

8-2017

## Design, Development and Implementation of Intelligent Algorithms to Increase Autonomy of Quadrotor Unmanned Missions

Diego F. Garcia Herrera

Follow this and additional works at: <https://commons.erau.edu/edt>



Part of the [Aerospace Engineering Commons](#)

---

### Scholarly Commons Citation

Herrera, Diego F. Garcia, "Design, Development and Implementation of Intelligent Algorithms to Increase Autonomy of Quadrotor Unmanned Missions" (2017). *Dissertations and Theses*. 330.

<https://commons.erau.edu/edt/330>

This Thesis - Open Access is brought to you for free and open access by Scholarly Commons. It has been accepted for inclusion in Dissertations and Theses by an authorized administrator of Scholarly Commons. For more information, please contact [commons@erau.edu](mailto:commons@erau.edu).

DESIGN, DEVELOPMENT AND IMPLEMENTATION OF INTELLIGENT  
ALGORITHMS TO INCREASE AUTONOMY OF QUADROTOR UNMANNED  
MISSIONS

A Thesis Submitted to the Faculty of Embry-Riddle Aeronautical University

by

Diego F. Garcia Herrera

In Partial Fulfillment of the Requirements for the Degree of

Master of Science in Aerospace Engineering

Embry-Riddle Aeronautical University

Daytona Beach, Florida

August 2017

DESIGN, DEVELOPMENT AND IMPLEMENTATION OF INTELLIGENT  
ALGORITHMS TO INCREASE AUTONOMY OF QUADROTOR UNMANNED  
MISSIONS

by


Diego F. Garcia Herrera

A Thesis prepared under the direction of the candidate's committee chairman, Dr. Hever Moncayo, Department of Aerospace Engineering, and has been approved by the members of the thesis committee. It was submitted to the School of Graduate Studies and Research and was accepted in partial fulfillment of the requirements for the degree of Master of Science in Aerospace Engineering.

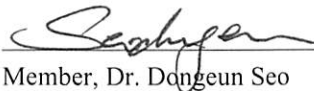
THESIS COMMITTEE



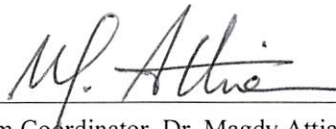
Chairman, Dr. Hever Moncayo



Member, Dr. Richard Prazenica



Member, Dr. Dongeun Seo



Graduate Program Coordinator, Dr. Magdy Attia

6.21.2017

Date



Dean of College of Engineering, Dr. Maj Mirmirani

27 June 2017

Date



Vice Chancellor, Academic Support, Dr. Christopher Grant

6/29/2017

Date

## ACKNOWLEDGMENTS

It is a pleasure for me to recognize and thank those who made this thesis possible. First and foremost I wish to extend my utmost gratitude to my mother Wilma, my father Carlos, my brother Carlos and my sister Emilia for all their motivation and unconditional support. All their hard work has been my motor of life and inspiration throughout these years; thanks to their infinite love and courage, I have become the person who I am today and none of this work would be possible without them. Special thanks to my beloved girlfriend Sayuri for her unconditional love and patience.

To Dr. Moncayo, I would like to show my gratitude for his guidance and support throughout the last three years and for giving me the opportunity of being part of the Flight Dynamics and Control Research Lab, place where I have gathered a good amount of knowledge by participating in multiple research projects. In the same way, I would like to thank to my thesis' committee members, Dr. Prazenica and Dr. Seo, for their valuable input towards improving this final manuscript.

Many thanks to all the people who I had the opportunity to work with, especially, to all the people from the FDCRL. Special thanks to Andres Perez, Chirag Jain, and Sergio Bacca for their friendship and for making this Master's degree more enjoyable. I am indebted to them for their hard work and for making this thesis possible. Last but not least, I would like to thank my roommates Javier Cisneros and Angelo Fonseca for their sincere friendship and constant support. Finally, I would like to say something in Spanish... Gracias a Dios por hacer esto posible. Gracias a mi hermosa FAMILIA, esto es para ustedes.

## TABLE OF CONTENTS

LIST OF TABLES .....	vi
LIST OF FIGURES .....	vii
SYMBOLS.....	ix
ABBREVIATIONS .....	x
ABSTRACT.....	xi
1. Introduction .....	12
1.1 Literature Review .....	17
1.1.1. Linear & Non-linear Control Architectures for Quadrotor Stabilization.....	18
1.1.2. Artificial Immune System.....	22
1.1.3. AIS as a Solution of the Failure Detection Problem .....	25
1.1.4. Vision-based Approach as an Alternative Solution for Autonomous Navigation in GPS Denied Environments.....	27
2. Quadcopter UAV Research Test-bed .....	32
2.1 Hardware Description.....	33
2.1.1 Pixhawk Autopilot.....	33
2.1.2 InvenSense MPU 6000 6-axis Accelerometer+Gyroscope .....	34
2.1.3 ST Micro LSM303D 14 bit Accelerometer / Magnetometer .....	34
2.1.4 ST Micro L3GD20H 16 bit Gyroscope.....	34
2.1.5 UBLOX LEA-6H GPS Receiver Module with Antenna.....	34
2.1.6 LightWare SF11/C (120 meter).....	35
2.1.7 Px4Flow Smart Camera .....	35
2.1.8 A 20 amp Electronic Speed Controller (ESC) with Simonk Firmware .....	36
2.1.9 Motor 850 Kv AC 2830 – 358.....	36
2.1.10 APC 10x4.7 SF Plastic Propellers .....	37
2.1.11 4S Lipo Battery .....	37
2.2 Software Description .....	38
2.2.1 Matlab/Simulink Simulation Environment.....	38
2.2.2 Support Package for Pixhawk Autopilot.....	38
2.3 Flight testing field (Daytona Beach Radio Control Association) .....	40
3. Non-linear Dynamic Inversion Control Architecture .....	41
3.1 Vehicle Dynamic Modeling .....	41
3.2 Dynamic Inversion Control Laws.....	46
3.2.1 Inner loop Dynamic Inversion .....	48
3.2.2 Outer loop Dynamic Inversion.....	51
3.3 Simulation Results .....	53
3.3.1 Autonomous Mission Results .....	54
3.4 Implementation Results .....	58
3.4.1 Attitude Controller Tuning Process .....	60

4.	AIS for Failure Detection & Identification Problem .....	68
4.1	Generation of Detectors .....	68
4.1.1	Generation of Single Data File: .....	71
4.1.2	Data Preprocessing: .....	71
4.1.3	Set Data Clustering: .....	72
4.1.4	Generation of Antibodies: .....	72
4.1.5	Detection Analysis: .....	74
4.2	Implementation Process .....	74
4.2.1	Design Constraints .....	75
4.3	Analysis of Detection Results .....	77
4.4	Fault Identification Capabilities .....	83
5.	Vision-based Alternative Solution for Autonomous Navigation in GPS Denied Environments .....	84
5.1	Optical Flow Theory .....	84
5.2	Motion Field Equations .....	85
5.2.1	Angular Rate Compensation .....	87
5.3	Implementation Process & Position Kalman Filtering Design using Optical Flow	88
5.3.1	Description of the Optical Flow Estimation Module and its Tasks .....	88
5.3.2	Position Kalman Filtering Design using Optical Flow: .....	90
5.4	Implementation Results .....	94
5.4.1	Approach Limitations: .....	97
6.	Conclusions .....	98
7.	Future Work & Recommendations .....	99
8.	References .....	100

## LIST OF TABLES

<b>Table 1</b> Main properties of the testbed (3DR X8).....	32
<b>Table 2</b> Calculated Baseline Control Gains .....	54
<b>Table 3</b> Tracking error Simulation results.....	57
<b>Table 4</b> Tracking error for the inner loop at implementation.....	59
<b>Table 5</b> Proportional Gains .....	61
<b>Table 6</b> Integral Gains .....	61
<b>Table 7</b> Tracking error for the outer loop at implementation.....	63
<b>Table 8</b> Tracking error for the outer loop at implementation (trajectory).....	66
<b>Table 9</b> Features used in AIS detection scheme.....	69
<b>Table 10.</b> List of 2D-Projections for the AIS Detection Scheme.....	70
<b>Table 11</b> Detection Performance.....	78
<b>Table 12</b> Maximum measurable velocities per specifications .....	90

## LIST OF FIGURES

<b>Figure 1</b> Cascade control configuration (Alfaro, Vilanova, Arrieta, 2009).....	20
<b>Figure 2</b> 3DR X8 Quadcopter Testbed with instrumentation .....	32
<b>Figure 3</b> Pixhawk Autopilot board.....	33
<b>Figure 4</b> MPU 6000.....	34
<b>Figure 5</b> 3DR GPS with compass module .....	35
<b>Figure 6</b> SF11/C Laser Range Finder .....	35
<b>Figure 7</b> Optical Flow sensor Px4Flow.....	36
<b>Figure 8.</b> 20 amp ESC .....	36
<b>Figure 9</b> Electric AC brushless motor.....	37
<b>Figure 10</b> APC plastic propeller.....	37
<b>Figure 11</b> 4S Lipo battery .....	37
<b>Figure 12</b> Sample blocks from Pixhawk Support package library.....	39
<b>Figure 13</b> Daytona Beach RC Association Field .....	40
<b>Figure 14</b> Mean Thrust vs PWM.....	43
<b>Figure 15</b> Mean rps vs PWM .....	43
<b>Figure 16</b> Control Architecture for fully autonomous flight.....	52
<b>Figure 17</b> 3DR-X8 Quadcopter Simulation Environment developed using Matlab/Simulink Software .....	53
<b>Figure 18</b> Inner loop-Fast mode tracking response for an autonomous mission with 5 waypoints .....	55
<b>Figure 19</b> Inner loop-Slow mode tracking response for an autonomous mission with 5 waypoints .....	56
<b>Figure 20</b> Outer loop tracking response for an autonomous mission with 5 waypoints..	57
<b>Figure 21</b> Inner loop tracking response for an autonomous mission with 5 waypoints...	59
<b>Figure 22</b> Outer loop tracking response for an autonomous mission with 5 waypoints..	63
<b>Figure 23</b> Inner loop tracking response for an autonomous trajectory .....	65
<b>Figure 24</b> Outer loop tracking response for an autonomous trajectory.....	66
<b>Figure 25</b> Raw Data Set Union Method (RDSUM) Phases .....	71
<b>Figure 26</b> Self Cluster and Antibodies generated using ENSA-RV algorithm.....	73
<b>Figure 27.</b> Trajectory Implementation for the AIS Detection Scheme Design for hovering flight.....	76
<b>Figure 28</b> Detection Rate and False Alarms for a 19 % Loss of Effectiveness in M1 ....	78



<b>Figure 29</b> Detection Rate and False Alarms for a 44 % Loss of Effectiveness in M1 ....	79
<b>Figure 30</b> Detection of Self # 42 during flight test with loss of effectiveness of 44% in M1 .....	79
<b>Figure 31</b> History of activated detectors for two flight test with loss of effectiveness of 44% and 19% in M1 .....	80
<b>Figure 32</b> Nominal validation flight.....	82
<b>Figure 33</b> Comparison between activated detectors of Self # 42 between failure at M1 and failure at M7.....	83
<b>Figure 34</b> Imaging model: perspective-central projection (Kendoul, Fantoni, Nonami, 2009) .....	85
<b>Figure 35</b> Optical-flow-based Autonomous Navigation Architecture.....	92
<b>Figure 36</b> Optical-flow based autonomous navigation for a drilling and sampling mission .....	93
<b>Figure 37</b> Comparison of Velocity Estimation in $x$ and $y$ axis .....	94
<b>Figure 38</b> KF noise reduction properties.....	95
<b>Figure 39</b> Comparison of Velocity Estimation in $z$ axis .....	95
<b>Figure 40</b> Comparison of the Position estimation in $x$ and $y$ axis.....	96

## SYMBOLS

$\boldsymbol{\omega}^b$	Angular rate vector
$a_x, a_y, a_z$	Body frame accelerations
$u, v, w$	Body frame velocities
$\xi$	Damping ratio
$\phi, \theta, \psi$	Euler angles
$\phi_{cmd}, \theta_{cmd}, \psi_{cmd}$	Euler angles command
$\dot{\phi}, \dot{\theta}, \dot{\psi}$	Euler rates
$f$	Focal length
$\bar{J}$	Inertia matrix
$\ddot{x}, \ddot{y}, \ddot{z}$	Inertia reference frame accelerations
$X, Y, Z$	Inertia reference frame positions
$\dot{x}, \dot{y}, \dot{z}$	Inertia reference frame velocities
$m$	Mass
$\omega_n$	Natural Frequency
$N_{self}$	Number of two dimensional projections
$v(\boldsymbol{x})$	Pseudo control vector
$p, q, r$	Roll rate, pitch rate, yaw rate
$p_{cmd}, q_{cmd}, r_{cmd}$	Roll rate command, pitch rate command, yaw rate command
$I_{xx}, I_{yy}, I_{zz}$	Symmetrical moments of inertia
$\vec{F}$	Vector of total forces
$\vec{M}$	Vector of total moments
$V^b$	Vector of velocities in body frame

## ABBREVIATIONS

AC	Alternating Current
AIS	Artificial Immune System
CAN	Controller Area Network
CMOS	Complementary metal-oxide semiconductor
CPU	Central Processing Unit
DBRCA	Daytona Beach Radio Control Association
DMA	Direct Memory Access
DR	Detection Rate
DC	Direct Current
ESC	Electronic Speed Controller
EKF	Extended Kalman Filter
ENSA-RV	Enhanced Negative Selection Algorithm for real-valued
ETHZ	Eidgenössische Technische Hochschule Zürich
FA	False Alarms
FAA	Federal Aviation Administration
FP	False Positive
FN	False Negative
GPS	Global Positioning System
IMU	Inertial Measurement Unit
KSC	Kennedy Space Center
LED	Light emitting diode
LQR	Linear Quadratic Regulator
MIMO	Multi-input multi-output
MSS	Multi-Self Strategy
NAS	National Airspace System
NLDI	Nonlinear Dynamic Inversion
PID	Proportional, Integral and Derivative
PWM	Pulse width modulation
RC	Radio Control
RDSUM	Raw Data Set Union Method
RTOS	Real Time Operating System
SAD	Sum of absolute differences
SLAM	Simultaneous Localization and Mapping
SPI	Serial Peripheral Interface
STARMAC	Stanford Testbed of Autonomous Rotorcraft for Multi-Agent Control
TP	True Positive
TN	True Negative
UAS	Unmanned Aerial System
UART	Universal Asynchronous Receiver/Transmitter
UAVs.	Unmanned Aerial Vehicles
US	United States
VO	Visual-based Odometry
VTOL	Vertical Take-off and Landing

## ABSTRACT

This thesis presents the development and implementation of intelligent algorithms to increase autonomy of unmanned missions for quadrotor type UAVs. A six-degree-of-freedom dynamic model of a quadrotor is developed in Matlab/Simulink in order to support the design of control algorithms previous to real-time implementation. A dynamic inversion based control architecture is developed to minimize nonlinearities and improve robustness when the system is driven outside bounds of nominal design. The design and the implementation of the control laws are described. An immunity-based architecture is introduced for monitoring quadrotor health and its capabilities for detecting abnormal conditions are successfully demonstrated through flight testing. A vision-based navigation scheme is developed to enhance the quadrotor autonomy under GPS denied environments. An optical flow sensor and a laser range finder are used within an Extended Kalman Filter for position estimation and its estimation performance is analyzed by comparing against measurements from a GPS module. Flight testing results are presented where the performances are analyzed, showing a substantial increase of controllability and tracking when the developed algorithms are used under dynamically changing environments. Healthy flights, flights with failures, flight with GPS-denied navigation and post-failure recovery are presented.

## **1. Introduction**

In recent years, increased research efforts have been conducted towards increasing the reliability and safety of unmanned aerial vehicles (UAVs). The control of quadrotors is of special interest due to minimum flying quality requirements that vehicles of this nature require to execute different mission tasks. Quadrotor systems have been widely used in multiple quadrotor applications due to their quick maneuvering, robust hovering and vertical take-off and landing (VTOL) capabilities. However, new commercial and non-commercial quadrotor applications demand more reliable and efficient systems able to optimize the endurance and performance of the vehicle within complex and unstructured operating environments. In addition, these systems are required to comply with the new regulations, policies, procedures and standards that the Federal Aviation Administration (FAA) is implementing to integrate unmanned aircraft systems (UAS) into the National Airspace System (NAS). Such autonomous aerial systems require on-board intelligent algorithms at different levels with the ability to perform timely status determination, diagnostics, prognostics, and decision making in order to increase the safety of missions, and guarantee successful mission completion. Moreover, such technologies are expected to increase autonomy by maintaining control of the vehicle while avoiding threats such as unrecoverable post-failure flight conditions caused by extreme weather, physical damage, equipment malfunction, and/or other unexpected nonlinearities. “Intelligent” decision making is also required to determine any available modifications to the mission or planned flight path.

In order to address the needs for more efficient and reliable autonomous systems able to be integrated within the NAS, this thesis proposes the study and implementation of three methodologies designed to increase autonomy of unmanned missions for quadrotor type UAVs. First, a nonlinear dynamic inversion (NLDI) control architecture is developed and implemented to increase stability of a quadrotor system by minimizing nonlinearities at nominal conditions. Second, an immunity-based health monitoring system is implemented to increase mission protection by enabling actuators-fault detection capabilities. Finally, an optical-flow vision-based navigation system is developed and implemented to enable navigation of the vehicle over Global Positioning System (GPS) denied environments. The implementation of these methodologies aims to accomplish the main objectives of this thesis:

1. To enhance tracking capabilities, robustness to model uncertainties and attitude stabilization control;
2. To test a bio-inspired fault detection scheme within quadcopter type UAVs;
3. To allow autonomous navigation under GPS denied environments.

The dynamics of a quadrotor are essentially a simplified form of helicopter dynamics that exhibits the basic problems including under actuation, strong coupling, multi-input and multi-output, and unknown nonlinearities (Abhijit, Lewis & Subbarao, 2011). This thesis utilizes a nonlinear controller architecture to handle these problems and improve the results obtained from traditional linear control methods that have been used in the past to control this type of UAV. Nonlinear controllers have shown many advantages over linear control methods in three main areas: analysis of hard nonlinearities,

management of model uncertainties and design simplicity (Slotine & Li, 1991). In this research, a NLDI type architecture is designed, tested in simulation and implemented on a quadrotor system at nominal and upset conditions. A six-degree-of-freedom (6-DOF) simulation environment developed by researchers at the Flight Dynamics and Research Lab has been modified and used to design and implement the NLDI architecture. This simulation environment is developed using Matlab/Simulink software. Accurate simulation results are crucial to provide a safe and reliable starting point for the implementation process, especially during the tuning process of the inner stability control loop. A great part of this thesis deals with implementation of algorithms on a 3DR-X8 quadrotor frame and accurate simulation results are essential for providing safe initial conditions for different purposes. At the same time, this simulation environment is utilized as a verification tool for the specific findings that are encountered during the implementation process. A NLDI control architecture is developed within this research for attitude stabilization and altitude control. Traditional proportional, derivative and integral (PID) control architectures are used for position control and navigation purposes. The performance of the NLDI control architecture is determined based on the tracking response of attitude angles as well as navigation states. This NLDI architecture is enhanced with an Artificial Immune System (AIS) approach that is designed to solve the actuator-fault detection problem.

One of the most promising candidates that has offered a solution to the problem of on-line health monitoring is the Artificial Immune System (AIS) paradigm (Dasgupta, 1999) (Dasgupta, Krishna, Wong & Berry, 2004). Previous research efforts have shown that the

AIS based schemes are capable of automatically acknowledging when an abnormal condition has affected the normal behavior of one of the vehicle's subsystems. AIS based schemes have been proven by the simulation results of fixed wings aircraft sub-systems under upset conditions over extended areas of the flight envelope (Moncayo, Perhinschi, Davis , 2010) (Moncayo, Perhinschi, Davis , 2011) (Dasgupta, Nino, 2009). This thesis is focused on the implementation of bio-inspired mechanisms on an UAV quadrotor type system that is stabilized using NLDI baseline control laws. The AIS approach for fault tolerance detection investigated in this research simulates biological immune mechanisms which protect living organisms. Negative selection process is one of the main principles emulated by this fault tolerant algorithm. Using this principle and a considerable amount of experimental data, this algorithm is able to determine the "self" or dynamic signature for the vehicle in the hyper-space. Similarly, sets of "detectors" are generated by covering the free regions of the hyper-space. This region of the hyper-space is generally known as the "non-self". This process is executed for different combinations of features or states. However, this algorithm requires some off-line training in order to determine the best combinations of features depending on the type of failure. Features used in order to define "self" and "non-self" hyperspaces can be classified as follows: pilot inputs, variables calculated by the control laws and vehicle states. A Hierarchical Multi-Self Strategy (HMSS) approach can be used to enhance the results in terms of fault detection (Moncayo, Perhinschi, Davis, 2010). Fault detection is the process in which a system fault is declared. In general, faults can be classified by hardware faults and software faults. The detection process must be designed thoroughly and it must cover aspects such as the vehicles' operational envelope, the sub-system targeted, and the nature and type of



the abnormal conditions that are expected to be detected (Moguel, 2014). A multi-self detection logic can be implemented in order to declare different type of failures in real time.

Ease of autonomous navigation due to quick maneuvering and robust hovering are dynamic characteristics that make quadrotor type UAVs versatile and suitable for a variety of applications. However, most of these applications rely on the availability of a GPS signal. Unfortunately, when UAVs are required to navigate through indoor or urban environments, GPS signals might be noisy or unavailable. Vision-based approaches have been the focus of many research efforts as an alternative way for autonomous navigation of UAVs (Quelin, 2011). This thesis studies an optical-flow vision-based algorithm used for autonomous navigation within GPS denied environments. Optical flow sensors designed for computer mice have been successfully used to measure velocity and, through integration, horizontal displacement of a UAV (Hui, Yousheng, & Shing, 2014). “Px4flow” is an optical flow sensor based on a machine vision CMOS image that is designed for indoor and outdoor applications (Honegger, Meier, Tanskanen, Pollefeys, 2013). CMOS image sensors are light sensitive which means this sensor requires minimum lighting conditions for being able to measure velocity accurately. A six-state extended Kalman filter (EKF) is designed and implemented for accurate position estimation. The “px4flow” sensor, a laser range finder, accelerometers and attitude angles are required within the EKF algorithm in order to provide a fully integrated alternative solution for autonomous navigation within GPS denied environments. Performance of this alternative solution for autonomous navigation is measured by comparing the results obtained from the optical flow EKF against measurements from a GPS sensor.

## 1.1 Literature Review

The first Quadcopter prototype was built by the Breguet Brothers in 1907. The Gyroplane No.1 was one of the first attempts to create a practical rotary wing aircraft (Naduvilakandy, 2016). This event would serve as the proof of the early concept design. After this event, most of the improvements would come from military applications. Also, the automatic gyroscopic stabilizer, invented by Dr. Peter Cooper and Elmer Sperry, helped to develop the first radio controller UAV with autonomous flight. The development of UAVs had a rise during the late 1950's when the US military used them largely during the Vietnam War and they successfully decreased the number of pilot casualties. Rotary wing UAV development would grow exponentially after the improvements that occurred regarding Micro Electro Mechanical Systems (MEMS) (Naduvilakandy, 2016). The great developments of MEMS allowed to have lighter and smaller sensors and microcontrollers which would set the perfect environment for UAVs development. Since then, most of the advances in quadrotor technology have taken place in academia, in industry and in the open-source project community.

Ascending Technologies (Asctec) is one of the leaders of multirotor UAS drone technology development & manufacturing of technology for professional, civil and research UAV / drone use. Their main focus of research are: UAV Flight dynamics, Simultaneous Localization and Mapping (SLAM), Swarming and computer vision. In the same way, the Stanford Testbed of Autonomous Rotorcraft for Multi-Agent Control (STARMAC) is the name of one of the oldest and most successful quadrotor projects in academia. Their main area of research is the use of multi-vehicle systems. On the open source project community area, Eidgenössische Technische Hochschule Zürich

(ETHZ) is an interdisciplinary team of graduate students from the Swiss Federal Institute of Technology and a Computer Vision and Geometry Lab. They have developed Pixhawk autopilot software and hardware as well as different interfaces for the Gumstix board, QGround Control software, MAVCONN aerial middleware. Their main area of research includes vision-guided flight and recognition of natural features using the FAST detector and sometimes using as many as four cameras on a single aircraft (Naduvilakandy, 2016).

Most of the research performed for quadrotor type UAVs falls in one of three main areas: effective control architecture design for stabilization and navigation, fault tolerance techniques to increase reliability and safety on unmanned missions, and vision-based algorithms for alternative autonomous navigation capabilities. The main concepts and principles behind these three research areas are presented in this section.

### **1.1.1. Linear & Non-linear Control Architectures for Quadrotor Stabilization**

Over the last decade, various control methodologies have been proposed to investigate the attitude control problem of UAVs (Wang, Man, Cao, & Zheng, 2016). Classical linear control techniques and non-linear control algorithms have been designed for attitude and position control (Younes, Drak, Noura, Rabhi & El Hajjaji, 2016). Within the linear control architectures, cascade PID feedback control and the Linear Quadratic Regulator (LQR) are some of the most successful techniques that have been implemented. In general, quadcopter type UAVs are treated as multiple input and multiple output (MIMO) systems. They present under actuated characteristics and they could normally adopt one of two configurations: plus or cross configuration. This type of vehicles are normally modelled as a 6-DOF system with coupling of rotational and

translational dynamics (Wang, Man, Cao, & Zheng, 2016).

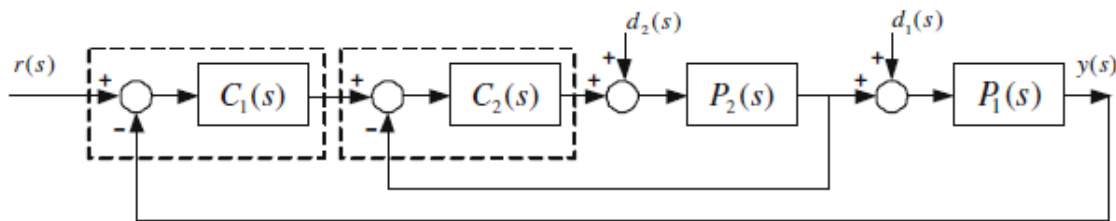
Linearization of a highly nonlinear model degrades the controller performance, and in such a situation, the linear control algorithms fail to control the vehicle at certain points of the flight envelope where the vehicle is not close to a linear region. This has been the reason for the development of alternative algorithms to control quadrotors using nonlinear control techniques.

### **Cascade PID Feedback Control**

Cascade control is a multi-loop control structure that is mostly used on industrial processes implemented to improve the disturbance rejection properties of the controlled system (Marlin, 2000) (Seborg, Edgar & Mellichamp, 2004). A cascade PID feedback controller can also be used to separate the fast and slow dynamics of the process resulting in a nested loop configuration (Alfaro, Vilanova & Arrieta, 2009). This is achieved by adding a new sensor which can provide state feedback for that additional control loop. Figure 1 presents a cascade control architecture. The idea of a cascade control architecture considers that the disturbances affecting the fast secondary loop are compensated before they affect the main process output. This is only possible if the inner loop exhibits faster dynamics which allow for early compensation (Alfaro, Vilanova & Arrieta, 2009). In (Wang, Man, Cao, & Zheng, 2016), strong robustness against external disturbances of a cascade PID control architecture is determined using simulation results.

## Linear Quadratic Regulator Architecture

The theory of optimal control is concerned with operating a dynamic system at minimum cost. Most LQR architectures used in control systems of quadrotor type UAVs are designed for attitude stabilization. In this context LQR is used as the solution of an optimization problem for linear systems with a quadratic performance index. In (Esteves,



**Figure 1** Cascade control configuration (Alfaro, Vilanova, Arrieta, 2009).

Moutinho, Azinheira, 2015), (Younes, Drak, Noura, Rabhi & El Hajjaji, 2016) and (Lan, Fei, Geng & Hu, xx) LQR architectures are used for attitude and altitude control of quadrotors. Simulation results have shown that an LQR architecture has superior performance compared to a classical PD architecture in terms of transient response (Lan, Fei, Geng & Hu, xx).

## Backstepping Architecture

Backstepping control is a popular nonlinear control architecture that has been integrated within many different control techniques to solve the attitude control of quadrotor vehicles. The backstepping technique is a recursive design methodology which uses Lyapunov stability theory to force certain nonlinear systems to follow a desired trajectory (Lan, Fei, Geng & Hu, xx), (Harkegard, 2011). However, these nonlinear

systems require some properties in order to be better handled by this control technique. Feedback from nonlinear systems and systems with nonlinear inputs is desirable to derive results using backstepping. The backstepping idea relies on letting certain states of the system act as “virtual controls” of other states. This idea is usually used in cascade control design and singular perturbation (Harkegard, 2011). In (Lan, Fei, Geng & Hu, xx), simulation results have shown that a backstepping law has better performance in terms of transient response and tracking response when applied to attitude control of a multirotor. Similarly, (Harkegard, 2011) evaluates a backstepping control architecture with respect to five major areas: stability, tuning, robustness, input saturation, and disturbance attenuation when used for a flight control of a UAV. Simulation results have shown that a backstepping technique improves the performance with respect to stability, tuning process, robustness and input saturation. However, the performance regarding disturbance attenuation has not been determined.

### **Sliding Mode Architecture**

Sliding mode control is a variable structure control algorithm. It includes many different continuous functions that map the plant state to a control surface. The switching among these functions is determined by the plant state which is represented by a switching function (Zeghlache, xx). Sliding mode architecture has been used to design robust nonlinear controllers for attitude stabilization and trajectory control. Attitude and trajectory control using a sliding mode control approach are studied in (Zeghlache, xx) for a 6 DOF quadrotor. Both efforts have shown that such architecture is capable of successfully dealing with problems such as: under actuation, strong coupling and multi-

input/ multi-output.

### **1.1.2. Artificial Immune System**

AIS is a vigorous field of research which investigates how immuno-biology can assist in technology, and along the way is beginning to help biologists understand their unique problems (Nicosia, Cutello, Bentley & Timmis, 2004). Many computer science techniques have been inspired from the working principles of immune systems to solve engineering problems in different areas of research such as machine learning, computer security, data mining, pattern recognition and anomaly detection (Nicosia, Cutello, Bentley & Timmis, 2004). Before further analyzing the artificial version of an immune system, a basic knowledge of the working principles of a biological immune systems would help appreciate the intelligence embedded within this system and should clarify the benefits behind the AIS approach. The human immune system on its own is an extremely effective system that can identify abnormal activities, solve the problem using existing knowledge, and generate new solutions for unseen events (Ko, Lau, & Lau, 2014). The human immune system consists of two subsystems: the innate immune system and the acquired or adaptive immune system.

**Innate Immune System:** It is formed by all the elements that are always present within the immune system, ready to fight against foreign virus action. Some of these elements include: the mucous membranes, the cough reflex and skin. In the same way, internal processes such as fever, interferons, macrophages and substances released by leukocytes help to nullify the effect of foreign viruses (Playfair J.H.L., Chain B.M., 2011).

**Acquired Immune System:** It is an aid to the innate system and works in parallel with it. The main recognition and reaction functions of the immune response are performed by T-lymphocytes (T-cells) and B-lymphocytes (B-cells) which exhibit specialization towards any antigen or virus. A process called Humoral immunity consists of the synthesis and secretion of antibodies to the bloodstream and is performed by the B-cells. In contrast, the T-cells seek out invaders to destroy. In addition, T-cells help B-cells to make antibodies and activate macrophages to eat foreign matter. The process performed by T-cells is called Cellular Immunity.

Within the AIS paradigm, different principles have been modeled in order to be emulated and directed to different applications. Some of the main models that have been developed are: Discrimination, the negative selection algorithm, Clonal Selection principle, Immune network, and Immunological memory. (Nicosia, Cutello, Bentley & Timmis, 2004).

**Discrimination:** One of the most important functions of the immune system is to discriminate Non-Self Cells from Self Cells. Non-self Cells are external elements that harm the system, often called antigens. On the other hand, Self cells are cells that work inside human bodies and do not have harmful effects against the body. The distinction process and antigen detection is performed by B-cells and T-cells which allow the immune system to fight against harmful viruses and keep the self cells.

**Negative Selection:** The negative selection algorithm is based on the principles of self-non-self discrimination in the immune system (Dasgupta, Krishna, Wong & Berry, 2004). During the maturation of T-cells, inside the thymus gland, if a T-cell in the thymus



recognizes any self cell, it is eliminated before deploying it for immune functionality. Only the T-cells that do not react with the self-patterns are free to proliferate and bind to any antigens (bacterial, virus, etc.) for destruction. The negative selection algorithm generates detectors set by eliminating any detector candidate that match elements from a group of self samples (Dasgupta, 2006).

**Clonal Selection Principle:** This theory holds that each B-cell produces antibodies that fit only one specific type of antigen, called its “cognate” antigen. B-cells proliferate to clone many copies of itself only after the specific B-cell binds with its cognate. The recently cloned cells become plasma B-cells which continue to produce and export huge quantities of antibodies (Ko, Lau, & Lau, 2014).

**Immune Network:** This theory was proposed by Jerne in the seventies and explains how B-cells survive even in the absence of antigenic stimulus. It suggested that B-cell memory was maintained via Idiotypic network (Bentley & Timmis, 2004). It is based on the assumption that B-cells can, in addition to being able to recognize antigens, recognize each other through interactions of idiotopes. This allows for the formation of a network structure of stimulating and suppressing signals which propagate through the network, boosting or decaying the concentration of a particular B-cell (Mohr, Ryan & Timmis, 2014).

**Immunological memory:** After B and T-cells have specialized and been able to proliferate to generate clones and destroy antigens, most of these cells eventually die.

However, some of them remain untouched to pass on their knowledge of the antigen. These cells are usually called Memory cells and they are much easier to activate than “naïve” cells (Ko, Lau, & Lau, 2014).

### **1.1.3. AIS as a Solution of the Failure Detection Problem**

In recent years, the development of fault tolerant flight controls emerged as a new methodology to increase safety and enhance performance not only for large scale aircraft but also small scale UAVs. Safety has been considered one of the main concerns of the Federal Aviation Administration (FAA) towards allowing commercial operations of UAVs within US aerospace. In addition, UAVs will face difficult new situations such as flight in urban environments where reliability is particularly critical. The poor reliability of current unmanned vehicles represents a serious obstacle to their success in demanding new flight environments (Drozeski, Saha, Vachtsevanos, 2005). The Office of the Secretary of Defense acknowledges this shortcoming in the UAV Roadmap 2002–2027. It determines that the development of self repairing, smart flight control systems is a crucial step in the overall advancement of UAV autonomy (Zhang, Chamseddine, Rabbath, Gordon, Su, Rakheja, Fulford, Apkarian & Gosselin, 2013). Many different approaches have been adopted towards increasing fault tolerant control of UAVs including: gain scheduling PID, fuzzy gain scheduling PID, adaptive controls (MRAC), sliding mode control, control allocation and relocation and AIS algorithms.

As mentioned before, one of the most promising candidates that has offered a solution to the problem of fault detection and identification is the AIS paradigm. Failure detection and identification of aircraft over extended areas of the flight envelope presents

a multi-dimensional and highly complex challenge that needs to be addressed by an integrated solution (Moguel, 2014). The basic idea of AIS-based FDI is that an abnormal condition can be declared when a current configuration of features does not match with any set of known normal conditions previously generated through simulation (Perhinschi, Moncayo & Al Azzawi, 2013) (Perhinschi, Moncayo, Wilburn, Bartlett, Davis & Karas, 2011). These features can include pilot inputs, estimated states, sensor readings, control law variables and system states. Projections of these features in multiple dimensions should be able to fully describe the dynamic behavior of the system within its flight envelope. In order to obtain a complete data base with information of the dynamic signature of the vehicle, large amounts of data need to be acquired and post processed. These data are then used to determine self and non-self hyperspaces depending on the dimensionality of the projection. Self and Non self hyperspaces determination are crucial for the generation of detector and identifiers.

In general, an AIS-based FDI subsystem is divide into two main processes: post processing of flight test or simulated data and online detection and identification. The former includes but is not limited to data acquisition, data reduction, detector and identifier generation and optimization. This process can be reproduced in a simulation environment or through flight test using the actual test-bed. The online FDI process includes the development and application of FDI logic. During this process, detectors are compared against sets of current values of identifiers measured in flight at a certain sampling rate. At each sample, a binary output (i.e. 0 for normal or 1 for abnormal) determines if the current values are inside a detector (abnormal condition) or outside a

detector (normal condition). The FDIE scheme utilizes sets of output values over moving time windows, reducing the number of false alarms (Moguel, 2014).

In (Moguel, 2014), the author shows simulation and experimental results of an AIS-based FDIE using a hierarchical multi-self strategy that is capable of detecting and identifying four different categories of abnormal conditions over extended areas of the flight envelope on a fixed wing type UAV. One of the main novelties of this thesis is the implementation of an AIS-based fault detection scheme using a multi-self strategy for quadrotor type UAVs.

#### **1.1.4. Vision-based Approach as an Alternative Solution for Autonomous Navigation in GPS Denied Environments**

A combination of inertial measuring devices and GPS systems have been traditionally utilized in order to effectively estimate the vehicle's velocity and its position in the world. Unfortunately, this sensing method is restricted to places where the positioning system signals are available. For this reason, different strategies have been proposed in order to successfully develop autonomous navigation algorithms using imaging sensors. However, achieving autonomous tasks becomes even more complicated if the vehicle operates in GPS denied environments (Garcia, Dzul, Lozano & Pegard, 2013). According to the different kinds of mission and the environments where the UAVs must interact, solutions based on monocular vision, stereo vision and even multiple views have been researched in recent years. Some of the latest findings in this area of research are presented here.

A visual-based position control called Visual Servoing is studied in (Saripalli, Sukhatme, Mejias & Campoy Cervera, 2005) (Guenard, Hamel & Mahony, 2008). In (Saripalli, Sukhatme, Mejias & Campoy Cervera, 2005) the vision system is designed for target detection and a fusion between vision and GPS measurements allowed the autonomous navigation. A relationship between the image-based task and the actuators is needed in order to accomplish UAV navigation. In (Guenard, Hamel & Mahony, 2008) the desired position of a quadrotor is deduced by using a specific position configuration of an on-ground target formed by four black circles. In (Proctor, Johnson & Apker, 2006) the authors used monocular imaging sensors to develop a vision-based navigation control system for a glider. The main contribution of this work is that this algorithm does not depend on inertial sensors for state estimation. (Garcia, Dzul, Lozano & Pegard, 2013). A Kalman filter, which estimates attitude angles, is designed to use information of an artificial target located in the image plane. The implementation of this idea is successful due to the stable nature of a fixed wing type UAV. Similar results are not possible if applied to a quadrotor UAV. Similarly, a vision system which allows autonomous navigation and object tracking is presented in (Ludington, Johnson & Vachtsevanous, 2007). This work presents implementation results of a Kalman filter, performing a fusion between inertial measurements and vision readings, programmed on a Pentium III PC that is installed in a helicopter vehicle.

The algorithms previously mentioned have an ideal applicability for autonomous landing or constant hover missions where object tracking is enough to accomplish most of this objectives. However, SLAM algorithms and visual-based odometry (VO)

techniques have also been the focus of considerable research. The idea behind these techniques is to increase the autonomy level on UAVs by allowing the vehicle to locate itself within an unknown environment and navigate through using vision-aided systems. At the same time the vehicle should be able to generate a map of the environment in which it is interacting. SLAM and VO techniques have been developed using two different approaches: single-camera vision systems and stereo vision systems. Single-camera vision systems require simplifications to the system by adopting different assumptions. On the other hand, stereo-vision approaches provide a more complete solution to the SLAM and VO problems by allowing a three dimensional analysis of the UAV motion at all times. This approach emulates the three-dimensional reconstruction process carried out by animals, birds and humans. Furthermore, the use of more views might be used in order to improve position estimation; however, weight constraints might be analyzed at the time of implementing such techniques.

A SLAM algorithm for localization of a quadrotor type UAV using a monocular camera is presented in (Blosch, Weiss, Scaramuzza & Siegwart, 2010). The camera pose estimation is used to stabilize the vehicle at a desired set point. In this way, maneuvers of take-off, hovering and landing are performed while building a map of the region travelled. On the other hand, (Achtelik, Bachrach, He, Prentice & Roy, 2009) presents results of an OV and SLAM algorithm used to estimate the relative displacement of the vehicle. The relative displacement of the vehicle is estimated using a double integration of the pose of a stereo-rig. A SLAM technique is used in this research to compensate for

position drift.

The main focus in this thesis is the study of optical flow techniques in the development of autonomous navigation systems for a quadrotor type UAV. Optical flow sensors have been successfully implemented to safely navigate UAVs through outdoor and indoor environments. However, many constraints have to be considered when implementing this technique: altitude, minimum lighting conditions and different type of terrains and features.

Optical flow algorithms are generally used to represent the motion of the objects as they appear in a sequence of images. This motion field is described by assigning to every point in the visual field a two-dimensional instantaneous velocity vector (Quelin, 2011). The movement of brightness patterns in a sequence of images can be used to estimate a velocity vector. Comparison of grey levels and intensities or features found in one image are usually used to match with a feature in the following image. Furthermore, many assumptions need to be considered in order to formulate optical flow principles. This algorithms assume that the appearance of a scene does not change considerably between frames; hence, smooth changes between scenes are ideal. Also, it is assumed that each scene contains mainly smooth surfaces which move rigidly or distort smoothly. In addition to that, movement between frames is assumed to be small compared to the size of the image which leads to small optic flow vectors for a pair of images (Chhaniyara, 2008). In this research, an optical flow camera is arranged perpendicular to the direction of motion which is favorable for velocity estimation on low speeds applications; however, flying at higher altitudes allows the camera to capture higher speeds if needed.

This thesis document is organized as follows: Chapter 1 presents a literature review of the topics addressed on this thesis. A brief description of the hardware and software utilized for the implementation stages of this thesis is presented in Chapter 2. Chapter 3 describes the dynamic model of the quadrotor UAV derived using a classical Newtonian approach and the NLDI control architecture. Chapter 4 presents the AIS architecture used to enable actuator-fault detection and health monitoring capabilities. The vision-based approach and EKF design used for autonomous navigation under GPS denied environments are presented in Chapter 5. Conclusions and recommendations for future work are provided in Chapters 6 and 7.

The research effort presented in this thesis has resulted in the publication and submission of:

**Garcia D. F.**, Moncayo H., Perez A., Jain C., (2016) “Low Cost Implementation of a Biomimetic Approach for UAV Health Management”. American Control Conference ACC 2016.

**Garcia D. F.**, Perez A. E., Moncayo H., Rivera K., DuPuis M, Robert, Mueller P., (2017). “Spacecraft Health Monitoring Using a Biomimetic Fault Diagnosis Scheme”. AIAA Conference, Grapevine, Texas.

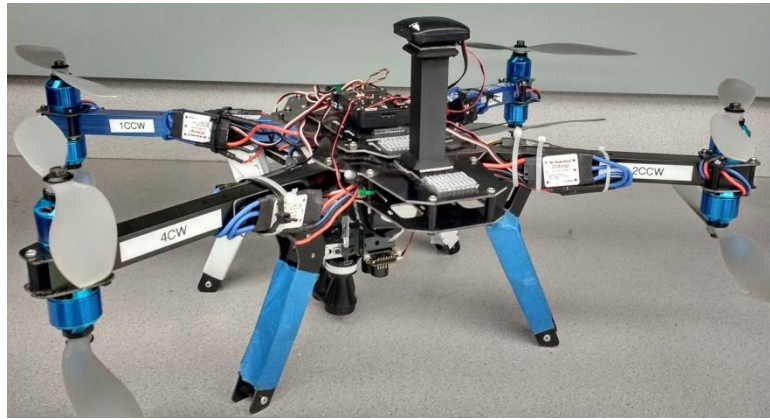
Journal under Revision

**Garcia D. F.**, Perez A. E., Moncayo H., Rivera K., DuPuis M, Robert, Mueller P., (2017). “Spacecraft Health Monitoring Using a Biomimetic Fault Diagnosis Scheme”. On progress to be submitted to the AIAA (Journal of Aerospace Information Systems (JAIS)).



## 2. Quadcopter UAV Research Test-bed

The testbed used for all implementation purposes of this thesis is a 3DR X8 frame quadcopter. This low-cost testbed meets minimum requirements needed to implement all the algorithms proposed in this thesis. This testbed has been used for multiple projects and the eight motors “X8” configuration is adopted in order to obtain extra lift capabilities. Figure 2 shows a close view of the testbed with the instrumentation onboard the vehicle.



**Figure 2** 3DR X8 Quadcopter Testbed with instrumentation

The main dimensions and mass properties of this vehicle used within the simulation environment, are presented in the following table:

**Table 1** Main properties of the testbed (3DR X8).

<b>Items</b>	<b>Dimensions</b>
<b>Total Weight</b>	2310 gr
<b>Thrust Arm - Roll</b>	23.5 cm
<b>Thrust Arm - Pitch</b>	15.56 cm
<b>I<sub>xx</sub></b>	0.0179 kg*m <sup>2</sup>
<b>I<sub>yy</sub></b>	0.0184 kg*m <sup>2</sup>
<b>I<sub>zz</sub></b>	0.0312 kg*m <sup>2</sup>

## 2.1 Hardware Description

The 3DR X8 quadcopter testbed requires a microcontroller, known as an onboard computer, in order to stabilize the system and allow for stable hover flight. For the purposes of this research a “Pixhawk Autopilot” board is used to test all the algorithms that require autonomous and manual flight routines. This low cost board is compatible with a series of analog and digital sensors that provide essential states utilized for feedback stabilization control loops as well as autonomous navigation control loops (GPS and optical flow sensor). Features and variables needed for generation of self clusters and non-self clusters for AIS detection scheme are also obtained through this autopilot.

### 2.1.1 Pixhawk Autopilot

This autopilot board is designed by an open hardware development team from The Computer Vision and Geometry Lab of ETH Zurich in collaboration with 3D Robotics and ArduPilot Group. This autopilot module uses a 168 MHz Cortex M4F CPU (256 KB RAM, 2MB Flash) which runs a very efficient real-time operating system (RTOS), which provides for better performance of flight control and vehicle management applications. It has 14 PWM/servo outputs and abundant connectivity options for additional peripherals (UART, I2C, and CAN).



**Figure 3** Pixhawk Autopilot board.

### **2.1.2 InvenSense MPU 6000 6-axis Accelerometer+Gyroscope**

The MPU 6000 Inertial sensor is featured with a 3-axis gyroscope and a 3-axis accelerometer inside a housing of 4x4x0.0 mm.



**Figure 4 MPU 6000**

### **2.1.3 ST Micro LSM303D 14 bit Accelerometer / Magnetometer**

The LSM303D is a system-in-package featuring a 3D digital linear acceleration sensor and a 3D digital magnetic sensor. It includes an I2C serial bus interface that supports standard and fast mode (100Hz and 400 Hz) and SPI serial standard interface.

### **2.1.4 ST Micro L3GD20H 16 bit Gyroscope**

The L3GD20H is a low-power three-axis angular rate sensor that includes a sensing element and an IC interface able to provide the measured angular rate to the external world through digital interface I2C/SPI). It also has a full scale of +/- 245, +/- 500, +/- 2000 degrees per second and is capable of measuring rates at different bandwidths.

### **2.1.5 UBLOX LEA-6H GPS Receiver Module with Antenna**

This GPS module incorporates the HMC5883L digital compass, providing a convenient method of mounting the compass away from sources of interference that may

be present in the confines of the vehicle especially at the brushless AC motors. It features a u-blox LEA-6H GPS module, with a 5 Hz update rate and a low noise 3.3V regulator. The full module dimensions are: 38 x 38 x 8.5 mm and weight 16.8 gr.



**Figure 5** 3DR GPS with compass module

### **2.1.6 LightWare SF11/C (120 meter)**

The SF11/C is a compact, lightweight laser altimeter for above-ground-level measurement from small fixed wing or multi-rotor craft. The SF11/C laser altimeter makes accurate distance measurements to solid surfaces up to an altitude of 120 meters and water up to 40 meters. It includes digital (serial and I2C) and analog (12bit) outputs along with a micro USB configuration port. Its dimensions are 30 x 56.5 x 50 mm and weights 35 gr.

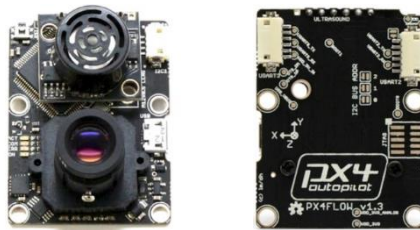


**Figure 6** SF11/C Laser Range Finder

### **2.1.7 Px4Flow Smart Camera**

The Px4Flow smart camera is essentially an optical flow sensor. It has a resolution of 752 x 480 pixels and calculates optical flow on a 4x binned and cropped area at 400 Hz. It is designed to work indoors as well as outdoors along low light

condition environments without the need of an artificial source of light. It uses a 168 MHz Cortex M4F CPU (128 + 64 KB RAM), a 752×480 MT9V034 image sensor, a 16 mm M12 lens and is featured with a L3GD20 3D Gyroscope. Its overall dimensions are: 45.5 mm x 35 mm.



**Figure 7** Optical Flow sensor Px4Flow

### **2.1.8 A 20 amp Electronic Speed Controller (ESC) with Simonk Firmware**

This ESC can be used with 2-4 cell LiPo batteries and it provides a continuous current of 20 Amp and a burst current of 25 amp for 10 seconds. It has a voltage regulator at 5V and 2amp. It is designed to support a motor speed of 210000 RPM (Max) with (2 poles), 70000 RPM with (6 poles), and 35000 RPM with (12 poles). It weighs 21 gr.



**Figure 8.** 20 amp ESC

### **2.1.9 Motor 850 Kv AC 2830 – 358**

This electric AC brushless motor is designed to develop 850 RPM/V [Kv]. Its overall dimensions are 28 x 30 mm with a shaft of 3.17mm and total weight of 62 gr. Its

best performance is featured for 10×47 propellers and it requires 2-4 cells LiPo batteries.



**Figure 9** Electric AC brushless motor

### **2.1.10 APC 10x4.7 SF Plastic Propellers**

Its overall dimensions are: diameter 10 inches and geometric pitch 4.7 inches.



**Figure 10** APC plastic propeller

### **2.1.11 4S Lipo Battery**

This power module is featured with 4 cells and a nominal voltage of 14.8 V. The discharge rate is rated at 5000 mAh and its overall dimensions are: 136 x 43 x 32 mm.



**Figure 11** 4S Lipo battery

## **2.2 Software Description**

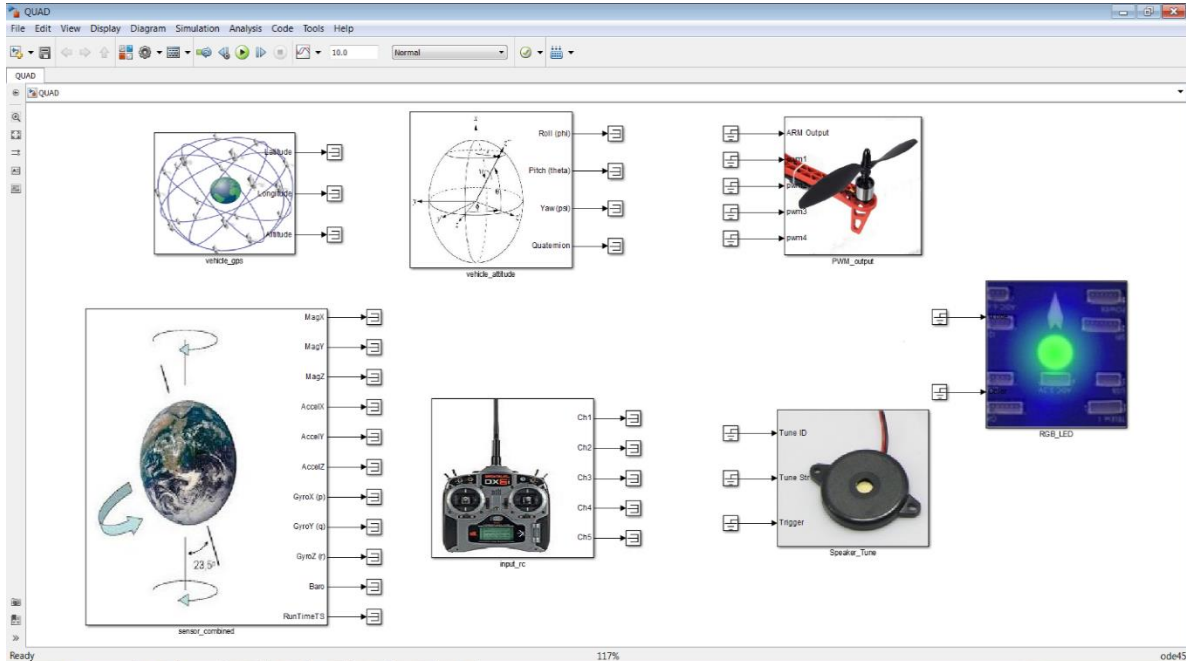
### **2.2.1 Matlab/Simulink Simulation Environment**

A 6-DOF dynamic model of a quadrotor is designed within Simulink in order to design and test the baseline NLDI controller. This simulation environment has an interface with Flight Gear software used to display attitude and position of the simulated vehicle at all times. These capabilities allow the user to have a better visualization of the controller performance and can be used as a tool for debugging purposes. In addition to the design of a baseline inner controller, this environment can be used to design the outer loop controller for autonomous navigation. An accurate simulation of the dynamic vehicle not only provides a tool for designing and simulating new control algorithms, it also provides a safe starting point for the tuning process during implementation stage. Implementation of all the algorithms discussed throughout this thesis is possible by deploying the Simulink codes into the Pixhawk Autopilot board. This process is performed through Simulink software and the Embedded Coder tool.

### **2.2.2 Support Package for Pixhawk Autopilot**

The Pixhawk support package from Matlab/Simulink allows the user to read several sensors featured for Pixhawk Autopilot including those embedded inside this board. It includes some libraries that allow the user to interact with inertial measurements, GPS, vehicle estimation, light emitting diode (LED), PWM output and serial Rx/Tx for communication purposes. In addition, it provides capabilities to log and record flight data from sensors, actuators, or any control signal created within the Simulink environment. This capability is essential during the implementation stage since

it allows the user to save flight test data used for post-flight analysis and redesign purposes. A specific Simulink code must be developed for implementation purposes in order to allow for stable flight of the quadrotor. This Simulink code integrates parts from the NLDI baseline controller simulation, blocks for reading required sensors, blocks for reading pilot input commands and blocks for sending PWM signals to the electronic speed controllers that go into the electric motors. The embedder coder tool is designed to translate Simulink code into a readable, compact, and fast C and C++ code for use on embedded processors, on-target rapid prototyping boards, and microprocessors. This feature allows Simulink codes to be loaded into the autopilot board. Figure 12 shows some of the blocks from the Pixhawk Support package library used for implementation purposes (Pixhawk Support Package User Guide from Mathworks, 2017).



**Figure 12** Sample blocks from Pixhawk Support package library



### 2.3 Flight testing field (Daytona Beach Radio Control Association)

The flight tests necessary to gather the data and results presented in this research effort were performed at the Daytona Beach Radio Control Association field (DBRCA). Figure 13 shows a top view of the actual field. The designated area for multirotor testing is identified with a red “X” in figure 13. The RC field elevation is about 46 feet above sea level and the overall area is around 1500 square feet. Its latitude and longitude coordinates are respectively:  $29^{\circ} 6' 34.2612''$  and  $81^{\circ} 5' 14.4846''$ . The open field presents suitable conditions for flight tests designed for tuning inner and outer loop controllers. It allows enough space for sudden recoveries during emergency situations. Furthermore, this field meets the minimum safety requirements to perform the required autonomous missions using the GPS sensor module and the optical flow sensor.



**Figure 13** Daytona Beach RC Association Field

### 3. Non-linear Dynamic Inversion Control Architecture

#### 3.1 Vehicle Dynamic Modeling

The full dynamic model of the quadrotor consists of three main parts: the propulsion system model, the 6-DOF model of the quadrotor and the sensors model. The propulsion system produces the required forces and moments that will drive the quadrotor to a desired attitude and position. These forces and moments depend on parameters, such as diameter of the propeller, propeller pitch, velocity of the propeller and atmosphere density. For simplicity, the forces and torques of the propellers are assumed to be proportional to the square of the angular velocity (Das, Subbarao & Lewis, 2008).

$$F = k_f n^2 \quad (2.1)$$

$$\zeta = k_M n^2 \quad (2.2)$$

$k_f$  and  $k_M$  are force and moments coefficients, respectively; and  $n$  is the angular velocity. The vehicle chosen for all implementation purposes in this thesis is an X8 quadrotor with four arms and two actuators per arm. A cross configuration is chosen as a reference, but plus type configurations will work in a similar fashion. Since the revolutions per seconds (rps) on each motor can be controlled independently, the total sum of forces, moments, and torques can be expressed in the following matrix form:

$$\begin{bmatrix} T \\ M_X \\ M_Y \\ M_Z \end{bmatrix} = \begin{bmatrix} k_f & k_f & k_f & k_f & k_f & k_f & k_f & k_f \\ k_f L_1 & k_f L_1 & -k_f L_1 & -k_f L_1 & -k_f L_1 & -k_f L_1 & k_f L_1 & k_f L_1 \\ k_f L_2 & k_f L_2 & k_f L_2 & k_f L_2 & -k_f L_2 & -k_f L_2 & -k_f L_2 & -k_f L_2 \\ -k_M & k_M & k_M & -k_M & -k_M & k_M & k_M & -k_M \end{bmatrix} \begin{bmatrix} n_1 \\ n_{1*} \\ n_2 \\ n_{2*} \\ n_3 \\ n_{3*} \\ n_4 \\ n_{4*} \end{bmatrix} \quad (2.3)$$

$T$  is the total thrust of the system,  $M_X$  is the total rolling moment,  $M_Y$  is the total pitching

moment, and  $M_Z$  is the total yawing moment.  $L_1$  represents the perpendicular distance between body x-axis and the motors y-position. Similarly,  $L_2$  represents the perpendicular distance between body y-axis and the motors x-position. In order to obtain the required rps for each of the four motors, the previous matrix system needs to be inverted as follows:

$$n_1^2 = \frac{T}{8 k_f} + \frac{M_X}{8 k_f L_1} + \frac{M_Y}{8 k_f L_2} - \frac{M_Z}{8 k_M} \quad (2.4)$$

$$n_{1*}^2 = \frac{T}{8 k_f} + \frac{M_X}{8 k_f L_1} + \frac{M_Y}{8 k_f L_2} + \frac{M_Z}{8 k_M} \quad (2.5)$$

$$n_2^2 = \frac{T}{8 k_f} - \frac{M_X}{8 k_f L_1} + \frac{M_Y}{8 k_f L_2} + \frac{M_Z}{8 k_M} \quad (2.6)$$

$$n_{2*}^2 = \frac{T}{8 k_f} - \frac{M_X}{8 k_f L_1} + \frac{M_Y}{8 k_f L_2} - \frac{M_Z}{8 k_M} \quad (2.7)$$

$$n_3^2 = \frac{T}{8 k_f} - \frac{M_X}{8 k_f L_1} - \frac{M_Y}{8 k_f L_2} - \frac{M_Z}{8 k_M} \quad (2.8)$$

$$n_{3*}^2 = \frac{T}{8 k_f} - \frac{M_X}{8 k_f L_1} - \frac{M_Y}{8 k_f L_2} + \frac{M_Z}{8 k_M} \quad (2.9)$$

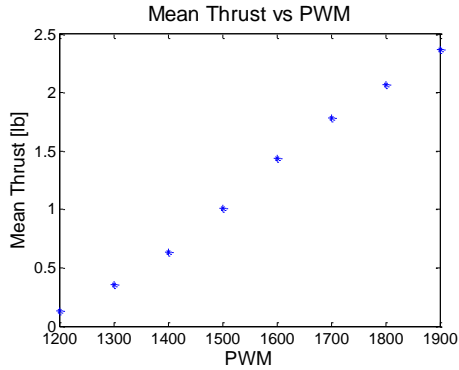
$$n_4^2 = \frac{T}{8 k_f} + \frac{M_X}{8 k_f L_1} - \frac{M_Y}{8 k_f L_2} + \frac{M_Z}{8 k_M} \quad (2.10)$$

$$n_{4*}^2 = \frac{T}{8 k_f} + \frac{M_X}{8 k_f L_1} - \frac{M_Y}{8 k_f L_2} - \frac{M_Z}{8 k_M} \quad (2.11)$$

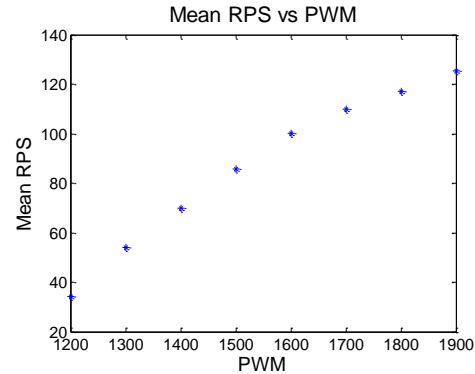
Equations 2.4 through 2.11 will be required to derive the control laws using the dynamic inversion approach.

Experimental tests are performed to characterize the propulsion system of the vehicle. Data from the tests are used to find an accurate relation between motors rps, thrust, torque, and pulse width modulation signals (PWM). The propulsion system, direct current (DC) motor and propeller, is characterized using a counter-torque setup. As a result of this characterization, the mean values for thrust, torque and rps are determined

against PWM signals. Figure 14 and Figure 15 show an example of mean thrust and rps for different values of PWMs.



**Figure 14** Mean Thrust vs PWM



**Figure 15** Mean rps vs PWM

The equations of motion that describe the dynamics of the vehicle are developed using a Newtonian approach. The general dynamics of a rigid body under external forces and moments with respect to its center of gravity in the body reference frame can be written as (Wang, He, Zhang & He, 2013):

$$\text{Force equations: } m \frac{dV^b}{dt} + \omega^b \times m V^b = \vec{F} \quad (2.12)$$

$$\text{Moment equations: } \vec{J} \frac{d\omega^b}{dt} + \omega^b \times \vec{J} \omega^b = \vec{M} \quad (2.13)$$

where  $V^b = [u \ v \ w]^T$  is the relative linear velocity of the center of mass of the rigid body with respect to an inertial frame and  $\omega^b = [p \ q \ r]^T$  represents angular velocity in the body frame with respect to the inertial reference frame. Consequently expanding Eq. (2.12) yields:

$$\begin{bmatrix} \sum F_x \\ \sum F_y \\ \sum F_z \end{bmatrix} = m \left( \begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} + \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \begin{bmatrix} u \\ v \\ w \end{bmatrix} \right) = m \begin{bmatrix} \dot{u} + qw - rv \\ \dot{v} + ru - pw \\ \dot{w} + pv - qu \end{bmatrix} \quad (2.14)$$

Since  $F_X = 0, F_Y = 0$  and  $F_Z = -T$ , we can rewrite Eq. (2.12) as (including the gravitational force),

$$\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} = - \begin{bmatrix} qw - rv \\ ru - pw \\ pv - qu \end{bmatrix} + \frac{1}{m} \begin{bmatrix} 0 \\ 0 \\ -T \end{bmatrix} + \begin{bmatrix} -g \sin\theta \\ g \cos\theta \sin\phi \\ g \cos\theta \cos\phi \end{bmatrix} \quad (2.15)$$

In Eq. (2.15),  $[u \ v \ w]$  and  $[p \ q \ r]$  represent linear velocities and angular rates in body reference frames, respectively.  $m$  is the mass of the quadrotor.  $\phi$  and  $\theta$  represent roll angle and pitch angle between body and earth reference frames.

In the moment equation Eq. (2.13),  $\vec{J}$  represents the inertia tensor of the rigid body and  $\vec{M}$  represents the external sum of all the moments in body reference frame. In general, quadrotors present the following inertia tensor due to symmetrical characteristics.

$$J = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix} \quad (2.16)$$

$I_{xx}, I_{yy}, I_{zz}$  represent symmetrical moments of inertia of the quadrotor. Then Eq. (2.13)

can be expanded as follows:

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}^{-1} \left\{ - \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \begin{bmatrix} I_{xx} p \\ I_{yy} q \\ I_{zz} r \end{bmatrix} + \begin{bmatrix} M_x \\ M_y \\ M_z \end{bmatrix} \right\} \quad (2.17)$$

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = - \begin{bmatrix} qr(I_{zz} - I_{yy})/I_{xx} \\ pr(I_{xx} - I_{zz})/I_{yy} \\ pq(I_{yy} - I_{xx})/I_{zz} \end{bmatrix} + \begin{bmatrix} M_x/I_{xx} \\ M_y/I_{yy} \\ M_z/I_{zz} \end{bmatrix} \quad (2.18)$$

In order to develop expressions for rotational motion, transformations from body reference frame to Earth reference frame are used. To find a relationship between angular

rates vector  $\boldsymbol{\omega}$  in the body reference frame and the Euler angle rates the following rotations are required (Sidi, 1997):

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = R_\phi R_\theta R_\psi \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} + R_\phi R_\theta \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + R_\phi \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} \quad (2.19)$$

where  $R_\phi R_\theta R_\psi$  are defined as follows:

$$R_\phi = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{bmatrix}, R_\theta = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix}, R_\psi = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (2.20)$$

These transformations are usually used to rotate a vector in the fixed Earth ( $\mathbf{E}$ ) reference frame to a vector in the body ( $\mathbf{B}$ ) reference frame or vehicle reference frame as follows:

$$\begin{bmatrix} \hat{i} \\ \hat{j} \\ \hat{k} \end{bmatrix}_B = R_\phi R_\theta R_\psi \begin{bmatrix} \hat{i} \\ \hat{j} \\ \hat{k} \end{bmatrix}_E \quad (2.21)$$

Similarly, a rotation from body reference frame to Earth reference frame can be obtained by using the transpose of the previous rotations as follows.

$$\begin{bmatrix} \hat{i} \\ \hat{j} \\ \hat{k} \end{bmatrix}_E = (R_\psi)^T (R_\theta)^T (R_\phi)^T \begin{bmatrix} \hat{i} \\ \hat{j} \\ \hat{k} \end{bmatrix}_B \quad (2.22)$$

This transformation is usually referred in the literature as the Direct Cosine Matrix (DCM). It can be expanded as follows:

$$DCM_B^E = \begin{bmatrix} c\theta c\psi & s\theta s\theta c\psi - c\theta s\psi & c\theta s\theta c\psi + s\theta s\psi \\ c\theta s\psi & s\theta s\theta s\psi + c\theta c\psi & c\theta s\theta s\psi - s\theta c\psi \\ -s\theta & s\theta c\theta & c\theta c\theta \end{bmatrix} \quad (2.23)$$

Eq. (2.13) is expanded as follows after all the matrix multiplications:

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \sin\phi \cos\theta \\ 0 & -\sin\phi & \cos\phi \cos\theta \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \quad (2.24)$$

The previous equation can be solved for Euler angle rates  $[\dot{\phi} \ \dot{\theta} \ \dot{\psi}]^T$  by calculating the inverse of the corresponding matrix which leads to:

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin\phi \tan\theta & \cos\phi \tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi / \cos\theta & \cos\phi / \cos\theta \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \quad (2.25)$$

where  $\psi$  represents yaw angle in earth reference frame. This set of three equations is known as the Kinematic Equations.

Similarly, translational equations of motion or navigation equations can be derived by defining the following relationships between the velocities of the vehicle in the body reference frame  $V^b$  and the velocities in the fixed Earth reference frame  $[\dot{x} \ \dot{y} \ \dot{z}]$ .

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = DCM \begin{bmatrix} u \\ v \\ w \end{bmatrix} \quad (2.26)$$

Equations (2.15), (2.18), (2.25) and (2.26) form a set twelve non-linear equations that describe the quadrotor dynamics. These equations are used within this research as part of a simulation environment, to accurately model and simulate the quadrotor dynamics.

### 3.2 Dynamic Inversion Control Laws

The NLDI technique is used as a baseline controller for attitude stabilization of the UAV system. The NLDI performance is highly dependent on the accuracy of the modeling of the non-linear system. If the model is well characterized, then most of the non-linear parameters of the system will be cancelled out during a feedback linearization process. The main task of the feedback linearization is to change a non-linear system into a traditional linear system. Several classical control techniques can be then used within

the new linear system for stabilization purposes.

NLDI is applied to the vehicle by inverting the equations of motion that describe its dynamics. The general structure for the inner stability controller of the quadrotor is separated into two inversion phases: a slow mode and a fast mode. This distinction is made based on the response time of the vehicle dynamics. This inner controller requires state feedback from angular rates and attitude angles. Angular rates can be measured using MEMs gyroscopes while attitude angles are estimated using an EKF. The slow mode inversion takes attitude angles  $[\phi \ \theta \ \psi]^T$  and outputs the desired angular rates  $[p \ q \ r]^T$ . The fast mode uses the desired angular rates and outputs the required moments to stabilize the system. Force required for each pair of motor-propellers can then be determined using the moments from the fast mode. As a result, this inner loop dynamic inversion controller takes care of the attitude stabilization of the quadrotor. Attitude control can be commanded from pilot inputs at this point. Regarding inner controller performance, simulation results as well as implementation results from a 3DR X8 quadcopter are discussed in this section.

For autonomous flight, the inner stability controller is augmented with an outer controller that provides the desired attitude to the inner controller based on predetermined waypoints. The main goal of this outer loop is to allow the quadrotor to change position  $[X \ Y \ Z]^T$  autonomously within an earth reference frame. A NLDI architecture can be adopted to design this outer loop controller and translational equations of motion need to be used for this purpose. In order to accomplish this task, state feedback for positions and velocities are required and they can be used directly from a GPS sensor or they can be estimated by using vision-based EKF as it will be presented in Chapter 4. Regarding



outer controller performance, simulation results using a NLDI architecture are presented in this section. However, implementation results for X and Y position control are obtained using a cascade PID control architecture for simplicity. Regarding Z position control or altitude hold mode flight, implementation results are obtained using the NLDI architecture described in this section.

### 3.2.1 Inner loop Dynamic Inversion

#### A. Slow mode

The slow mode outputs angular rates that can be calculated by inverting Eq. (2.25). The Kinematic equations in state space form are defined below:

$$\dot{\mathbf{x}}_1 = g(\mathbf{x})_1 u_1 \quad (3.1)$$

where,  $\mathbf{x}_1 = [\dot{\phi} \ \dot{\theta} \ \dot{\psi}]^T$  is the state vector of Euler rates and  $u_1 = [p \ q \ r]^T$  can represent the desired states. A dynamic inversion control input to invert the slow mode dynamics can be expressed as follows:

$$u_1(\mathbf{x}) = \begin{bmatrix} p_D \\ q_D \\ r_D \end{bmatrix} = \begin{bmatrix} 1 & \sin\phi \tan\theta & \cos\phi \tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi/\cos\theta & \cos\phi/\cos\theta \end{bmatrix}^{-1} \begin{bmatrix} U_\phi \\ U_\theta \\ U_\psi \end{bmatrix} \quad (3.2)$$

where,  $v(\mathbf{x})_1 = [U_\phi \ U_\theta \ U_\psi]^T$  is the pseudo control vector that will stabilize the closed loop system.  $[p_D \ q_D \ r_D]^T$  is the vector of desired angular rates that will drive the linear behavior of the Kinematics. The pseudo control vector  $v(\mathbf{x})_1$  can be determined using a linear controller, as follows:

$$v(\mathbf{x})_1 = \begin{bmatrix} U_\phi \\ U_\theta \\ U_\psi \end{bmatrix} = \begin{bmatrix} k_\phi(\phi_D - \phi) \\ k_\theta(\theta_D - \theta) \\ k_\psi(\psi_D - \psi) \end{bmatrix} \quad (3.3)$$

The closed loop feedback linearized dynamics of the slow mode can be finally written as:

$$\dot{\mathbf{x}}_1 \cong v(\mathbf{x})_1 \rightarrow \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \cong \begin{bmatrix} k_\phi(\phi_D - \phi) \\ k_\theta(\theta_D - \theta) \\ k_\psi(\psi_D - \psi) \end{bmatrix} \quad (3.4)$$

### B. Fast mode

The fast mode utilizes the desired angular rates and outputs the desired moments. Eq. (2.18) can be written in state space form as:

$$\dot{\mathbf{x}}_2 = f(\mathbf{x})_2 + g(\mathbf{x})_2 u_2 \quad (3.5)$$

where,  $\mathbf{x}_2 = [\dot{p} \ \dot{q} \ \dot{r}]^T$  is the state vector of angular accelerations and  $u_2 = [M_{xD} \ M_{yD} \ M_{zD}]^T$  are the required moments that can be treated as the inputs of the system. Functions  $f(\mathbf{x})_2$  and  $g(\mathbf{x})_2$  can be defined as:

$$f(\mathbf{x})_2 = - \begin{bmatrix} qr(I_{zz} - I_{yy})/I_{xx} \\ pr(I_{xx} - I_{zz})/I_{yy} \\ pq(I_{yy} - I_{xx})/I_{zz} \end{bmatrix} \quad (3.6)$$

$$g(\mathbf{x})_2 = \begin{bmatrix} M_x/I_{xx} & 0 & 0 \\ 0 & M_y/I_{yy} & 0 \\ 0 & 0 & M_z/I_{zz} \end{bmatrix} \quad (3.7)$$

Based on Eq. (3.2), a dynamic inversion control input that inverts fast mode dynamics can be expressed as follows:

$$u_2(\mathbf{x}) = \begin{bmatrix} M_{xD} \\ M_{yD} \\ M_{zD} \end{bmatrix} = g^{-1}(\mathbf{x})_2 [v(\mathbf{x})_2 - f(\mathbf{x})_2] \quad (3.8)$$

where,  $v(\mathbf{x})_2 = [U_p \ U_q \ U_r]^T$  is a fast mode pseudo controller.

$$u_2(\mathbf{x}) = \begin{bmatrix} M_{xD} \\ M_{yD} \\ M_{zD} \end{bmatrix} = \begin{bmatrix} U_p I_{xx} \\ U_q I_{yy} \\ U_r I_{zz} \end{bmatrix} = \begin{bmatrix} k_{p\omega_x} (p_D - p) I_{xx} \\ k_{p\omega_y} (q_D - q) I_{yy} \\ k_{p\omega_z} (r_D - r) I_{zz} \end{bmatrix} \quad (3.9)$$

Differential inertia terms have been left out to ensure zero dynamics while the angular velocity error is zero (Wang, He, Zhang & He, 2013). A proportional controller can be used as a pseudo controller for the system as follows:

$$\dot{\mathbf{x}}_2 \cong \mathbf{v}(\mathbf{x})_2 \rightarrow \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} \cong \begin{bmatrix} k_{p\omega_x} (p_D - p) \\ k_{p\omega_y} (q_D - q) \\ k_{p\omega_z} (r_D - r) \end{bmatrix} \quad (3.10)$$

After performing the slow mode and fast mode dynamic inversions, the second order rotational dynamics of the closed loop system are described in the following system of differential equations (Wang, He, Zhang & He, 2013):

$$\begin{aligned} \ddot{\phi} &= k_{p\omega_x} k_{\phi} (\phi_D - \phi) - k_{p\omega_x} \omega_x \cong -2\xi_{\phi} \omega_{n_{\phi}} \dot{\phi} - \omega_{n_{\phi}}^2 (\phi - \phi_D) \\ \ddot{\theta} &= k_{p\omega_y} k_{\theta} (\theta_D - \theta) - k_{p\omega_y} \omega_y \cong -2\xi_{\theta} \omega_{n_{\theta}} \dot{\theta} - \omega_{n_{\theta}}^2 (\theta - \theta_D) \\ \ddot{\psi} &= k_{p\omega_z} k_{\psi} (\psi_D - \psi) - k_{p\omega_z} \omega_z \cong -2\xi_{\psi} \omega_{n_{\psi}} \dot{\psi} - \omega_{n_{\psi}}^2 (\psi - \psi_D) \end{aligned} \quad (3.11)$$

The proportional gains from Eqs. (3.4) and (3.10) can be calculated according to the desired natural frequency and damping of the dynamic system response:

$$\begin{aligned} k_{\phi} &= \frac{\omega_{n_{\phi}}}{2 \xi_{\phi}}, k_{p\omega_x} = 2 \xi_{\phi} \omega_{n_{\phi}} \\ k_{\theta} &= \frac{\omega_{n_{\theta}}}{2 \xi_{\theta}}, k_{p\omega_y} = 2 \xi_{\theta} \omega_{n_{\theta}} \\ k_{\psi} &= \frac{\omega_{n_{\psi}}}{2 \xi_{\psi}}, k_{p\omega_z} = 2 \xi_{\psi} \omega_{n_{\psi}} \end{aligned} \quad (3.12)$$

### 3.2.2 Outer loop Dynamic Inversion

The outer loop is designed to allow the vehicle navigate within a three dimensional space. For this reason the goal of the outer loop dynamic inversion controller is to cancel out the non-linearity in the translational equations that describe the translational dynamics of the quadrotor. To apply the feedback linearization we can recall equations (2.15) and (2.22). The force equations can be represented in the earth reference frame as follows:

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = DCM \begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{z} \end{bmatrix} \quad (3.13)$$

Equation 3.13 can be rewritten as follows:

$$\ddot{x} = \frac{-F_z}{m} (\sin\phi \sin\psi + \cos\phi \sin\theta \cos\psi) \quad (3.13)$$

$$\ddot{y} = \frac{-F_z}{m} (-\cos\phi \sin\theta \sin\psi + \sin\phi \sin\psi) \quad (3.14)$$

$$\ddot{z} = g - \frac{F_z}{m} (\cos\phi \cos\theta) \quad (3.15)$$

The inversion of equations (3.13) and (3.14) would output the required attitude angles (roll and pitch commands) that are used within the inner loop in order to change the quadcopter position. In addition, inversion of equation (3.15) yields to the required total thrust that the quadcopter needs to hold for a commanded altitude. The inversion of these equations are derived as follows (Ireland, Vargas & Anderson, 2015):

$$\phi_d = -\sin^{-1} \left\{ \frac{m [(\sin\psi)u_x(x) - (\cos\psi)u_y(x)]}{F_{zd}} \right\} \quad (3.16)$$

$$\theta_d = -\tan^{-1} \left\{ \frac{m [(\cos\psi)u_x(x) + (\sin\psi)u_y(x)]}{F_{zd}} \right\} \quad (3.17)$$

$$F_{zd} = \frac{m [u_z(x) - g]}{\cos\phi \cos\theta} \quad (3.18)$$

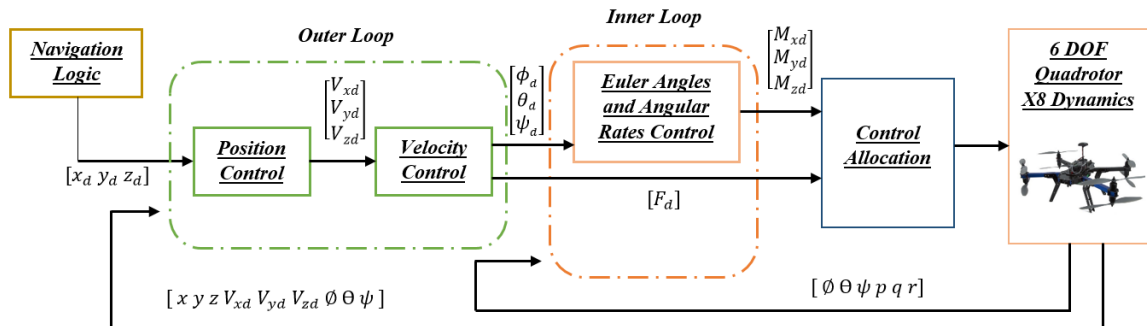
where  $u_x(x)$ ,  $u_y(x)$  and  $u_z(x)$  represent the virtual controllers that in this case are designed to produce specific second order closed loop dynamics which would govern translational motion in the three dimensional space. Once the feedback linearization is performed, the second order dynamics are defined as:

$$\begin{aligned}\ddot{x} &= u_x(x) = -2\xi_x \omega_{n_x} \dot{x} - \omega_{n_x}^2 (x - x_d) = -K_{V_x} \dot{x} + K_{V_x} K_{P_x} (x_d - x) \\ \ddot{y} &= u_y(x) = -2\xi_y \omega_{n_y} \dot{y} - \omega_{n_y}^2 (y - y_d) = -K_{V_y} \dot{y} + K_{V_y} K_{P_y} (y_d - y) \\ \ddot{z} &= u_z(x) = -2\xi_z \omega_{n_z} \dot{z} - \omega_{n_z}^2 (z - z_d) = -K_{V_z} \dot{z} + K_{V_z} K_{P_z} (z_d - z)\end{aligned}\quad (3.19)$$

The outer loop controller gains can be calculated according to the desired natural frequency and damping of the dynamic system response:

$$\begin{aligned}k_{P_x} &= \frac{\omega_{n_x}}{2 \xi_x}, k_{pV_x} = 2 \xi_x \omega_{n_x} \\ k_{P_y} &= \frac{\omega_{n_y}}{2 \xi_y}, k_{pV_y} = 2 \xi_y \omega_{n_y} \\ k_{P_z} &= \frac{\omega_{n_z}}{2 \xi_z}, k_{pV_z} = 2 \xi_z \omega_{n_z}\end{aligned}\quad (3.20)$$

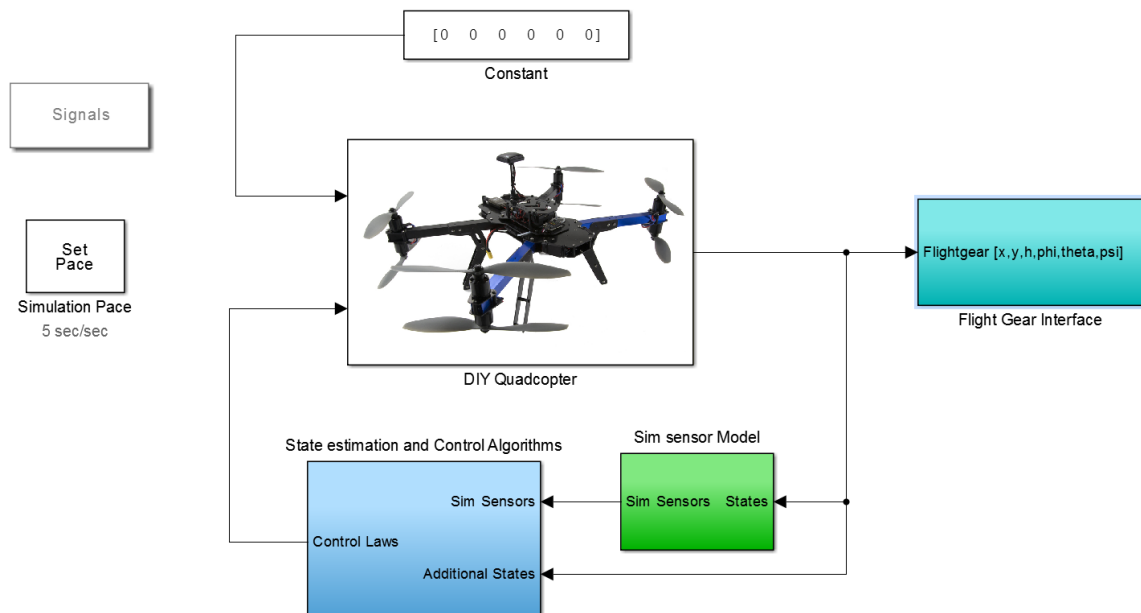
A fully autonomous control architecture can be designed by using the results from control allocation equations (2.4) to (2.11), the desired moment equations derived from the inner loop dynamic inversion (3.9) and the desired thrust and attitude command derived from the outer loop (3.16) to (3.17).



**Figure 16** Control Architecture for fully autonomous flight

### 3.3 Simulation Results

In this subsection, the simulation environment used to test the previously proposed control algorithms is described. This simulation environment is composed of four main blocks. The core of this simulation is represented by the dynamic model of a 6 DOF rigid body. The twelve non-linear equations of motion that describe a 6 DOF quadrotor dynamics system are being solved within this block. A sensor block is being used in this environment to model different types of sensors with different noise levels. The next block is where the control and state estimation algorithms are developed. Within this block, the estimated states are being used for designing the navigation and attitude control architectures. Finally, a Flight Gear Interface block is utilized for displaying purposes. This simulation environment allows the user to command the quadcopter using joystick inputs as well as autonomous navigation mission.



**Figure 17** 3DR-X8 Quadcopter Simulation Environment developed using Matlab/Simulink Software

This simulation environment is used to test the inner loop and outer loop controllers. Table 2 presents the pseudo controller gains for the inner and outer loop. Gains for the inner pseudo controller are estimated using a natural frequency of 10 rad/s and a damping ratio of 0.7 (Wang, He, Zhang & He, 2014) for the roll and pitch axis. As expected, the yaw axis has a slower response because the pseudo controllers are designed for a natural frequency of 5 rad/s and a damping ratio of 0.7. In contrast the outer pseudo controller is designed with a natural frequency of 6.3 rad/s and a damping ratio of 0.7. Values for natural frequency and damping ratio are chosen based on simulation experience. By replacing these parameters in equations 3.12 and 3.20, pseudo controller gains can be calculated.

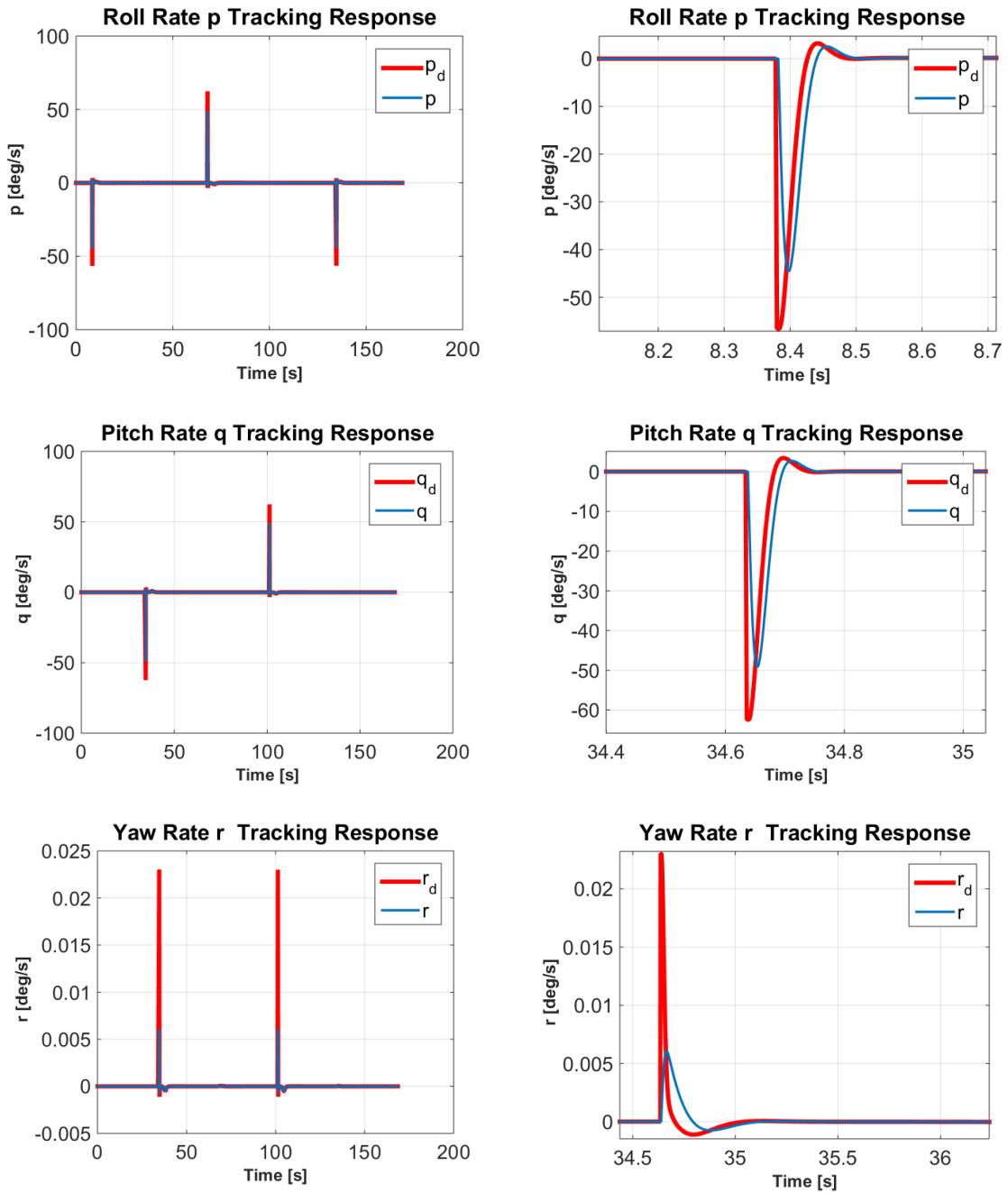
**Table 2** Calculated Baseline Control Gains

Outer loop (Slow mode)				Inner loop (Fast mode)			
Position		Velocity		Euler Angles		Angular rates	
$k_{px}$	4.487	$k_{pv_x}$	8.796	$k_{\phi}$	7.14	$k_{p\omega_x}$	14
$k_{py}$	4.487	$k_{pv_y}$	8.796	$k_{\theta}$	7.14	$k_{p\omega_x}$	14
$k_{pz}$	4.487	$k_{pv_z}$	8.796	$k_{\psi}$	3.57	$k_{p\omega_x}$	7

### 3.3.1 Autonomous Mission Results

Figure 18 through Figure 20 present results for tracking response of the inner and outer loop controller while the vehicle is performing an autonomous mission of five consecutive waypoints. This simulation incorporates a waypoint logic in which the next waypoint is targeted only after the vehicle has reached a region inside a circle of 1ft of radius from the actual waypoint. In this way, if the vehicle does not reach as close as 1ft to the waypoint, it will generate control commands to meet this requirement before the logic commands the next waypoint. Figure 20 shows a three dimensional representation of the vehicle path. The vehicle is expected to take off, perform a roll maneuver in order

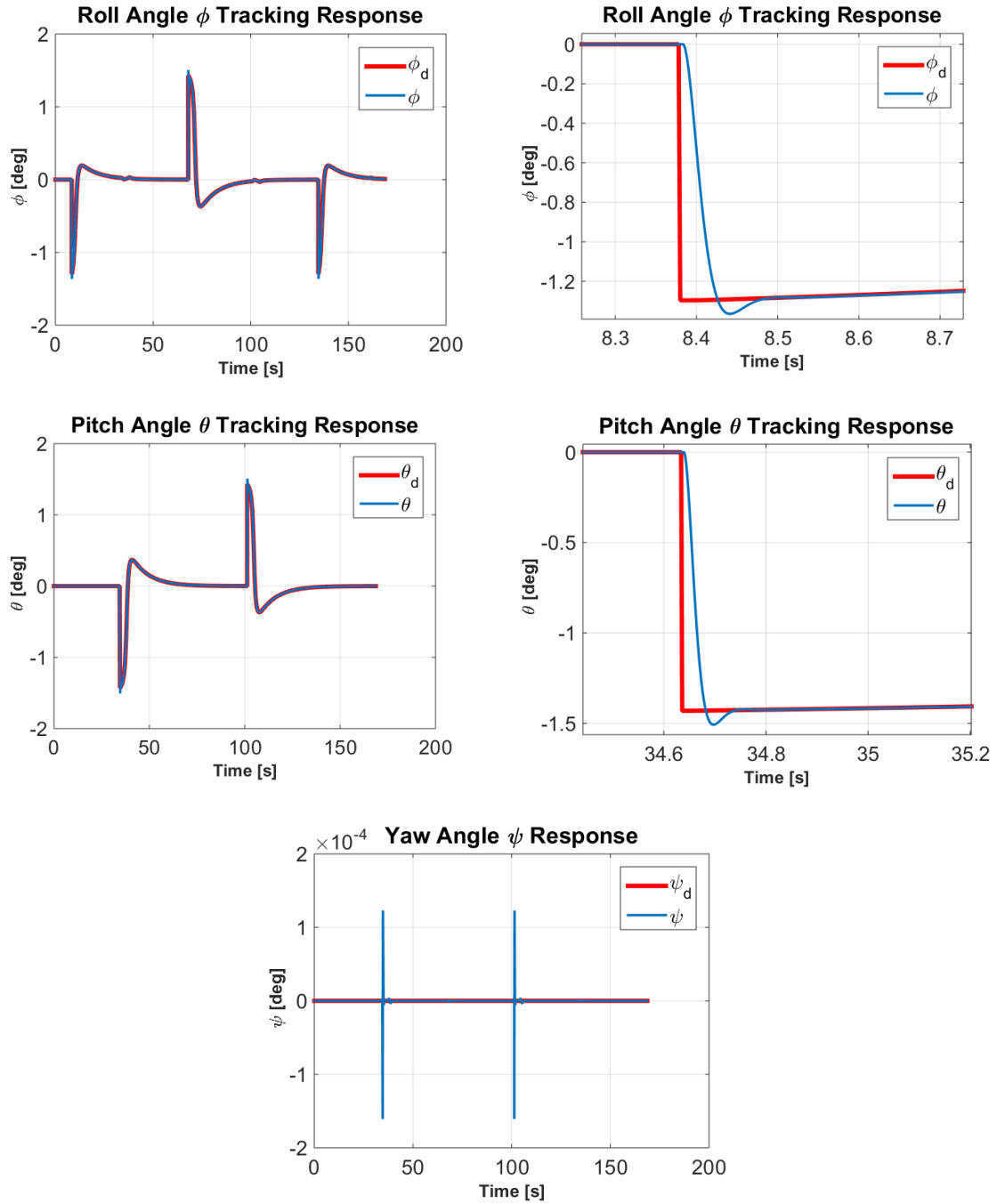
to move 15 ft in the Y-axis. Then, it moves 30 ft in the X axis for which a pitch maneuver is required. Finally the quadcopter completes a square trajectory and lands.



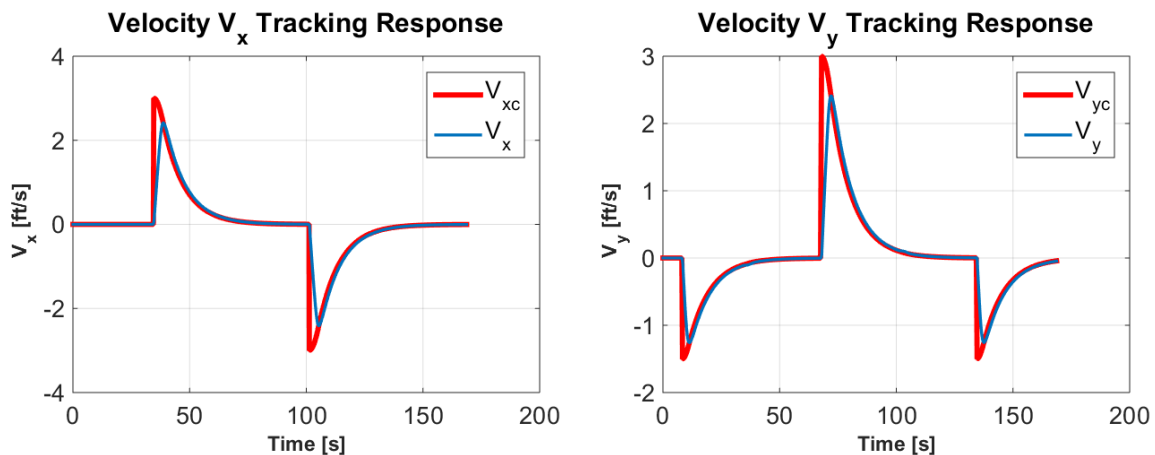
**Figure 18** Inner loop-Fast mode tracking response for an autonomous mission with 5

waypoints





**Figure 19** Inner loop-Slow mode tracking response for an autonomous mission with 5 waypoints



3D Position Autonomous Mission

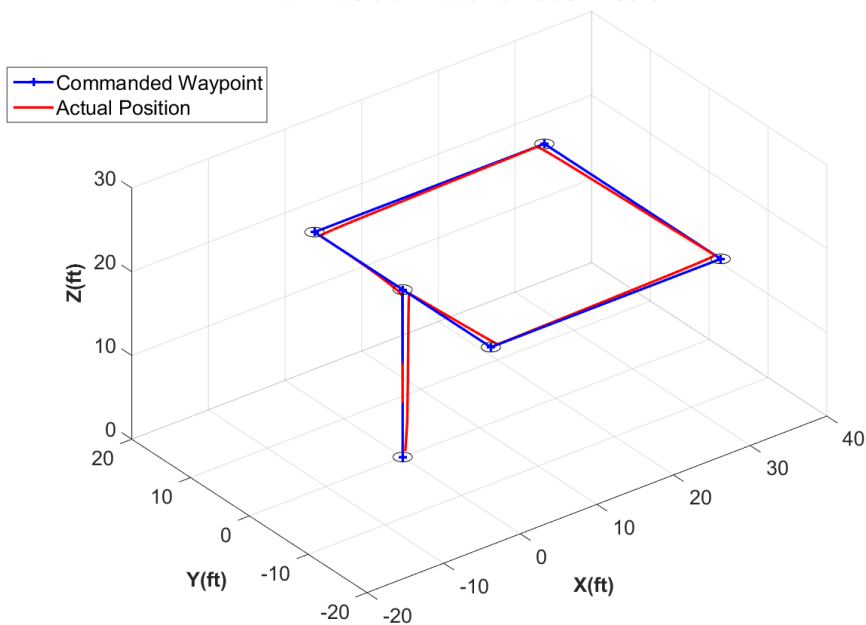


Figure 20 Outer loop tracking response for an autonomous mission with 5 waypoints

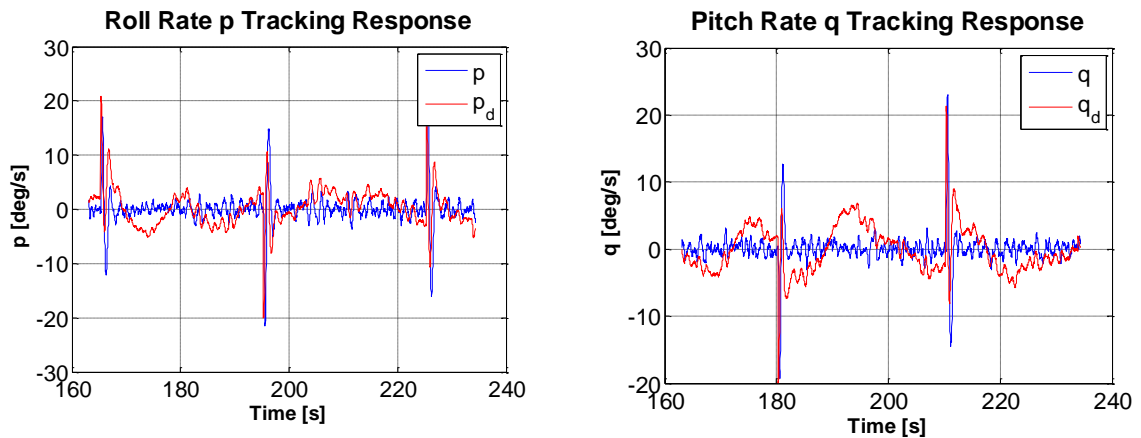
Table 3 Tracking error Simulation results

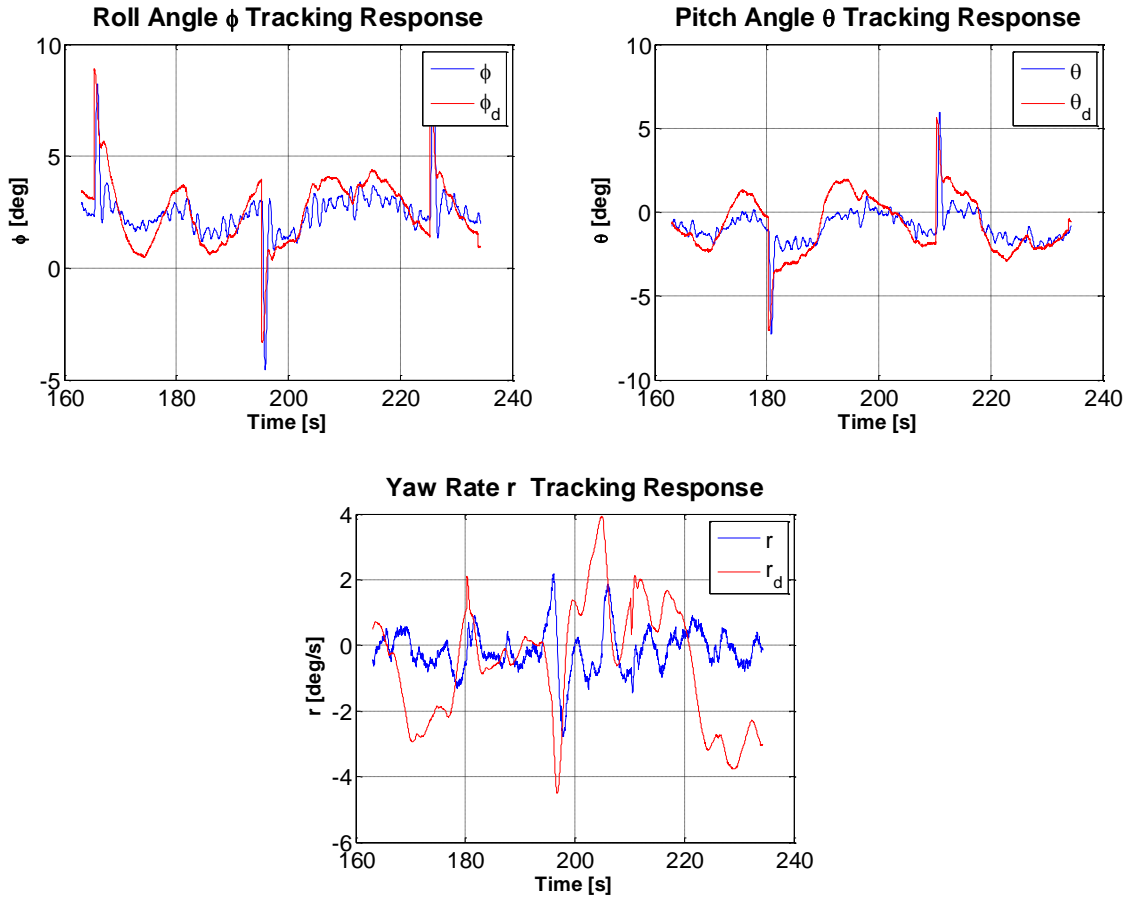
	$[\phi, p, V_x, P_x]$	$[\theta, q, V_y, P_y]$	$[\psi, r, V_z, P_z]$
Euler Angles	2.1178e-08	1.2137e-09	3.0007e-09
Angular Rates	1.0264e-05	5.2143e-07	7.0049e-08
Velocity	2.2834e-05	0.0120196	3.4534e-05
Position	4.9454e-03	7.3458e-04	2.6285e-06

Table 3 shows the tracking error results for the inner and outer control loops obtained from the simulated waypoint mission. It can be noticed that the tracking performance for Euler angles and angular rates is similar. In addition, the outer loop tracking error results follow a similar pattern; therefore, all the specified waypoints are reached by the vehicle and the mission is successfully completed.

### 3.4 Implementation Results

The previous controller is implemented on a 3DR quadcopter UAV as discussed in Chapter 1. The proposed controller is developed and modified within Matlab/Simulink to make it deployable on the Pixhawk Autopilot board. This testbed is equipped with the necessary sensor suite in order to provide state feedback that is required for the inner and outer loop dynamic inversions. Figure 21 shows results for inner controller tracking response during an autonomous flight test with 5 waypoints similar to the one analyzed in the simulation section of this thesis. This autonomous flight is achieved by using a GPS sensor which is used to obtain the velocity and position of the vehicle at every sample time.





**Figure 21** Inner loop tracking response for an autonomous mission with 5 waypoints

**Table 4** Tracking error for the inner loop at implementation

	$[\phi, p]$	$[\theta, q]$	$[\psi, r]$
Euler Angles	0.1692	0.1324	---
Angular Rates	0.2539	0.2330	4.4198

Table 4 shows the tracking error results for the inner control loop obtained from the implementation waypoint mission. It can be noticed that the tracking errors are higher than those obtained for Euler angles and angular rates in the simulation environment. This behavior is expected since the simulation only analyzed the ideal scenario without

external disturbances such as wind conditions or turbulence effects. As it can be seen, there is not a tracking error value for the yaw angle controller. This is a modification developed to the implementation code that allows the pilot to command an angular rate instead of an angle for the yaw axis. This was implemented for pilot convenience only in the yaw axis.

### **3.4.1 Attitude Controller Tuning Process**

In order to obtain a desirable behavior of the quadcopter during the implementation process, the gains from the inner loop controllers have to undergo a tuning process. The gains obtained from simulation cannot be directly implemented into the actual quadcopter for two main reasons. Assumptions made in order to simplify the dynamic modeling of the vehicle as well as the actuator dynamics model can lead to different responses between the simulation environment and the actual quadcopter behavior. Simulation gains are adjusted to account for these “un-modeled” dynamics and tuned according to the quadcopter behavior and transient response. In addition to this, the controller is required to be modified in such a way that control calculations inside the onboard computer can be estimated using units of degrees instead of radians. As a result, the controllers’ gains are scaled down. This modification is necessary in order to maintain a reasonable accuracy level in the calculations inside the onboard computer. Also, the range of gains in which the system would perform in a stable condition is increased. In this way, the tuning process becomes less demanding. During the tuning process, an integral term is included in order to obtain a better quadcopter performance. Table 5 and Table 6 show the gains used for the inner and outer loop pseudo controller that run inside the on board computer.

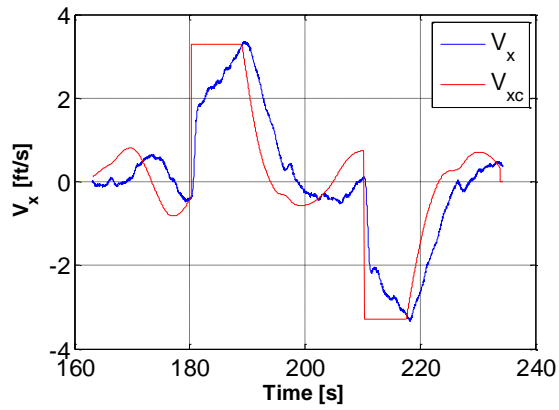
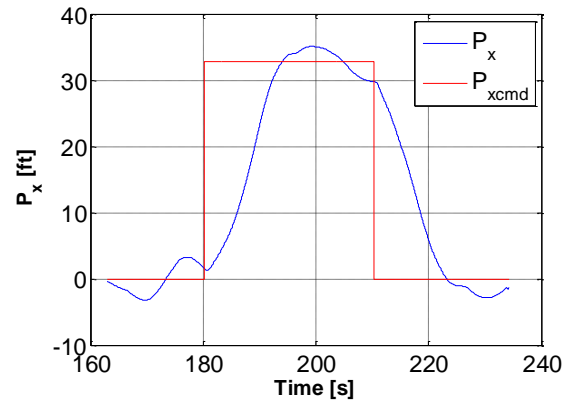
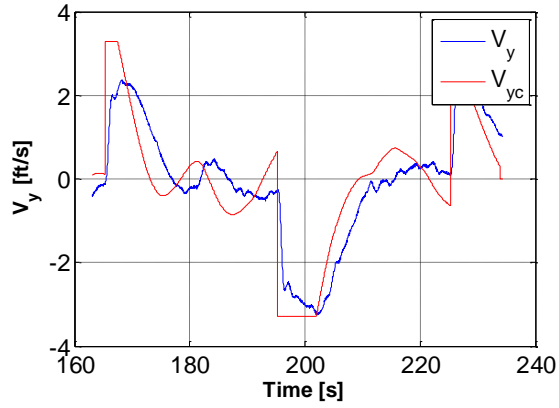
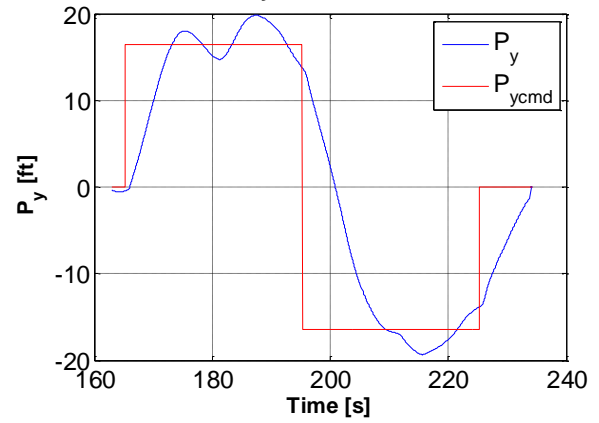
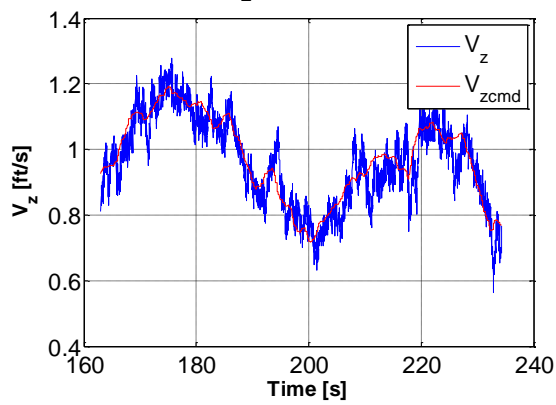
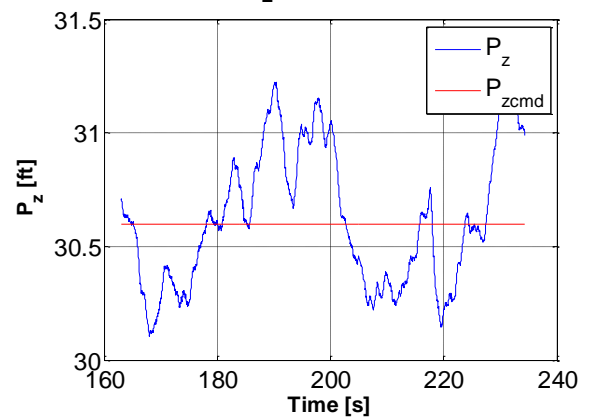
**Table 5** Proportional Gains

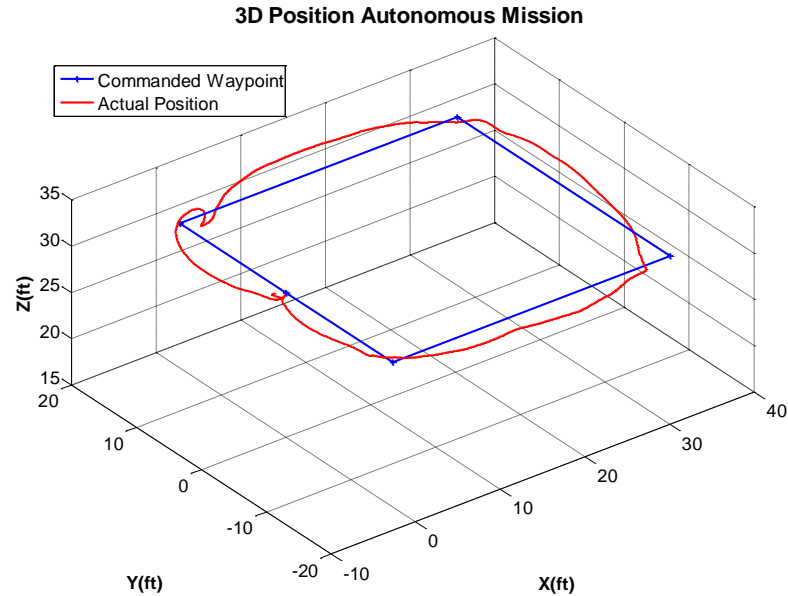
Outer loop				Inner loop			
Position		Velocity		Euler Angles		Angular rates	
$k_{px}$	0.5	$k_{pv_x}$	5.75	$k_{\phi}$	3.075	$k_{p\omega_x}$	0.48
$k_{py}$	0.5	$k_{pv_y}$	5.75	$k_{\theta}$	3.075	$k_{p\omega_y}$	0.48
$k_{pz}$	0.22	$k_{pv_z}$	0.28	$k_{\psi}$	1.3998	$k_{p\omega_z}$	0.48

**Table 6** Integral Gains

Outer loop				Inner loop			
Position		Velocity		Euler Angles		Angular rates	
$k_{ix}$	0	$k_{iv_x}$	0.5	$k_{i\phi}$	0	$k_{i\omega_x}$	0.46
$k_{iy}$	0	$k_{iv_y}$	0.5	$k_{i\theta}$	0	$k_{i\omega_y}$	0.46
$k_{iz}$	0.06	$k_{iv_z}$	0.28	$k_{i\psi}$	0	$k_{i\omega_z}$	0.46

In addition, Figure 22 shows results for outer controller tracking response during an autonomous flight test with 5 waypoints similar to the one analyzed in the simulation section of this thesis. The quadcopter would take-off following pilot commands and the autonomous waypoint mission would start after a manual switch enables the autonomous mode. At the end of the autonomous mission, the pilot would recover manual control by moving the switch back to manual position. The autonomous mission is based on a 15 feet side square trajectory which starts at coordinates (0, 0, altitude at which the switch is enabled) in the three dimensional axis.

Velocity  $V_x$  Tracking ResponsePosition  $P_x$  Tracking ResponseVelocity  $V_y$  Tracking ResponsePosition  $P_y$  Tracking ResponseVelocity  $V_z$  Tracking ResponsePosition  $P_z$  Tracking Response



**Figure 22** Outer loop tracking response for an autonomous mission with 5 waypoints

**Table 7** Tracking error for the outer loop at implementation

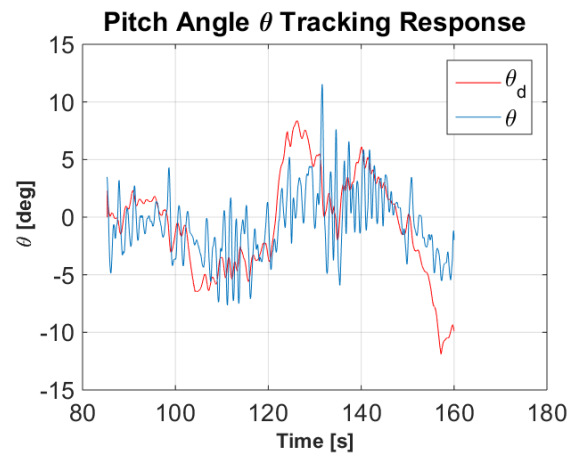
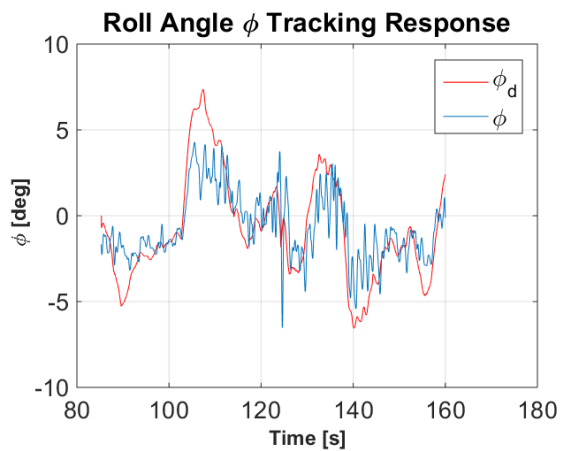
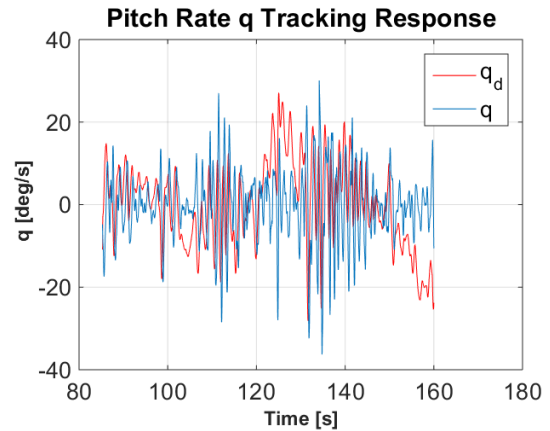
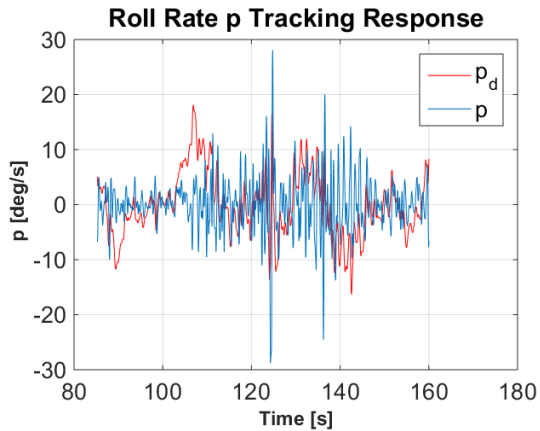
	$[V_x, P_x]$	$[V_y, P_y]$	$[V_z, P_z]$
Velocity	0.0748	0.0308	0.0133
Position	0.8903	0.7412	0.0261

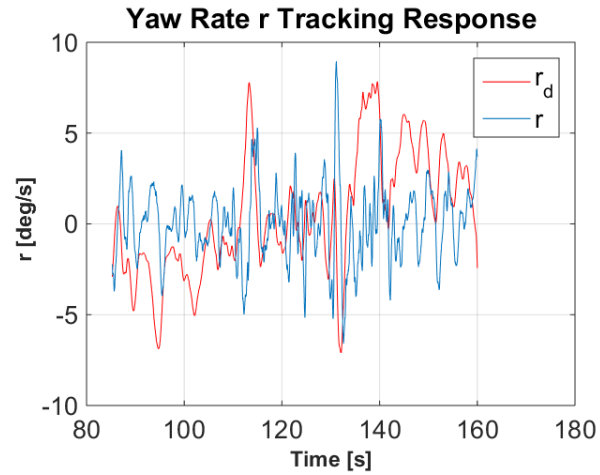
Table 7 shows the tracking error results for the outer control loop obtained from the implementation waypoint mission. It can be noticed that the tracking errors are higher than the ones obtained for velocities and position in the simulation environment. This behavior is expected since the simulation only analyzed the ideal scenario, as it was suggested before.

In addition to a waypoint type controller, results for a trajectory controller are presented in this section. The following results correspond to an autonomous flight that is following a desired trajectory. In contrast to the waypoint mission previously analyzed, this trajectory is constantly updating the commanded position with respect to time

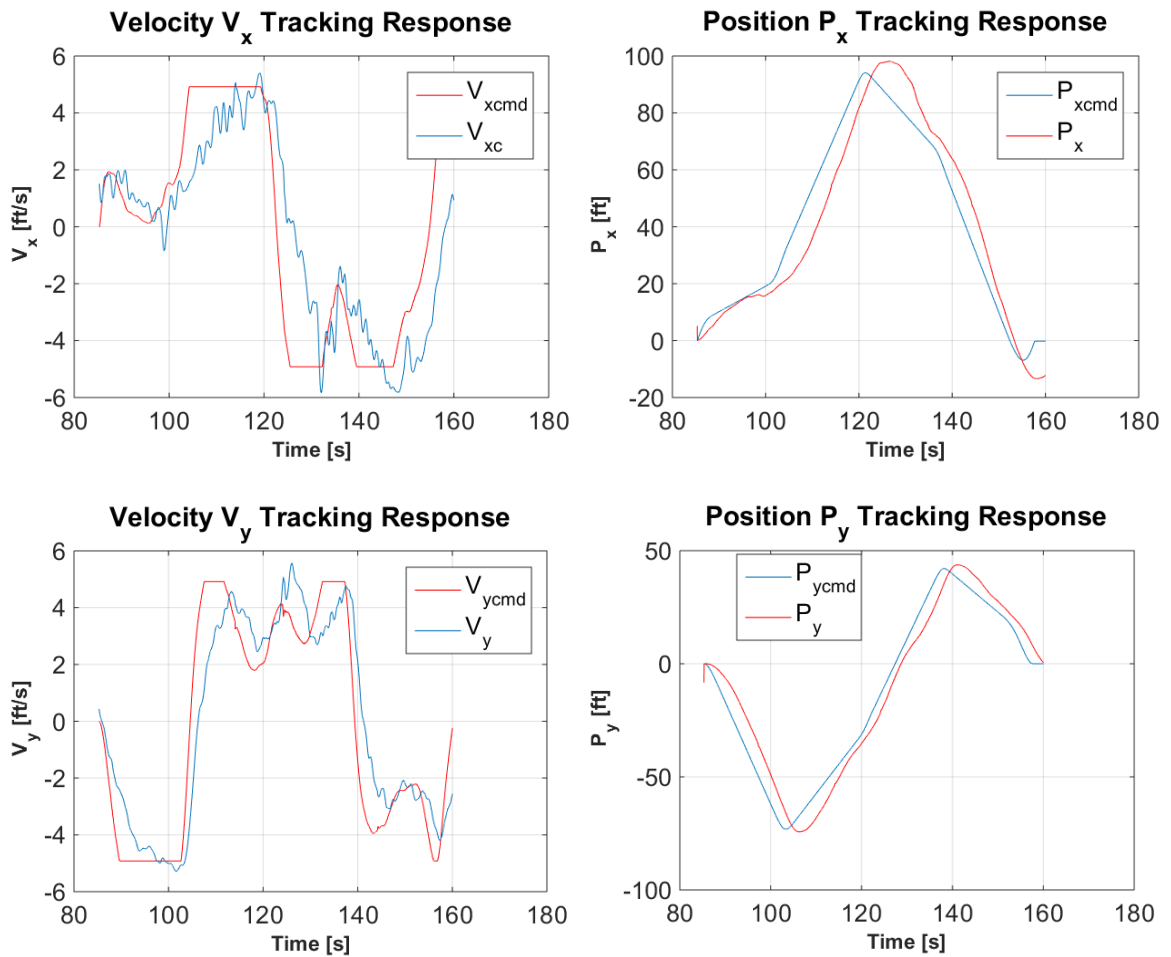


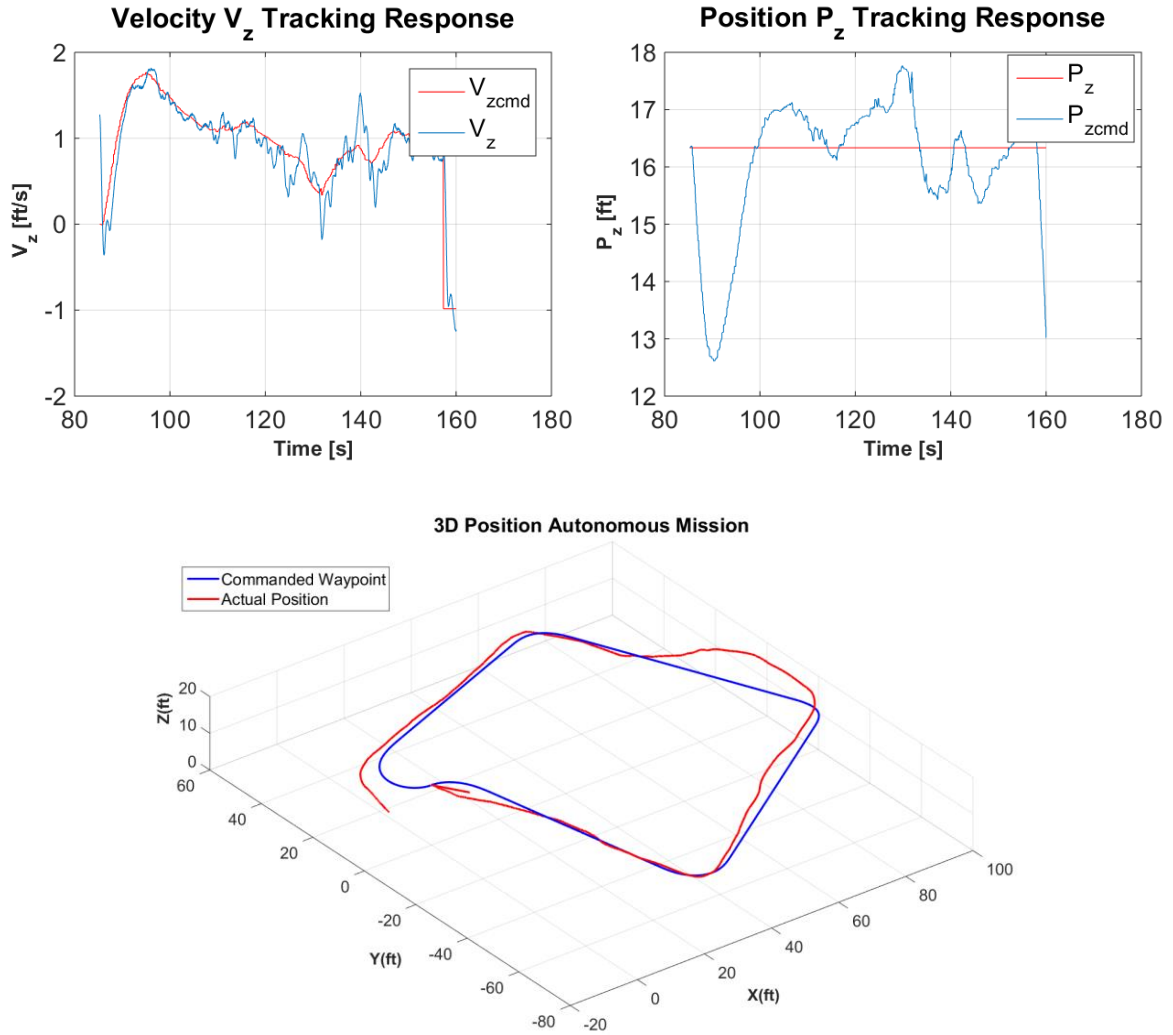
without verification of the vehicle distance to the desired trajectory. However, this trajectory controller provides a smoother navigation response than the waypoint type mission and logic. Results for the inner and outer loop tracking response are presented in the following figures:





**Figure 23** Inner loop tracking response for an autonomous trajectory





**Figure 24** Outer loop tracking response for an autonomous trajectory

**Table 8** Tracking error for the outer loop at implementation (trajectory)

	$[V_x, P_x]$	$[V_y, P_y]$	$[V_z, P_z]$
Velocity	0.2990	0.1857	0.0369
Position	0.6130	1.0113	0.2733

Table 8 shows the tracking error results for the outer control loop obtained from the implementation trajectory mission. It can be noticed that the tracking error results are higher than the ones obtained for velocities and position when using a waypoint

controller. However the tracking performance that has been reached allows the vehicle to successfully complete the mission.

The results presented in this section show that the proposed NLDI control architecture can be implemented on a quadrotor type UAV in order to perform a fully autonomous flight. The advantages of a nonlinear controller regarding analysis of hard nonlinearities, management of model uncertainties and design simplicity will then increase autonomy of this type of UAVs.

## **4. AIS for Failure Detection & Identification Problem**

### **4.1 Generation of Detectors**

The most fundamental idea of the AIS used for failure detection is that the presence of an abnormal situation can be detected when a certain projection of “features” does not match with any projection of features at nominal conditions that are pre-established through an off-line training process of the AIS system. This off-line training process is the key point in the design of the AIS detection scheme. The detection scheme developed in this thesis utilizes 47 two-dimensional projections. The nominal pre-established projections can be represented as clusters that will constitute self cells which represent the dynamic signature of the vehicle for that specific set of features. In a similar way, detectors are originated through a clustering process of the empty space or non-self two-dimensional space of each projection. Detectors are analogous to the antibodies or specialized cells generated during the humoral immunity process performed by the B-cells in the acquired immune system of living organisms. These antibodies known also as T cells are responsible for seeking intruders (viruses), binding them and marking them for destruction under the principle of positive selection. Using the same analogy, detectors are implemented within this detection scheme to detect an abnormal situation based on features projections that fall inside the two dimensional space represented by the detectors clusters. A detection logic is then implemented in order to declare an abnormal situation based on the detectors activation history.

In order to successfully determine the best features projections for capturing the abnormal conditions, this AIS detection scheme must be designed considering information about the operational envelope of the vehicle, the targeted faulty systems and

type of abnormal conditions. Table 9 presents the features that are utilized in this AIS detection scheme.

**Table 9** Features used in AIS detection scheme

$\varphi$	Roll attitude	$\varphi\_cmd$	Roll reference command
$\theta$	Pitch attitude	$\theta\_cmd$	Pitch reference command
$\Psi$	Yaw attitude	$\Psi\_cmd$	Yaw reference command
$p$	Roll rate	$p\_cmd$	Roll rate reference command
$q$	Pitch rate	$q\_cmd$	Pitch rate reference command
$r$	Yaw rate	$r\_cmd$	Yaw rate reference command
$A_x$	x body acceleration		
$A_y$	y body acceleration		
$A_z$	z body acceleration		

In theory, these 15 features could be used to generate a 15-dimension hyperspace which will define self and non-self hyperspaces. However this higher dimensionality would consume a high processing power which is not practical for implementation purposes. This research is focused only on two-dimensional projections that can provide high detection activity and low processing power consumption. The maximum number of two-dimensional projections that can be obtained from combinations of 15 features are calculated as follows:

$$N_{self} = C_N^{N_{max}} = \frac{N_f!}{N_{max}!(N_f - N_{max})!} = \frac{15!}{2!13!} = 105$$

Out of the 105 possible two-dimensional projections only 47 projections are analyzed in this effort in order to exclude repetitive projections and save processing time.

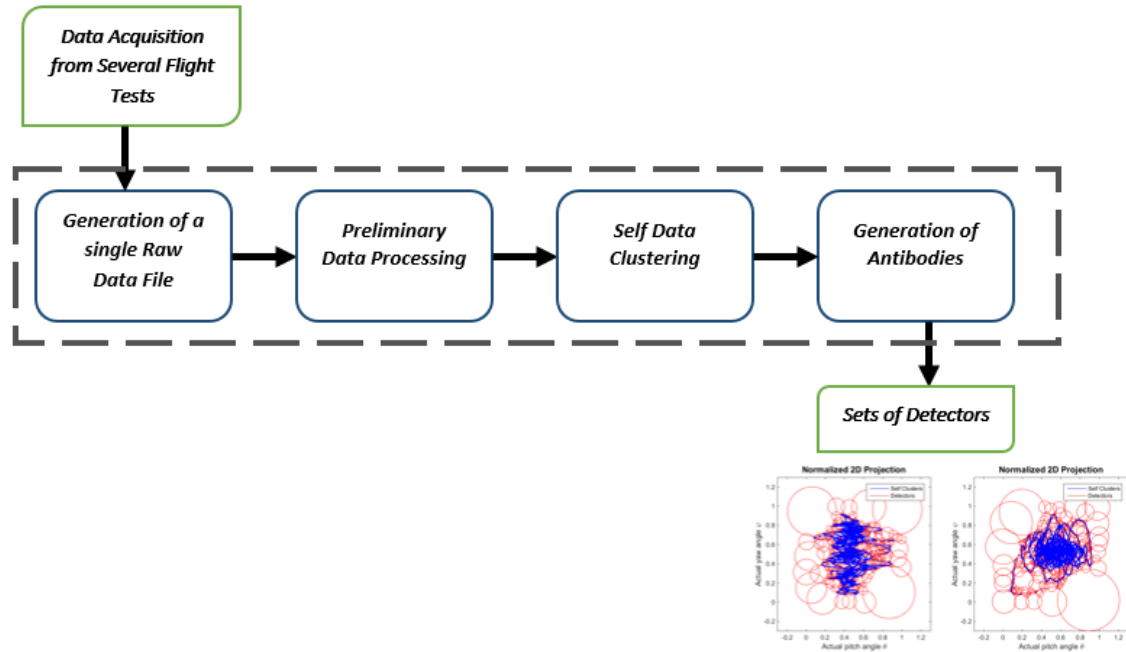
Table 10 present the list of two dimensional projections that are analyzed in this thesis:

**Table 10.** List of 2D-Projections for the AIS Detection Scheme

Self #	Features		Self #	Features		Self #	Features	
<b>1</b>	$\varphi$	$\varphi\_cmd$	<b>17</b>	$\theta$	$Ay$	<b>33</b>	$q$	$Az$
<b>2</b>	$\varphi$	$\theta$	<b>18</b>	$\theta$	$Az$	<b>34</b>	$r$	$Ax$
<b>3</b>	$\varphi$	$q\_cmd$	<b>19</b>	$\Psi$	$p$	<b>35</b>	$r$	$Ay$
<b>4</b>	$\varphi$	$r\_cmd$	<b>20</b>	$\Psi$	$q$	<b>36</b>	$r$	$Az$
<b>5</b>	$\varphi$	$p$	<b>21</b>	$\Psi$	$r$	<b>37</b>	$Ax$	$Ay$
<b>6</b>	$\varphi$	$q$	<b>22</b>	$\Psi$	$Ax$	<b>38</b>	$Ax$	$Az$
<b>7</b>	$\varphi$	$r$	<b>23</b>	$\Psi$	$Ay$	<b>39</b>	$Ay$	$Az$
<b>8</b>	$\varphi$	$Ax$	<b>24</b>	$\Psi$	$Az$	<b>40</b>	$r\_cmd$	$p$
<b>9</b>	$\varphi$	$Ay$	<b>25</b>	$p$	$q$	<b>41</b>	$r\_cmd$	$q$
<b>10</b>	$\varphi$	$Az$	<b>26</b>	$p$	$r$	<b>42</b>	$r\_cmd$	$r$
<b>11</b>	$\theta$	$\theta\_cmd$	<b>27</b>	$p$	$Ax$	<b>43</b>	$r\_cmd$	$Ax$
<b>12</b>	$\theta$	$\Psi$	<b>28</b>	$p$	$Ay$	<b>44</b>	$r\_cmd$	$Ay$
<b>13</b>	$\theta$	$p$	<b>29</b>	$p$	$Az$	<b>45</b>	$r\_cmd$	$Az$
<b>14</b>	$\theta$	$q$	<b>30</b>	$q$	$r$	<b>46</b>	$r\_cmd$	$\theta$
<b>15</b>	$\theta$	$r$	<b>31</b>	$q$	$Ax$	<b>47</b>	$r\_cmd$	$\Psi$
<b>16</b>	$\theta$	$Ax$	<b>32</b>	$q$	$Ay$			

Prior to generating detectors, many flight tests need to be performed at nominal conditions in order to log all the features specified in Table 9. These features are saved to the micro SD card that is inside the onboard computer. In order to successfully determine the nominal two-dimensional space for each of the projections, several flight tests are required. This fact guarantees that the dynamic signature of the vehicle is fully covered within the nominal two-dimensional space. After each flight test, these sets of features are saved into a data bank that is required during the detectors generation process. The process of detectors generation takes place during the off-line training process. This process is usually time consuming due to the data processing algorithms that multiple sets of data must undergo. These algorithms are based on the “Raw Data Set Union Method” (RDSUM). The RDSUM is implemented in order to represent the self and non self two-dimensional spaces of each projection as a group of clusters. As mentioned before, the sets of clusters that cover the non-self two-dimensional space are known as the detectors.

The RDSUM is composed of 4 single phases: generation of a single data file, data preprocessing, set data clustering and generation of detectors (Perhinschi, Moncayo, Al Azzawi, Moguel, 2014). Figure 25 shows the schematic of the RDSUM algorithm phases.



**Figure 25** Raw Data Set Union Method (RDSUM) Phases

#### 4.1.1 Generation of Single Data File:

The data banks created from several flight tests are combined together in one single data file during this step. This file would contain 15 columns corresponding to the history of all features logged during all the flight tests performed.

#### 4.1.2 Data Preprocessing:

This phase is composed of two algorithms: normalization and duplicates elimination. The single file of raw data is normalized between 0 and 1 based on the maximum and minimum values of each of the features analyzed plus a percentage



margin. In addition, points that duplicate within any of the columns are eliminated through the duplicates elimination process. This allows to reduce the size of the data file which decreases the amount of storage and processing power needed for implementation purposes.

#### **4.1.3 Set Data Clustering:**

The data produced from the last phase is organized according to Table 10 in order to conform 47 unit planes which contain self points of nominal conditions within them. These self points need to be represented by a definite number of geometric hyper-bodies, referred to as clusters (Perhinschi, Moncayo, Al Azzawi, Moguel, 2014). Since this AIS detection scheme is analyzing two-dimensional projections, clusters in this case are defined as circles. This process is based on a modified version of a “k\_means” clustering algorithm used to represent clusters as hyper-spheres.

#### **4.1.4 Generation of Antibodies:**

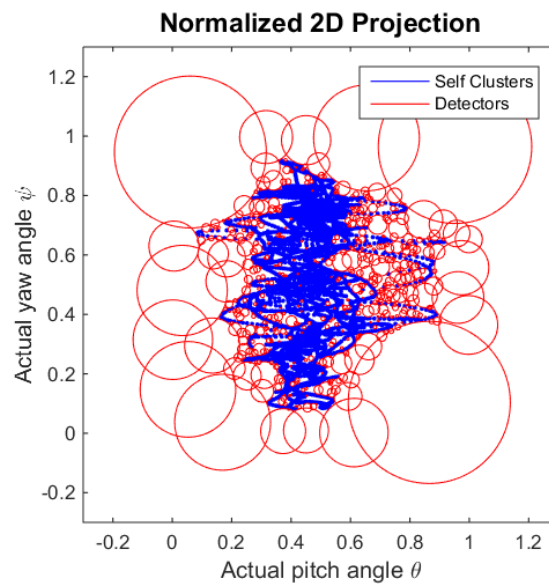
An Enhanced Negative Selection Algorithm for real-valued (ENSA-RV) representation with variable non-self radius is utilized to generate the 47 sets of detectors. This algorithm is responsible for eliminating the cluster overlap between the self and non-self clusters sets. It requires some parameters that must be selected by the operator such as: number of initial set of detectors, the minimum radius permitted for a detector, number of detectors inserted at every iteration to explore new non-self space not covered and the number of rejected detectors selected to be moved at every iteration. The detectors or antibodies generation process can be stopped if one of these requirements is accomplished; a predetermined number of iterations and a preset maximum number of

detectors is reached or when a desired coverage of the non-self space is reached.

Moreover, this algorithm is designed to optimize the following criteria:

- No overlapping between non-self and self detectors.
- Minimum empty space in the self clusters
- Minimum un-covered areas in the non self
- Minimum overlapping among self clusters
- Minimum overlapping among non-self detectors
- Minimum number of detectors

Figure 26 presents a sample two-dimensional projection with self and non-self clusters generated using ENSA-RV algorithm. This particular two-dimensional projection analyzes two attitude angles as features: pitch angle and yaw angle.



**Figure 26** Self Cluster and Antibodies generated using ENSA-RV algorithm

#### 4.1.5 Detection Analysis:

Detection rate and false alarms are two quantitative values used to evaluate the detection performance of this health monitoring scheme. These values can be calculated as follows:

$$DR = \frac{TP}{TP + FN} \times 100$$

$$FA = \frac{FP}{TN + FP} \times 100$$

where, TP, TN, FP and FN represent different conditions of the detection logic:

True Positive (TP): A failure is detected and declared as failure

True Negative (TN): Nominal conditions are declared as nominal

False Positive (FP): Nominal conditions are declared as failures

False Negative (FN): Failure condition is not detected

Detection rates and false alarms are calculated by dividing TP and FP by the total amount of data points considered to be of that particular condition respectively.

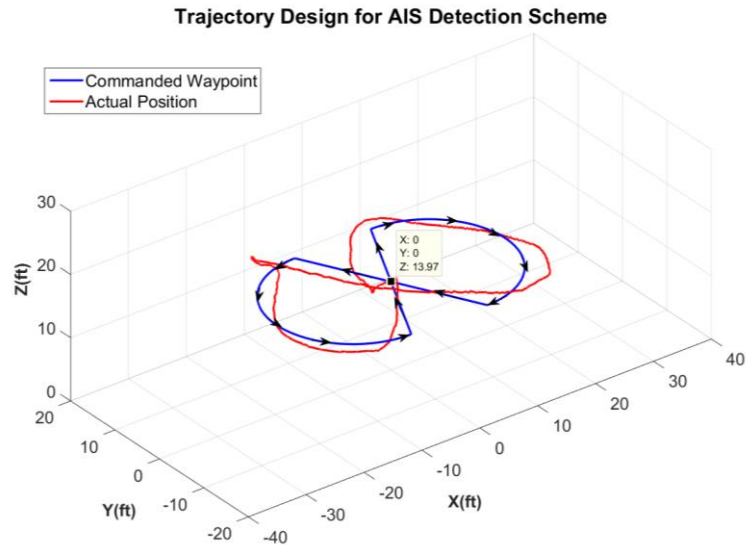
#### 4.2 Implementation Process

This thesis focuses on an AIS fault detection scheme at a hardware level. Efficiency reduction on different actuators is analyzed within this research. For these purposes, this AIS approach utilizes two dimensional projections in order to determine “self” and “non-self” regions. By using two-dimensional projections the number of data that needs to be processed decreases considerably. Two-dimensional projections are used not only for simplicity but they also decrease the computational power required. Savings on computational power can guarantee the effective operation of this AIS algorithm on real time applications. A total of 47 two-dimensional projections are generated in order to

define different sets of detectors. The process of generating detectors consists of a clustering algorithm and a process called Raw Data Set Union Method (RDSUM) (Perhinschi, Moncayo, Al Azzawi & Moguel, 2014). As a result of the RDSUM process a geometrical and numerical representation of every set of detectors is obtained. The group of all detectors contain information about the dynamic signature of the vehicle when it is outside the nominal condition. The performance of this algorithm can be measured based on the history of activated detectors. Validation flight tests are required in order to verify the robustness of this AIS scheme against false alarms. This AIS scheme is meant to increase safety of any autonomous mission programmed in this vehicle.

#### **4.2.1 Design Constraints**

Prior to generating detectors, extensive data acquisition is required through several flight tests. Flight tests must be designed in such a way that the dynamic signature of the quadcopter can be captured inside the two dimensional projections presented in Table 10. In general, quadcopters spend most of their flight time in a hovering flight regime; therefore, this AIS detection scheme is intended to sense system failures within this regime. In this way, take-off and landing flight attitudes are not analyzed within this AIS detection scheme. In order to capture most of the quadcopter dynamics, a flight trajectory with multiple roll and pitch maneuvers is designed. Figure 27 depicts the flight trajectory designed in order to capture most of the vehicle dynamics at nominal conditions.



**Figure 27.** Trajectory Implementation for the AIS Detection Scheme Design for hovering flight

All flight tests are performed using a trajectory controller which will provide a smoother navigation response. This maneuver has an eight shaped trajectory and it starts once the autonomous switch is enabled from the pilot command. At the same time, the initial position is set to the current position. After this, the trajectory is commanded and the quadcopter performs a negative pitch maneuver and a negative roll maneuver to reach position  $[-13.1 \text{ ft}, 6.56 \text{ ft}]$  as depicted in Figure 27. Once position  $[-13.1 \text{ ft}, 6.56 \text{ ft}]$  is reached, a semicircle shaped trajectory is commanded in order to send the quadcopter to the next position  $[13.1 \text{ ft}, 6.56 \text{ ft}]$ . Just after this part of the trajectory, the quadcopter performs a positive pitch maneuver and a negative roll maneuver to come back to the initial point  $[0, 0]$ . At this point, half of the trajectory has been completed. The rest of the trajectory follows a similar pattern and the eight shaped trajectory is completed. It is important to notice that this controller is performing in a body reference frame.

This quadcopter vehicle is composed of two main systems: airframe system and propulsion system. Within the airframe system, there are three main subsystems: main structure, electrical and electronic subsystems. In the same way, the propulsion system is composed of every motor and propeller combination within the vehicle. Every subsystem that conforms the quadcopter has a certain probability of failing under certain conditions. However, this research effort is focused on failures within the propulsion system. Because this a critical area which sustains the quadcopter flight, an AIS detection scheme is studied in this thesis to determine its feasibility to detect performance faults within the propulsion system. The feasibility study is based on the detection activity and the fault detection time obtained. At this point is important to notice that this AIS detection scheme is not considering the risk of failure of the AIS scheme itself. This study is only focused on faults regarding the propulsion system and a complete risk management of the vehicle is outside the scope of this thesis. This study analyzes loss of effectiveness on a single motor/propeller combination at two levels of severity:

- 44 % loss of effectiveness on single motor/propeller combination
- 19 % loss of effectiveness on single motor/propeller combination

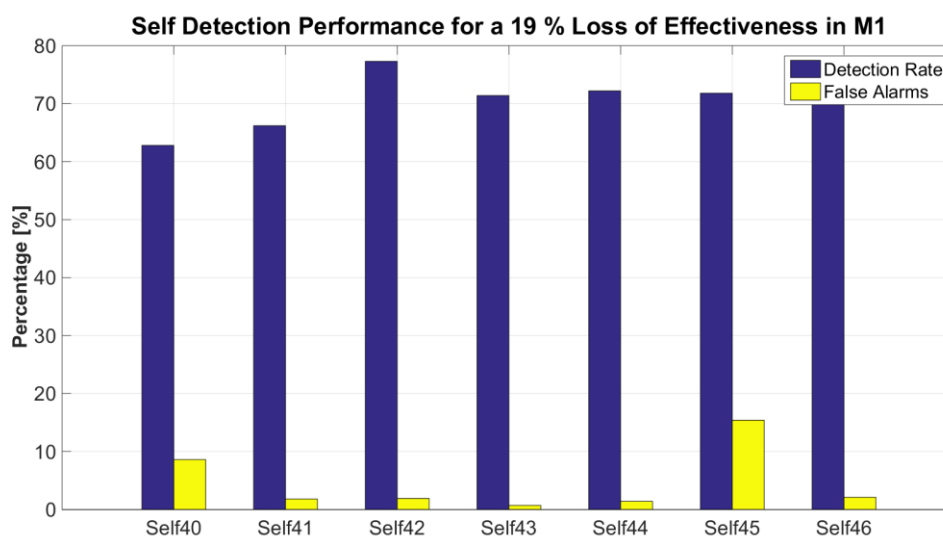
### **4.3 Analysis of Detection Results**

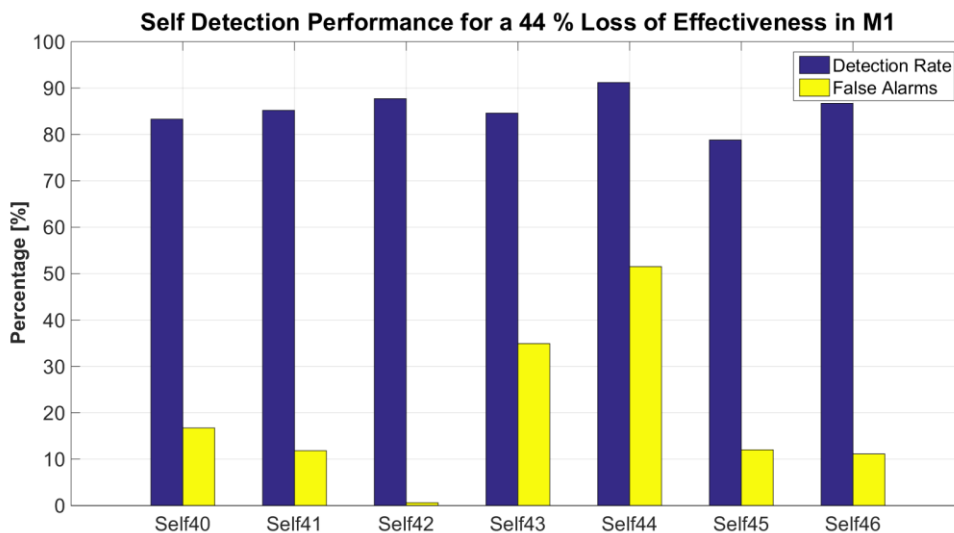
The detection rates and false alarms are calculated for the 47 different selves under these two levels of failure severity. Table 11 presents results for 6 out of the 47 selves that are selected for showing acceptable detection performance. As it can be noticed, results from Table 11 show that the detection performance of these selves is acceptable especially for the case in which the failure is more severe.

**Table 11** Detection Performance

<i>Self #</i>	M1 19 % loss of effectiveness		M1 44 % loss of effectiveness		M3* 19 % loss of effectiveness	
	DR	FA	DR	FA	DR	FA
<b>40</b>	62.8	8.6	83.3	16.7	55.9	11.3
<b>41</b>	66.2	1.8	85.2	11.8	52.6	5.2
<b>42</b>	77.3	1.9	87.7	0.6	55.6	2.8
<b>43</b>	71.4	0.7	84.6	34.9	48.6	4.4
<b>44</b>	72.2	1.4	91.2	51.5	45.2	0.8
<b>45</b>	71.8	15.4	78.8	12.0	40.8	13.2
<b>46</b>	71.5	2.1	86.7	11.1	58.1	7.5

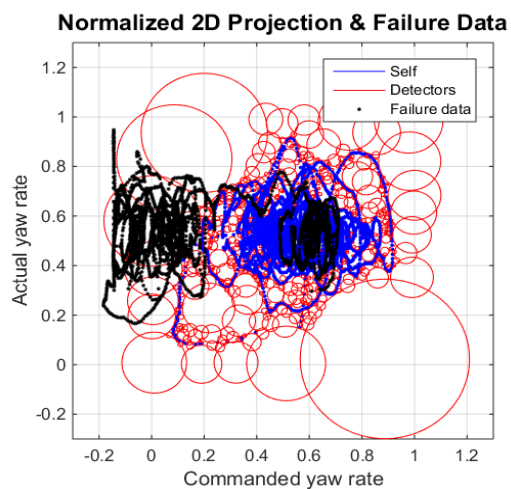
Also, it can be determined that the amount of false alarms varies considerably between different flight test for some selves. This is expected since the nominal portion of the flight is not always the same for all the flight test. Flight time and wind condition can affect false alarms considerably. Therefore, an optimum AIS detection scheme is designed based on the self or selves which provide the highest detection rate and the lowest false alarms at the same time. Figure 28 and Figure 29 present comparative plots that show the correlation between detection rate, false alarms, and failure severity.

**Figure 28** Detection Rate and False Alarms for a 19 % Loss of Effectiveness in M1



**Figure 29** Detection Rate and False Alarms for a 44 % Loss of Effectiveness in M1

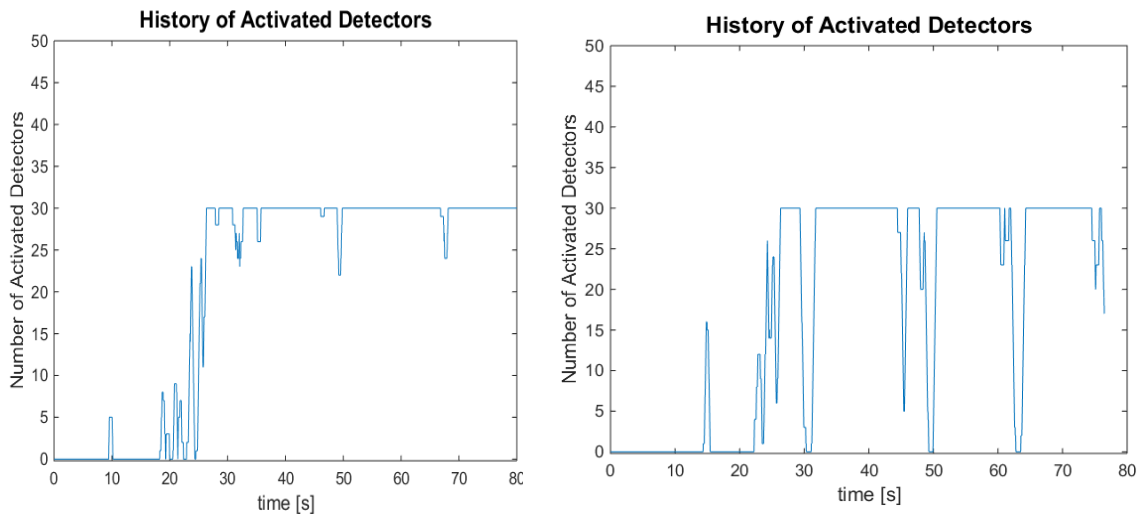
From figures 28 and 29, it is appreciated that Self # 42 provides the highest DR and the lowest FA. Self # 42 is originated from the two dimensional projection of the control signal  $r_{cmd}$  (desired roll rate) and the gyroscope signal  $r$  (actual roll rate). This result suggests that a loss of effectiveness in one motor/propeller combination can be best detected by analyzing states which relate to the attitude response of the quadrotor in the z or yaw axis.



**Figure 30** Detection of Self # 42 during flight test with loss of effectiveness of 44% in M1.



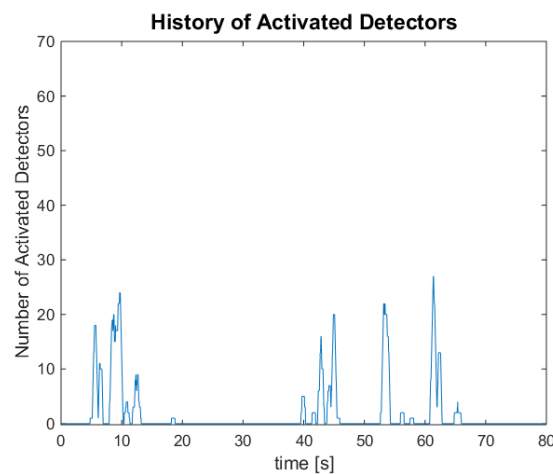
Figure 30 presents the self clusters, non self clusters (antibodies) and flight data points obtained during a flight test with an injected failure. As it can be seen, most of the data points fall outside the region occupied by self clusters (nominal). This allows the flight test data points to match with some of the antibodies of Self #22 which will be then activated. This idea is used within this AIS detection system to design a health monitoring system able to alert the pilot or autopilot about the presence of a fault. Figure 31 shows the history of activated detectors that is being used to declare a failure in the system. This type of plots is generated by considering all the 530 antibodies that compose Self # 42. Every sample time, a window of 30 flight test data points is analyzed against the 530 antibodies. Activated detectors are assigned to have a value of 1 while inactivated detectors are assigned a value of 0. Activated detectors are added together and plotted against the flight time to create a time history of activated detectors which is used to declare a failure. Failure is declared if the number of activated detectors are greater than a certain threshold value.



**Figure 31** History of activated detectors for two flight test with loss of effectiveness of 44% and 19% in M1

Faults are incrementally injected into the vehicle after 10 seconds of starting the autonomous mission. These failures are injected incrementally during 10 seconds in order to avoid abrupt attitude changes. The entire failure is fully injected after 20 seconds of starting the autonomous mission. The plot from the left in Figure 31 shows the response of activated detectors for a flight test with a loss of effectiveness of 44% in M1. From this plot, the fault detection time can be estimated based on a predefined threshold value at which the failure is declared. The threshold value is assigned to be 25 in order to stay on the conservative side and avoid false alarms. Considering this threshold, a failure detection time of 7 seconds can be estimated based on the history of activated detectors from the left plot. In the same way, the plot from the right in Figure 31 shows the response of activated detectors for a flight test with a loss of effectiveness of 19% in M1. Considering the same threshold as before, a failure detection time of 7 seconds can be estimated based on the history of activated detectors from this plot. From these results, it can be determined that failure detection time remains constant within this range of failure. It is important to consider that most of the actuators failures can be treated as non-linear disturbances; consequently, the NLDI controller in this case will minimize the effect of these disturbances to maintain system stability, minimizing the failure effect and driving the system towards a pseudo nominal behavior. This explains why the activation activity of the antibodies within the health monitoring system does not remain constant all the time while facing an upset condition. However, this phenomenon enhances the performance of the on-line health monitoring system that is able to detect the presence of a failure even if the robustness of the NLDI controller allows its partial rejection.

Validation flights are also conducted in order to evaluate the performance under nominal conditions for which low or no activation of detectors is expected. Nominal flight tests are performed based on the same autonomous mission for this purpose. Figure 32 shows the activation activity for that specific flight test. The low activation activity presented is evidence of the acceptable performance of the biomimetic system with a maximum number of activated detectors of no more than 25. This, of course, can be translated to very low false alarms.

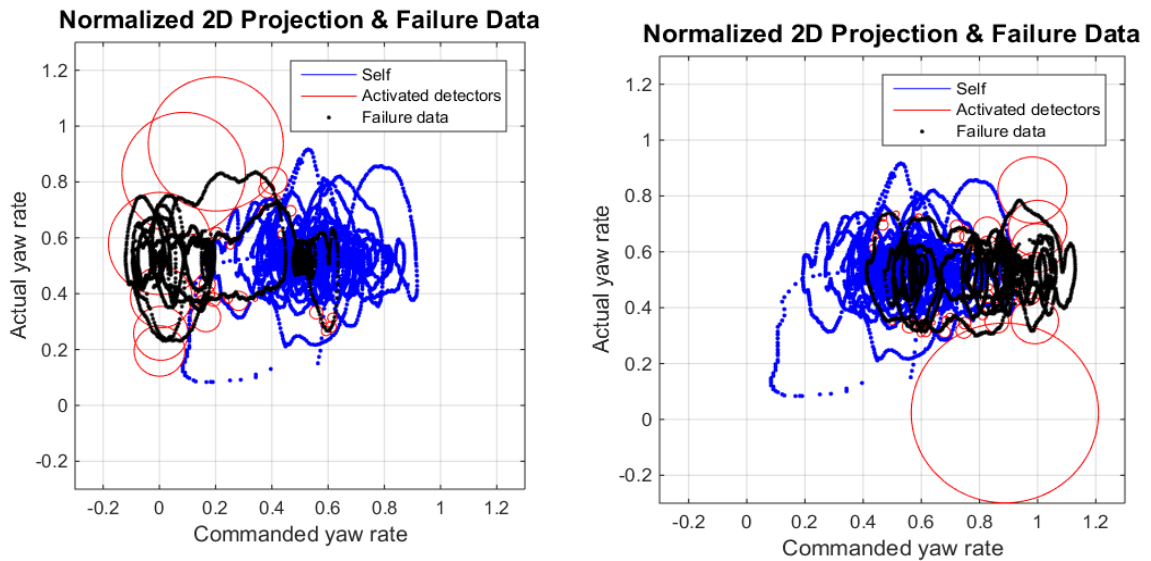


**Figure 32** Nominal validation flight

Based on the overall results, an acceptable performance of the AIS detection scheme has been demonstrated while implemented off-line on a quadrotor type UAV. Such a health monitoring system can be implemented for real time applications if a powerful enough onboard computer is programmed with the proposed architecture and algorithms. In this way, reliability of the vehicle and safety of the missions would be increased, and this would contribute to the overall autonomy enhancement for this type of UAVs.

#### 4.4 Fault Identification Capabilities

Fault identification is not the main focus of this AIS detection scheme, however, based upon the obtained results, high possibilities of an identification phase implementation have been determined.



**Figure 33** Comparison between activated detectors of Self # 42 between failure at M1 and failure at M7.

From Figure 33, it can be noticed that opposite sets of antibodies are being activated depending on the position of the actual failed actuator. This characteristic can be used in order to identify the faulty actuator. However, the use of a unique Self is not enough for identifying precisely the faulty actuator. By using only Self# 42, the probability of identifying a faulty actuator is increased in a 50%, nonetheless, a multi-self approach needs to be implemented to successfully identify the faulty actuator accurately. Further research and analysis is required to determine a group of selves capable of identifying a specific faulty motor for this type of UAV.

## **5. Vision-based Alternative Solution for Autonomous Navigation in GPS Denied Environments**

### **5.1 Optical Flow Theory**

Optical flow is the pattern of apparent motion of objects, surfaces, and edges in a visual scene caused by the relative motion between an observer (the image sensor), and the scene (Garcia, Dzul, Lozano, Pegard, 2013). In general, the optical flow techniques can be classified into four main groups according to the assumptions they contain: differential of gradient methods, correlation and block matching schemes, energy and phase based methods and sensor-based approaches (Kendoul, Fantoni, Nonami, 2009). Even though these approaches are driven by different assumptions, they are conceptually organized in three stages of processing:

- Pre-filtering or smoothing with low-pass/band-pass filters. This enhances the signal to noise ratio and extracts the structure of interest.
- The extraction of basic measurements such as spatiotemporal derivatives or local correlation surfaces
- The integration of measurements to produce a two dimensional flow field which often involves assumptions about the smoothness of the underlying flow field (Barron, Fleet, Beauchemin, 1994).

This thesis focuses on the study of the correlation and matching block methods. This method is one of the less complex and most used techniques in today's applications. In addition, this approach provides good accuracy level and good performance against the aperture problem and large displacements. Unfortunately, some drawbacks related to this approach require special consideration. Some of these drawbacks include: lack of sub-pixel precision, quadratic computational complexity and inaccuracy in presence of image

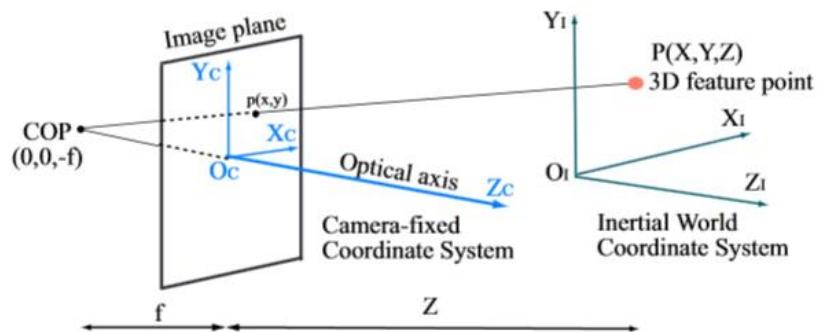
deformation due to rotation. In general, some of these drawbacks are compensated by using coarse-to-fine frameworks and refinement algorithms (Kendoul, Fantoni, Nonami, 2009).

## 5.2 Motion Field Equations

This section is intended to develop a relation between the pixel-based motion field and the metric velocity of the camera fixed to the UAV frame. The motion field can be derived by projecting the 3D velocity field (translational and rotational) on the image plane. Let  $\mathbf{P}=[X \ Y \ Z]^T$  be a point in the three dimensional camera frame and the projected pixel coordinates of  $\mathbf{P}$  on the image plane can be defined by the following relation:

$$\mathbf{p} = \frac{f \mathbf{P}}{Z} \quad (4.1)$$

where  $f$  represents the distance between the image plane and the origin otherwise known as the focal length. In this way,  $\mathbf{p}$  can be expressed as  $[x \ y \ f]^T$ . Figure 34 shows a perspective-central projection of the camera plane



**Figure 34** Imaging model: perspective-central projection (Kendoul, Fantoni, Nonami, 2009)

In general, the relative motion between the camera and P is described by the following expression:

$$V = -T - \omega \times \mathbf{P} \quad (4.2)$$

where,  $\omega$  is the angular velocity and T is the translational component of the motion. A relation between the velocity of P in the camera reference frame and the velocity or flow of p in the image plane must be developed in order to calculate optical flow. Taking the derivative with respect to time of (4.1) yields:

$$\frac{flow}{\Delta t} \cong v = f \frac{ZV - V_z P}{Z^2} \quad (4.3)$$

The motion field in the x axis can be derived by substituting (4.2) into (4.3) as follows.

$$v_x = f \left[ \frac{Z(-T_X - (Z\omega_Y - Y\omega_Z)) - (-T_Z - (Y\omega_X - X\omega_Y))X}{Z^2} \right] \quad (4.4)$$

From (4.1),  $\mathbf{P}$  can be extended as:

$$X = \frac{xZ}{f} \quad (4.5)$$

$$Y = \frac{yZ}{f} \quad (4.6)$$

Substituting (4.5) and (4.6) into (4.4) yields to:

$$v_x = f \left[ \frac{Z(-T_X - (Z\omega_Y - \frac{yZ}{f}\omega_Z)) - (-T_Z - (\frac{yZ}{f}\omega_X - \frac{xZ}{f}\omega_Y))\frac{xZ}{f}}{Z^2} \right] \quad (4.8)$$

$$v_x = f \left[ -\frac{T_X}{Z} - \omega_Y + \frac{y}{f}\omega_Z + T_Z \frac{x}{Zf} + \frac{xy}{f^2}\omega_X - \frac{x^2}{f^2}\omega_Y \right] \quad (4.9)$$

Reorganizing the terms yields to:

$$v_x = \frac{T_Z x - T_X f}{Z} - \omega_Y f + \omega_Z y + \frac{\omega_X xy - \omega_Y x^2}{f} \quad (4.10)$$

Similarly the motion field in the y axis can be expressed as

$$v_y = \frac{T_Z y - T_Y f}{Z} + \omega_X f - \omega_Z x + \frac{\omega_X y^2 - \omega_Y xy}{f} \quad (4.11)$$

In general, the optical flow can be expressed in terms of image coordinates  $[x \ y \ f]$ , the

UAV body-axis velocities and angular rates  $[T_X \ T_Y \ T_Z]$ ,  $[\omega_X \ \omega_Y \ \omega_Z]$  as follows:

$$\begin{bmatrix} v_x \\ v_y \end{bmatrix} = \begin{bmatrix} \frac{-1}{1+\tau Z} & 0 & \frac{\tau x}{1+\tau Z} \\ 0 & \frac{-1}{1+\tau Z} & \frac{\tau y}{1+\tau Z} \end{bmatrix} \begin{bmatrix} T_X \\ T_Y \\ T_Z \end{bmatrix} + \begin{bmatrix} \tau x y & -\left(\frac{1}{\tau} + \tau x^2\right) & y \\ \left(\frac{1}{\tau} + \tau y^2\right) & -\tau x y & -x \end{bmatrix} \begin{bmatrix} \omega_X \\ \omega_Y \\ \omega_Z \end{bmatrix} \quad (4.12)$$

where  $\tau$  represents the inverse of the focal length  $\frac{1}{f}$ .

The translational velocity part and plus the rotational part are easily identified in Equation (4.12). For autonomous navigation purposes, only the translational velocity is required. In order to calculate the translational velocity only, the rotational velocity can be measured using gyroscopes and compensated inside the motion field.

### 5.2.1 Angular Rate Compensation

If a constant distance to the scene is maintained during a hover flight, equations of the motion field can be written as:

$$v_x = \frac{-T_X f}{Z} - \omega_Y f + \omega_Z y \quad (4.13)$$

Similarly, the motion field in the y axis can be expressed as

$$v_y = \frac{-T_Y f}{Z} + \omega_X f - \omega_Z x \quad (4.14)$$



The terms divided by the focal length are neglected since they are more than one order of magnitude smaller compared to the other summands (Honegger, Meier, Tanskanen, Pollefeys, 2013). Satisfying these conditions, the effects of the angular rates on the motion field can be subtracted from the motion field. After compensation, the translational velocity in metric scale can be calculated as follows:

$$T_y = -\frac{v_y Z}{f} \quad (4.15)$$

$$T_x = -\frac{v_x Z}{f} \quad (4.16)$$

### **5.3 Implementation Process & Position Kalman Filtering Design using Optical Flow**

#### **5.3.1 Description of the Optical Flow Estimation Module and its Tasks.**

The sensor system performs optical flow calculations from images received from the Complementary metal-oxide semiconductor (CMOS) machine vision sensor. This vision sensor is directly connected to the ARM Cortex M4 microcontroller. The Cortex M4F microcontroller has a fully parametrizable camera interface which allows the use of an 8 bit resolution per pixel in order to process 4 pixels at a time with special 32 bit instructions. The first stage during the flow computation is image frames storage. Pixel data is streamed into the microcontroller and a frame grabber module samples pixel values at the corresponding pixel clock of the CMOS sensor. The incoming pixel data from the camera is saved in the embedded main system memory using Direct Memory Access. (DMA).

Optical flow is calculated between the current and the preceding frame by using the sum of absolute differences (SAD) block matching algorithm. This algorithm compares SAD values of blocks of pixels of the current and previous frame within a certain area. Depending on the best match within the search area, the resulting flow is selected. This SAD value of a 8x8 pixel block is calculated inside a search area of +/-4 pixels in both directions (Honegger, Meier, Tanskanen, Pollefeys, 2013). In order to account for the lack of sub-pixel precision of this method, a refinement step is necessary. This refinement process estimates optical flow with half pixel step size in all directions from the pixel with the best matching properties. As a result, the final refined optical flow is selected based on the best match of the directions around the best match including the previous result. After the flow is estimated, the angular rate compensation can be incorporated using equations (4.13) and (4.14). With this compensation taken into consideration, the translational velocity of the vehicle can be calculated from equations (4.15) and (4.16), where  $Z$  is the altitude of the vehicle being measured by the laser sensor and  $f$  represents the focal length.

The maximum measurable velocity is influenced by the focal length and the ground distance of the vehicle. This parameter can be calculated using relations derived for equations 4.15 and 4.16 as follows:

$$T_{max} = \frac{v_{max} Z}{f} \quad (4.17)$$

where  $v_{max}$  is obtained from the maximum internal update rate or the maximum number of frames per second analyzed.

$$v_{max} = \frac{\text{search range [pixels]}}{(1/\text{max update rate [Hz]})} \quad (4.18)$$

In a similar way, equation 4.17 can be rewritten as follows:

$$T_{max} = \frac{\text{search range [pixels]} \cdot Z[m]}{(1/\text{max update rate [Hz]}) \cdot f[\text{pixels}]} \quad (4.19)$$

The maximum update rate search range and pixel size can be obtained from the sensor specifications:

Maximum internal update rate: 400 Hz

Search range: +/- 4 pixels

Pixel size: 24 micro meters

Focal length: 0.016 meters

According to the specifications maximum measurable velocity can be calculated at different altitudes.

**Table 12** Maximum measurable velocities per specifications

Ground distance	1 [m]	3 [m]	10 [m]
16mm lens	2.4 [m/s]	7.2 [m/s]	24 m/s]

### 5.3.2 Position Kalman Filtering Design using Optical Flow:

The design of a complete autonomous navigation control architecture requires not only velocity terms to be feedback into the control system; an accurate position estimation is desirable to guarantee accurate navigation of the vehicle at all times. A 6 states extended Kalman filter is designed to accurately estimate position based on velocity measurements sensed by the optical flow sensor. The model equations for the Kalman filter algorithm are expressed below:

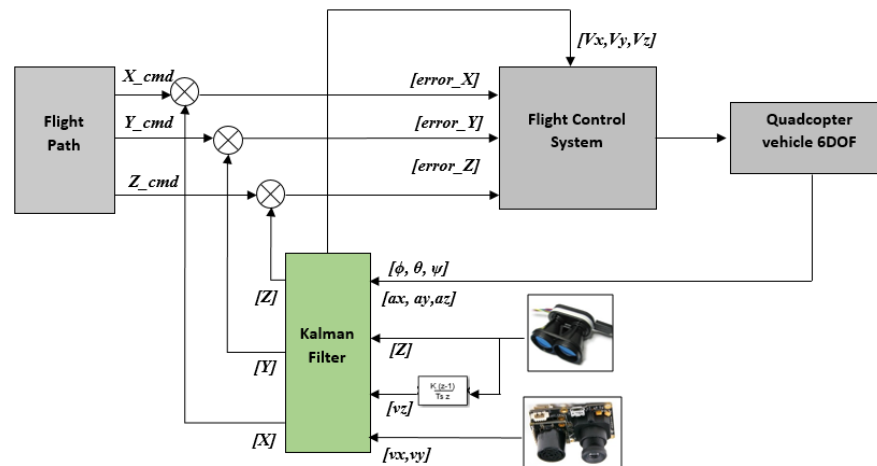
$$\begin{aligned}
X_{(k+1)} &= X_k + v_{xk} dt \\
Y_{(k+1)} &= Y_k + v_{yk} dt \\
Z_{(k+1)} &= Z_k + v_{zk} dt \\
v_{x(k+1)} &= v_{xk} + a_{xk}^E dt \\
v_{y(k+1)} &= v_{yk} + a_{yk}^E dt \\
v_{z(k+1)} &= v_{zk} + a_{zk}^E dt
\end{aligned} \tag{4.17}$$

where  $[X \ Y \ Z]^T$  represent the position of the vehicle, and  $[v_x \ v_y \ v_z]^T$  represent the linear velocities on earth reference frame. Equations in (4.17) are based on a discrete version of Euler integration method, where velocity measurements are integrated to calculate positions and acceleration measurements are integrated to calculate velocities. Acceleration measurements are rotated from body reference frame into earth reference frame can be using the expression below:

$$\begin{bmatrix} ax \\ ay \\ az + g \end{bmatrix}_E = \begin{bmatrix} c\theta \ c\psi & s\theta \ s\theta \ c\psi - c\theta \ s\psi & c\theta \ s\theta \ c\psi + s\theta \ s\psi \\ c\theta \ s\psi & s\theta \ s\theta \ s\psi + c\theta \ c\psi & c\theta \ s\theta \ s\psi - s\theta \ c\psi \\ -s\theta & s\theta \ c\theta & c\theta \ c\theta \end{bmatrix} \begin{bmatrix} ax \\ ay \\ az \end{bmatrix}_B \tag{4.18}$$

From equation (4.18), it can be seen that attitude angles are being considered within this KF algorithm. This is suggesting that by having an accurate attitude estimation is going to indirectly improve the velocity and position estimation. This KF algorithm estimates values for  $X$ ,  $Y$  positions and the  $v_z$  component of the translational velocity in  $Z$  axis. Measurements for  $Z$  position,  $v_x$  and  $v_y$  velocities are introduced into the KF algorithm as correction terms. The  $Z$  position measurement is taken from an

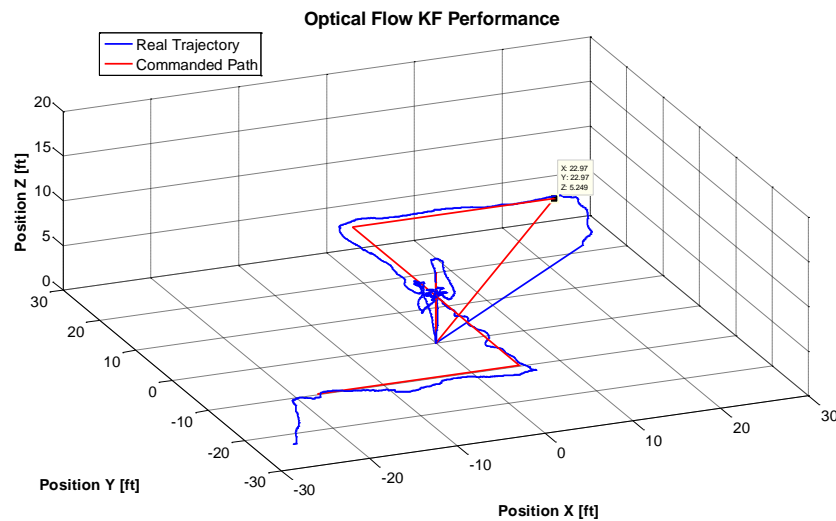
infrared laser sensor while the  $v_x$  and  $v_y$  velocities are obtained from the optical flow sensor. Besides estimating unknown states as  $X$  and  $Y$  positions, the second main function of this KF is to filter out the measurement noise especially for the optical flow sensor and the accelerometers. An accurate estimation for the vertical component of the translational velocity  $v_z$  is crucial for the autonomous take off, landing and altitude hold modes. These last three modes of flight are essential stages on a fully autonomous mission. Moreover, a robust altitude hold flight mode will contribute to better optical flow sensor performance. It is important to notice that this solution does not incorporate any pressure sensor measurements nor GPS measurements for  $v_z$  estimation. Figure 35 shows a general overview of this autonomous navigation alternative for GPS denied environments.



**Figure 35** Optical-flow-based Autonomous Navigation Architecture

An autonomous drilling and sampling mission is designed to test the accuracy and performance of the optical-flow autonomous navigation system. The intent of this mission is to test the autonomous capabilities of the quadcopter while performing an exploration mission in GPS denied environments. The quadcopter is expected to take off

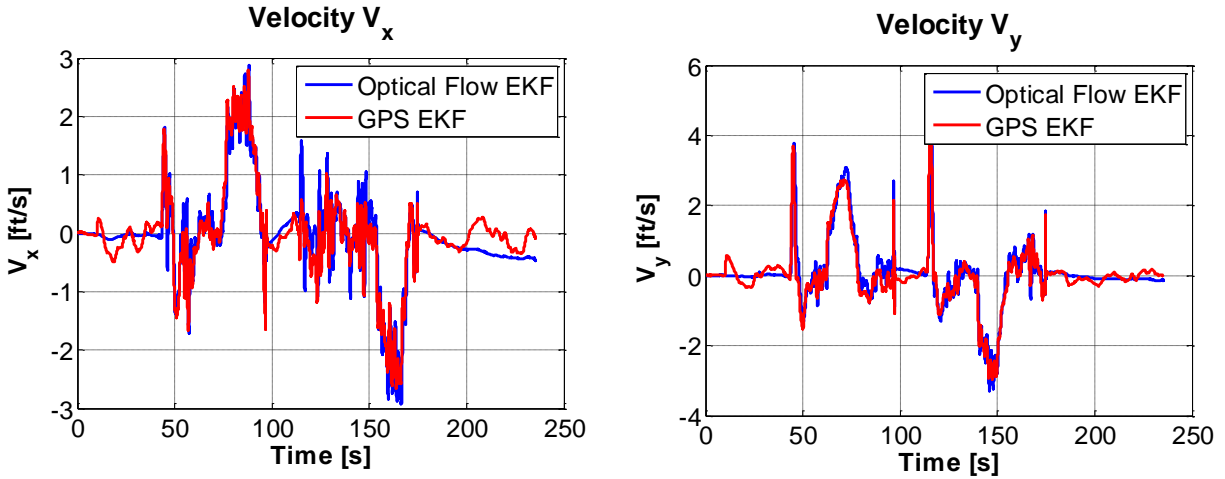
and fly to a desirable waypoint where a sample needs to be obtained from the ground and returned to the initial waypoint. This sampling mission includes six different autonomous stages in one mission. First, an autonomous take-off routine is implemented followed by a short position-hold flight mode. Then, autonomous navigation to a pre-set waypoint is commanded while flying in an altitude-hold mode. After the target waypoint is reached, an autonomous landing routine is performed. Once landed, the quadcopter can command a sampling and drilling action in order to collect the desired sample. It is important to note that, at this point the sampling and drilling action is only simulated by switch activation. At this point the integration of the drilling and sampling system is still under development. After the sampling action is completed, the quadcopter repeats the autonomous take off routine and returns back to the initial point or home. Figure 36 depicts the trajectory followed by the quadcopter using the optical flow autonomous navigation system with the autonomous drilling and sampling system.



**Figure 36** Optical-flow based autonomous navigation for a drilling and sampling mission

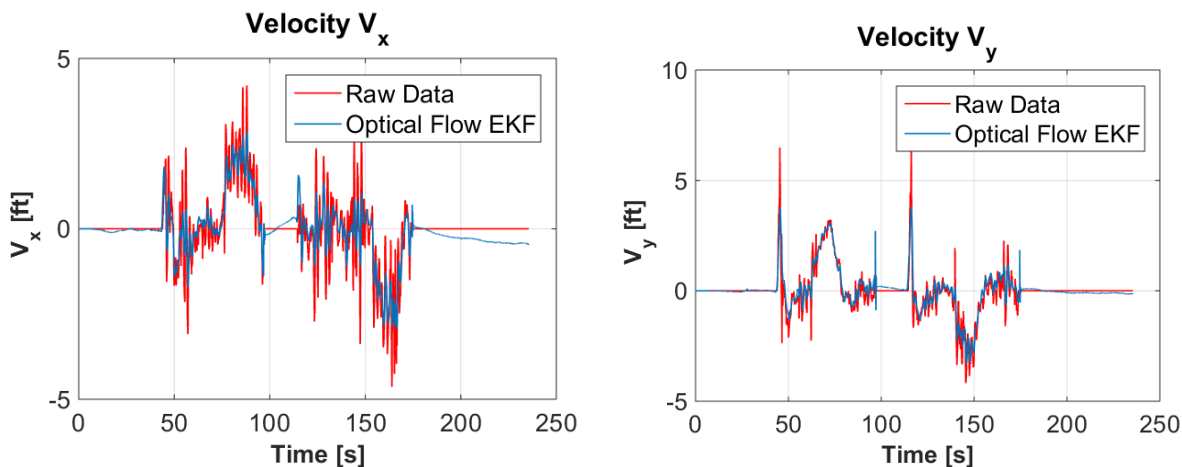
## 5.4 Implementation Results

The performance of this optical-flow vision based algorithms is analyzed using comparison plots between GPS Kalman filter data and the optical-flow Kalman filter data.



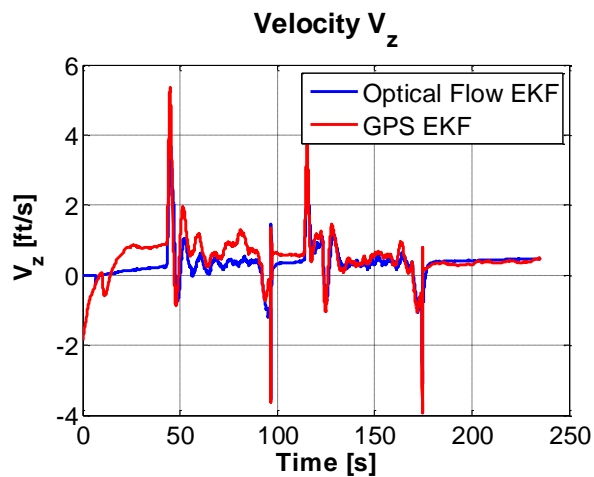
**Figure 37** Comparison of Velocity Estimation in  $x$  and  $y$  axis

From Figure 37, it can be seen that the estimation of velocity performed by the optical flow KF is similar to the velocity estimated using the GPS sensor KF. This result is achieved due to the accuracy of the optical flow sensor and the noise reduction provided by the Kalman Filter.



**Figure 38** KF noise reduction properties

Results shown in Figure 38 depict noise cancellation characteristics of the optical flow KF. As it can be seen, the noise present in the optical flow sensor is considerably decreased by the KF action. Noise filtering improves the outer loop performance by allowing a smoother navigation. In addition, noise reduction decreases the integration error induced during the position estimation.

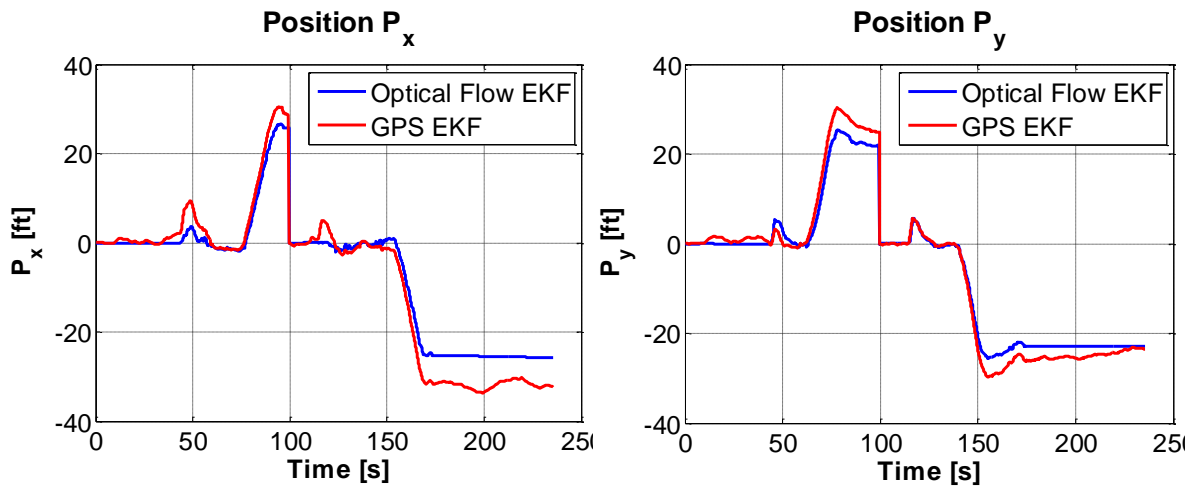


**Figure 39** Comparison of Velocity Estimation in  $z$  axis

Figure 39 depicts the performance of the estimated  $V_z$  velocity. The GPS KF often has some error induced by the low accuracy of the GPS sensor in the  $z$  axis. In this case, the



velocity  $V_z$  estimation of the optical flow KF provides a more accurate estimation than the GPS KF. Accurate  $V_z$  estimation contributes for a robust altitude hold flight mode which ensures an accurate navigation.



**Figure 40** Comparison of the Position estimation in  $x$  and  $y$  axis.

Figure 40 shows the performance of the position estimation in the  $x$  and  $y$  axes. A small position estimation error is induced within the optical flow KF. Possible sources of error include and are not limited to the following: the on-board computer is not able to maintain a fixed sample time for real time integration due to the amount of applications it runs at the same time. Also, the error in velocity calculation is induced when the assumption of constant distance to ground is not satisfied. Even though a robust altitude hold flight mode is implemented, small changes in altitude can induce a small velocity component in the  $z$  axis which will nullify the assumption of constant distance to ground.

#### **5.4.1 Approach Limitations:**

Distance from vehicle to ground is a constraint that has been determined during the flight testing process. It has been found that the optical flow sensor produces more accurate measurements when the distance to ground is between 1 to 2 meters. It is worth mentioning that all tests have been performed over a grass surface. In addition to this, accurate results from the optical flow sensor are only obtained when noticeable contrast or distinguishable features form part of the surface at which the optical flow camera is pointing. The optical flow sensor requires to have minimum conditions of light in order to measure optical flow accurately.

## 6. Conclusions

A NLDI control architecture has been successfully implemented on a quadcopter type UAV to increase its stability by minimizing nonlinear effects and improve robustness against model uncertainties. Simulation and implementation results have been used to demonstrate that a NLDI controller is robust enough to stabilize a quadrotor with a certain transient response. In addition, a health monitoring system inspired by an AIS paradigm has been successfully designed and tested through flight testing on a quadcopter type UAV. Its promising performance has been evaluated and analyzed based on antibodies activation activity present in flight tests with an efficiency loss in different actuators. Several nominal flight tests and flight test with failures have been analyzed in order to characterize the dynamic signature of the vehicle.

An optical-flow-based vision system has been successfully implemented on a quadcopter type UAV to allow for autonomous navigation in GPS denied environments. The optical-flow-based vision system performance has been analyzed by comparing optical flow EKF data against GPS EKF data. The proposed methodologies and their results have demonstrated the capabilities to increase autonomy on quadrotor type UAVs within three areas: attitude and position control stability, health management, and versatility of navigation environments.

These intelligent algorithms have been successfully integrated and implemented into a low cost testbed which provides minimum capabilities to perform autonomous missions.

## **7. Future Work & Recommendations**

A NLDI control architecture can be implemented within the autonomous navigation control loop. Migrating from a PID controller into a NLDI control architecture is a critical process and has to be performed systematically to avoid accidents. This process would require several flight tests and tuning routines. Once the NLDI architecture is fully implemented on the outer loop, comparison between PID results and NLDI results can be determined in terms of tracking error and robustness to nonlinearities.

Fault identification capabilities of the proposed health monitoring scheme can be the subject of further research which will require extensive flight testing and a Multi-Self Strategy analysis. Furthermore, a real time health monitoring system can be implemented in order to determine system health while the vehicle is performing any mission. Computer science techniques and efficient data processing algorithms can be the subject of research in order to make the real time application demand less processing power from the onboard computer.

The optical-flow-based vision system can be characterized and tested at higher altitudes in order to increase range for different applications. Different altitudes would translate into different allowable speed ranges and maximum navigation speeds can be determined for different scenarios. In addition, multiple optical flow sensors can be implemented in order to analyze advantages and disadvantages.

## 8. References

- Abhijit Das, Frank L. Lewis and Kamesh Subbarao (2011). Sliding Mode Approach to Control Quadrotor Using Dynamic Inversion, Challenges and Paradigms in Applied Robust Control, Prof. Andrzej Bartoszewicz (Ed.), ISBN: 978-953-307-338-5, InTech, Available from: <http://www.intechopen.com/books/challenges-and-paradigms-in-applied-robust-control/sliding-mode-approach-to-control-quadrotor-using-dynamic-inversion>
- Achtelik M., Bachrach A., He R., Prentice S., Roy N., (2009). ‘Autonomous Navigation and Exploration of a quadrotor helicopter in GPS-denied indoor environments. First Symposium Indoor Flight Issues, Mayaguez, Puerto Rico.
- Barron J., Fleet D., Beauchemin S., (1994). Performance of optical flow techniques, International Journal of Computer Vision 12 (1) (1994) 43–77.
- Bentley P., Timmis J. (2004). “A Fractal Immune Network” in Proceedings of the Third International Conference ICARIS, Catania, Sicily, Italy.
- Blosch M., Weiss S., Scaramuzza D., Siegwart R. (2010). “Vision based MAVin unknown and unstructures environments. IEEE International Conference on Robotics and Automation, Anchorage, Alaska, USA.
- Alfaro V.M., Vilanova R., Arrieta O. (August, 2009). “Robust tuning of Two-Degree-of-Freedom (2-DoF) PI/PID based cascade control systems”. Journal of Process Control 19 (2009) 1658–1670.
- Chhaniyara S., Bunnun P., Seneviratne L. D., Althoefer K. (2008). “Optical Flow Algorithm for Velocity Estimation of Ground Vehicles: A Feasible Study”. Journal on Smart Sensing and Intelligent Systems” Vol. 1, No.1
- Das A., Subbarao K., Lewis F., (2008) “ Dynamic Inversion of Quadrotor with Zero-Dynamics Stabilization”, 17<sup>th</sup> IEEE.
- Dasgupta D. (1999). *Artificial Immune Systems and Their Applications*, New York: Springer Verlag, New York, USA.
- Dasgupta D., (2006). “Advances in Artificial Immune Systems” in IEEE Computational Intelligence Magazine.
- Dasgupta D., Krishna Kumar K., Wong D., Berry M. (2004), “Negative Selection Algorithm for Aircraft Fault Detection”, in Proceedings of ICARIS 2004, edited by G. Nicosia et al., LNCS 3239, pp.1–13.

- Dasgupta D., Nino L. F. (2009). *Immunological Computation – Theory and Applications*, CRC Press, Auerbach Publications, Taylor & Francis Group, Boca Raton FL, USA.
- Drozeski G.R., Saha B., Vachtsevanos G.J. (2005), “A fault detection and reconfigurable control architecture for unmanned aerial vehicles”, in: *IEEE Aerospace Conference*, Big Sky, MT, USA, 5–12 March 2005, pp. 1–9.
- Esteves D. J., Moutinho A., Azinheira J. R. (2015), “Stabilization and altitude control of an indoor low-cost quadrotor: design and experimental results”. *2015 IEEE International Conference on Autonomous Robot Systems and Competitions*.
- Garcia L. R., Dzul A. E., Lozano R., Pegard C., (2013). *Quad rotorcraft control: vision-based hovering and navigation*. New York:Springer, London
- Garcia D. F., Moncayo H., Perez A., Jain C., (2016) “Low Cost Implementation of a Biomimetic Approach for UAV Health Management”. *American Control Conference ACC 2016*.
- Guenard, N. Hamel T., Mahony R., (2008). “A practical visual servo control for a unmanned aerial vehicle. *IEEE Trans. Robot.* 24(2), 331-341.
- Harkegard, Ola (2001) (Unpublished Master’s Thesis). *Flight Control Design Using Backstepping*. Linköping university, SE-581 83 Linköping, Sweden.
- Honegger D., Meier L., Tanskanen P., Pollefeys M. (2013). “An Open Source and Open Hardware Embedded Metric Optical Flow CMOS Camera for Indoor and Outdoor Applications”, *International Conference on Robotics and Automation (ICRA) Karlsruhe, Germany*.
- Hui C., Yousheng C., Shing W. W. (August 2014), “Trajectory Tracking and Formation Flight of Autonomous UAVs in GPS-Denied Environments Using Onboard Sensing\*”, *Proceedings of 2014 IEEE Chinese Guidance, Navigation and Control Conference*. Yantai China
- Ireland M. L., Vargas A., & Anderson, D., (2015). “A comparison of Closed-Loop Performance of Multirotor Configurations Using Non-Linear Dynamic Inversion Control”. *Aerospace*, 325.
- Kendoul F., Fantoni I., Nonami K (2009). *Optic flow-based vision system for autonomous 3D localization and control of small aerial vehicles*. *Robotics and Autonomous Systems*, Elsevier, 57 (6-7), pp.591-602.
- Ko A., Lau H.Y.K., and Lau T.L. (2014) “An Immuno Control Framework for Decentralized Mechatronic Control” in *Proceedings of the Third International Conference ICARIS, Catania, Sicily, Italy*.

- Ludington B., Johnson, E.N, Vachtsevanous G.J., (2007). Vision based navigation and target tracking for unmanned aerial vehicles. In: *Advances in Unmanned Aerial Vehicles*. Springer, Berlin.
- Marlin T., (2000) *Process Control: Designing Processes and Control Systems for Dynamic Performance*, McGraw-Hill.
- Moguel I. (2014) (Unpublished Master's Thesis). *Bio-inspired Mechanism for Aircraft Assessment under Upset Conditions*. Embry-Riddle Aeronautical University Daytona Beach, Florida.
- Mohr P. H., Ryan N., Timmis J. (2014). "Exploiting Immunological Properties for Ubiquitous Computing Systems" in *Proceedings of the Third International Conference ICARIS, Catania, Sicily, Italy*.
- Moncayo H., Perhinschi M. G., Davis J., (July/August 2011). "Artificial Immune System Based Aircraft Failure Evaluation over Extended Flight Envelope", *AIAA Journal of Guidance, Control, and Dynamics*, Volume 34, No. 4.
- Moncayo H., Perhinschi M. G., Davis J. (Jul.–Aug. 2010). "Artificial Immune System Based Aircraft Failure Detection and Identification Using an Integrated Hierarchical Multi-Self Strategy", *AIAA Journal of Guidance, Control, and Dynamics*, Vol. 33, No. 4, , pp. 302–320.
- Lan T., Fei Q., Geng Q., Hu, Q., (xx) *Control Algorithm Research of Quadrotor system based on 3-DOF Simulation Platform*. Beijing Institute of Technology.
- Naduvilakandy G., (2016). *Dynamics and Control of a Quadcopter* (Master's thesis, Texas A&M University, Kingsville, Texas).
- Nicosia G., Cutello V., Bentley P. J., Timmis J. (2004). *Artificial Immune Systems*. Berlin: Springer-Verlag, Berlin Heidelberg.
- Office of the Secretary of Defense, *Unmanned Aerial Vehicles Roadmap (2002–2027)*, Technical Report, Washington, DC, 2002.
- Perhinschi M. G., Moncayo H., Al Azzawi D., Moguel I. (March 2014). "Generation of Artificial Immune System Antibodies Using Raw Data and Cluster Set Union", *International Journal of Immune Computation*.
- Perhinschi M. G., Moncayo H., Al Azzawi D., (2013). "Development of Immunity-Based Framework for Aircraft Abnormal Conditions Detection, Identification, Evaluation, and Accommodation", in *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, Boston, MA.

- Perhinschi M. G., Moncayo H., Wilburn B., Bartlett A., Davis J., Karas O., (2011). "Testing of Immunity-Based Failure Detection Scheme with the NASA Generic Transport Model", in the Proceedings of the AIAA Guidance, Navigation, and Control Conference, Portland, OR.
- Perhinschi M. G., Porter J., Moncayo H., Davis J., Wayne W. S., (2011). "Artificial Immune System-Based Detection Scheme for Aircraft Engine Failures," AIAA Journal of Guidance, Control, and Dynamics, Vol. 34, No. 5, pp.1423-1440.
- Pilot Engineering Group, MathWorks. (2014). "*Pixhawk Pilot Support Package (PSP) User Guide*". Retrieved from <https://www.mathworks.com/hardware-support/pixhawk.html>
- Playfair J.H.L., Chain B.M., (2011). Immunology at a Glance. Blackwell Science, Bodmin, Cornwall.
- Proctor A.A., Johnson E.N., Apker T.B., (2006). Vision only control and guidance for aircraft. J. Field Robot. 23(10), 863-890.
- Quelin M., (2011) Optical flow estimation using insect vision based parallel processing, Master by Research thesis, University of Wollongong. School of Electrical, Computer and Telecommunications Engineering, University of Wollongong. <http://ro.uow.edu.au/theses/3410>
- Saripalli S., Sukhatme G., Mejias L.O., Campoy Cervera, P., (2005) Detection and tracing of external features in an urban environment using autonomous helicopter. In Proceedings of the 2005 IEEE International Conference on Robotics and Automation, Barcelona, Spain, pp. 3972-3977
- Seborg D.E., Edgar T.F., Mellichamp D.A. (2004), Process Dynamics and Control, John Wiley and Sons.
- Sidi, M. J. (1997). Spacecraft Dynamics and Control A Practical Engineering Approach. New York: Cambridge University Press.
- Slotine J.-J. & Li, W. (1991). *Applied Non-Linear Control*. New Jersey: Prentice Hall, New Jersey.
- Younes A. Y., Drak A., Noura J., Rabhi A., El Hajjaji A. (February 2016) "Robust Model-Free Control Applied to a Quadrotor UAV", Springer Science+Business Media Dordrecht.
- Wang L., He Y., Zhang Z., He C., (2014). "Adaptive Dynamic Inversion Controller for Quadrotor Aerial Robot", JCET Journal of Control Engineering and Technology.
- Wang L., Zhang Z., He C., (2014). "Trajectory tracking of Quadrotor Aerial Robot Using Improved Dynamic Inversion Method", ICA Journal, Vol. 4, pp. 343-348.



Wang P., Man Z., Cao Z., Zheng J (2016). “Dynamics Modelling and Linear Control of Quadcopter”, in Proceedings of the 2016 International Conference on Advanced Mechatronic Systems, Melbourne, Australia.

Zeglache S. (xx) “Sliding Mode Control Strategy for a 6 DOF Quadrotor Helicopter”, Journal of Electrical Engineering.

Zhang Y.M., Chamseddine A., Rabbath C. A., Gordon B.W., Su C.-Y., Rakheja S., Fulford C., Apkarian J., Gosselin P., (2013). “Development of advanced FDD and FTC techniques with application to an unmanned quadrotor helicopter testbed” Journal of The Franklin Institute 350 (2013) 2396-2422.