The code can reliably be used to improve accuracy of the rover position in DGPS mode for the scenarios where rover true coordinates are valuable, e.g. accurate geo-referencing in case of aerial photography.

Chapter 8

Conclusion

MUAVs have replaced manned aircraft in many fields and are even capable to perform novel assignments which cannot be performed by manned platforms. Utilization of MUAVs is expected to rise steadily particularly for remote sensing missions while exploiting emerging technologies. Their utilization is foreseen ranging from emergency situations handling, e.g. fire fighting and volcanoes monitoring etc., to routine tasks like postal and parcel services, film shooting, patrolling and wild life survey etc. Incorporation of a robotic arm to a MUAV will enhance their applications to further extent. There is still a lot to explore for this fascinating field. It definitely has the potential to serve the humanity in an even better and advanced fashion. Unmanned is therefore unmatched.

An elementary real-time formation flying test set-up for 3DOF has been conceived at Institute of Aerospace Information Technology, University of Würzburg. Proposed approach appears to be useful for further research projects as well as for education purpose. It served as a basis for realization of synchronized attitude of two quadcopters in real-time.

For centralized formation flying we are interested for followers to track varying output of leader in order to smoothly maintain relative 3D distances. For LQR PI control scheme based on model following, extensive simulations were realized under a number and types of disturbances including input disturbance on control values, output disturbances (e.g. a wind gust) and communication delays. It revealed the follower systems to be quite robust in terms of maintaining the desired formation geometry. This technique has promising results in terms of stability and leader output tracking even in the presence of significant perturbations and under arbitrary switching formation geometries during flight. The approach is appropriate for the scenarios where tightly coupled formation flight is desired like cooperative grasping, joint load transportation etc. Proposed approach for followers in the formation, is simple from implementation point of view. Although this scheme is suitable for small formations, however appropriate arrangements may be made to extend it to medium sized formations.

As a further step, position controllers for two formations have been implemented using the control schemes LQR PI based on model following and LQR PI servomechanism, incorporating the notion of virtual leader. These control schemes are compared in terms of convergence to desired tracking values and the control effort. Model following controller behaves well in terms of transient response and exhibits no overshoot, while servomechanism reacts faster to the commands but at the cost of high control effort. These two control schemes are able to track the desired trajectory while maintaining relative 3D separations. Each formation is being controlled through respective ground station. Altitude information is shared between the two stations to ensure safe operation. Extensive simulations proved efficacy of the proposed schemes while giving promising results. Proposed architecture is scalable and can be expanded easily. Notion of virtual leader enables the proposed scheme to be robust against any node failure. Load of information (command values) from ground station to the units is quite low, as only position information is transmitted from ground station to the quadcopters. Working range of communication media needs to be considered that allows the maximum distance between ground station and the quadcopter formation.

A simplest approach for cluster reconfiguration of quadcopters has been presented that also ensures collision avoidance during this phase. However this technique is realizable when we have ample of space available. Minimum and maximum height separation limitations may be imposed in order to avoid collisions and to remain in communication zone of other vehicles respectively.

It is already established that communication topology for multi agent system plays an important role, through eigenvalues of Laplacian matrix, towards stability and performance [91]. Modeling of communication topology is facilitated through graph theory in general and Laplacian matrix in particular. In this context, Euler's formula has been generalized to make it applicable to multiple subgraphs while exploiting the eigenvalues of underlying Laplacian matrix. Application of presented results may be found in swarm of aerial vehicles, formation flight and Kirchoff's current law etc. These are envisaged to be helpful for analysis of the network of networks. It is concluded that network dynamics and control for large number of units and multiple fleets of aerial vehicles may be conveniently investigated using the notion of graph theory.

Consensus algorithms in combination with different control schemes, including LQR PI servomechanism and LQR PI based on model following, have been simulated and evaluated. Extensive simulations under different conditions validated the efficacy of proposed schemes. The schemes are simple to implement for cooperative control of agents. Proposed framework is scalable for large formations as well. However using the consensus algorithms for large formations may cause some time delay. Depending on the application, consensus algorithm may be tailored accordingly. The work presented here assumes that states of quadcopter are available. In case of non-availability of states, a suitable *observer* may be designed to estimate the states of quadcopters.

The presented control approaches for formation flights are simple from implementation point of view. However it necessitates knowing the exact dynamic model and the states

of aerial vehicles. As it is quite laborious to model all the dynamics of rotorcraft flying in close formation, adaptive and robust control techniques, like LQR PI, may be explored to their full potential to cater for such scenarios. Advanced designs may be realized while introducing the network dynamics that is a new area of research.

Accurate positions of agents in a fleet are of paramount importance when the agents are operating in near vicinity and collisions are naturally not desired. Therefore, a part of this thesis has been dedicated to implementation of DGPS. For this purpose, RINEX files have been exploited for DGPS positioning of rover. Correction factors have been exploited instead of computing double differences due to the reasons highlighted in subsection 7.4.3 of Chapter 7. All computations have been performed in MATLAB using C/A code and L1 frequency only to make the code compatible with low-cost GPS receivers. Almost all error factors which are common to base station and rover (in case of short baseline) are cancelled out. The residual errors are due to multi-path reflections and receiver noise. It is an easy to use tool and is quite flexible for processing of RINEX files. The code can reliably be used to improve the accuracy of the rover position in DGPS mode for the scenarios where rover true coordinates are valuable like in case of aerial photography for accurate geo-referencing.

8.1 Future Works

The formation flying test setup can be extended to multiple quadcopters making synchronized motions in 3DOF (roll / pitch / yaw). As it is possible for quadcopter on rod to gain height up to a certain level, so with the addition of a height sensor the setup may even be extended for fourth degree of freedom. Existing ground station for controlling one quadcopter may also be extended for multiple quadcopters. Transmitted data integrity algorithm may also be improved.

For future works, damping effects may also be incorporated in the simulations of quadcopter dynamic model to correspond to real world scenarios. Presented control schemes may be implemented in real-time using the formation flying test setup developed at Institute of Aerospace Information Technology, University of Würzburg [2]. A suitable collision avoidance mechanism may also be implemented for safe operations.

In order to enhance the usage of DGPS, some improvements are suggested; for e.g. the code may be modified to read and process Galileo GNSS data and its performance may be compared versus GPS. It is envisaged that easy availability of Galileo would play an important role in the development of many systems. Code may be extended to read other versions of RINEX too (e.g. version 2.10 and 3.00) and may be improved to read the RINEX files with diverse sequence of GPS observables. Carrier phase measurements may also be used to improve navigation solution.

DGPS services are either to be hired through some agency providing correction signals or we need to set-up an indigenous reference station where correction factors are

computed and then sent to quadcopter. Prior knowledge of reference station exact coordinates is mandatory that restricts the use of indigenous reference station in the field. It is proposed to undertake a feasibility study for setting up an indigenous local reference station in the field to provide DGPS correction factors exploiting GPS raw data. Following DGPS techniques / methodologies are proposed as potential candidates for future study in this regard:

1. Using software *rtklib*.
2. Using SBAS services for quadcopter and local reference station.
3. Using a combination of SBAS + PPP services for local reference station.

A study may be undertaken to implement and compare these methods in terms of feasibility, accuracy, time and effort.

8.2 Areas Requiring Attention

In case of MUAVs, we are to restrict ourselves to the sensors and hardware which have small volume, weight and power consumption. For some applications, real time on-board voluminous data processing may be required, e.g. PMD camera, that necessitates more powerful on-board processors. There is still a lot of room for improvement in the domain of more sophisticated algorithms for distributed task allocation and autonomous decision making by the agents keeping in view the computational intensity. Limited bandwidth of communication data link is another limitation that poses challenges for real-time implementation of some techniques like GPS carrier-phase double-difference and live video streaming from MUAV. Robust algorithms are required to cater for communication packets loss.

Autonomy is still a challenging field of research, when it comes to substitute external positioning systems with on-board sensors, which suffer from low accuracy because of limited resources in terms of size, weight, power, battery capacity and data processing. However these problems are unavoidable when a MUAV needs to fly autonomously in an unknown, GPS-denied and adverse environments (e.g. smoke). This is an up to date field of research, as MUAVs are being designed to help fire fighters inside burning buildings. Docking for air refuelling by autonomous UAVs is still an open topic for future research. MUAVs potential for agriculture applications may be quite interesting for agricultural sectors, for example spraying the insecticides to prevent the humans from toxicity. Endurance of MUAV is limited by the life of battery so these can operate within a short distance from the launching point. Some alternatives to extend the mission duration have been suggested like automated battery changing/ charging capability for autonomous UAV beyond the life of a single UAV battery [137]. Batteries with extended

life will open the door for many other useful applications. Exploitation of solar energy [138] to enhance the MUAV endurance may also be investigated.

As it is quite laborious to model all the dynamics of VTOLS flying in close formation, adaptive and robust control techniques may be explored to their full potential to cater for such scenarios. Robust control techniques will not only cater for vast variations in parameter changes but also the measurement noise. These control techniques may also be quite valuable to control swarm behaviour. Recent research is also aimed towards the uncertainties associated with quadcopter dynamics. Collision avoidance for a swarm of MUAVs is still an area that requires attention. In this context, velocity matching and flock joining algorithms may be improved. Fault tolerant control methods may further increase the integrity and reliability of MUAVs. Convertible MUAVs are envisaged to be another interesting area for researchers thereby combining the merits of fixed wings as well as VTOLS. Innovative air vehicle structures may be proposed and developed for this purpose. Flapping wing MUAVs while exploiting synthetic muscles is another domain of research to mimic the birds and to perform novel manoeuvres. Multi-discipline research may also be undertaken to develop biological-inspired systems while learning from nature, for example optic flow vision of flying insects. Low Reynolds number aerodynamics for small fixed wing airplanes may also be focussed.

Appendix A

Matrices Used for Simulations

Desired poles for leader = $[-1.9054+1.6368i, -1.9054-1.6368i, -0.2350+0.0835i, -0.2350-0.0835i, -0.1737+0.0720i, -0.1737-0.0720i, -1.9054+1.6368i, -1.9054-1.6368i, -0.2350+0.0835i, -0.2350-0.0835i, -3.1214+0.0000i, -0.5065+0.0000i]$;

Q matrix (for LQR PI) = $\text{diag}([4000, 4000, 5000, 4000, 1, 4000, 1, 5000, 20, 0.25, 1, 1000, 50, 1000, 50, 4000, 1, 4000, 1, 5000, 20, 0.25, 1, 1000, 50, 1000, 50])$;

R matrix (for LQR PI) = $\text{diag}([100, 0.1, 25, 25])$;

Gain matrix for leader (using pole placement method) =

$$\begin{bmatrix} 0.0557 & 0.1602 & 0.1641 & 0.4351 & 0.5483 & 2.5271 & 0.3093 & 1.9635 & 0.6765 & -0.2422 & -0.2322 & 0.1653 \\ -0.0172 & 0.4065 & 0.1405 & 0.3571 & 0.3443 & 1.1077 & 0.4117 & 2.4180 & -1.3149 & -0.4698 & -0.2773 & 0.1314 \\ -0.0381 & -0.3742 & -0.0096 & -0.0279 & -0.0248 & -0.0458 & -0.0387 & -0.1789 & 8.4567 & 4.4017 & -0.1616 & -0.1089 \\ 0.0008 & 0.0591 & 0.0559 & 0.3689 & 0.0445 & 0.1450 & 0.0472 & 0.2508 & -0.3756 & -0.1682 & 8.4043 & 4.4536 \end{bmatrix}$$

Gain matrix for follower (using LQR PI control scheme),

$$\begin{bmatrix} 0 & 0 & 7.07 & 0 & 0 & 0 & 0 & 10.06 & 3.62 & 0 & 0 & 0 & 0 & 0 & 0 & -0.10 & -0.17 & 0.19 & 0.61 & -9.18 & -5.07 & 0.17 & 1.15 & 2.08 & 0.42 & 2.15 & 0.44 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.58 & 3.63 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -12.65 & 0 & 0 & -22.73 & -14.1 & 0 & 0 & 0 & 0 & 0 & 48.41 & 9.94 & 0 & 0 & 22.13 & 20.71 & -0.02 & -0.03 & -0.1 & 0.1 & -0.24 & -0.89 & -82.26 & -15.49 & -0.24 & -0.34 \\ 0 & 12.65 & 0 & 0 & 0 & 22.73 & 14.1 & 0 & 0 & 0 & 0 & 0 & 0 & 48.41 & 9.94 & 0.02 & -22.06 & -20.06 & -20.85 & 0.17 & 0.57 & 0.21 & 1.0 & -1.48 & -0.66 & -83.16 & -15.47 \end{bmatrix}$$

Bibliography

- [1] Qasim Ali, Nils Gageik, and Sergio Montenegro. A review on distributed control of cooperating mini UAVs. *International Journal of Artificial Intelligence & Applications*, 5(4):1–13, 2014.
- [2] Qasim Ali and Sergio Montenegro. A simple approach to quadcopter formation flying test setup for education and development. In *9th International Technology, Education and Development Conference*, pages 2776–2784, Madrid, Spain, 2015.
- [3] Qasim Ali and Sergio Montenegro. Explicit model following distributed control scheme for formation flying of mini UAVs. *IEEE Access Journal*, 4:397–406, 2016.
- [4] Qasim Ali and Sergio Montenegro. Decentralized control for scalable quadcopter formations. *International Journal of Aerospace Engineering*, 2016 (Accepted).
- [5] Qasim Ali and Sergio Montenegro. Role of graphs for multi-agent systems and generalization of Euler’s formula. In *8th IEEE International Conference on Intelligent Systems*, Sofia, Bulgaria, 2016 (Accepted).
- [6] Qasim Ali and Sergio Montenegro. A Matlab implementation of differential GPS for low-cost GPS receivers. *TransNav: International Journal on Marine Navigation and Safety of Sea Transportation*, 8(3):343–350, 2014.
- [7] Tal Shima and Steven J Rasmussen. *UAV cooperative decision and control: challenges and practical approaches*, volume 18. SIAM, 2009.
- [8] By Randal W Beard, Timothy W McLain, Derek B Nelson, Derek Kingston, and David Johanson. Decentralized cooperative aerial surveillance using fixed-wing miniature UAVs. *Proceedings of the IEEE*, 94(7):1306–1324, 2006.
- [9] Vijay Kumar and Nathan Michael. Opportunities and challenges with autonomous micro aerial vehicles. *The International Journal of Robotics Research*, 31(11):1279–1291, 2012.
- [10] Guadalupe Flores and Rogelio Lozano. Transition flight control of the quad-tilting rotor convertible MAV. In *Unmanned Aircraft Systems (ICUAS), 2013 International Conference on*, pages 789–794. IEEE, 2013.

- [11] Zhou Chao, Shao-Lei Zhou, Lei Ming, and Wen-Guang Zhang. UAV formation flight based on nonlinear model predictive control. *Mathematical Problems in Engineering*, 2012, 2012.
- [12] Wei Ren, Randal W Beard, and Timothy W McLain. Coordination variables and consensus building in multiple vehicle systems. In *Cooperative Control*, pages 171–188. Springer, 2005.
- [13] Joongbo Seo, Chaeik Ahn, and Youdan Kim. Controller design for UAV formation flight using consensus based decentralized approach. In *Proceedings of AIAA Infotech@ Aerospace Conference, AIAA Press*, pages 1–11, 2009.
- [14] Brian L Stevens and Frank L Lewis. *Aircraft control and simulation*. John Wiley & Sons, 2003.
- [15] Stefania Tonetti, Markus Hehn, Sergei Lupashin, and Raffaello D’Andrea. Distributed control of antenna array with formation of UAVs. In *World Congress*, volume 18, pages 7848–7853, 2011.
- [16] Uwe Kiencke, Lars Nielsen, Robert Sutton, Klaus Schilling, Markos Papageorgiou, and Hajime Asama. The impact of automatic control on recent developments in transportation and vehicle systems. *Annual reviews in control*, 30(1):81–89, 2006.
- [17] Christopher Lehnert and Peter Corke. μ AV-design and implementation of an open source micro quadrotor. *AC on Robotics and Automation, Eds*, 2013.
- [18] Ole Falkenberg, Jonas Witt, Ulf Pilz, Uwe Weltin, and Herbert Werner. Model identification and H_∞ attitude control for quadrotor MAVs. In *Intelligent Robotics and Applications*, pages 460–471. Springer, 2012.
- [19] Nils Gageik, Thilo Müller, and Sergio Montenegro. Obstacle detection and collision avoidance using ultrasonic distance sensors for an autonomous quadrocopter. *University of Würzburg, Aerospace Information Technology (Germany), Würzburg September*, 2012.
- [20] Allistair Moses, Matthew J Rutherford, and Kimon P Valavanis. Radar-based detection and identification for miniature air vehicles. In *Control Applications (CCA), 2011 IEEE International Conference on*, pages 933–940. IEEE, 2011.
- [21] Javier Moyano Cano. Quadrotor UAV for wind profile characterization. 2013.
- [22] Nils Gageik, Julian Rothe, and Sergio Montenegro. Data fusion principles for height control and autonomous landing of a quadrocopter, 2012.
- [23] HaiYang Chao, YongCan Cao, and YangQuan Chen. Autopilots for small unmanned aerial vehicles: a survey. *International Journal of Control, Automation and Systems*, 8(1):36–44, 2010.

- [24] L Vaillon, B Taffet, J Degeselle, L Giulicchi, and A Pasetti. Attitude determination using GPS: Multipath reduction through GPS antenna design. In *Spacecraft Guidance, Navigation and Control Systems*, volume 425, page 355, 2000.
- [25] E Glenn Lightsey and Jared Madsen. Three-axis attitude determination using global positioning system signal strength measurements. *Journal of guidance, control, and dynamics*, 26(2):304–310, 2003.
- [26] Rui Li, Yuanyuan Jiao, Yong Li, and Chris Rizos. Simulation platform for relative navigation using GPS carrier phase measurements for satellite formation flying missions. In *IGNSS Symposium*, pages 15–17. Citeseer, 2011.
- [27] Eric A Olsen, CHAN-WOO PARK, and Jonathan P How. 3D formation flight using differential carrier-phase GPS sensors. *Navigation*, 46(1):35–48, 1999.
- [28] Wen Zhang, Mounir Ghogho, and L Enrique Aguado. GPS short-distance baseline estimation from RINEX files under Matlab environment. In *Proc. of the 13th IAIN World Congress and Exhibition*, pages 1–10, 2009.
- [29] David Potere. Horizontal positional accuracy of Google Earth’s high-resolution imagery archive. *Sensors*, 8(12):7973–7981, 2008.
- [30] Eric Alan Olsen. *GPS sensing for formation flying vehicles*. Stanford University, 2000.
- [31] Navipedia. Precise Point Positioning, 2015. URL http://www.navipedia.net/index.php?title=Precise_Point_Positioning&oldid=13291.
- [32] Navipedia. EGNOS performances, 2015. URL http://www.navipedia.net/index.php?title=EGNOS_Performances&oldid=13243. [Online; accessed 7-September-2015].
- [33] U-blox, . URL <http://www.u-blox.com/en>. [Online; accessed 7-September-2015].
- [34] Jinling Wang, Matthew Garratt, Andrew Lambert, Jack Jianguo Wang, S Han, and David Sinclair. Integration of GPS/INS/vision sensors to navigate unmanned aerial vehicles. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 37:963–970, 2008.
- [35] Andrew Robertson Tobe’Corazzini, John Carl Adams, Arash Hassibi, Jonathan P How, Andrew Robertson, and Jonathan How. GPS sensing for spacecraft formation flying, 1997.
- [36] Rudolf Schwarte, B Buxbaum, H Heinol, Z Xu, J Schulte, H Riedel, P Steiner, M Scherer, B Schneider, and T Ringbeck. New powerful sensory tool in automotive safety systems based on PMD-technology. In *Advanced Microsystems for Automotive Applications 2000*, pages 181–203. Springer, 2000.

- [37] Nils Gageik, Michael Strohmeier, and Sergio Montenegro. Waypoint flight parameter comparison of an autonomous UAV. *International Journal of Artificial Intelligence & Applications*, 4(3):39, 2013.
- [38] Shahid Mahmood. Unmanned Aerial Vehicle (UAV) communications. 2007.
- [39] Andrey Popov and Herbert Werner. Robust stability of a multi-agent system under arbitrary and time-varying communication topologies and communication delays. *IEEE transactions on automatic control*, 57(9):2343–2347, 2012.
- [40] Nils Gageik, Atheel Redah, and Sergio Montenegro. Avionics control systems for education and development. *INTED2012 Proceedings*, pages 935–942, 2012.
- [41] Belal H Sababha, Hong Chul Yang, and Osamah A Rawashdeh. An RTOS-based run-time reconfigurable avionics system for UAVs. *AIAA Infotech@ Aerospace*, 2010.
- [42] S Craciunas, C Kirsch, Harald Röck, and Rainer Trummer. The JAviator: A high-payload quadrotor UAV with high-level programming capabilities. *Proc. GNC*, 2008.
- [43] Joakim Tosteberg and Thomas Axelsson. Development of a wireless video transfer system for remote control of a lightweight UAV. 2012.
- [44] Sergio Montenegro. Network centric core avionics. In *2009 First International Conference on Advances in Satellite and Space Communications*, pages 197–201. IEEE, 2009.
- [45] T Takasu. RTKLIB: An open source program package for GNSS positioning, 2011.
- [46] B Wisniewski, K Bruniecki, and M Moszynski. Evaluation of RTKLIBs positioning accuracy using low-cost GNSS receiver and ASG-EUPOS. *TransNav, the International Journal on Marine Navigation and Safety of Sea Transportation*, 7(2):79–85, 2013.
- [47] Tomoji Takasu and Akio Yasuda. Development of the low-cost RTK-GPS receiver with an open source program package RTKLIB. In *international symposium on GPS/GNSS*, pages 4–6. International Convention Centre Jeju, Korea, 2009.
- [48] J Gordon Leishman. The breguet-richet quad-rotor helicopter of 1907. *Vertiflite*, v. 47, no. 3 (Summer 2001), p. 58-60: ill, 2001.
- [49] Geno Wagner, Dave Jacques, William Blake, and Meir Pachter. Flight test results of close formation flight for fuel savings, 2002.
- [50] Chan-Woo Park. *Precise relative navigation using augmented CDGPS*. PhD thesis, Citeseer, 2001.

- [51] Sanghyuk Park. *Avionics and control system development for mid-air rendezvous of two unmanned aerial vehicles*. PhD thesis, Draper Laboratory, 2003.
- [52] Steven M Ross. Formation flight control for aerial refueling. Technical report, DTIC Document, 2006.
- [53] Koushil Sreenath, Taeyoung Lee, and Vipin Kumar. Geometric control and differential flatness of a quadrotor UAV with a cable-suspended load. In *Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on*, pages 2269–2274. IEEE, 2013.
- [54] Nathan Michael, Daniel Mellinger, Quentin Lindsey, and Vijay Kumar. The GRASP multiple micro-UAV testbed. *Robotics & Automation Magazine, IEEE*, 17(3):56–65, 2010.
- [55] Daniel Mellinger, Michael Shomin, Nathan Michael, and Vijay Kumar. Cooperative grasping and transport using multiple quadrotors. In *Distributed autonomous robotic systems*, pages 545–558. Springer, 2013.
- [56] Mark Müller, Sergei Lupashin, and Raffaello D’Andrea. Quadrocopter ball juggling. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 5113–5120. IEEE, 2011.
- [57] Mark Cutler and Jonathan P How. Actuator constrained trajectory generation and control for variable-pitch quadrotors. In *AIAA Guidance, Navigation, and Control Conference (GNC)*, 2012.
- [58] H Van Dyke Parunak, Sven Brueckner, and James J Odell. Swarming coordination of multiple UAVs for collaborative sensing. In *Proceedings of Second AIAA” Unmanned Unlimited” Systems, Technologies, and Operations Conference*, 2003.
- [59] Michael Angermann, Martin Frassl, and Michael Lichtenstern. Autonomous formation flying of micro aerial vehicles for communication relay chains. *San Diego, CA*, 2011.
- [60] JS Ardaens and S DAMico. Formation flying testbed. *DLR-GSOC TN*, pages 09–01, 2009.
- [61] Jonathan How, Ellis King, and Yoshiaki Kuwata. Flight demonstrations of cooperative control for UAV teams. In *AIAA 3rd unmanned unlimited technical conference, workshop and exhibit*, 2004.
- [62] Gabe Hoffmann, Dev Gorur Rajnarayan, Steven L Waslander, David Dostal, Jung Soon Jang, and Claire J Tomlin. The stanford testbed of autonomous rotorcraft for multi agent control (STARMAC). In *Digital Avionics Systems Conference, 2004. DASC 04. The 23rd*, volume 2, pages 12–E. IEEE, 2004.
- [63] Timothy W McLain and Randal W Beard. Unmanned air vehicle testbed for cooperative control experiments. 2004.

- [64] Long Di, Haiyang Chao, Jinlu Han, and YangQuan Chen. Cognitive multi-UAV formation flight: principle, low-cost UAV testbed, controller tuning and experiments. In *ASME 2011 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, pages 917–927. American Society of Mechanical Engineers, 2011.
- [65] Lars Cremean, William B Dunbar, Dave Van Gogh, Jason Hickey, Eric Klavins, Jason Meltzer, and Richard M Murray. The Caltech multi-vehicle wireless testbed. In *Decision and Control, 2002, Proceedings of the 41st IEEE Conference on*, volume 1, pages 86–88. IEEE, 2002.
- [66] Lucas M Argentim, Willian C Rezende, Paulo E Santos, Renato Aguiar, et al. PID, LQR and LQR-PID on a quadcopter platform. In *Informatics, Electronics & Vision (ICIEV), 2013 International Conference on*, pages 1–6. IEEE, 2013.
- [67] Arthur Richards and Jonathan How. Decentralized model predictive control of cooperating UAVs. In *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, volume 4, pages 4286–4291. IEEE, 2004.
- [68] Ulf Pilz and Herbert Werner. An H_∞/l_1 approach to cooperative control of multi-agent systems. In *Decision and Control (CDC), 2012 IEEE 51st Annual Conference on*, pages 5930–5935. IEEE, 2012.
- [69] Jose Alfredo Guerrero, Isabelle Fantoni, Sergio Salazar, and Rogelio Lozano. Flight formation of multiple mini rotorcraft via coordination control. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 620–625. IEEE, 2010.
- [70] Dohyeon Kim and Youdan Kim. Optimized feedforward design scheme unifying regulator and command generator tracker. *Journal of guidance, control, and dynamics*, 19(4):899–904, 1996.
- [71] Kenneth M Sobel and Elizer Y Shapiro. A design methodology for pitch pointing flight control systems. *Journal of Guidance, Control, and Dynamics*, 8(2):181–187, 1985.
- [72] Shuang Li and Yuming Peng. Command generator tracker based direct model reference adaptive tracking guidance for Mars atmospheric entry. *Advances in Space Research*, 49(1):49–63, 2012.
- [73] Biao Wang, Xiangxu Dong, Ben M Chen, Tong H Lee, and Swee King Phang. Formation flight of unmanned rotorcraft based on robust and perfect tracking approach. In *American Control Conference (ACC), 2012*, pages 3284–3290. IEEE, 2012.
- [74] Matthew Turpin, Nathan Michael, and Vijay Kumar. Decentralized formation control with variable shapes for aerial robots. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 23–30. IEEE, 2012.

- [75] Zdzisław Gosiewski and Leszek Ambroziak. Formation flight control scheme for unmanned aerial vehicles. In *Robot Motion and Control 2011*, pages 331–340. Springer, 2012.
- [76] Reza Olfati-Saber and Richard M Murray. Distributed cooperative control of multiple vehicle formations using structural potential functions. In *IFAC World Congress*, volume 15, pages 242–248. Citeseer, 2002.
- [77] Airlie Chapman and Mehran Mesbahi. UAV swarms: Models and effective interfaces. In *Handbook of Unmanned Aerial Vehicles*, pages 1987–2019. Springer, 2015.
- [78] Simone Martini, Magnus Egerstedt, and Antonio Bicchi. Controllability decompositions of networked systems through quotient graphs. In *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*, pages 5244–5249. IEEE, 2008.
- [79] Sean H Breheny, Raffaello D’Andrea, and Jeremy C Miller. Using airborne vehicle-based antenna arrays to improve communications with UAV clusters. In *Decision and Control, 2003. 42nd IEEE Conference on*, volume 4, pages 4158–4162. IEEE, 2003.
- [80] Marcus Bartels and Herbert Werner. Cooperative and consensus-based approaches to formation control of autonomous vehicles. In *World Congress*, volume 19, pages 8079–8084, 2014.
- [81] Alberto Bemporad and Claudio Rocchi. Decentralized hybrid model predictive control of a formation of unmanned aerial vehicles. In *Proc. 18th IFAC World Congress, Milano, Italy*, pages 11900–11906, 2011.
- [82] Reza Olfati-Saber, Alex Fax, and Richard M Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233, 2007.
- [83] Bogdan Niemoczynski, Santosh Biswas, James Kollmer, and Frank Ferrese. Hovering synchronization of a fleet of quadcopters. In *Resilient Control Systems (IS-RCS), 2014 7th International Symposium on*, pages 1–5. IEEE, 2014.
- [84] Oetomo Oetomo and Bambang Trilaksono Riyanto. Distributed formation control using consensus for multiple humanoid robots. *Journal of Instrumentation, Automation and Systems*, 1(3):78–83, 2014.
- [85] Xiwang Dong, Bocheng Yu, Zongying Shi, and Yisheng Zhong. Time-varying formation control for unmanned aerial vehicles: theories and applications. *IEEE Transactions on Control Systems Technology*, 23(1):340–348, 2015.
- [86] Gerald Alexanderson. About the cover: Euler and königsbergs bridges: A historical view. *Bulletin of the american mathematical society*, 43(4):567–573, 2006.

-
- [87] Andrey P Popov and Herbert Werner. A robust control approach to formation control. In *Control Conference (ECC), 2009 European*, pages 4428–4433. IEEE, 2009.
- [88] Reza Olfati-Saber and Richard M Murray. Consensus problems in networks of agents with switching topology and time-delays. *Automatic Control, IEEE Transactions on*, 49(9):1520–1533, 2004.
- [89] Phillip R Chandler, Meir Pachter, Dharba Swaroop, Jeffrey M Fowler, Jason K Howlett, Steven Rasmussen, Corey Schumacher, and Kendall Nygard. Complexity in UAV cooperative control. In *American Control Conference, 2002. Proceedings of the 2002*, volume 3, pages 1831–1836. IEEE, 2002.
- [90] Louis M Pecora and Thomas L Carroll. Master stability functions for synchronized coupled systems. *Physical Review Letters*, 80(10):2109, 1998.
- [91] J Alexander Fax and Richard M Murray. Information flow and cooperative control of vehicle formations. *Automatic Control, IEEE Transactions on*, 49(9):1465–1476, 2004.
- [92] Yoonsoo Kim and Mehran Mesbahi. On maximizing the second smallest eigenvalue of a state-dependent graph laplacian. *Automatic Control, IEEE Transactions on*, 51(1):116–120, 2006.
- [93] Arpita Ghosh and Stephen Boyd. Growing well-connected graphs. In *Decision and Control, 2006 45th IEEE Conference on*, pages 6605–6611. IEEE, 2006.
- [94] Chai Wah Wu. Synchronization in networks of nonlinear dynamical systems coupled via a directed graph. *Nonlinearity*, 18(3):1057, 2005.
- [95] T Takasu. RTKLIB ver. 2.4. 2 manual. 2013 ed. *Tokyo University of Marine Science and Technology*, 2013.
- [96] Kai Borre. The GPS Easy Suite–Matlab code for the GPS newcomer. *GPS solutions*, 7(1):47–51, 2003.
- [97] Eugenio Realini and Mirko Reguzzoni. goGPS: open source software for enhancing the accuracy of low-cost receivers by single-frequency relative kinematic positioning. *Measurement Science and technology*, 24(11):115010, 2013.
- [98] Wen Zhang, Mounir Ghogho, and L Enrique Aguado. GPS single point positioning and velocity computation from RINEX files under Matlab environment. In *13th IAIN World Congress, Stockholm, Sweden*, pages 27–30, 2009.
- [99] B Hofmann, H Lichtenegger, and J Collins. GPS theory and practice. *Wien: Springer-Verlag*, 5, 2001.
- [100] US Air Force. NAVSTAR GPS space segment/navigation user interfaces. *Interface Specification IS-GPS-200D*, US Air Force, 3, 2006.

-
- [101] Michael Gaeb. GNSS receiver, 2012. URL <http://http://www.gnssreceiver.de/>.
- [102] Atmel corporation, . URL <http://www.atmel.com/>.
- [103] IMU LSM303DLM. URL <https://www.pololu.com>. [Online; accessed 11-April-2016].
- [104] Technical reference manual, . URL <http://www.atmel.com/Images/doc32002.pdf/>.
- [105] Michael A Johnson and Mohammad H Moradi. *PID control*. Springer, 2005.
- [106] Antonio Visioli and Qingchang Zhong. *Control of integral processes with dead time*. Springer Science & Business Media, 2010.
- [107] Gabriel Hugh Elkaim, Fidelis Adhika Pradipta Lie, and Demoz Gebre-Egziabher. Principles of guidance, navigation, and control of UAVs. In *Handbook of Unmanned Aerial Vehicles*, pages 347–380. Springer, 2015.
- [108] Zhang Peng and Liu Jikai. On new UAV flight control system based on Kalman & PID. In *Intelligent Control and Information Processing (ICICIP), 2011 2nd International Conference on*, volume 2, pages 819–823. IEEE, 2011.
- [109] M Imran Rashid and Suhail Akhtar. Adaptive control of a quadrotor with unknown model parameters. In *Applied Sciences and Technology (IBCAST), 2012 9th International Bhurban Conference on*, pages 8–14. IEEE, 2012.
- [110] Pedro Castillo, Alejandro Dzul, and Rogelio Lozano. Real-time stabilization and tracking of a four-rotor mini rotorcraft. *Control Systems Technology, IEEE Transactions on*, 12(4):510–516, 2004.
- [111] David Lara, Anand Sanchez, Rogelio Lozano, and P Castillo. Real-time embedded control system for VTOL aircrafts: Application to stabilize a quad-rotor helicopter. In *Computer Aided Control System Design, 2006 IEEE International Conference on Control Applications, 2006 IEEE International Symposium on Intelligent Control, 2006 IEEE*, pages 2553–2558. IEEE, 2006.
- [112] Eugene Lavretsky. Robust and adaptive control methods for aerial vehicles. In *Handbook of Unmanned Aerial Vehicles*, pages 675–710. Springer, 2015.
- [113] Gene F Franklin, J David Powell, and Abbas Emami-Naeini. *Feedback control of dynamic systems*. Addison-Wesley Reading (Ma) etc., 1988.
- [114] Huibert Kwakernaak and Raphael Sivan. *Linear optimal control systems*, volume 1. Wiley-interscience New York, 1972.
- [115] Frank L Lewis, Hongwei Zhang, Kristian Hengster-Movric, and Abhijit Das. *Co-operative control of multi-agent systems: optimal and adaptive design approaches*. Springer Science & Business Media, 2013.

-
- [116] Miroslav Fiedler. Algebraic connectivity of graphs. *Czechoslovak mathematical journal*, 23(2):298–305, 1973.
- [117] Gilbert Strang and Wellesley-Cambridge Press. *Introduction to linear algebra*, volume 4. Wellesley-Cambridge Press Wellesley, MA, 2009.
- [118] Andries E Brouwer and Willem H Haemers. *Spectra of graphs*. Springer Science & Business Media, 2011.
- [119] Wei Ren, Randal W Beard, and Ella M Atkins. Information consensus in multi-vehicle cooperative control. *IEEE Control systems magazine*, 2(27):71–82, 2007.
- [120] Bernhard Hofmann-Wellenhof, Herbert Lichtenegger, and Elmar Wasle. *GNSS—global navigation satellite systems: GPS, GLONASS, Galileo, and more*. Springer Science & Business Media, 2007.
- [121] Elliott Kaplan and Christopher Hegarty. *Understanding GPS: principles and applications*. Artech house, 2005.
- [122] Christian Rocken, Teresa Van Hove, and Randolph Ware. Near real-time GPS sensing of atmospheric water vapor. *Geophysical research letters*, 24(24):3221–3224, 1997.
- [123] RB Langley, H Jannasch, B Peeters, and S Bisnath. The GPS broadcast orbits: an accuracy analysis. *33rd COSPAR Scientific Assembly, Warsaw, Poland*, 2000.
- [124] Neil Ashby. Relativity in the global positioning system. *Living Rev. Relativity*, 6, 2003.
- [125] International GNSS service, . URL <https://igscb.jpl.nasa.gov/>.
- [126] Alfred Kleusberg. Analytical GPS navigation solution. In *Geodesy-The Challenge of the 3rd Millennium*, pages 93–96. Springer, 2003.
- [127] Jan Leyssens. GNSS positioning for UAV applications. In *International Symposium Light Weight Unmanned Aerial Vehicle Systems and Subsystems, Oostende (Belgium), marzec*, 2009.
- [128] Richard B Langley. GPS receiver system noise. *GPS world*, 8(6):40–45, 1997.
- [129] W Gurtner. RINEX: The receiver independent exchange format version 2, 1993.
- [130] W Gurtner and L Estey. The receiver independent exchange format-version 3.00. *Central Bureau of the EUREF Permanent Network (EPN)*, 2007.
- [131] Kai Borre. Easy Suite II: easy17-visualizing satellite orbits, easy18-computing range and range rate corrections. *Inside GNSS*, 5(4):50–51, 2010.

-
- [132] JM Zogg. GPS compendium: Essentials of satellite navigation. *Internet: [http://zogg-jm.ch/Dateien/GPS Compendium \(GPS-X-02007\). pdf](http://zogg-jm.ch/Dateien/GPS%20Compendium%20(GPS-X-02007).pdf)*, [Nov. 20, 2012], 2009.
- [133] Richard B Langley. Dilution of precision. *GPS world*, 10(5):52–59, 1999.
- [134] Javier G García, PI Mercader, and Carlos H Muravchik. Use of GPS carrier phase double differences. *Latin American applied research*, 35(2):115–120, 2005.
- [135] Richard B Langley. Why is the GPS signal so complex. *GPS world*, 1(3):56–59, 1990.
- [136] Richard D Fontana, Wai Cheung, and Tom Stansell. The modernized L2 civil signal. *GPS world*, 12(9):28–35, 2001.
- [137] Tuna Toksoz. *Design and implementation of an automated battery management platform*. PhD thesis, Massachusetts Institute of Technology, 2012.
- [138] Giuseppe Della Penna, Benedetto Intrigila, Daniele Magazzeni, and Fabio Mercurio. Resource-optimal planning for an autonomous planetary vehicle. *arXiv preprint arXiv:1007.5130*, 2010.

Affidavit

I hereby confirm that my thesis entitled "Distributed Control of Cooperating Mini UAVs" is the result of my own work. I did not receive any help or support from commercial consultants. All sources and / or materials applied are listed and specified in the thesis.

Furthermore, I confirm that this thesis has not yet been submitted as part of another examination process neither in identical nor in similar form.

Würzburg, 29.07.2016
Place, Date

Signature

Eidesstattliche Erklärung

Hiermit erkläre ich an Eides statt, die Dissertation "Verteilte Regelung von Kooperierenden Mini UAVs" eigenständig, d.h. insbesondere selbständig und ohne Hilfe eines kommerziellen Promotionsberaters, angefertigt und keine anderen als die von mir angegebenen Quellen und Hilfsmittel verwendet zu haben.

Ich erkläre außerdem, dass die Dissertation weder in gleicher noch in ähnlicher Form bereits in einem anderen Prüfungsverfahren vorgelegen hat.

Würzburg, 29.07.2016
Ort, Datum

Unterschrift