FACULTY OF ELECTRICAL ENGINEERING

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

LAPORAN PROJEK SARJANA MUDA

MODELING OF A SMALL SCALE UAV QUADROTOR SYSTEM

Ng Mei Chin

B011010337

Bachelor of Mechatronic Engineering

MAY 2014

I hereby declare that I have read through this report entitle “Modeling of a Small Scale UAV Quadrotor System” and found that it has comply the partial fulfillment for awarding the degree of Bachelor of Mechatronics Engineering.

Signature: .... ……….

Supervisor’s Name: ....

Date: ....

MODELING OF A SMALL SCALE UAV QUADROTOR SYSTEM

NG MEI CHIN

A report submitted in partial fulfillment of the requirements for the degree of Bachelor of Mechatronics Engineering

Faculty of Electrical Engineering

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

YEAR 2014

I declare that this report entitle “Modeling of a Small Scale UAV Quadrotor System” is the result of my own research except as cited in the references. The report has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.

Signature: ....

Name: ....

Date: ....

ACKNOWLEDGMENT

First of all, I would like to express my deepest gratitude to my family, lecturers and friends that being supportive to me all the time. Without them, I would never complete my project.

I would like to take the opportunity to express my sincere gratefulness my project supervisor, Puan Norafizah binti Abas for giving good guidance to me in completing the research. She would never be tired of answering multiple questions that I have. Thank you.

Besides that, to my fellow friends those are willing to help me throughout the research, especially to Ahmad Mahadi bin Razali, thanks for all the helps.

Last but not least, my greatest appreciation to my family, my dad, my mum and my sisters, without their support, I would not able to go this far. Thank you.

ABSTRACT

Quadrotor is a flying robot that is highly nonlinear, multi variable and strongly coupled because it has 6 Degree Of Freedom of the system translational and rotational motions, which are $x, y, z, \phi, \theta, \psi$ with only four inputs from rotors. The inspection work at the gas, petroleum and specialty chemical plant side is very risky due to the height of chimneys. Currently the industries is using real helicopter to perform the inspection work, this method is high cost and require the plant side to shut down for a period of time which will cause lost to the company. To overcome this problem, quadrotor system is proposed and workers will able to collect information from more angles from high position more easily. The objective of this research is to derive and validates the mathematical modeling for autonomous quadrotor system. Based on the findings of other researcher, there are different types of derivation method for mathematical modeling and several tests need to be carried out to identify unknown parameter. The modeling is derived using Newtonian method, and several tests are conducted such as thrust factor and drag factor. The complementary filter is implemented in the on-board system of quadrotor to obtain more accurate reading from the sensors. In order to validate the modeling of quadrotor system, simulation test for the open loop system is performed using the Matlab Simulink. Then, real time implementation test is carried out as well. The data is recorded. The response of axes x, y, z, roll, pitch and yaw from an open loop system during real time implementation is recorded. Based on the analysis of the performance, the simulation generate similar pattern of signal as in real time.

ABSTRAK

Quadrotor adalah sebuah robot terbang yang *nonlinear*, pembolehubah berbilang dan kuat ditambah pula kerana ia mempunyai 6 darjah kebebasan, tetapi hanya ada empat *input* daripada rotor. Kerja-kerja pemeriksaan di cerobong-cerobong dalam kawasan kilang gas, petroleum dan loji kimia khusus adalah sangat berisiko kerana ketinggian cerobong sekurange-kurangnya 3 meter ke atas. Buat masa ini industri menggunakan helikopter sebenar untuk melaksanakan kerja pemeriksaan, kaedah ini melibatkan kos yang tinggi dan memerlukan kilang tersebut menghentikan operasi selama suatu tempoh masa yang akan menyebabkan syarikat tersebut mengalami kerugian. Untuk mengatasi masalah ini, quadrotor sistem adalah dicadangkan supaya pekerja akan dapat mengumpul maklumat dari lebih sudut dan kedudukan yang tinggi dengan lebih mudah. Objektif kajian ini adalah untuk mendapatkan dan mengesahkan model matematik bagi sistem quadrotor. Berdasarkan dapatan kajian penyelidik yang lain, terdapat berlainan jenis kaedah asal untuk pemodelan methamatikal dan beberapa ujian perlu dijalankan untuk mengenal pasti parameter yang tidak diketahui. Pemodelan dalam kajian menggunakan kaedah *Newton- Euler*, beberapa ujian dijalankan seperti faktor teras dan faktor seretan. Penapis pelengkap dilaksanakan di atas sistem quadrotor untuk mendapatkan bacaan yang lebih tepat dari sensor. Bagi mengesahkan pemodelan sistem quadrotor, ujian simulasi untuk sistem gelung terbuka dilakukan menggunakan Matlab Simulink. Kemudian, ujian pelaksanaan masa sebenar dijalankan juga dan data direkodkan. Hasil paksi *x, y, z, roll, pitch* dan *yaw* dari sistem gelung terbuka semasa pelaksanaan masa sebenar direkodkan. Berdasarkan analisis prestasi, simulasi menjana corak yang sama isyarat seperti dalam masa nyata.

CONTENTS

CHAPTER	TITLE	PAGE
	ACKNOWLEDGEMENT	**iv**
	ABSTRACT	**v**
	ABSTRAK	**vi**
	CONTENTS	**vii**
	LIST OF TABLES	**x**
	LIST OF FIGURES	**xi**
	LIST OF ABBREVIATION	**xiv**
	LIST OF APPENDICES	**xv**
1	**INTRODUCTION**	**1**
	1.1 History	1
	1.2 Movement of Quadrotor	3
	1.3 Problem Statement	4
	1.4 Research Motivation	5
	1.5 Objective	6
	1.6 Scope	6
	1.7 Report Outline	7
2	**LITERATURE REVIEW**	**8**
	2.1 Mathematical Modeling	8
	2.2 Sensors at On-board System	9
	2.3 Filter Implementation	11
	2.4 Summary	12
3	**METHODOLOGY**	**16**
	3.1 Methodology Flow Chart	16
	3.2 Mathematical Modeling	18

	3.2.1 Introduction to Quadrotor System	18
	3.2.2 Physical Measurement	20
	3.2.3 Speed Test	20
	3.2.4 Force-lift Test	22
	3.2.5 Bifilar Pendulum Test	23
	3.3 System Estimation with Complementary Filter	25
	3.3.1 Sensors	25
	3.3.2 Complementary filter	27
	3.3.3 IMU Read Data Conversion	30
	3.3.4 IMU Data Collection	37
	3.4 Validation of Modeling of Quadrotor	38
	3.4.1 Simulation in Simulink for Open-loop system	39
	3.4.2 Real Time Implementation	40
	3.4.3 Comparison on signals generated	41
4	**RESULTS AND DISCUSSION**	**42**
	4.1 Mathematical Modeling	42
	4.1.1 Rotational Matrix	43
	4.1.2 Force and Torque	44
	4.1.3 Kinematic Movement of Quadrotor	45
	4.1.4 Thrust Force	46
	4.1.5 Moment Equations	47
	4.1.6 Calculation for Moment of Inertia	48
	4.1.7 State –space Equation	52
	4.2 Physical Measurement and Testing	56
	4.2.1 Speed Test	56
	4.2.2 Force lift Test	61
	4.2.3 Bifilar Pendulum Test	72
	4.3 Complementary Filter	76
	4.4 Quadrotor Mathematical Modeling Validation	79
	4.4.1 Simulation Result	79
	4.4.2 Real time Implementation Result	82
	4.4.3 Data Analysis	87

5	**CONCLUSION AND RECOMMENDATION**	**88**
	5.1 Conclusion	88
	5.2 Recommendation	88
	REFERENCE	**89**
	APPENDIX	**92**

LIST OF TABLES

TABLE	DESCRIPTION	PAGE
2.1	Summary table of literature review	13
4.1	Quadrotor properties measurement	56
4.2	Speed test Result for rotor 1	57
4.3	Speed test Result for rotor 2	58
4.4	Speed test Result for rotor 3	59
4.5	Speed test Result for rotor 4	60
4.6	Force-Lift Test Result for rotor 1	62
4.7	Force-Lift Test Result for rotor 2	64
4.8	Force-Lift Test Result for rotor 3	66
4.9	Force-Lift Test Result for rotor 4	68
4.10	Quadrotor parameters	71
4.11	Time taken from bifilar pendulum test for each movement	72
4.12	Quadrotor bifilar pendulum test result	75
4.13	Summary explanation of the data read by IMU sensor	77

LIST OF FIGURES

FIGURE	DESCRIPTION	PAGE
1.1	Convertawings Model A quad-rotor Design, 1956	2
1.2	Full Throttle Movement of Quadrotor	3
3.1	Flow chart of methodology	17
3.2	Quadrotor Inertial Frame	18
3.3	Quadrotor Body Frame	19
3.4	Quadrotor testbed	20
3.5	Turnigy tachometer	21
3.6	Setup of speed test	21
3.7	Experiment setup for speed test and force-lift test	22
3.8	Setup for yawing motion	24
3.9	Setup for rolling and pitching motion	24
3.10	accelerometer symbol	26
3.11	Symbol of accelerometer movement	26
3.12	Gyroscope symbol	27
3.13	Theory of complementary filter	28
3.14	Object in no gravitational field environment	30
3.15	Friction force applied on object in no gravitational field environment	31
3.16	Object in gravitational field environment	31
3.17	Object in gravitational field environment and contact with two axes	32
3.18	Vectors R in 3D	32
3.19	Angle between x, y, z axis and vector R	34
3.20	Rotation angle	35

3.21	Set up for IMU sensor data collection experiment	37
3.22	Comparison of angle data collected from gyroscope and accelerometer without filter with data from complementary filter	38
3.23	Block diagram of quadrotor system on Matlab Simulink	39
3.24	Block diagram of quadrotor testbed on real time implementation	40
3.25	Setup for quadrotor open loop test	41
4.1	Quadrotor frame for inertia calculation	48
4.2	Speed test graph of rotor 1	57
4.3	Speed test graph of rotor 2	58
4.4	Speed test graph of rotor 3	59
4.5	Speed test graph of rotor 4	60
4.6	Force lift test graph of rotor 1	63
4.7	Force lift test graph of rotor 2	65
4.8	Force lift test graph of rotor 3	67
4.9	Force lift test graph of rotor 4	69
4.10	Force lift test graph of all rotors	70
4.11	Rolling and pitching motion graph extracted from serial chart	73
4.12	Pitching and rolling motion graph of quadrotor bifilar pendulum test	73
4.13	Yawing motion graph extracted from serial chart	74
4.14	Yawing motion graph of quadrotor bifilar pendulum test	74
4.15	Comparison result of raw data from accelerometer and gyroscope with estimated data from complementary filter	76
4.16	Closer examination on the final estimated result by the complementary filter and the accelerometer data	76
4.17	Simulink diagram of quadrotor system	79
4.18	Subsystem of quadrotor system's modeling	80
4.19	Simulation result of open loop test for x-axis	80
4.20	Simulation result of open loop test for roll movement	81
4.21	Open loop test - x output value of quadrotor at speed 60	83
4.22	Open loop test - Rolling angle of quadrotor at speed 60	83
4.23	Open loop test - x output value of quadrotor at speed 100	84
4.24	Open loop test - Rolling angle of quadrotor at speed 100	84

4.25	Open loop test - x output value of quadrotor at speed 130	86
4.26	Open loop test - Rolling angle of quadrotor at speed 130	86
D.1	Simulation result of open loop test for y-axis	102
D.2	Simulation result of open loop test for z-axis	102
D.3	Simulation result of open loop test for pitch movement	103
D.4	Simulation result of open loop test for yaw movement	103
E.1	Result of open loop test for y-axis at speed = 60	104
E.2	Result of open loop test for y-axis at speed = 100	104
E.3	Result of open loop test for y-axis at speed = 130	105
E.4	Result of open loop test for z-axis at speed = 60	105
E.5	Result of open loop test for z-axis at speed = 100	106
E.6	Result of open loop test for z-axis at speed = 130	106
E.7	Result of open loop test for pitch movement at speed = 60	107
E.8	Result of open loop test for pitch movement at speed = 100	107
E.9	Result of open loop test for pitch movement at speed = 130	108
E.10	Result of open loop test for pitch movement at speed = 60	108
E.11	Result of open loop test for pitch movement at speed = 100	109
E.12	Result of open loop test for pitch movement at speed = 130	109

LIST OF ABBREVIATIONS

UAV	Unmanned Aerial Vehicle
VTOL	Vertical Take-off and Landing
IMU	Inertial Measurement Unit
DOF	Degree of Freedom
RRT	Rapidly-exploring Random Tree
GPS	Global Positioning Device
DGPS	Differential Global Positioning Device
OEM	Original Equipment Manufacturer
UART	Universal Asynchronous Receiver/ Transmitter
PIC	Peripheral Interface Controller
PID	Proportional Integral Derivative
COTS	Commercial Off the Shelf
FOV	Field of View
Gyro	Gyroscope
UKF	Unscented Kalman Filter
PI	Proportional Integral
PD	Proportional Derivative
ESC	Electronic Speed Controller

LIST OF APPENDICES

APPENDIX	TITLE	PAGE
A	Gantt Chart	92
B	Coding of Single Rotor Run	93
C	Coding of Complementary Filter Implementation	95
D	Simulation Test Result	102
E	Real time Implementation Result	104

CHAPTER 1

INTRODUCTION

Recently unmanned aerial vehicles (UAV's) have attracted considerable interest for a wide variety of different applications. Quadrotor is a certain kind of UAV. It is considered to be more preferable for surveillance, precise delivery and some other missions requiring agility and accuracy. It is a flying robot that is highly nonlinear due to it has 6-DOF with only four inputs. Compared to fixed-wing airplanes, quadrotor has several advantages such as it can easily move in any direction, able to hover, fly in low speeds, allows disposition in almost any ground, simple construction with made up of a light frame, four motors, speed controllers, batteries, a control board and/or a receiver. [1] [2]

1.1 History

In 1907, the Breguet Brothers built their first human carrying helicopter–they called it the Breguet-Richet Gyroplane No. 1, which is a quadrotor. The issue of stability is a consideration in the design, but the first requirement for the machine is simply to lift itself and a pilot off the ground under its own power. However, there is no means of control provided to the pilot other than a throttle for the engine to change the rotor speed, and the stability of the machine is found to be very poor. [3]

In year 1922, Etienne Oehmechen developed his quadrotor named as Oehmichen No.2 with six full-scale rotary-winged vehicles. The attitude and orientation of craft could not be control as only a single 120HP engine is used to power up all four rotor. four propellers are added to each rotor in order to change the speed of individual rotor and control the craft's attitude and orientation.

The prototype of quadrotor design by Marc Adam Kaplan in year 1956, Convertawings Model “A” quadrotor is the most successful rotorcraft early design as it is power up and control by varying thrust between rotors. It featured wings for additional lift in forward flight. This craft able to hover and maneuver using its two 90HP motors, each capable of driving all four rotors in backup mode. It is the first successful quadrotor that able to fly forward. [4] [5]

Quadrotor design is not competitive as the conventional aircraft performance specification in term of speed, range, payload and others at early stage. However, in recent decades, this design idea shows its great potential after the Stanford Testbed of Autonomous Rotorcraft for Multi Agent Control (STARMAC) project at Stanford University performed some of the initial work on making small-scale quadrotors autonomous. [4] [5]

Figure 1.1: Convertawings Model A quad-rotor Design, 1956 [5]

1.2 Movement of Quadrotor

The frame structure of quadrotor is built in a cross intersection way, forming two diagonal square shapes and attached on a fixed body. Each end of the square has an individual rotor that is driven by a motor. The two rotors that in same diagonal will rotate in same direction, which means motor pair (1,3) rotates clockwise and the other pair (2,4) rotates counter-clockwise in order to balance the moments. The altitude and orientation of a quadrotor is affected by changing the speed of each rotor. Figure 1.2 shows the full throttle movement of quadrotor system.

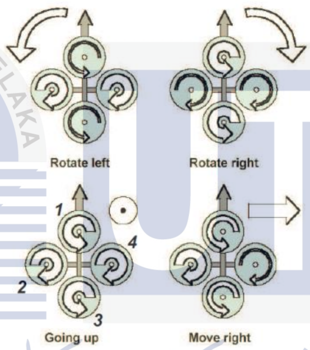

Figure 1.2: Full throttle movement of quadrotor [6]

When the motor’s speed and motion is changed, quadrotor will generate 6-DOF movement. The changes in motion include translational motion along three coordinate axis (x,y and z) and rotational motion around three axis (roll, pitch and yaw). The vertical motion of z-axis in quadrotor is accomplished by increasing or decreasing speed of four motors at the same time and same amount. When the total of thrust of the quadrotor is same as its weight, it will become hoverable. The pitch angle is changed by a difference in thrust between the front and the back rotors to maintain the total thrust of the system, while the change of roll angle is a result from differences between the left and right rotor. Yaw rotation can achieved by the difference in the Counter-torque between each pair (1, 3 and 2, 4) of rotors, the total thrust remain unchanged to avoid the up-down motion. [6] [7]

1.3 Problem Statement

At the gas, petroleum and specialty chemical plant side, there are many chimneys used as ventilation structure to discharge the hot flue gases or smoke from a boiler, stove, furnace or fireplace in the factory to the outside atmosphere. The height of those chimneys are very tall, generally it should not less than 3 meters plus the building height. It is too risky to lift the worker with crane or fly them with helicopter to the top of chimney for every inspection work.

In last decades, one of the solutions to perform the inspection work is by using real helicopter. There are still limitation as the height of chimneys are very tall, the view is not clear for the person controlling the helicopter to continue monitor the direction of the helicopter when it reach certain level. Besides, issue of human component like capability has to be considered. [8] [9]

As a solution, quadrotor is proposed to overcome this problem. The mathematical modeling for quadrotor system is derived for developing the quadrotor system to work autonomously. With the development of on-board system of quadrotor and system estimation through implementation of filters, the system would able to get a more accurate state estimation and these can be features in quadrotor to vertical take-off and landing with hovering autonomously. By implementing a VTOL system into a quadrotor, it will become more functional for small area inspection purpose. The specialties of quadrotor are it can hover and control precisely, the design are more simple compare to the helicopter with mechanical linkage. This system will make the inspection work at plant side easier and also reduce the cost of renting the helicopter. The plant side maintenance team will able to collect information from more angles from high position.

1.4 Research Motivation

The purpose of this research is to model the quadrotor system. Nowadays, quadrotor is been widely used in many sector as it is a better choice compare to remotely-controlled helicopter. The inspection on the chimney is very dangerous due to the high temperature. From The energy newspaper, March 2013, Gary Livingstone, senior mechanical integrity engineer, said: "This was much harder because the flare tip – used to burn off waste gas – is in constant use. So usually the only way to inspect it manually is during a shutdown." [10]

On the other side, the inspection environment is also being an issue for the workers. The bad weather will affect their vision and inspection works have to stop immediately to encounter the safety of worker. Malcolm Connolly, company founder and technical director, said: "The biggest challenges when working offshore are weather and working in a dangerous environment. We planned four days for this inspection but it took six, despite being able to continue when even a full-sized helicopter couldn't get out to the platform because of bad weather." [10]

Thus, it is stated that inspection work would cause an amount of lost to the company due to the entire shutdown. Besides that, inspection through the small scale quadrotor system can be carried out frequently compared to helicopter as the issues for example the cost of renting, impact of shutdown the entire station has to be considered.

1.5 Objective

The objectives of this research are:

1. To derive the mathematical modeling of quadrotor system through Newtonian derivation.
2. To perform system estimation through the implementation of complementary filter.
3. To validate the modeling of quadrotor via simulation and real time implementation and comparison on the signals generated.

1.6 Scope

This research is limited to several scopes:

1. The testbed used in this research is assembled and calibrated by Ahmad Mahadi bin Razali, who are working on another research topic, "Design and Development of a Small Scale UAV Quadcopter".
2. Control design for quadrotor system does not included in this research. The flight test is performed under an open-loop system.
3. The real time implementation testing is done under indoor environment; the testbed is tied with rope for safety purpose during testing.
4. Arduino board is used in this research for retrieve the result from sensor reading and to perform open loop test.

1.7 Report Outline

This report begins with Chapter 1, the introduction of this research, history background of quadrotor system, the construction and the movement of quadrotor, the problem faced by inspection work on plant side, the motivation of carry out this research, lastly the objective and scope of this research.

In Chapter 2, literature review on the quadrotor system is done and separated into several subtopics, which are the method used to derive the mathematical modeling of the system, sensors that other researcher used at the on-board system and its functions, type of filter used for system estimation.

Methodology to carry out the research in Chapter 3, it is divided into several part, which are the mathematical modeling of the quadrotor system, experiment procedure to carry out several tests to identify unknown parameter, includes speed test, force-lift test and bifilar pendulum test; implementation of filter for system estimation, design procedure for Mitlab Simulink block diagram for simulation test, open loop test and data analysis.

All the results and discussion comes under Chapter 4, the derivation of the mathematical modeling, includes the rotational matrix, force and torque equation, the kinematic movement of quadrotor, thrust force of all rotors, moment of inertia calculation, differential equation and finally state-space equation. To determine all the unknown parameters, physical measurement on quadrotor is carried out, speed test, force-lift test and bifilar pendulum test as well. Besides that, the estimated data from complementary filter result for open loop test from Matlab simulation and real time implementation is recorded and discussed. Lastly Chapter 5 will conclude this research and some recommendation are made.

Appendix includes Gantt chart of this research, the coding for single rotor to operate in speed test and force-lift test, coding for complementary filter implementation, the results of simulation and real time open loop test.

CHAPTER 2

LITERATURE REVIEW

The literature review is conducted to determine the information of quadrotor system overview that done by other researchers. The topics included in this chapter are method of mathematical modeling of quadrotor, implementation of filter for system estimation and sensors used at on-board system and its function.

2.1 Mathematical Modeling

Hui Bai *et al.* [11] presents a quadrotor platform and applies a path planning method for multi-UAVs. The path planning method implement in this platform is Rapidly-exploring Random Tree (RRT) method, which is to find a path to travel from a starting position to a goal position through a configuration space. The algorithm developed using the RRT Method was implemented in C language tested on an ARM9 platform run under embedded Linux system for real-time implementation.

The avionic system by Mustafa Ilarslan *et al.* [12] implements the non-linear control algorithms to have a degree of autonomous control. The integrated modular avionic architecture will enable the rapid realization and testing of these algorithms.

2.2 Sensors at On-Board System

The quadrotor platform presents by Hui Bai *et al.* [11] consists of several sensors, an Inertial Measurement Unit (IMU), ultrasonic and Novatel OEM star GPS module. The IMU is to estimate the full state of quadrotor. It provides three-axis attitude, attitude rate and acceleration. Besides that, six ultrasonic ranging finders with accuracy of 1cm are implemented to detect the obstacles. Two Novatel OEM star GPS modules are used as carrier phase differential GPS (DGPS). The DGPS and IMU work together as an integrated navigation system, which provides the three-dimensional position, velocity, attitude, attitude rate and acceleration's information to the platform.

Wang Qi *et al.* [13] present a new searching and saving quadrotor aircraft system to solve the on–site detection and rescue after major natural disasters and biological and chemical pollution. This system is consist of controller, gesture detection, quadrotor motor and drive, wireless controller, wireless video capturing and date transmission module, etc. In the gesture detection module, triaxial gyroscope will collect data of aircraft's angular velocity respectively in three directions, which are pitch rate, roll angle rate, and yaw angular rate; triaxial accelerometer will measure and collect data when temperature rising for correction and compensation; and height barometer is use the atmospheric pressure sensor and the MPX4115 sensor to control the aircraft's altitude.

The prototype of quadrotor presented by James F. Roberts *et al.* [14] consists of three rate gyroscopes, three accelerometers, one ultrasonic sensor, four infrared triangulation-based sensors, a high speed motor controller and a flight computer. The quadrotor is naturally a highly non-linear and unstable platform which requires stability controllers to deal with its fast dynamics. Three accelerometers are providing the automatic leveling. Three rate gyroscopes are for rotational dampening control. The altitude of the platform is measured using an ultrasonic sensor. The ultrasonic sensor is connected via a UART interface and the four infrared sensors are connected directly to the dsPICs analogue inputs. The MATLAB script is used to track the trajectory of the platform.

Zul Azfar *et al.* [15] discussed about a simple approach for implementing well-known PID controller in microcontroller for quadrotor project. By implementing the IMU sensor fusion, the quadrotor will stabilize while in flight. Sensor fusion is a common method to increase the accuracy and remove noise in sensory system because just using a low pass filter will not gain accuracy from the sensor.

G. Angeletti *et al.* [16] presents the development of hardware and software of quad-rotor’s indoor environment navigation, by controlling the 6 DOF of vehicle with different sensors. In laser based approach for quadrotor indoor hovering, sensors involved are ultrasonic sensor to control the quad-rotor height and laser range finder is to measure the angle and range objects. In vision based approach, monocular camera is used for capture a large area; on the other hand, this lens introduces a (easy-to-correct) distortion, which is quadratic with the distance from the center of the image.

The avionic system by Mustafa Ilarslan *et al.* [12] is designed to study the flight dynamics of such the development of on-board electronics that consists of a sensor payload, embedded computer, and other hardware, along with the embedded software used a set of ultrasonic sensor for altitude sensing, collision avoidance and automatic take-off and landing operation, supported by a set of barometric sensor. The sets of laser scanner (range finder) function as a provision for complimentary collision avoidance aid. The avionic system of quadrotor uses a common object-oriented software language, COTS Real-time operating system.

Nick Chamnong [17] presents the development of a localization system that incorporates visual information from consumer grade camera on a VTOL UAV. Visual sensor is used to extract 3D trajectory information for VTOL UAV in real time. GPS data and other sensory information can be used together with visual sensor’s information in order to increase the accuracy of the navigation system. A vision algorithm detects and tracks visual features based on catadioptric images obtained from an on-board omni-directional camera. By continuously tracking ground objects that appear in the camera Field Of View (FOV), combined with UAV attitude estimated from IMU and various sensor's data including GPS itself, it is possible to better estimate the position and velocity of the UAV than with GPS alone. IMU is the main component of inertial navigation; the data collected from the IMU's sensors allows a computer to track a craft's position, using a method known as dead reckoning. It consists of 3 axes in- vensense Gyros, Analog Devices ADX330 Accelerometer connected directly to the flight control system and HMC5843 - Triple Axis Magnetometer for reduce noise interference. All flight data and telemetry will be sent to the ground base station;

command and configuration can also be sent to the flight controller via Digi XBEE Pro Data link.

2.3 Filter Implementation

The saving quadrotor aircraft system presents by Wang Qi *et al.* [13] use the digital incremental PID control technology based on Kalman filtering algorithms to achieve effective control for the quadrotor.

James F. Roberts *et al.* [14] presents a quadrotor prototype implemented with complementary filter will takes the integrated angular rate of the gyroscope and the measured Euler angle from the accelerometers and fed the output into a proportional-integral-derivative controller. Four infrared triangulation-based sensors are for correcting the drifting of gyro run-away and external acceleration. The infrared sensor will also provide reference for maneuvering in 2D space and allow for collision detection of large object.

Zul Azfar *et al.* [15] presents that the algorithms using the sensor fusion which is the simplified Kalman Filter can be coded in C language to be embedded in the microcontroller. IMU is a combination of accelerometer and gyroscope, where accelerometer measure acceleration and used to measure tilting by measuring the inertial force due to gravity applied in x, y, and z axis; gyroscope is a sensor to measure angular rate in an axis and rotation rate of body.

Norafizah Abas *et al.* [18] presents the identification method of unknown parameter for quadrotor. The unknown parameter will be identified using state estimation method with the implementation of Unscented Kalman Filter (UKF). The main processes involved are dynamic modeling of quadrotor and implementation of UKF algorithms. UKF with two phases, predict and update. The predict phase estimate based on the previous estimated time step to produce the estimation of current time step state. The update phase used current step measurement information to correct the prediction and get a new and more accurate state estimation. The thrust factors and drag factor are defined from force lift test. The performance of UKF is demonstrated based n flight test applied on quadrotor.

Kong Wai Weng *et al.* [19] presents the control system for quadrotor in term of VTOL. The experimental flying platform is flying robot that controlled by a pilot using radio frequency. The sensors used are compass sensor, Gyro sensor and accelerometer sensor. Due to the drift error and earth magnetic field that effects the readings of sensor, complementary filter is used to combine all the sensors' output together. The controller system implemented is PID control system, which is used to stabilize the quadrotor flying robot.

2.4 Summary

Based on the literature review conducted, details are illustrated as in table 2.1. In a nut shell, the mathematical modeling for the quadrotor system is derived using Newtonion derivation to obtain six differential equations. To identify the unknown parameter, speed test, force-lift test and bifilar pendulum test is carried out. The sensor that chooses to be used in this research is IMU sensor. IMU is used to measure the acceleration, angle of rotation and angular rate. In order to eliminate the noise from affect the reading of sensors, complementary filter is implemented. In order to validate the mathematical modeling, open loop test is carried out to compare its result shape with simulation test.

Table 2.1: Summary table of literature review

No.	Title	Author	Characteristics				
			Hardware	Dynamic Modelling and system ID	Controller	Sensors	Results
1	A VTOL Quadrotor Platform for Multi-UAV Path Planning [11]	Hui Bai, Shihuang Shao, and Hongyu Wang	ARM9 platform	Rapidly-exploring Random Tree (RRT) method	PID Controller	IMU, Ultrasonic, Novatel OEM star GPS module (DGPS)	obtain optimal path for travel with obstacle avoidance aid
2	System Design and Control Research for New Quadrotor Searching and Saving Aircraft [13]	Wang Qi, Yuan Mingxin, Zhang Peng, Cheng Fei, and Cao Yuqiang	Quadrotor's testbed	Pitching, roll angle, deviation of x, y, z direction; Kalman Filter	PID Controller	Triaxial Gyroscoper, Triaxial Accelerometer and Height Barometer	Feasibility - Get remote mifi control and feedback

3	Quadrotor Using Minimal Sensing For Autonomous Indoor Flight [14]	James F. Roberts, Timothy S. Stirling, Jean-Christophe Zufferey, and Dario Floreano	Quadrotor's prototype	Pitch & roll control method; infrared anti-drift control method ,infrared range detection	PID Controller	gyroscope, accelerometers, ultrasonic, infrared	Altitude control, collision avoidance, anti-drift control and automatic VTOL
4	A Simple Approach on Implementing IMU Sensor Fusion in PID Controller for Stabilizing Quadrotor Flight Control [15]	Zul Azfar and D. Hazry	-	Theorem Pythagoras in 3D, inclination angle estimation method, Simpified Kalman filter	PID Controller	IMU(accelerometer and gyroscope)	Stabilization
5	Autonomous Indoor Hovering with a Quadrotor [16]	G. Angeletti, J. R. Pereira Valente, L. Iocchi, and D. Nardi	The quadrotor X-3D-BL, developed by Ascending Tecnologies.	-	PI Controller (sonar); PD controller (laser)	sonar, ultrasonic, laser, monocular camera, accelerometer	Maintain a stationary pose (hovering)

6	Avionics System Design of a Mini VTOL UAV [12]	Mustafa Ilarslan, M. Kemal Bayrakceken, and Aydemlr Arboy	Quadrotor's prototype	-	PID controller	ultrasonic, barometric, laser	VTOL and collision avoidance aid
7	3D Visual Odometry in Real Time VTOL UAV Navigation [17]	Nick Chamnong	DIY Drone's ArduPilot Mega	-	PID controller	IMU (gyroscope, accelerometer, magnetormeter) and GPS	VTOL, navigation
8	Parameter Identification of an Autonomous Quadrotor [18]	Norafizah Abas, Ari Legowo, and Rini Akmeliawati	DIY Quadrotor Flying Robot Testbed	Newton-Euler formalism; Unscented Kalman Filter (UKF)	-	6DOF IMU and ultrasonic sensor	identify and estimate the needed parameter for an autonomous quadrotor.
9	Design and Control of a Quad-Rotor Flying Robot For Aerial Surveillance [19]	Kong Wai Weng and Mohamad Shukri b. Zainal Abidin	Modified Dragonflyer quadrotor	Complementary Filter	PID Controller	Compass sensor, accelrometer, gyro sensor	VTOL

CHAPTER 3

METHODOLOGY

3.1 Methodology Flow Chart

Figure 3.1 shows the flow chart of methodology of this research. First of all, research begins with literature review by study on others journal in order to do comparison and find the best way and method to carry out the research. After that, Newtonion derivation through Newton-Euler method is used to derive the mathematical modeling of quadrotor system. Several tests are carried out to identify the unknown parameter, include direct physical measurement on the testbed, speed test and force-lift test of rotors, and bifilar pendulum test. Next, to obtain a more reliable result from raw data of IMU sensor, complementary filter is implemented for system estimation. Conversion on raw data of sensor is carried out to centralize the data obtained. To prove the mathematical modeling derived is identical to the quadrotor testbed, the open loop of quadrotor system is designed in Matlab Simulink and the six equations are inserted into it. Simulation test is run and the output result is recorded. On the other hand, open loop test on quadrotor testbed is carried out to obtain the result from real time implementation. Both result from simulation test and real time implementation is analyzed for its data pattern.

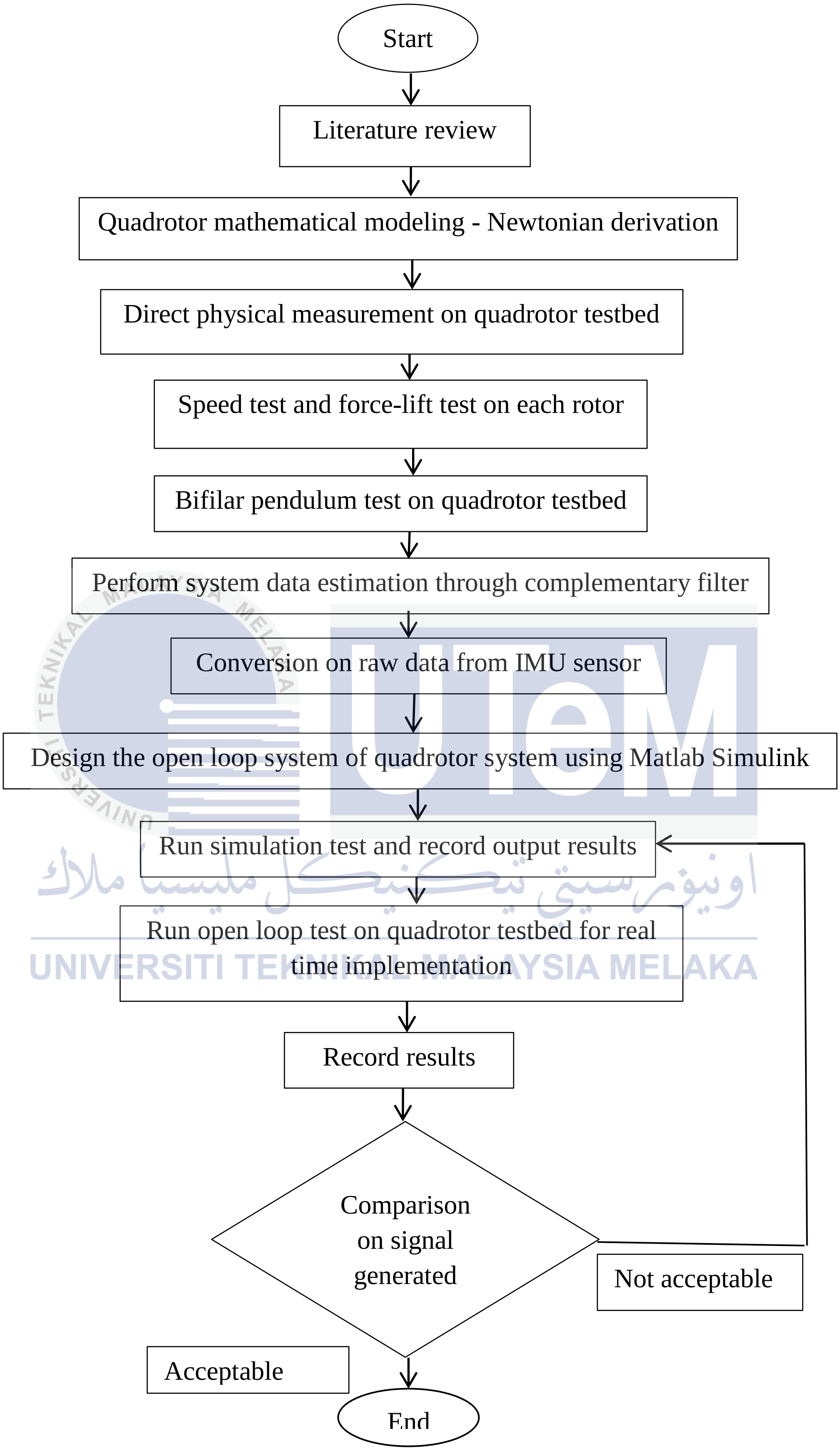

Figure 3.1: Flow chart of methodology

3.2 Mathematical Modeling

The mathematical modeling of quarotor can be derived through Jacobian derivation and Newtonian derivation. The following dynamic model equation of quadrotor is derived through Newton-Euler Method. Beside of directly take measurement on quadrotor testbed, several tests are carried out in order to obtain unknown parameter, which are speed test, force-lift test and bifilar pendulum test.

3.2.1 Introduction to Quadrotor System

There are two frame operate in quadrotor, inertial frame and body frame. The inertial frame is earth fixed and the gravity point to negative z direction. The body frame's origin is the center of gravity which defined by orientation, arms of quadrotor pointed to x and y directions and rotor axes pointed to positive z direction. [20]

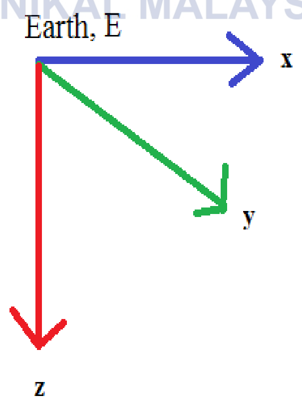

Figure 3.2: Quadrotor Inertial Frame

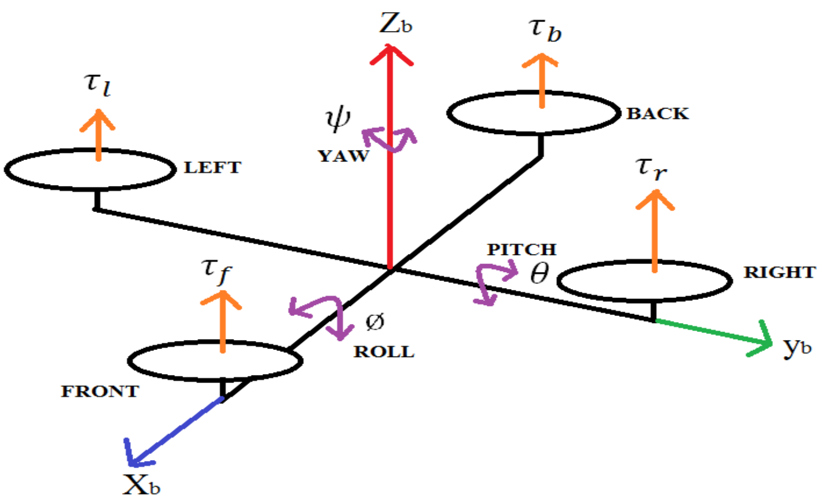

Figure 3.3: Quadrotor Body Frame

Newton-Euler method is used in derivation for 6DOF equation for quadrotor. The orientation of quadrotor is shown by Euler angles, which are: roll (Ø), pitch (θ) and yaw (ψ). The vector X_b, Y_b, Z_b shows the position of quadrotor in body frame. The mathematical modeling is derived and several unknown parameter is identified through the test under section 4.2.

3.2.2 Physical Measurement

Figure 3.4 illustrates the quadrotor testbed of the research. The measurements are taken directly from the quadrotor testbed. The properties that are measured directly from it are the weight of the quarotor testbed, weight of a single rotor, distance from center body of quadtrotor to rotor and distance between the front rotor to back rotor. All of these measurements will be used in calculation for other parameter. The data is recorded in table 4.1. This quadrotor testbed is used for physical measurement and real time implementation.

Figure 3.4: Quadrotor testbed

3.2.3 Speed Test

Speed test is carried out to identify the relationship of timing pulse in Pulse Width Modulation and the programmed signal from Arduino board that would trigger the rotor to start to rotate. Tachometer is the device that measures the speed of rotation of the rotor once it is triggered. Tachometer is attached to the rotor perpendicularly to the propeller. Once the propeller starts to rotates, tachometer will count the total number of rotation per second. The setup of speed test is illustrated in figure 3.6.

Figure 3.5: Turnigy tachometer

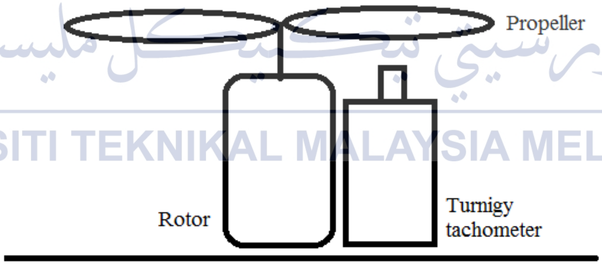

Figure 3.6: Setup of speed test

3.2.4 Force-lift Test

Force-lift test is carried out to measure the force generated by propeller on certain rotational speed. As the disturbance is assumed to be negligible, the force acting on body are propeller thrust force, rotor torque, and gravitational force. The four rotors are initially calibrated using AeroQuad Configurator. The experiment setup is shown in Figure 3.7. The rotor-propeller is fixed on the weighting machine. Weighting machine is set to zero at initial stage. Experiment begins by give a signal to rotor from 0 to 160 which refers to PWM value to control the speed of rotor and corresponding to rotational speed of propeller. The value is set in coding and programmed to Arduino board. The reading on weighting machine as the balance measurements for weight (in grams) is recorded. Force is calculated based on Newton's second law, F=mg, where m is the weight and g is the gravity of earth. Experiment is repeated for all four rotor. The coding to run the rotor attached in Appendix B.

Figure 3.7: Experiment setup for speed test and force-lift test

3.2.5 Bifilar Pendulum Test

To obtain the moment inertia parameter of the quadrotor system, bifilar pendulum test is carried out. The parts that needed are quadrotor test bench, quadrotor testbed, stopwatch and ropes. First, quadrotor testbed is hanged on the test bench and secure with ropes. Stop watch is ready to record the time taken for the quadrotor testbed swing about the axis until it stops at stationary position. Experiment begins with releasing the quadrotor testbed and let it to rotate freely along the x axis. The same steps are repeat for two time for each axis to obtain more accurate measurement.

Mass moment of inertia about rotation axis is calculated based on below formula:

$$J = \left(\frac{T}{2\pi}\right)\frac{mgR^2}{L} \qquad (kgm^2/s) \tag{3.1}$$

Where:

T = time taken of the quadrotor system swing about the axis (s)

m = mass of the quadrotor system (kg)

g = gravity (m/s^2)

R = radius of the quadrotor system (m)

L = length of ropes (m)

Angular rotation data of the quadrotor testbed is taken from IMU sensor. The values obtained were present in the table and plotted into graph using Serial Chart. The inertial acting reference to axis x, y, and z is determined through calculations:

$$I_x = I_y = \frac{2MR^2}{5} + 2l^2\overline{m} \quad (\text{kg}m^2) \tag{3.2}$$

$$I_z = \frac{2MR^2}{5} + 4l^2\overline{m} \quad (\text{kg}m^2) \tag{3.3}$$

Where:

M= Mass (m)

R= Radius (m)

l = Lever length (m)

$\overline{m}$ = Mass of each rotor (kg)

Experiment setup:

Figure 3.8: Setup for yawing motion

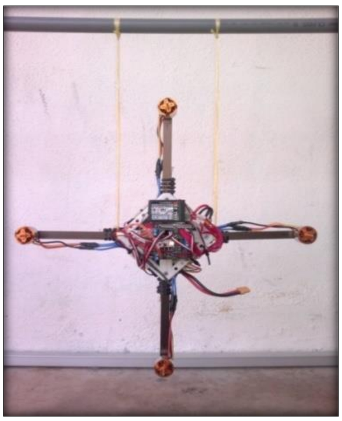

Figure 3.9: Setup for rolling and pitching motion

3.3 System estimation with complementary filter

Complementary filter is a simple data estimation method which often applied in flight control field by combining the data measured by sensors. In this research, the filter is implemented because the raw data from IMU sensor consist of too many noise and drift. It would eliminate the horizontal acceleration and external disturbance value that read by accelerometer, correct the gyroscopic drift from gyroscope. The data from accelerometer and gyroscope is combined to obtain more reliable data for the quadrotor's x, y, z, roll, pitch and yaw during flight. The data after filtered does not contain external disturbance and drift which make these data more reliable.

From gyroscope, the integrated sensor signal is referred as orientation angle of quadrotor. However the integration is lead to drift, the small rate will cause the new angle far away from actual angle. Besides that, the accelerometer will read any horizontal acceleration of the quadrotor as the change of orientation angle. Therefore, complementary filter is implemented to combine the reading of these two sensors and eliminate the problem. [21]

3.3.1 Sensors

The sensors used are 6DOF IMU sensor. This sensor is interfaced with Arduino board, and placed at the center of gravity for quadrotor. IMU sensor consists of accelerometer and gyroscope.

Acceleration and the force of gravity are measured by the accelerometer. Figure 3.1 shows that the x-axis of accelerometer read as 0g for the force of gravity, while y-axis read as -1g.

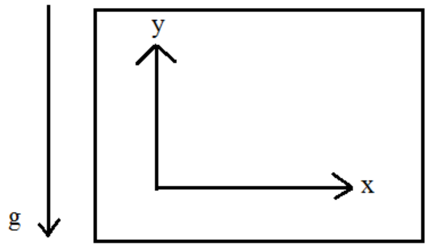

Figure 3.10: accelerometer symbol

It can be used to measure the slope or the angle of rotation of quadrotor during the flight. When the accelerometer is rotate to the right, x-axis shows a slight positive value. As the accelerometer is rotate to the left, x-axis value will be a negative value. Reading on y-axis does not contribute much as its sensitivity to the direction of rotation is low. X-axis is more sensitive to even slight changes when any movement occurs.

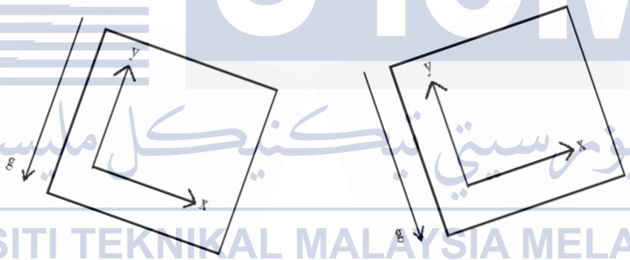

Figure 3.11: Symbol of accelerometer movement

However, the raw data read by accelerometer is not reliable as the x-axis will read any reading includes horizontal acceleration and disturbance on the quadrotor, it is not able to differentiate out the needed data, which is the force of gravity.

On the other hand, gyroscope is used to measure the angular rate or angular velocity of quadrotor. Theoretically this sensor will give the zero reading when the quadrotor is in stationary.

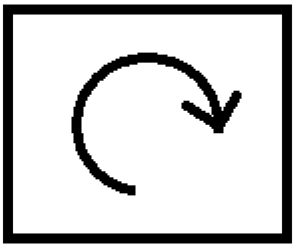

Figure 3.12: Gyroscope symbol

In order to obtain the orientation angle, the signal from gyro sensor has to be integrated. However, when the signal is integrated, noise will be integrated as well. Gyroscopic drift is occurred. Therefore, a small rate will cause the angle grow far away from actual angle.

Angular velocity, $\bar{\theta} = \frac{d\theta}{dt}$ (3.4)

By integrating the angular velocity, the angular position or the angle can be obtained.

Angle, $\theta(t) = \int_0^t \bar{\theta}\,(t)\, dt = \bar{\theta} * t$ (3.5)

3.3.2 Complementary Filter

To overcome the problems occurred on sensors mentioned in section 3.3.1, complementary filter is implemented. The average reading from accelerometer is taken to be represented as actual orientation angle of quadrotor. The gyroscopic drift is corrected by the combination data of integrated gyro signal for short period and reading from accelerometer.

At stationary, the reading from gyroscope is not perfectly zero due to the gyroscopic drift; the reading is far different from the actual reading as the small rate is keep adding into it. Therefore the gyroscope reading is only reliable on short term; the data is filter with high-pass filter. The accelerometer can detect any disturbance and noise easily, the reading will be manipulated. The data is reliable only on the long term, therefore it is filter with low-pass filter.[22]

Figure 3.13 shows the concept of complementary filter.

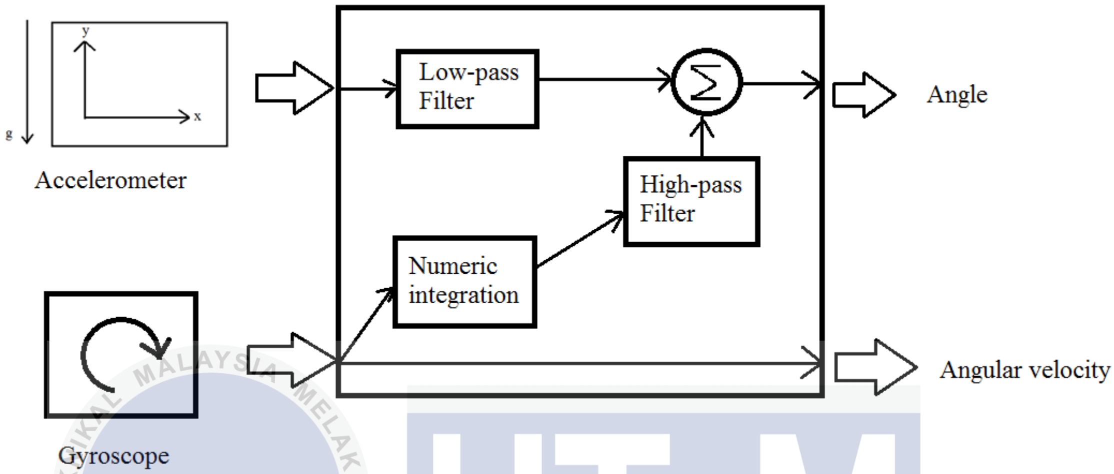

Figure 3.13: Theory of complementary filter

The numeric integration method on gyroscope data is to obtain the new position of the quadrotor every motion by taking the previous position and add with the current change in position. The position is calculated through multiplying the velocity with each time of the velocity obtained. In another way, we express in equation form as below:

$$\text{Position} = \text{velocity} * dt \tag{3.6}$$

$$\text{Angle} = \text{angular velocity} * dt \tag{3.7}$$

Low –pass filter is used to only let the long term and stable data be accepted and eliminates all the short term which is the fluctuations. This filter is implemented in this research by coding on the Arduino board program loop. For example,

$$\text{Angle} = (0.98)*\text{angle} + (0.02)*\text{x\_acc} \tag{3.8}$$

Where the accelerometer reading is suddenly increased to higher reading compared to the previous estimated angle reading, and drop back to the previous range, then the sudden rise reading is known as the fluctuation and will be ignore. The 0.98 and 0.02 is the filter coefficient of the loop.

The high-pass filter is works as allow short term data from gyroscope and filter out the data which steady all time. This is because the all-time steady data will be referred as the constantly vibration and disturbance reading that detected by the sensors. By filter it out, it means that the gyroscopic drift is removed as well.

To completely combine all the above together, one more aspect we have to pay attention is the time constant of each filter as this related to the data duration. For the low-pass filter, the data which shorter than time constant will be filter out, we calculate the time constant with the below formula:

$$\tau = \frac{a.dt}{1-a} \tag{3.10}$$

Where *a* is the filter coefficient, *dt* is sample period.

Sample period is the total time passed between the program loops. As the sample rate is 100Hz, sample period will be 0.01 second.

Time constant for the low pass filter and high pass filter is

$$\tau = \frac{a.dt}{1-a} = \frac{0.98(0.01\ sec)}{1-0.98} = 0.49\ sec \tag{3.11}$$

Therefore, when the time period is shorter than 0.49 sec, gyroscope data will be accepted while accelerometer data will be filtered out. On the other way, when the time period is longer than 0.49 sec, data accelerometer will be accepted and gyroscope data get to be filtered out.

The complementary filter combines the data with the following equation:

Angle = (0.98)*(angle + gyro*dt) + (0.02)*(x_acc) (3.12)

The constants (0.98 and 0.02) have to be sum up as 1; it can be changed to tune the filter properly. [23]

3.3.3 IMU Read Data Conversion

Accelerometer is to detect the friction force that directed in opposite direction from acceleration. The object is in a box, where the wall of box is referred as axes. Initially, the object is float at the center of box as the box is in no gravitational field environment. The reading of each axis are -x = 0g, -y=0g and -z=0g.

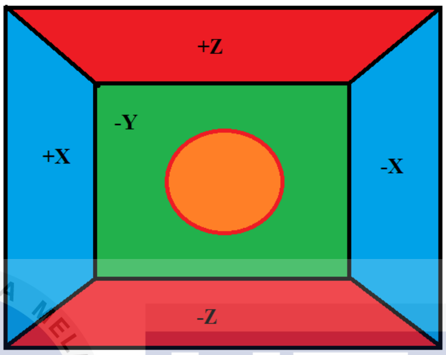

Figure 3.14: Object in no gravitational field environment

Once the box is shift to left, a force is created on the object and cause it be dragged and hit the wall –x. The friction force is cause by the acceleration. The friction force that applied by the object to the wall is measured by accelerometer. The reading of each axis are -x = -1g, -y=0g and -z=0g.

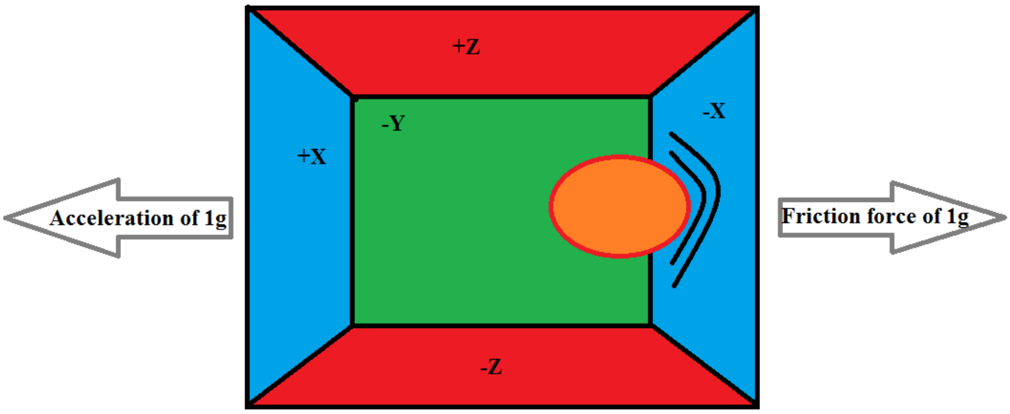

Figure 3.15: Friction force applied on object in no gravitational field environment

When the box is on earth, the object will hit wall –z when it is in stationary due to the gravitational force. Accelerometer still measures the gravitational force as acceleration. The reading of each axis are -x = 0g, -y=0g and -z=-1g.

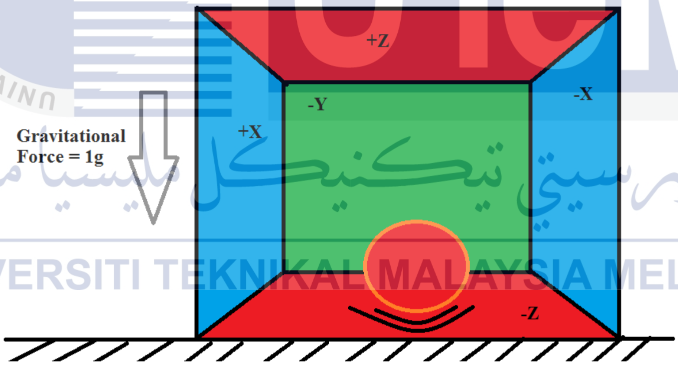

Figure 3.16: Object in gravitational field environment

When the box is rotated at 45degree to the right, the object will hit wall -x and –z. This situation shows the output for accelerometer with two axes. The reading of each axis are -x =$-\sqrt{2}g$, -y = 0g and -z=$-\sqrt{2}g$.

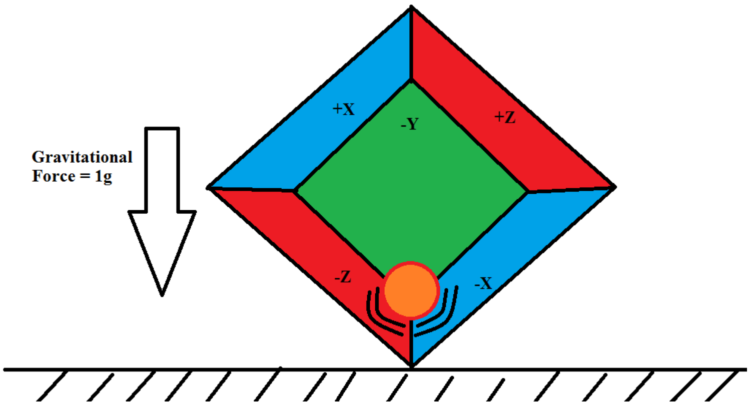

Figure 3.17: Object in gravitational field environment and contact with two axes

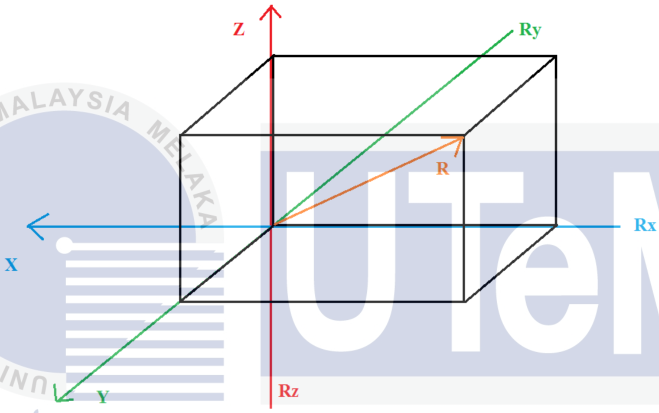

Figure 3.18: Vectors R in 3D

Refer to the Figure 3.18, based on the Pythagorean Theorem in 3D,

Vector R is the force vector that accelerometer measured. The relations are:

$$R^2 = R_x^2 + R_y^2 + R_z^2 \tag{3.13}$$

Therefore, on earth, the object is touched two axis, which is x and z, when substitute the value into the equation above, the gravitational force is 1g.

$$1^2 = \left(\sqrt{\frac{1}{2}}\right)^2 + 0 + \left(\sqrt{\frac{1}{2}}\right)^2 \tag{3.14}$$

When the digital accelerometer displayed the reading through 10bit ADC (Analog-to-Digital Converter) module, there are several steps to determine the force value. The below example only showed forR_x, R_y and R_z also use similar steps.

Step 1: given ADC R_x = 586, convert back to voltage by

$$V_{R_x} = \frac{\text{ADC } R_x \times V_{ref}}{1023} \quad [V] \tag{3.15}$$

Where Vref is from ADC module and 10bit ADC module is 1023 = $2^{10} - 1$

Step 2: Zero-gravitational voltage level from specs of accelerometer is used to find the voltage shift from 0g voltage.

$$\delta V_{R_x} = V_{R_x} - V_{Zero-G} \; [V] \tag{3.16}$$

Step 3: apply accelerometer sensitivity [mV/g] from specs of accelerometer.

$$R_x = \frac{\delta V_{R_x}}{Sensitivity} \; [g] \tag{3.17}$$

The inertial force vector x, $$R_x = \frac{\frac{ADCR_x \times V_{ref}}{1023} - V_{Zero-G}}{Sensitivity} \; [g] \tag{3.18}$$

After all inertial force vector expressed in g, angle can be calculated easily.

Angle to be calculated are the angles between X, Y, Z axis and force vector R.

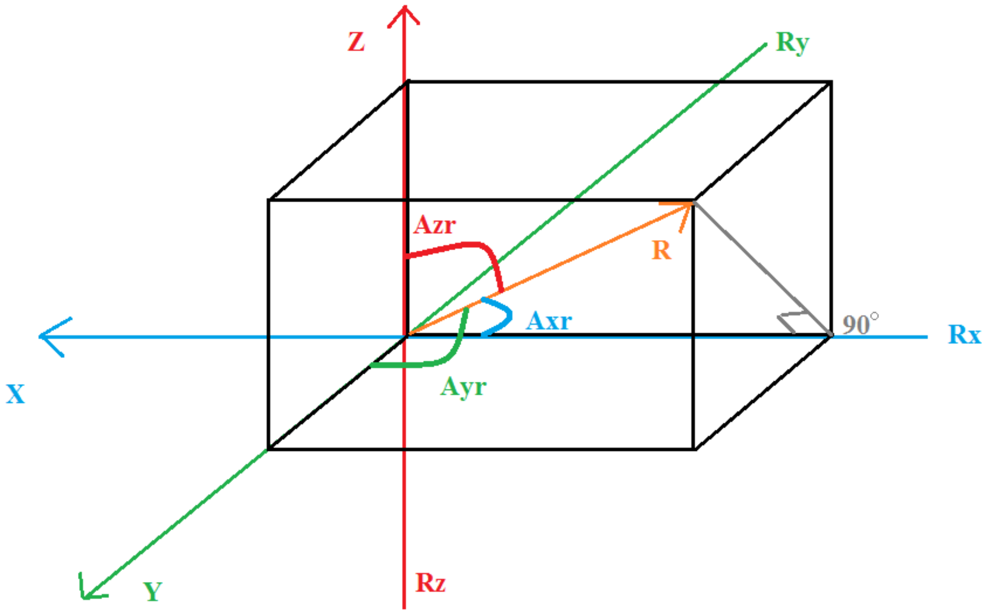

Figure 3.19: Angle between x, y, z axis and vector R

From the Figure 3.19, we can notice that:

$$\cos X = \cos(A_{Xr}) = \frac{R_x}{R} \quad (3.19)$$

$$\cos Y = \cos(A_{Yr}) = \frac{R_y}{R} \quad (3.20)$$

$$\cos Z = \cos(A_{Zr}) = \frac{R_z}{R} \quad (3.21)$$

They are same direction with R vector as:

$$\sqrt{(\cos X)^2 + (\cos Y)^2 + (\cos Z)^2} = 1 \quad (3.22)$$

For gyroscope, it is used to measure the rate of change of angle.

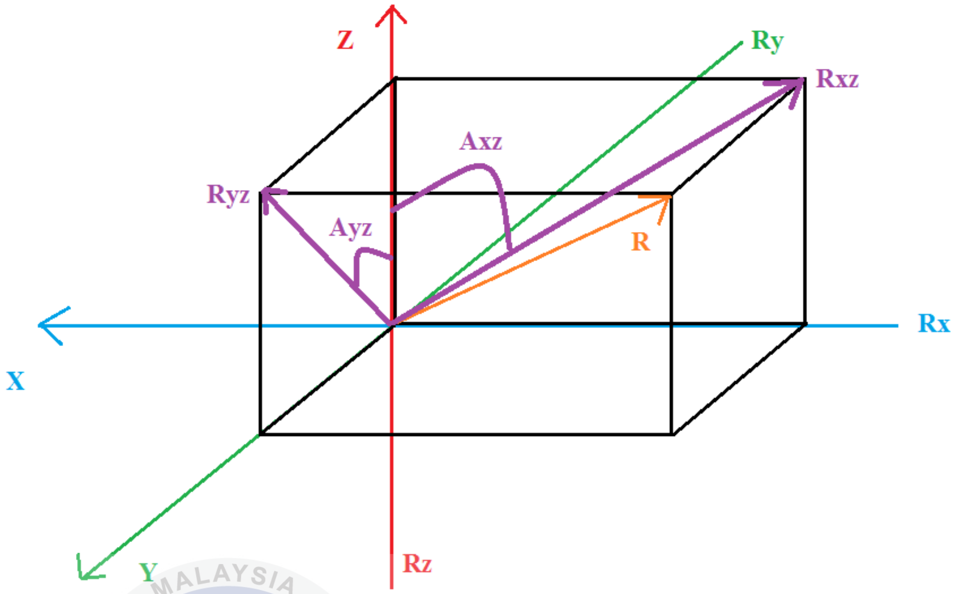

Figure 3.20: Rotation angle

The 2-axis gyro measure the rotation about X and Y axis.

R_{XZ} = Projection of friction force vector R on XZ plane

R_{YZ} = Projection of friction force vector R on YZ plane

Angles between Z axis and vectors are

A_{XZ} – Angle between R_{XZ} and Z axis

A_{YZ} – Angle between R_{YZ} and Z axis

From Pythagorean Theorem:

$$R_{XZ}^2 = R_x^2 + R_Z^2 \tag{3.23}$$

$$R_{YZ}^2 = R_Y^2 + R_Z^2 \tag{3.24}$$

$$\text{Rate } A_{XZ} = \frac{(A_{XZ1} - A_{XZ0})}{(t1 - t0)} \, [°/s] \tag{3.25}$$

Note that the rate value can be calculated from the 10bit gyro ADC module with the similar way as accelerometer.

$$\text{The angle rate, } A_{XZ} = \frac{\frac{ADCA_{XZ} \times V_{ref}}{1023} - V_{Zero-Rate}}{Sensitivity} \text{ [g]} \tag{3.26}$$

Where

$ADCA_{XZ}$ – Value from gyro ADCmodule

$Vref$ – ADC reference voltage

$V_{Zero-Rate}$ – The voltage that gyro output when there is no rotation

Sensitivity – Sensitivity of gyro in $\left[\frac{mV}{°/s}\right]$

3.3.4 IMU Data Collection

To test out equation of combining the raw data from accelerometer and gyroscope with complementary filter, following steps is carried out in order to observe the pattern of the result before and after filtered.

1. IMU sensor is placed at the center of the quadrotor testbed. USB cable used to connect laptop to Arduino board. The experiment set up is shown in Figure 3.21.
2. The source code to integrate the gyroscope and accelerometer is uploaded into Arduino Board.
3. The sensor is move in x, y and z orientation to obtain the result.
4. Serial chart is used to display the result of sensor's orientation angle.
5. Data is obtained from serial chart and plot using graph chart. Discussion on the result is shown in section 4.2.

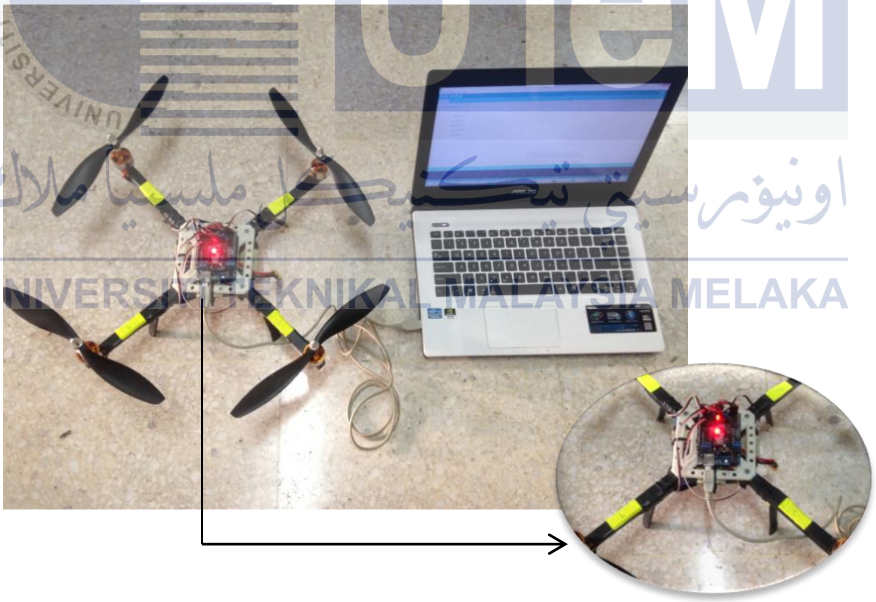

Figure 3.21: Set up for IMU sensor data collection experiment

Graph in figure 3.22 shows the result of the difference between angle data collected from gyroscope and accelerometer with and without complementary filter.

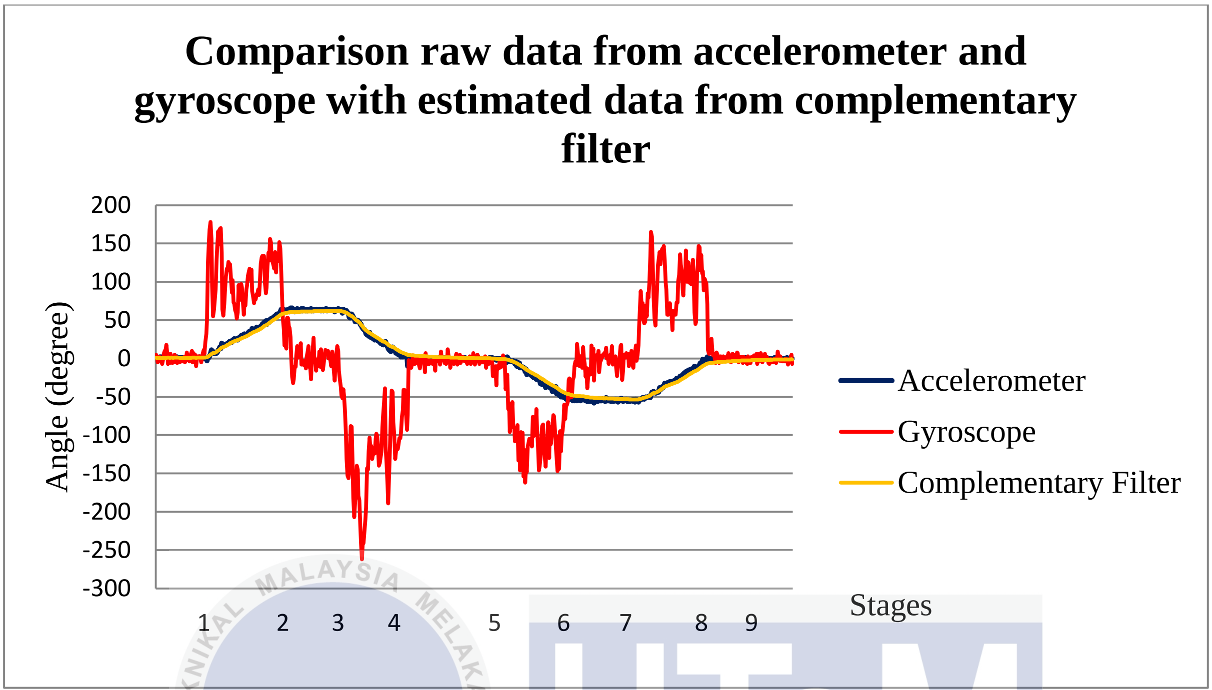

Figure 3.22: Comparison of angle data collected from gyroscope and accelerometer without filter with data from complementary filter

3.4 Validation of Modeling of Quadrotor

After derivation of the mathematical modeling of quadrotor system using Newton-Euler method, it need to be validate through testing to confirm it is identical with the available quadrotor testbed. In order to validate the modeling of quadrotor system, simulation test with Matlab Simulink and flight test through real time implementation on quadrotor testbed are carried out to collect reading and perform data pattern analysis.

3.4.1 Simulation in Simulink for Open-loop System

Block diagram of open-loop system for quadrotor is designed in Matlab Simulink. Simulation test is carried out in order to get output result from the modeling of quadrotor. First, the inputs block for the modeling is irregularities block to replicate the input signal of a real quadrotor that constantly effected by external disturbance and hard to be control. Irregularities input block is a better choice than constant input block because the input to the quadrotor system is continuously and hard to model or control since it is an open-loop. Then, the subsystem of the mathematical modeling is construct and the six differential equations ($\ddot{x},\ddot{y},\ddot{z},\ddot{\phi},\ddot{\theta},\ddot{\psi}$) which describe the acceleration and angular acceleration of the quadrotor is inserted into the function block as the parameters. The values of unknown parameters in the equation are listed in model initialization function so that when the simulation is run, the values will be called into the differential functions.

The result of the state output is displayed in scope block. In order to compare the data pattern of input and output, both signals is combined to display in one scope block for analysis. The block diagram of quadrotor system is showed in Figure 3.23.

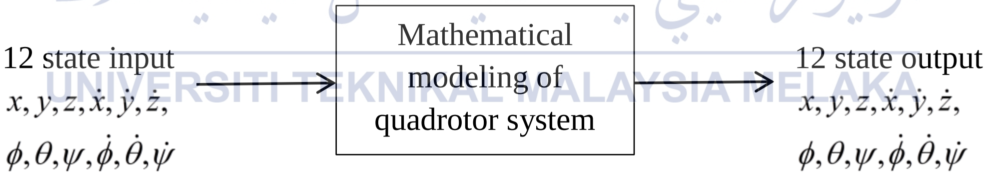

Figure 3.23: Block diagram of quadrotor system on Matlab Simulink

3.4.2 Real Time Implementation

Open loop test on quadrotor testbed is carried out to obtain the result from real time implementation. The result is used in comparison on data pattern with the simulation test result. The experiment begins with the quadrotor is fixed at a position with four edge is tied to fixed object which is the table that is stable and remain at the position during the test. This is a precaution step to hold the quadrotor testbed and prevent damage happened. The programming code is uploaded into the Arduino board through laptop. The quadrotor testbed is connected to laptop for command and data collection purpose. The next step is to run all four rotors by giving the program speed to the Arduino board using laptop. The data is transmitted to laptop through the USB cable, collected and recorded using Serial Chart. Result then is plotted in graph chart of Microsoft Excel and used for data pattern analysis.

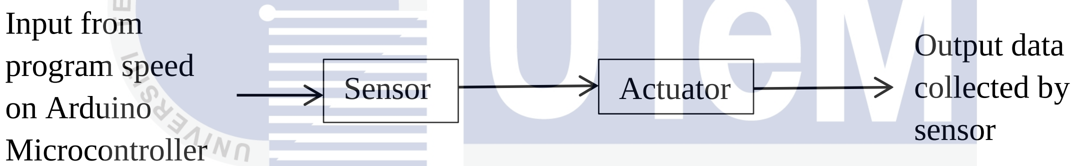

Figure 3.24: Block diagram of quadrotor testbed on real time implementation

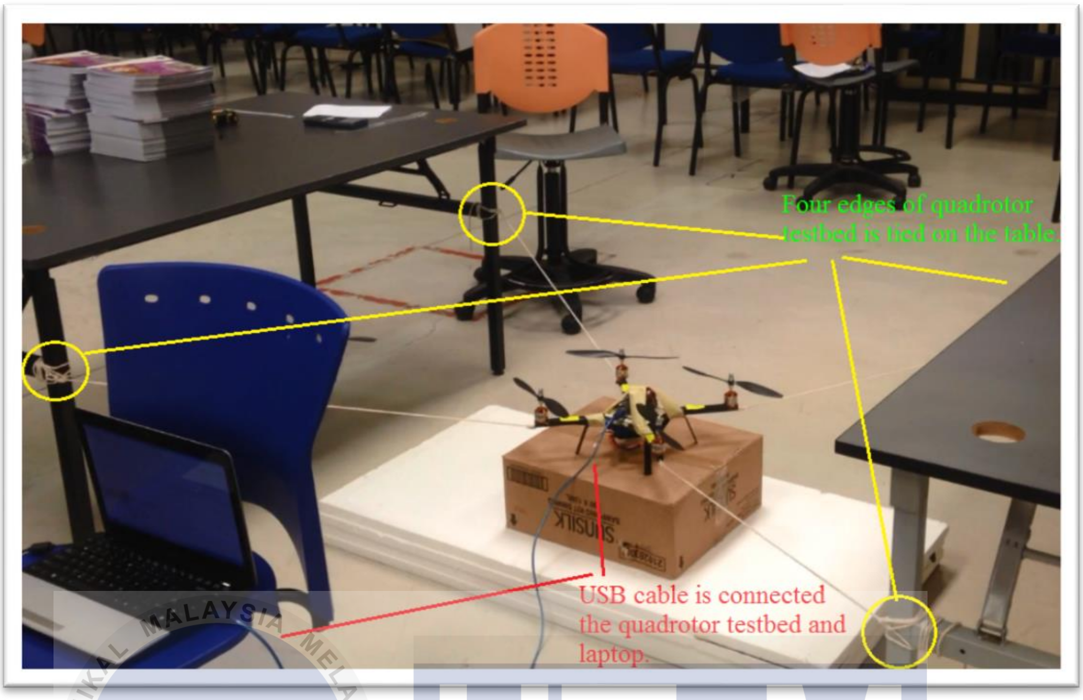

Figure 3.25: setup for quadrotor open loop test

3.4.3 Comparison on signals generated

The result from simulation test and real time implementation recorded and plotted in graph. First, the simulation test result compared in term of x, y, z, $\emptyset, \theta, \psi$ of its input and output to identify the shape form of a nonlinear quadrotor system. Next, result from open loop test of quadrotor testbed is compared in term of x, y, z, $\emptyset, \theta, \psi$ of its different value of the program speed. Data analysis will be carried out by comparing the results shape from both simulation test on Matlab Simulink simulation test and also the open loop test.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Mathematical Modeling

The following dynamic model equation of quadrotor shows the Newtonian derivation through Newton-Euler Formalism Method. Newton-Euler method is used in derivation for 6DOF equation for quadrotor. The orientation of quadrotor is showed by Euler angles, which are: roll (Ø), pitch (θ) and yaw (ψ). The vector x, y, and z shows the position of quadrotor in inertial frame. There are two frame operate in quadrotor, inertial frame and body frame. The inertial frame is earth fixed and the gravity point to negative z direction. The body frame's origin is the center of gravity which defined by orientation, arms of quadrotor pointed to x and y directions and rotor axes pointed to positive z direction.

The illustration of the body frame and inertial frame is shown in Figure 3.2.

The assumptions of this modeling are:

1. The structure of the quadrotor is rigid and symmetrical.
2. The body frame and center of mass is assumed coincide.
3. The pitch is fixed and the propeller is rigid.

4.1.1 Rotational Matrix

The right handed rotation of coordinate system about x-axis gives:

$$R^B_{Vx}(\emptyset) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\emptyset & \sin\emptyset \\ 0 & -\sin\emptyset & \cos\emptyset \end{pmatrix} \tag{4.1}$$

The right handed rotation of coordinate system about y-axis gives:

$$R^B_{Vy}(\theta) = \begin{pmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{pmatrix} \tag{4.2}$$

The right handed rotation of coordinate system about z-axis gives:

$$R^B_{Vz}(\psi) = \begin{pmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{4.3}$$

The overall transformation vector from body to inertial frame [20] is given by

$$\begin{aligned} & R^B_V(\emptyset, \theta, \psi) \\ &= R^B_{Vx}(\emptyset) R^B_{Vy}(\theta) R^B_{Vz}(\psi) \\ &= \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\emptyset & \sin\emptyset \\ 0 & -\sin\emptyset & \cos\emptyset \end{pmatrix} \begin{pmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{pmatrix} \begin{pmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \\ &= \begin{pmatrix} \cos\theta\cos\psi & \cos\theta\cos\psi & -sin\theta \\ sin\emptyset\ \sin\theta cos\psi - cos\emptyset sin\psi & sin\emptyset\ \sin\theta sin\psi + cos\emptyset cos\psi & sin\emptyset cos\theta \\ cos\emptyset\ \sin\theta cos\psi + sin\emptyset sin\psi & cos\emptyset\ \sin\theta sin\psi - sin\emptyset sin\psi & cos\emptyset cos\theta \end{pmatrix} \end{aligned} \tag{4.4}$$

4.1.2 Force and Torque

The total force acting on quadrotor is

$$F = F_f + F_r + F_b + F_l \tag{4.5}$$

Rolling torque produced by the forces of left and right rotors:

$$\tau_{\emptyset} = l\,(F_l - F_r) \tag{4.6}$$

Pitching torque produced by the forces of front and back rotors:

$$\tau_{\theta} = l\,(F_f - F_b) \tag{4.7}$$

Yawing torque is produced by the drag of propellers due to Newton's third law:

$$\tau_{\psi} = \tau_l + \tau_r - \tau_f - \tau_b \tag{4.8}$$

The direction of torque is in opposite direction of motion of propeller.

4.1.3 Kinematic Movement of Quadrotor

The 4 control inputs for quadrotor are from four rotors, which are u_1, u_2, u_3 and u_4.

Throttle movement: increase or decrease same amount of the velocity of all rotors.

$$u_1 = b(\omega_l^2 + \omega_r^2 + \omega_f^2 + \omega_b^2) \tag{4.9}$$

Roll movement: the angular velocity of rotor right increased and rotor left decreased while keeping whole trust constant.

$$u_2 = b(\omega_l^2 - \omega_r^2) \tag{4.10}$$

Pitch movement: the angular velocity of rotor back increased and rotor front decreased.

$$u_3 = b(\omega_f^2 - \omega_b^2) \tag{4.11}$$

Yaw movement: the angular velocity of rotor front and back increased; rotor left and right decreased.

$$u_4 = d(\omega_l^2 + \omega_r^2 - \omega_f^2 - \omega_b^2) \tag{4.12}$$

4.1.4 Thrust Force

The thrust force (F_i) of a propeller is proportional to the square of its rotational speed (ω_i), where $i = l, r, f, b$ refer to left, right, front and back rotor, τ is the thrust factor.

$$F_i = \tau \, \omega_i^2 \tag{4.13}$$

Total thrust force applied on four rotors are

$$F_T = \tau \sum \omega_i^2 = u_1 \tag{4.14}$$

Acceleration of quadrotor is written as:

$$\begin{pmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{pmatrix} = \begin{pmatrix} -(\cos\phi \sin\theta \cos\varphi + \sin\phi \sin\psi).(\frac{U_1}{m}) \\ -(\cos\phi \sin\theta \cos\varphi + \sin\phi \cos\psi).(\frac{U_1}{m}) \\ g - (\cos\phi \cos\theta).(\frac{U_1}{m}) \end{pmatrix} = \begin{pmatrix} -(\cos\phi \sin\theta \cos\varphi + \sin\phi \sin\psi).(\frac{U_1}{0.816}) \\ -(\cos\phi \sin\theta \cos\varphi + \sin\phi \cos\psi).(\frac{U_1}{0.816}) \\ 9.81 - (\cos\phi \cos\theta).(\frac{U_1}{0.816}) \end{pmatrix} \tag{4.15}$$

Where m is the total mass of quadrotor and U_1 is the control input from rotor.

4.1.5 Moment Equations

The equation of angular momentum at body frame is derived as:

$$I \cdot \ddot{\Omega} = -(\dot{\Omega} \times I \cdot \dot{\Omega}) - M_G + M \tag{4.16}$$

Where I = Inertia matrix

M = Torque applied to the quadrotor

M_G = Gyroscopic torque

Torque on quadrotor system is given by:

$$M = \begin{bmatrix} Lb(\omega_4^2 - \omega_2^2) \\ Lb(\omega_3^2 - \omega_1^2) \\ d(\omega_2^2 + \omega_4^2 - \omega_1^2 - \omega_3^2) \end{bmatrix} \tag{4.17}$$

Where d = drag factor

L = length from rotor to opposite rotor

Gyroscopic torque is obtained by the moment produced by every single pair of rotors:

$$M_G = I_R \cdot \left(\dot{\Omega} \times \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \cdot (\omega_2 + \omega_4 - \omega_1 - \omega_3) \right) \tag{4.18}$$

Where I_R = Rotor inertia

By substituting equation (4.17) and equation (4.18) into equation (4.16):

$$I \cdot \ddot{\Omega} = -(\dot{\Omega} \times I \cdot \dot{\Omega}) - \left[I_R \cdot \left(\dot{\Omega} \times \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \right) \cdot (\omega_2 + \omega_4 - \omega_1 - \omega_3) \right] + \begin{bmatrix} Lb(\omega_4^2 - \omega_2^2) \\ Lb(\omega_3^2 - \omega_1^2) \\ d(\omega_2^2 + \omega_4^2 - \omega_1^2 - \omega_3^2) \end{bmatrix} \tag{4.19}$$

The final differential equation for orientation is shown as below:

$$\ddot{\Omega} = \begin{bmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{bmatrix} = -I^{-1}\left[\begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} \cdot I \begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix}\right] - I^{-1}\left[I_R \cdot \begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} \times \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \cdot (\omega_2 + \omega_4 - \omega_1 - \omega_3)\right]$$
$$+ I^{-1}\begin{bmatrix} Lb(\omega_4^2 - \omega_2^2) \\ Lb(\omega_3^2 - \omega_1^2) \\ d(\omega_2^2 + \omega_4^2 - \omega_1^2 - \omega_3^2) \end{bmatrix} \tag{4.20}$$

4.1.6 Calculation for Moment of Inertia

From Newton's law, the translational motion of quadrotor is

$$m\frac{dv}{dt_i} = f \tag{4.21}$$

Where m = mass of quadrotor

f = total applied on the quadrotor

v = velocity vector

Constant inertia matrix, I is given by

$$I = \begin{pmatrix} I_x & -I_{xy} & -I_{xz} \\ -I_{xy} & I_y & -I_{yz} \\ -I_{xz} & -I_{yz} & I_z \end{pmatrix} \tag{4.22}$$

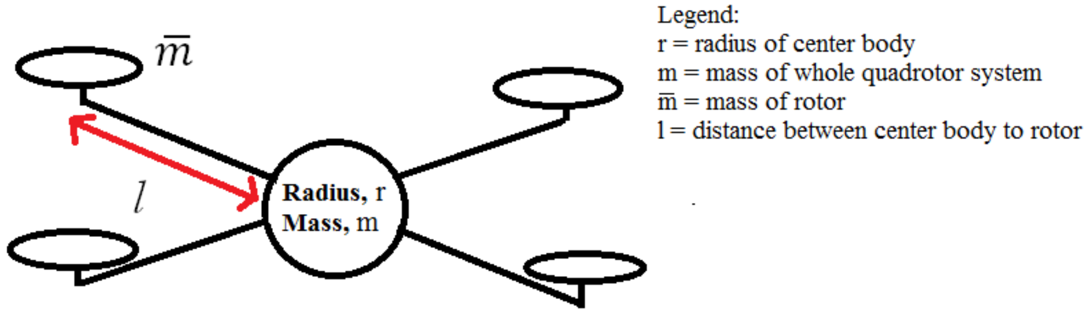

Figure 4.1: Quadrotor frame for inertia calculation

Moments of inertia for quadrotor are calculated by assuming the center body with a spherical shape with mass M and radius r. End of each frame has a mass of $\bar{m}$(the rotor mass) with a distance of l from the center. $\bar{m}$ is the mass of each rotor.

As the quadrotor is symmetric about all three axes, therefore $I_{xy} = I_{xz} = I_{yz} = 0$

$$I = \begin{pmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{pmatrix} \tag{4.23}$$

Therefore,

$$I^{-1} = \begin{pmatrix} \frac{1}{I_x} & 0 & 0 \\ 0 & \frac{1}{I_y} & 0 \\ 0 & 0 & \frac{1}{I_z} \end{pmatrix} \tag{4.24}$$

The inertia for a solid sphere is given by $I = \dfrac{2mr^2}{5}$. Therefore,

$$I_x = \frac{2mr^2}{5} + 2l^2\bar{m} \tag{4.25}$$

$$I_y = \frac{2mr^2}{5} + 2l^2\bar{m} \tag{4.26}$$

$$I_z = \frac{2mr^2}{5} + 4l^2\bar{m} \tag{4.27}$$

Where

Mass, m = 0.816kg

Radius of center body, r =0.07m

Distance between center body to rotor, l = 0.21m

Mass of each rotor, $\bar{m}$ =0.055 kg

$$I_x = I_y = \frac{2mr^2}{5} + 2l^2\overline{m}$$

$$= \frac{2(0.816)(0.07)^2}{5} + 2(0.21)^2(0.055) \tag{4.28}$$

$= (1.59936 \text{ x } 10^{-3}) + (4.851\text{x } 10^{-3})$

$= 6.45036 \text{ x } 10^{-3} \text{ kgm}^2$

$$I_z = \frac{2mr^2}{5} + 4l^2\overline{m}$$

$$= \frac{2(0.816)(0.07)^2}{5} + 4(0.21)^2(0.055) \tag{4.29}$$

$= (1.59936 \text{ x } 10^{-3}) + (9.702 \text{ x } 10^{-3})$

$= 0.0113 \text{ kgm}^2$

Take the four kinematic movement of quadrotor from section 4.1.3, where rolling movement is due to the angular velocity of rotor right increased and rotor left decreased while keeping whole trust constant; pitching movement due to the angular velocity of rotor back increased and rotor front decreased, while yawing movement is due to the angular velocity of rotor front and back increased; rotor left and right decreased. Therefore angular acceleration can be written as:

$$\begin{pmatrix} \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{pmatrix} = \begin{pmatrix} \dot{\theta}\dot{\psi}\left(\frac{I_y - I_z}{I_x}\right) - \frac{I_R}{I_x}\dot{\theta}\cdot g(u) + \frac{L}{I_x}u_2 \\ \dot{\phi}\dot{\psi}\left(\frac{I_z - I_x}{I_y}\right) - \frac{I_R}{I_y}\dot{\phi}\cdot g(u) + \frac{L}{I_y}u_3 \\ \dot{\theta}\dot{\phi}\left(\frac{I_x - I_y}{I_z}\right) + \frac{L}{I_z}u_4 \end{pmatrix}$$

$$= \begin{pmatrix} \dot{\theta}\dot{\psi}\left(\frac{6.45\times10^{-3} - 0.0113}{6.45\times10^{-3}}\right) - \frac{I_R}{6.45\times10^{-3}}\dot{\theta}\cdot g(u) + \frac{0.42}{6.45\times10^{-3}}u_2 \\ \dot{\phi}\dot{\psi}\left(\frac{0.0113 - 6.45\times10^{-3}}{6.45\times10^{-3}}\right) - \frac{I_R}{6.45\times10^{-3}}\dot{\phi}\cdot g(u) + \frac{0.42}{6.45\times10^{-3}}u_3 \\ \dot{\theta}\dot{\phi}\left(\frac{6.45\times10^{-3} - 6.45\times10^{-3}}{0.0113}\right) + \frac{0.42}{0.0113}u_4 \end{pmatrix} \tag{4.30}$$

$$= \begin{pmatrix} (-0.752)\dot{\theta}\dot{\psi} - \frac{I_R}{6.45\times10^{-3}}\dot{\theta}\cdot g(u) + 65.116u_2 \\ (0.752)\dot{\phi}\dot{\psi} - \frac{I_R}{6.45\times10^{-3}}\dot{\phi}\cdot g(u) + 65.116u_3 \\ 37.168u_4 \end{pmatrix}$$

4.1.7 State –space Equation

State space model able to combine both force and moment differential equation to best describe the quarotor system modeling. Therefore, the Quadrotor system differential equation derived in equation 4.15 and equation 4.31 are rewritten in state space equation of below form:

$$\dot{x} = Ax + Bu \tag{4.31}$$

$$y = Cx + Du \tag{4.32}$$

Where

x = state vector

$\dot{x}$ = First derivative of state vector with respect to time

y = output vector

u = input/control vector

A = system matrix

B = input matrix

C = output matrix

D = feed forward matrix

Let

$x_1 = \dot{x}$ $\quad$ $x_2 = \dot{y}$ $\quad$ $x_3 = \dot{z}$

$\dot{x_1} = \ddot{x} = x_4$ $\quad$ $\dot{x_2} = \ddot{y} = x_5$ $\quad$ $\dot{x_3} = \ddot{z} = x_6$

$x_7 = \dot{\emptyset}$ $\quad$ $x_8 = \dot{\theta}$ $\quad$ $x_9 = \dot{\psi}$

$\dot{x_7} = \ddot{\emptyset} = x_{10}$ $\quad$ $\dot{x_8} = \ddot{\theta} = x_{11}$ $\quad$ $\dot{x_9} = \ddot{\psi} = x_{12}$

The 12 state equations of quadrotor system are as below:

$$(\dot{x}) = \begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \ddot{x} \\ \ddot{y} \\ \ddot{z} \\ \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \\ \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{pmatrix} = \begin{pmatrix} \mathrm{x}_1 \\ x_2 \\ x_3 \\ -(\cos\phi\sin\theta\cos\varphi + \sin\phi\sin\psi).(\frac{\mathrm{U}_1}{0.816}) \\ -(\cos\phi\sin\theta\cos\varphi + \sin\phi\cos\psi).(\frac{\mathrm{U}_1}{0.816}) \\ 9.81 - (\cos\phi\cos\theta).(\frac{\mathrm{U}_1}{0.816}) \\ x_7 \\ x_8 \\ x_9 \\ (-0.752)\dot{\theta}\dot{\psi} - \frac{I_R}{6.45\times10^{-3}}\dot{\theta}\cdot g(u) + 65.116u_2 \\ (0.752)\dot{\phi}\dot{\psi} - \frac{I_R}{6.45\times10^{-3}}\dot{\phi}\cdot g(u) + 65.116u_3 \\ 37.168u_4 \end{pmatrix} \tag{4.33}$$

After substitution into equation 4.31, state space model shown as below:

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \\ x_{11} \\ x_{12} \end{pmatrix} = \begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \ddot{x} \\ \ddot{y} \\ \ddot{z} \\ \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \\ \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & a_0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & a_1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & a_2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & a_3 & 0 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \\ \phi \\ \theta \\ \psi \\ \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ B_{11} & 0 & 0 & 0 \\ B_{21} & 0 & 0 & 0 \\ B_{31} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & B_{42} & 0 & 0 \\ 0 & 0 & B_{53} & 0 \\ 0 & 0 & 0 & B_{64} \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{pmatrix}$$

(4.34)

Where

$$a_0 = g \qquad (4.35)$$

$$= 9.81$$

$$a_1 = \dot{\psi}\left(\frac{I_y - I_z}{I_x}\right) - \frac{I_R}{I_x} \cdot g(u) \qquad (4.36)$$

$$= -0.75184\psi - \frac{I_R}{(6.45036 \times 10^{-3})} \cdot g(u)$$

$$a_2 = \dot{\psi}\left(\frac{I_z - I_x}{I_y}\right) - \frac{I_R}{I_y} \cdot g(u) \qquad (4.37)$$

$$= 0.75184\dot{\psi} - \frac{I_R}{(6.45036 \times 10^{-3})} \cdot g(u)$$

$$a_3 = \dot{\theta}\left(\frac{I_x - I_y}{I_z}\right) \qquad (4.38)$$

$$= 0$$

$$\mathrm{B}_{11} = \frac{-(\cos\phi \sin\theta \cos\psi + \sin\phi \sin\psi)}{m} \qquad (4.39)$$

$$= \frac{-(\cos\phi \sin\theta \cos\psi + \sin\phi \sin\psi)}{0.816}$$

$$\mathrm{B}_{21} = \frac{-(\cos\phi\sin\theta\cos\psi + \sin\phi\cos\psi)}{m} \tag{4.40}$$

$$= \frac{-(\cos\phi\sin\theta\cos\psi + \sin\phi\cos\psi)}{0.816}$$

$$\mathrm{B}_{31} = (\frac{\cos\phi\cos\theta}{m}) \tag{4.41}$$

$$= (\frac{\cos\phi\cos\theta}{0.816})$$

$$B_{42} = \frac{L}{I_x} \tag{4.42}$$

$$= 65.11265$$

$$B_{53} = \frac{L}{I_y} \tag{4.43}$$

$$= 65.11265$$

$$B_{64} = \frac{L}{I_z} \tag{4.44}$$

$$= 37.16814$$

4.2 Physical Measurement and Testing

Table 4.1 shows the details of properties measurement of quadrotor testbed.

Table 4.1: Quadrotor properties measurement

No.	Description	Parameter	Unit	Value
1	Total mass of quadrotor	m	kg	0.816
2	Mass of rotor	$\overline{m}$	kg	0.055
3	Distance from rotor to center of body for quadrotor system	l	m	0.21
4	Length from rotor to opposite rotor	L	m	0.42

4.2.1 Speed test

The following table shows the speed test of each rotor and the result is plotted in graph with x-axis is the program speed versus the y-axis is the rotor speed in RPM.

a. Rotor 1

Table 4.2: Speed test result for rotor 1

Microcontroller		Speed		
Program speed	Speed percentage (%)	1st test (RPM)	2nd test (RPM)	Average (RPM)
0	0	0	0	0
10	0	0	0	0
20	0	0	0	0
30	0	0	0	0
40	0	0	0	0
50	21.48	1560	1500	1530
60	37.5	2860	2910	2885
70	43.75	3955	3810	3883
80	50	4751	4560	4655
90	56.25	5277	5100	5189
100	62.5	5598	5760	5679
110	68.75	6257	6330	6294
120	75	6822	6840	6831
130	100	7075	7170	7123
140	100	7073	7172	7123
150	100	7071	7174	7123
160	100	7075	7170	7123

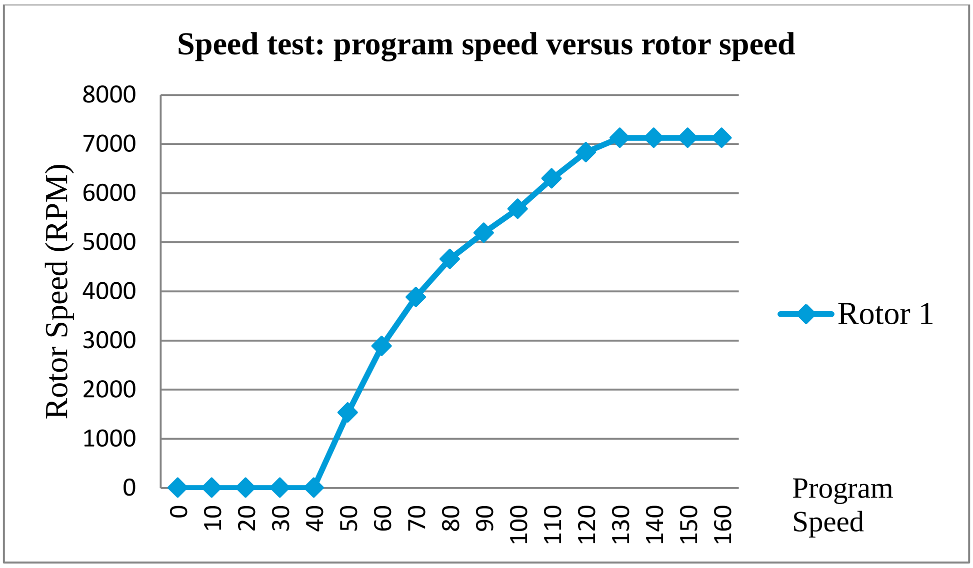

Figure 4.2: Speed test graph of rotor 1

b. Rotor 2

Table 4.3: Speed test result for rotor 2

Microcontroller		Speed		
Program speed	Speed percentage (%)	1st test (RPM)	2nd test (RPM)	Average (RPM)
0	0	0	0	0
10	0	0	0	0
20	0	0	0	0
30	0	0	0	0
40	0	0	0	0
50	20.13	1510	1321	1416
60	39.79	2849	2749	2799
70	53.47	3888	3634	3761
80	63.96	4605	4394	4499
90	71.64	5161	4917	5039
100	84.15	6188	5649	5919
110	90.99	6485	6316	6400
120	94.26	6803	6458	6630
130	100	7082	6987	7034
140	100	7081	6988	7034
150	100	7083	6986	7034
160	100	7081	6988	7034

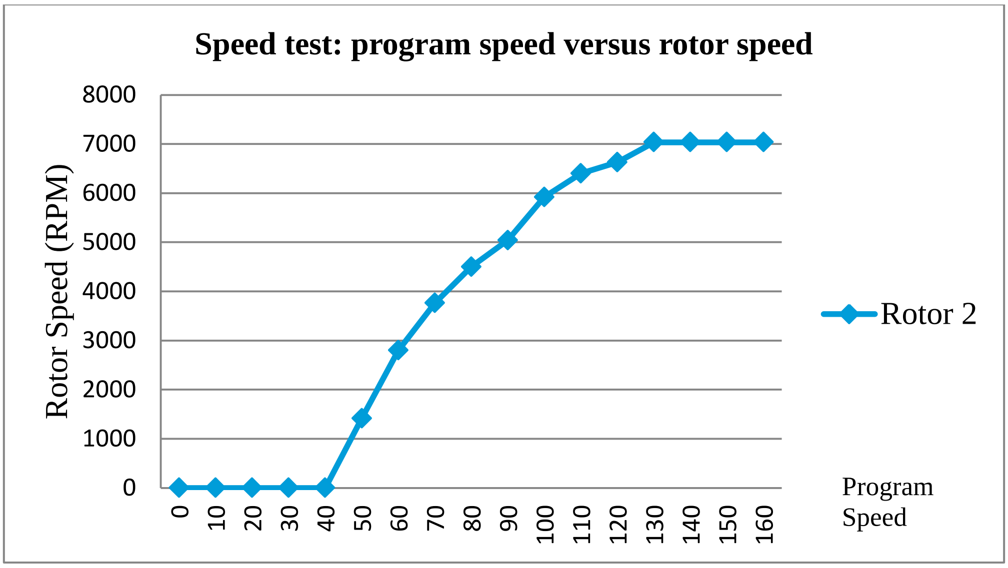

Figure 4.3: Speed test graph of rotor 2

c. Rotor 3

Table 4.4: Speed test Result for rotor 3

Microcontroller		Speed		
Program speed	Speed percentage (%)	1st test (RPM)	2nd test (RPM)	Average (RPM)
0	0	0	0	0
10	0	0	0	0
20	0	0	0	0
30	0	0	0	0
40	0	0	0	0
50	21.61	1562	1501	1532
60	40.53	2860	2885	2873
70	54.23	3883	3884	3884
80	65.36	4675	4650	4663
90	73.0	5163	5186	5174
100	80.12	5667	5690	5679
110	89.24	6330	6320	6325
120	95.78	6746	6830	6788
130	100	7057	7138	7088
140	100	7058	7137	7088
150	100	7057	7138	7088
160	100	7059	7136	7088

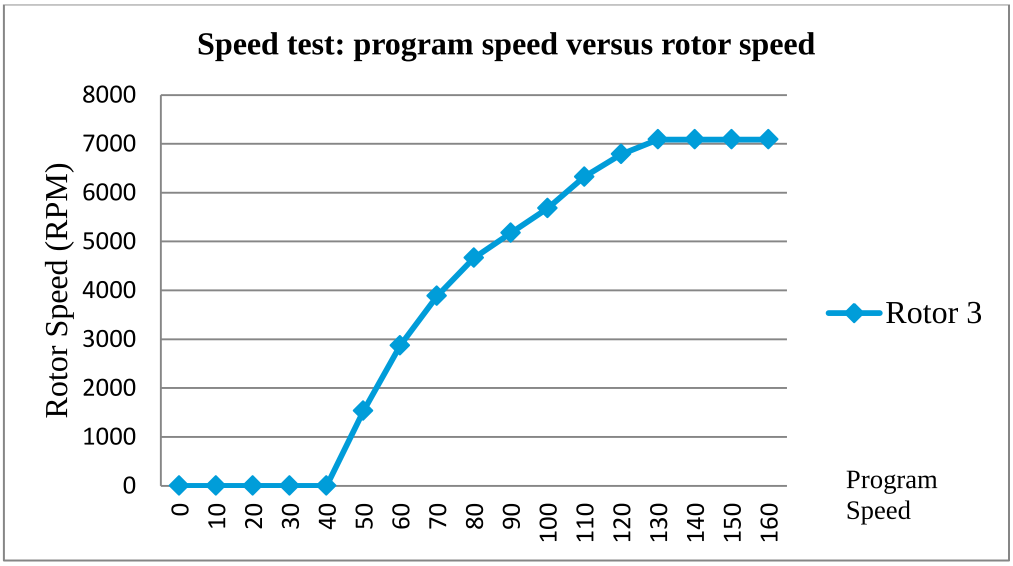

Figure 4.4: Speed test graph of rotor 3

d. Rotor 4

Table 4.5: Speed test Result for rotor 4

Microcontroller		Speed		
Program speed	Speed percentage (%)	1st test (RPM)	2nd test (RPM)	Average (RPM)
0	0	0	0	0
10	0	0	0	0
20	0	0	0	0
30	0	0	0	0
40	0	0	0	0
50	22.11	1500	1620	1560
60	40.7	2885	2860	2872
70	55.17	3886	3900	3893
80	64.83	4650	4500	4575
90	72.44	5188	5037	5112
100	80.37	5689	5655	5672
110	90.07	6288	6424	6356
120	96.22	6830	6749	6790
130	100	7118	6996	7057
140	100	7120	6994	7057
150	100	7117	6997	7057
160	100	7118	6996	7057

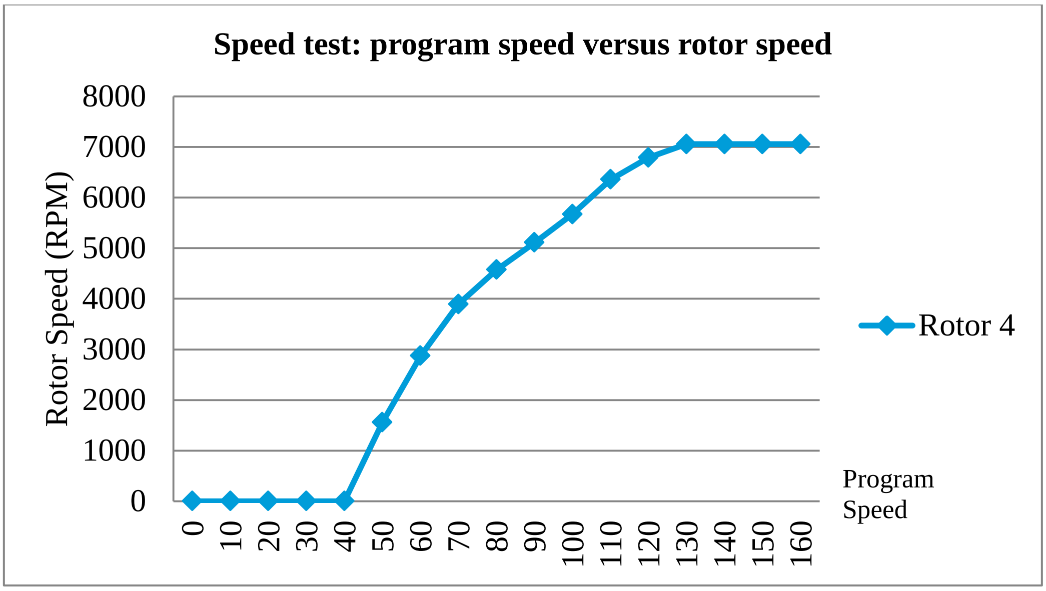

Figure 4.5: Speed test graph of rotor 4

From the results tabulated above, it can be conclude that all 4 rotors will only starts to rotate when the program speed is 50, which it generated in the range of 20% to 22% of total speed. The maximum program speed for the rotors is 130, where it generates 100% of total speed. As the program speed is increase, the total speed generated is the same therefore we can conclude that the program speed range is from 50 to 130.

4.2.2 Force-lift Test

Force lift test of all 4 rotors is carried out to obtain the balance measurements for the weight generated by propeller of different PWM speed.

$$\text{Force, } F = mg\ [kgms^{-2}] \qquad (4.45)$$

Where m is mass and g refer to gravity (9.81 ms^{-2}). Based on Newton's second law above, lift force generated by propeller is calculated. The coding to run the rotor attached in Appendix B. The tables below show the result for rotors. The speed are defined from minimum to maximum by value 50 is minimum speed and 130 is maximum speed.

a. Rotor 1

Table 4.6: Force-Lift Test Result for rotor 1

Microcontroller		Mass				Force = mg (N)
Program speed	Speed percentage (%)	1st test (g)	2nd test (g)	Average (g)	Kilogram (kg)	
0	0	0	0	0	0	0
10	0	0	0	0	0	0
20	0	0	0	0	0	0
30	0	0	0	0	0	0
40	0	0	0	0	0	0
50	21.48	26	25	25.5	0.0255	0.2502
60	37.5	114	116	115.0	0.1150	1.1282
70	43.75	218	210	214.0	0.2140	2.0993
80	50	315	303	309.0	0.3090	3.0313
90	56.25	417	403	410.0	0.4100	4.0221
100	62.5	483	497	490.0	0.4900	4.8069
110	68.75	596	603	599.5	0.5995	5.881
120	75	750	752	751.0	0.7510	7.3673
130	100	816	827	821.5	0.8215	8.0589
140	100	816	827	821.5	0.8215	8.0589
150	100	816	827	821.5	0.8215	8.0589
160	100	816	827	821.5	0.8215	8.0589

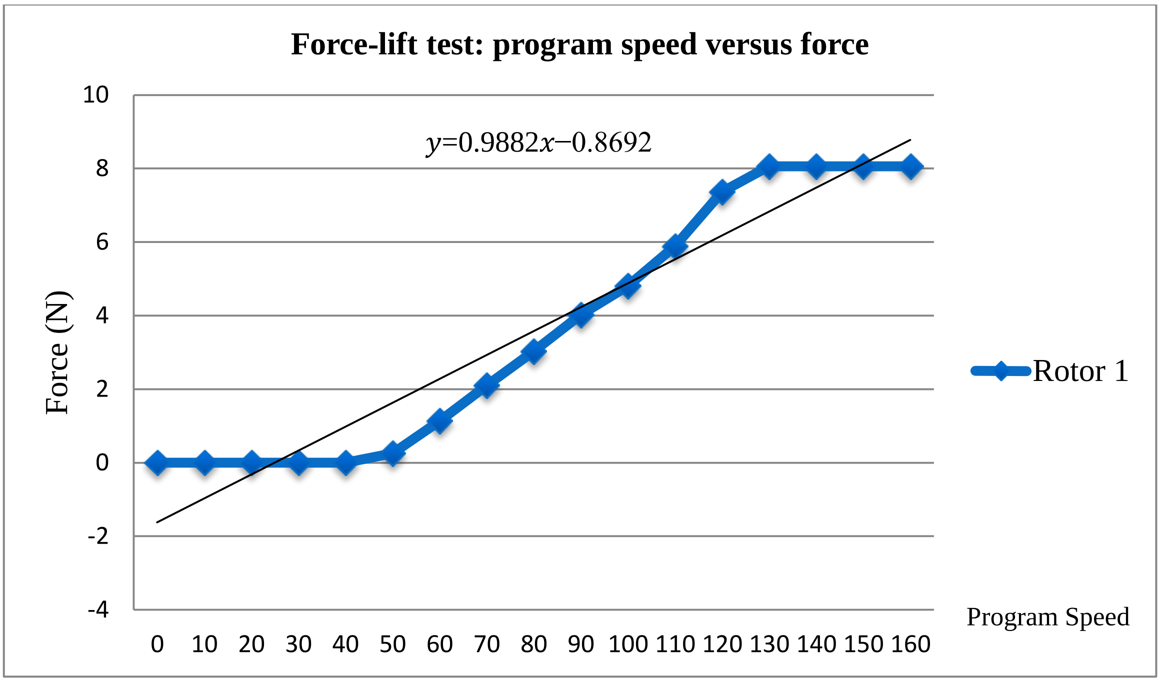

Figure 4.6: Force lift test graph of rotor 1

From the speed test result for rotor 1, the maximum program speed for rotor to generate total speed is 130, where the generated speed is 7123 RPM, equals to 746 rad/s. the force during maximum generated speed is 8.0589 N.

$$b = \frac{F}{\omega^2} = \frac{8.0589}{746^2} = 1.448 \times 10^{-5} \tag{4.46}$$

$$y = 0.9882x - 0.8692 \tag{4.47}$$

$$T_1 + b = 0.9882 \tag{4.48}$$

$$b = 1.448 \times 10^{-5} \tag{4.49}$$

$$T_1 = 0.988 \tag{4.50}$$

b. rotor 2

Table 4.7: Force-Lift Test Result for rotor 2

Microcontroller		Mass				Force = mg (N)
Program speed	Speed percentage (%)	1st test (g)	2nd test (g)	Average (g)	Kilogram (kg)	
0	0	0	0	0	0	0
10	0	0	0	0	0	0
20	0	0	0	0	0	0
30	0	0	0	0	0	0
40	0	0	0	0	0	0
50	20.13	25	22	23.5	0.0235	0.2305
60	39.79	114	110	112.0	0.1120	1.0987
70	53.47	214	200	207.0	0.2070	2.0307
80	63.96	306	292	299.0	0.2990	2.9332
90	71.64	407	387	397.0	0.3970	3.8946
100	84.15	489	447	468.0	0.4680	4.5912
110	90.99	602	545	573.5	0.5735	5.6260
120	94.26	748	710	729.0	0.7290	7.1515
130	100	817	807	812.0	0.8120	7.9657
140	100	817	807	812.0	0.8120	7.9657
150	100	817	807	812.0	0.8120	7.9657
160	100	817	807	812.0	0.8120	7.9657

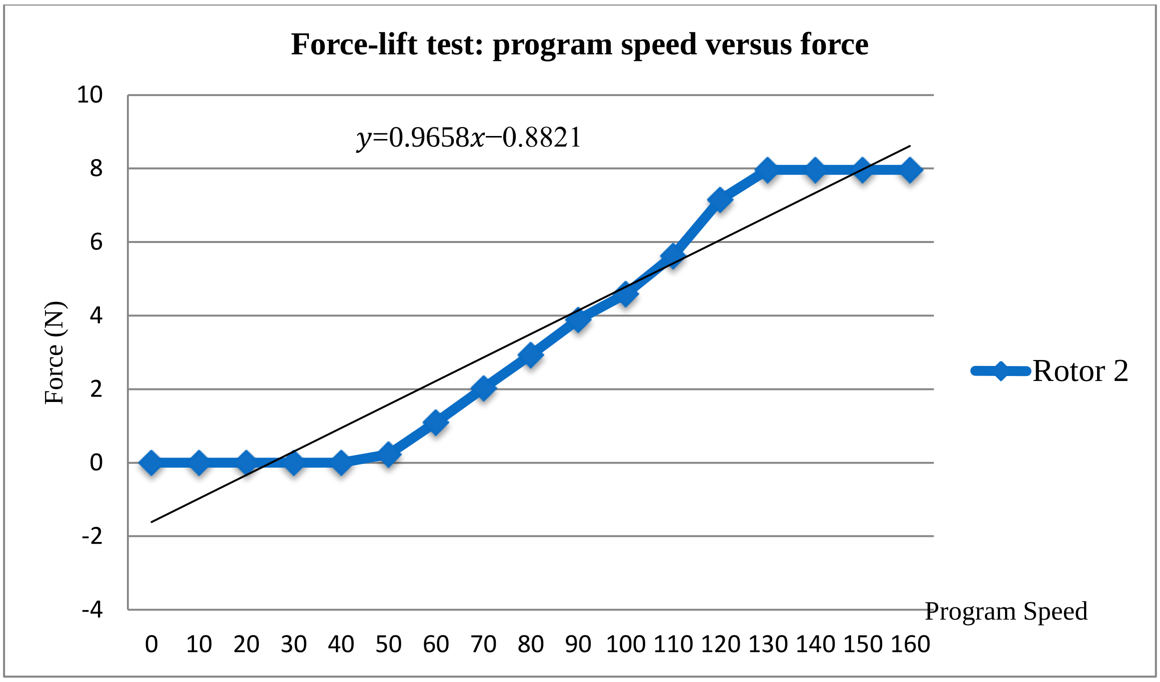

Figure 4.7: Force lift test graph of rotor 2.

From the speed test result for rotor 2, the maximum program speed for rotor to generate total speed is 130, where the generated speed is 7034 RPM, equals to 737rad/s. The force during maximum generated speed is 7.9657 N.

$$b = \frac{F}{\omega^2}$$

$$= \frac{7.9657}{737^2} \tag{4.51}$$

$$= 1.467 \times 10^{-5}$$

$$y = 0.9658x - 0.8821 \tag{4.52}$$

$$T_2 + b = 0.9658 \tag{4.53}$$

$$b = 1.467 \times 10^{-5} \tag{4.54}$$

$$T_2 = 0.968 \tag{4.55}$$

c. Rotor 3

Table 4.8: Force-Lift Test Result for rotor 3

Microcontroller		Mass				Force = mg (N)
Program speed	Speed percentage (%)	1st test (g)	2nd test (g)	Average (g)	Kilogram (kg)	
0	0	0	0	0	0	0
10	0	0	0	0	0	0
20	0	0	0	0	0	0
30	0	0	0	0	0	0
40	0	0	0	0	0	0
50	21.61	26	25	25.5	0.0255	0.2502
60	40.53	114	115	114.5	0.1145	1.1232
70	54.23	214	214	214.0	0.2140	2.0993
80	65.36	304	309	306.5	0.3065	3.0068
90	73.0	404	410	407.0	0.4070	3.9927
100	80.12	489	490	489.5	0.4895	4.8020
110	89.24	605	599	602.0	0.6020	5.9056
120	95.78	746	751	748.5	0.7485	7.3428
130	100	814	821	817.5	0.8175	8.0198
140	100	814	821	817.5	0.8175	8.0198
150	100	814	821	817.5	0.8175	8.0198
160	100	814	821	817.5	0.8175	8.0198

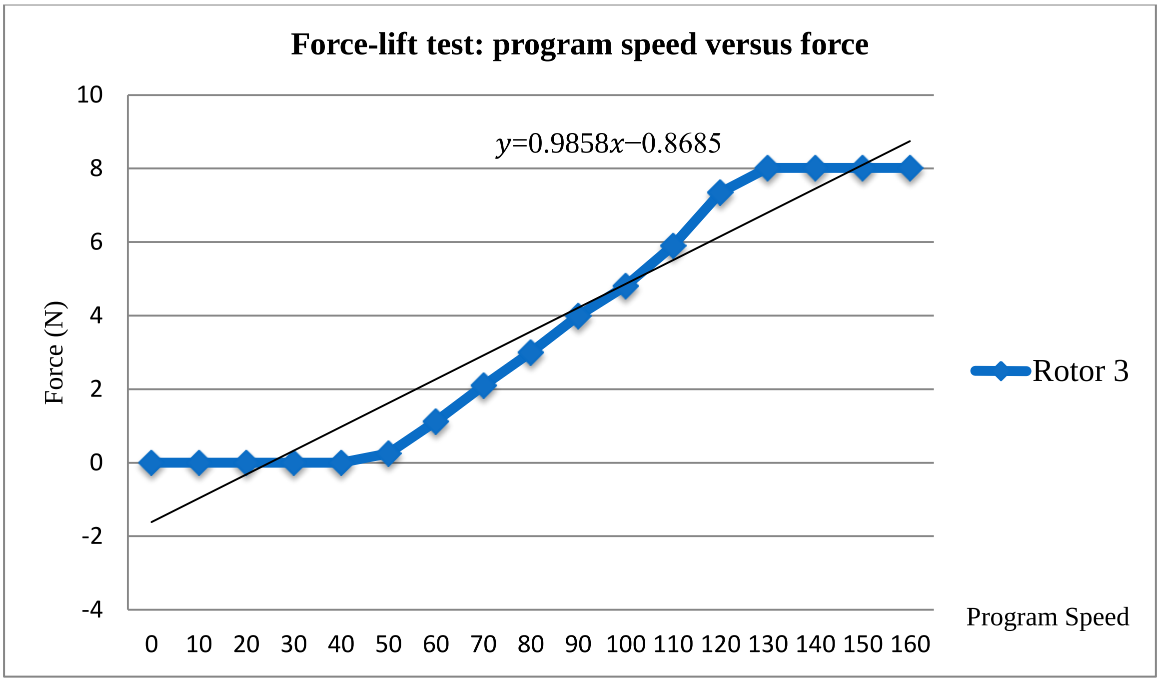

Figure 4.8: Force lift test graph of rotor 3.

From the speed test result for rotor 3, the maximum program speed for rotor to generate total speed is 130, where the generated speed is 7088 RPM, equals to 742rad/s. The force during maximum generated speed is 8.0198N.

$$b = \frac{F}{\omega^2}$$

$$= \frac{8.0198}{742^2} \tag{4.56}$$

$$= 1.457 \times 10^{-5}$$

$$y = 0.9858x - 0.8685 \tag{4.57}$$

$$T_3 + b = 0.9858 \tag{4.58}$$

$$b = 1.457 \times 10^{-5} \tag{4.59}$$

$$T_3 = 0.986 \tag{4.60}$$

d. Rotor 4

Table 4.9: Force-Lift Test Result for rotor 4

Microcontroller		Mass				Force = mg (N)
Program speed	Speed percentage (%)	1st test (g)	2nd test (g)	Average (g)	Kilogram (kg)	
0	0	0	0	0	0	0
10	0	0	0	0	0	0
20	0	0	0	0	0	0
30	0	0	0	0	0	0
40	0	0	0	0	0	0
50	22.11	25	27	26.0	0.0260	0.2551
60	40.7	115	114	114.5	0.1145	1.1232
70	55.17	214	215	214.5	0.2145	2.1042
80	64.83	309	299	304.0	0.3040	2.9822
90	72.44	410	398	404.0	0.4040	3.9632
100	80.37	490	488	489.0	0.4890	4.7971
110	90.07	599	612	605.5	0.6055	5.9400
120	96.22	751	742	746.5	0.7465	7.3232
130	100	821	807	814.0	0.8140	7.9853
140	100	821	807	814.0	0.8140	7.9853
150	100	821	807	814.0	0.8140	7.9853
160	100	821	807	814.0	0.8140	7.9853

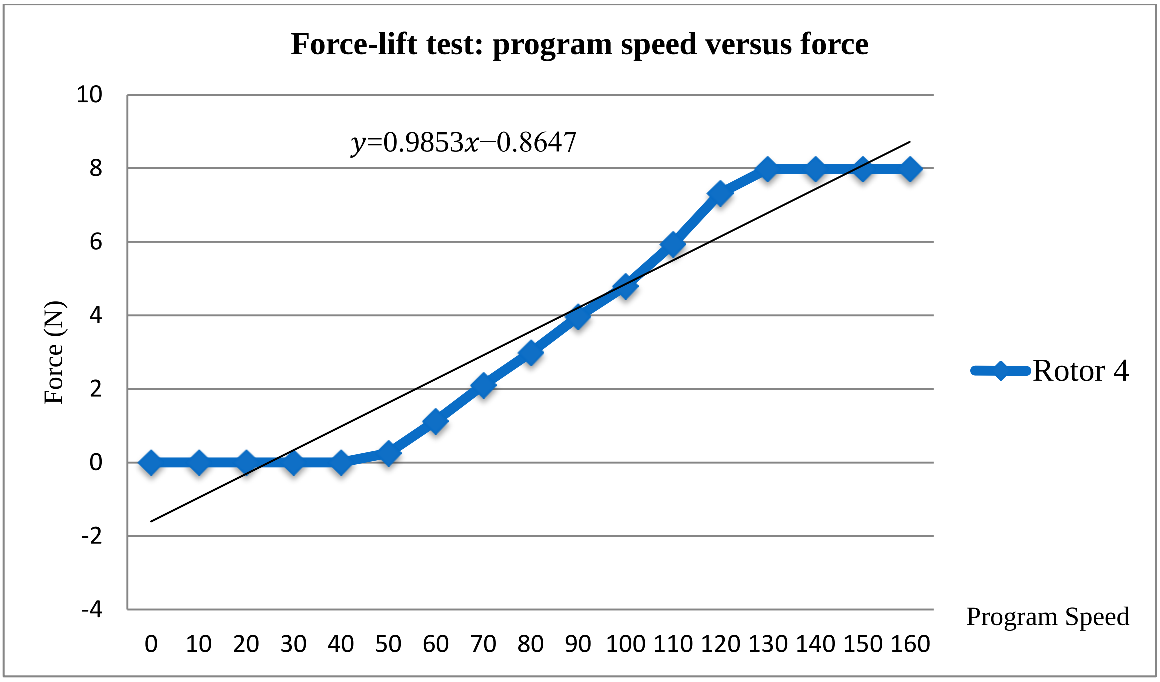

Figure 4.9: Force lift test graph of rotor 4.

From the speed test result for rotor 4, the maximum program speed for rotor to generate total speed is 130, where the generated speed is 7057RPM, equals to 739rad/s. The force during maximum generated speed is 7.9853N.

$$b = \frac{F}{\omega^2}$$

$$= \frac{7.9853}{739^2} \quad (4.61)$$

$$= 1.462 \times 10^{-5}$$

$$y = 0.9853x - 0.8647 \quad (4.62)$$

$$T_4 + b = 0.9853 \quad (4.63)$$

$$b = 1.462 \times 10^{-5} \quad (4.64)$$

$$T_4 = 0.985 \quad (4.65)$$

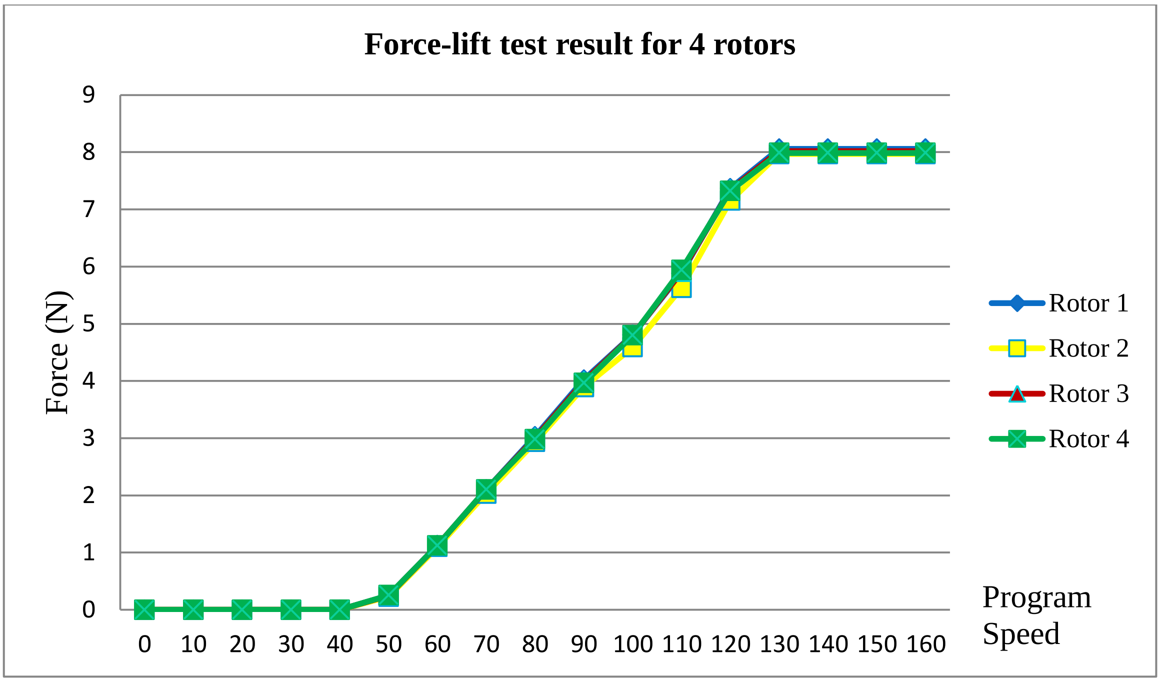

Figure 4.10: Force lift test graph of all rotors

The average value of thrust factor, b

$$= \frac{(1.448\times10^{-5} + 1.467\times10^{-5} + 1.457\times10^{-5} + 1.462\times10^{-5})}{4} \times 2\pi$$

$$= 9.164 \text{ x } 10^{-5} \; Ns^2 \qquad (4.66)$$

To calculate drag factor, d

$$d = bL \qquad (4.67)$$

$$= (9.164 \text{ x } 10^{-5})(0.23)$$

$$= 2.108 \times 10^{-5} \; Nms^2$$

Based on the momentum theory, the thrust force of propeller is proportional to square of rotational speed, therefore thrust factor; b of the maximum speed of each rotor is calculated with the formula below:

$$F = b\omega^2 \qquad (4.68)$$

$$b = \frac{F}{\omega^2} \qquad (4.69)$$

Where F is the force and ω is angular velocity.

The thrust factor of each rotor is the almost the same, this is due to the ESCs is calibrated.

Table 4.10: Quadrotor parameters

Parameter	Value	Unit	Remark
b	$9.164 \text{ x } 10^{-5}$	Ns^2	Thrust factor
d	2.108×10^{-5}	Nms^2	Drag factor
T_1	0.988	N	Thrust factor for rotor 1
T_2	0.968	N	Thrust factor for rotor 2
T_3	0.986	N	Thrust factor for rotor 3
T_4	0.985	N	Thrust factor for rotor 4

4.2.3 Bifilar Pendulum Test.

Table 4.11: Time taken from bifilar pendulum test for each movement

	First Test (s)	Second Test (s)	Third Test (s)	Average (s)
Roll	0.39	0.41	0.39	0.397
Pitch	0.39	0.41	0.39	0.397
Yaw	4.08	4.15	4.05	4.093

Calculation for moment of inertia for each motion:

$$J_{roll} = \left(\frac{T}{2\pi}\right)\frac{mgR^2}{L}$$

$$J_{roll} = \left(\frac{0.397}{2\pi}\right)\frac{0.816 \times 9.81 \times 0.155^2}{0.30} \tag{4.70}$$

$$J_{roll} = 0.0405\ kgm^2s^{-1}$$

$$J_{pitch} = \left(\frac{T}{2\pi}\right)\frac{mgR^2}{L}$$

$$J_{pitch} = \left(\frac{0.397}{2\pi}\right)\frac{0.816 \times 9.81 \times 0.155^2}{0.30} \tag{4.71}$$

$$J_{pitch} = 0.0405\ kgm^2s^{-1}$$

$$J_{yaw} = \left(\frac{T}{2\pi}\right)\frac{mgR^2}{L}$$

$$J_{yaw} = \left(\frac{4.093}{2\pi}\right)\frac{0.816 \times 9.81 \times 0.155^2}{0.30} \tag{4.72}$$

$$J_{yaw} = 0.4176\ kgm^2s^{-1}$$

Graphical result for rolling, pitching and yawing movement:

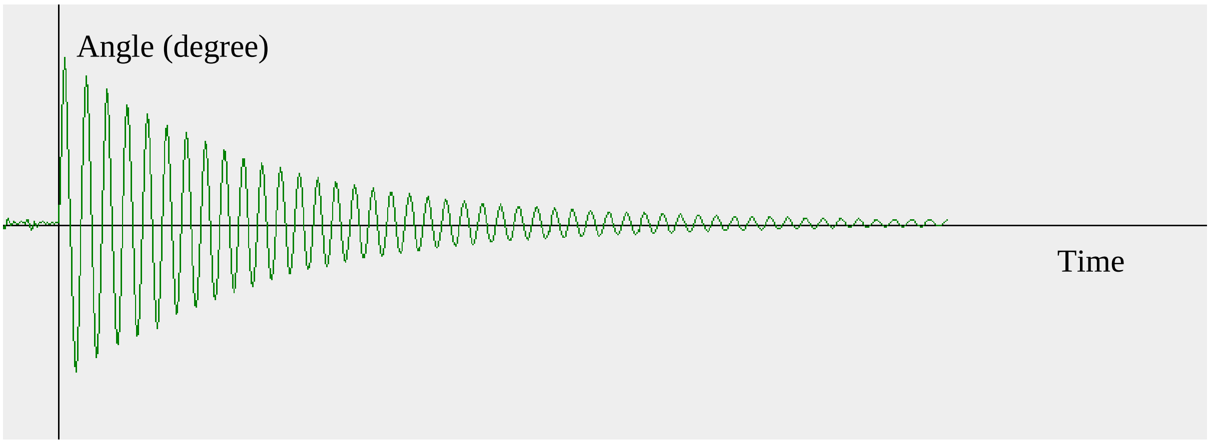

Figure 4.11: Rolling and pitching motion graph extracted from serial chart

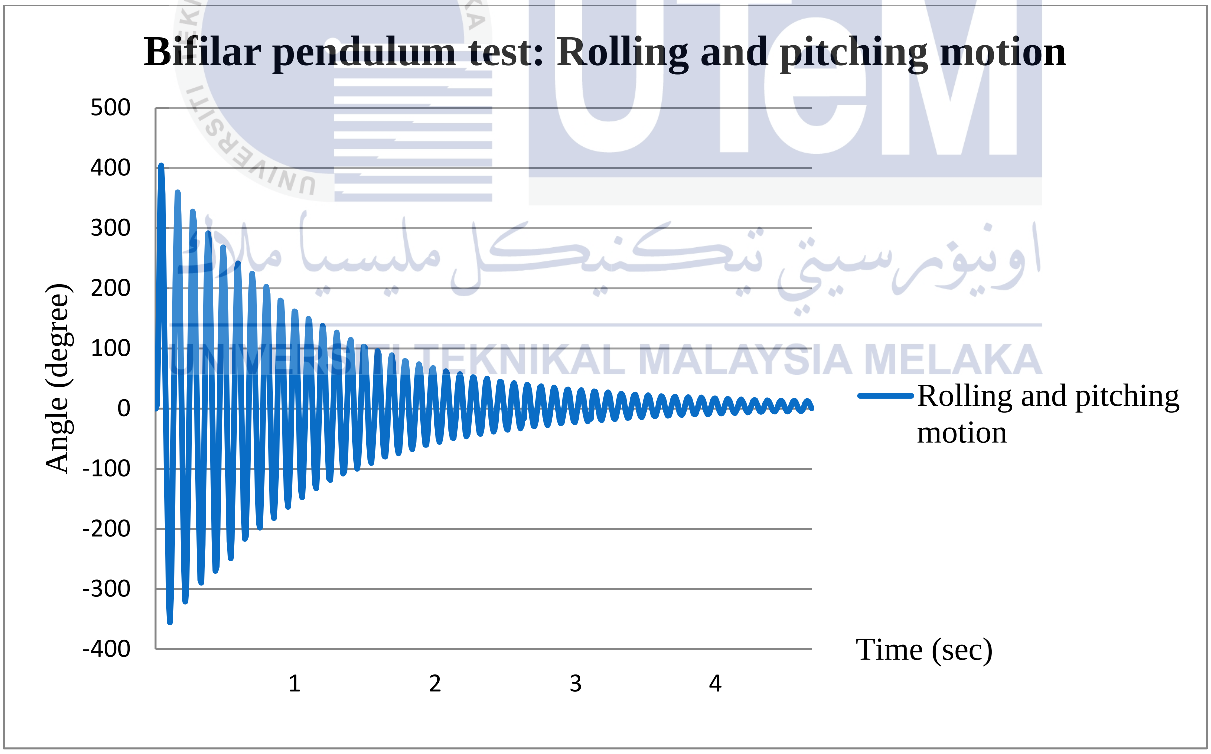

Figure 4.12: Pitching and rolling motion graph of quadrotor bifilar pendulum test

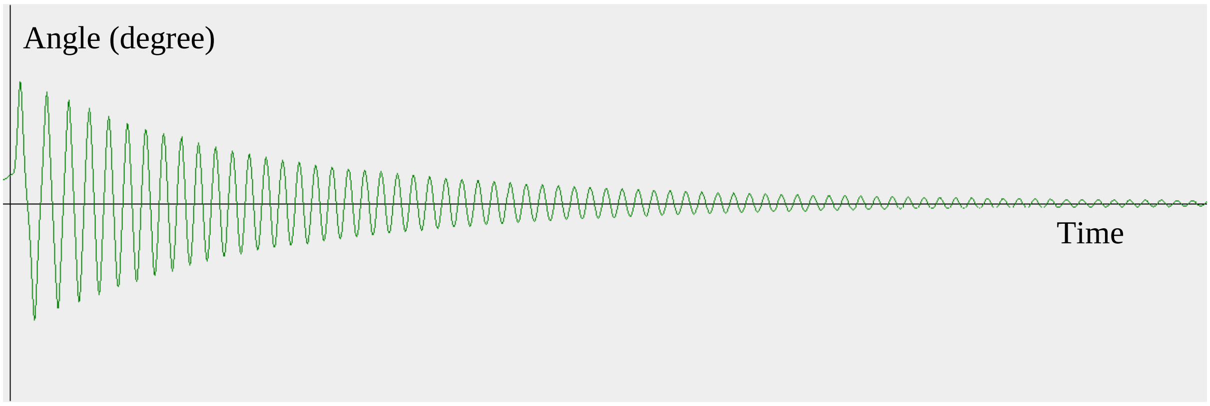

Figure 4.13: Yawing motion graph extracted from serial chart

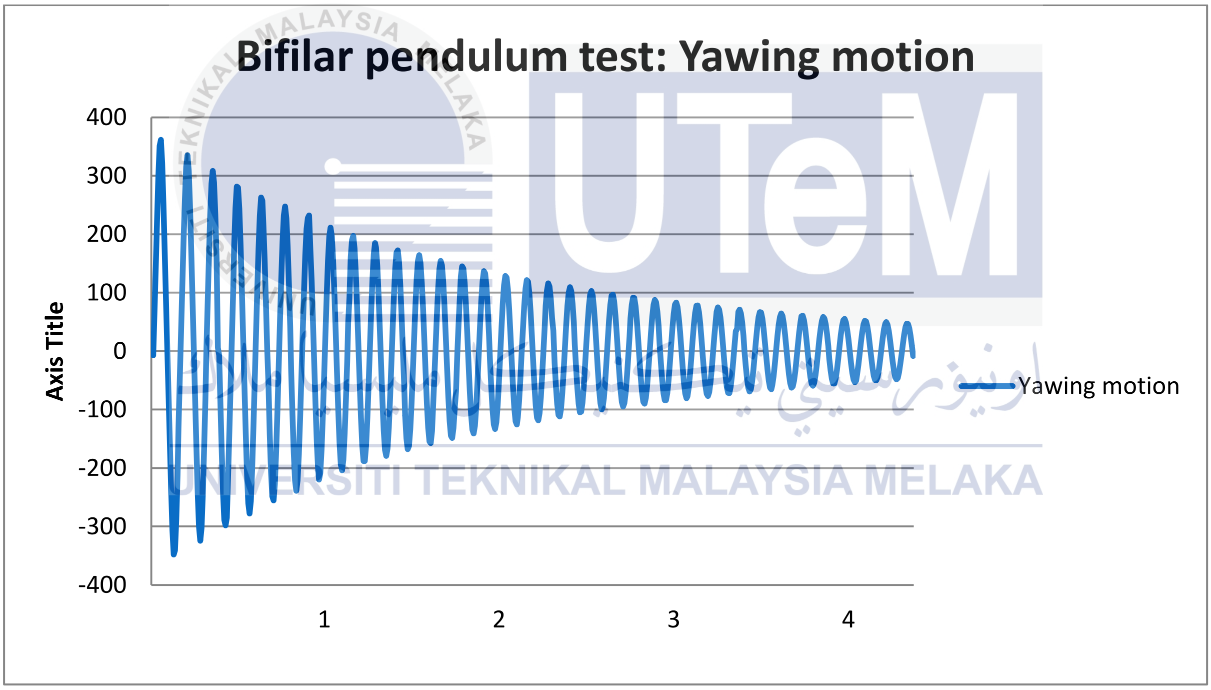

Figure 4.14: Yawing motion graph of quadrotor bifilar pendulum test

Table 4.12: Quadrotor bifilar pendulum test result

Parameter	Value	Unit	Remark
m	0.816	kg	Mass of quadrotor
R	0.155	m	Radius of the centre quadrotor
L	0.42	m	Length from rotor to rotor
l	0.200	m	Length from rotor to centre
$\overline{m}$	0.052	kg	Mass of single rotor
$I_{x\ cal}$	0.0645	Nm	Mass moment of inertia along x-axis (result from section 4.1.5)
$I_{y\ cal}$	0.0645	Nm	Mass moment of inertia along y-axis (result from section 4.1.5)
$I_{z\ cal}$	0.0113	Nm	Mass moment of inertia along z-axis (result from section 4.1.5)
$I_{x\ exp}$	0.0405	Nm	Mass moment of inertia along x-axis
$I_{y\ exp}$	0.0405	Nm	Mass moment of inertia along y-axis
$I_{z\ exp}$	0.4176	Nm	Mass moment of inertia along z-axis

The experimented value of the mass moment of inertia along x, y and z axes have differences compare to the value obtain from calculation in section 4.1.5. During the experiment, there are air disturbances that cause air friction effect to the propellers. Thus it will not give the ideal value as calculation from equation.

4.3 Complementary filter

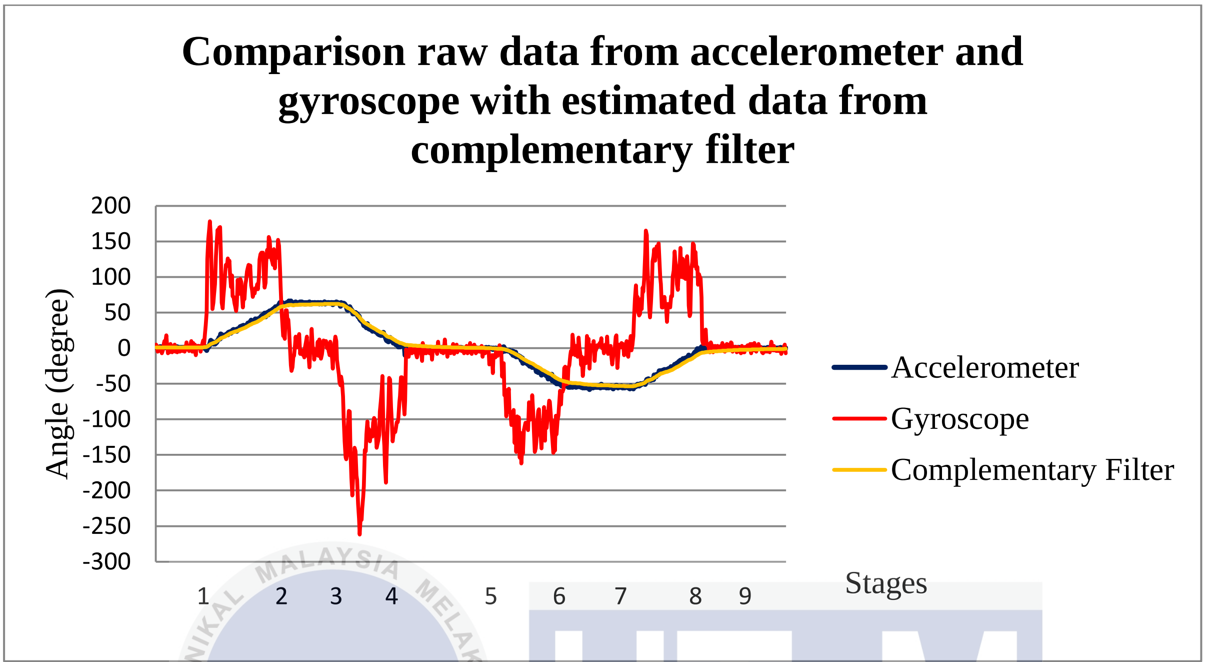

Figure 4.15: Comparison result of raw data from accelerometer and gyroscope with estimated data from complementary filter

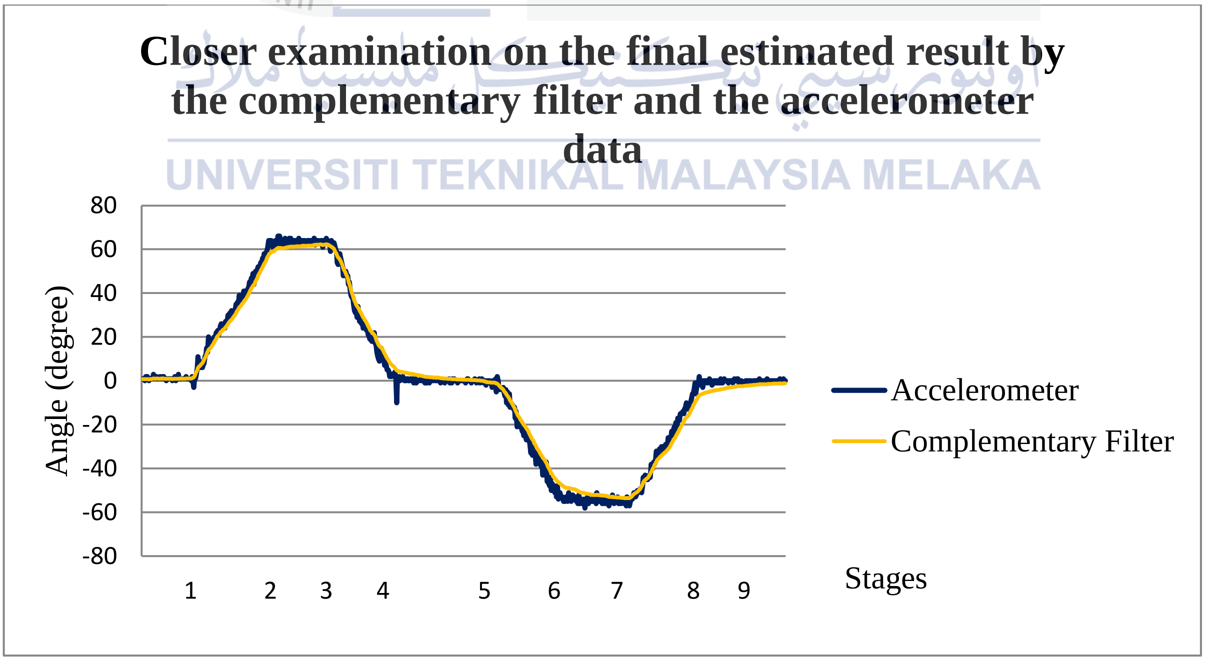

Figure 4.16: Closer examination on the final estimated result by the complementary filter and the accelerometer data

Figure 4.15 shows the combination of all data in a graph to study the pattern difference of the raw data and estimated data. It can be noticed about the data of accelerometer and complementary filter is overlapping. Therefore figure 4.16 give a closer examination on the final estimated result by the complementary filter and the accelerometer data. It shows that the complementary filter data have more smooth result compare to the accelerometer data.

Table 4.13: Summary explanation of the data read by IMU sensor

Stage	Position of IMU Sensor	Blue – accelerometer data	Red – gyroscope data	Orange – estimated data
1	Stationary position	Zero	Zero	Zero
2	Flip right (+90degree)	Positive(increase)	Positive (increase)	Positive (increase)
3	Stop	Positive (remain)	Decrease to zero	Positive(remain)
4	Flip left(back to stationary position)	Positive (decrease to zero)	Negative (decrease)	Positive (decrease to zero)
5	Stop	Zero	Increase to zero	Zero
6	Flip left (-90degree)	Negative (decrease)	Negative (decrease)	Negative (decrease)
7	Stop	Negative (remain)	Increase to zero	Negative (remain)
8	Flip right (back to stationary position)	Negative (increase to zero)	Positive (increase)	Negative (increase to zero)
9	Stop	Zero	Decrease to zero	Zero

Table 4.13 shows the explanation of the data recorded by the IMU sensor. The data generated are due to several movement and orientation detected by sensor itself. The IMU sensor is performing rolling phenomenon. At stage 1, it is at stationary position, accelerometer and gyroscope gives the zero reading, while complementary filter shows result at zero as well. At stage 2, from stationary position, IMU sensor is flip to the right, which is the turn to positive 90degree along x axis, data for accelerometer and gyro are positively increase. Complementary filter’s result is positively increased as well; the vibration and disturbance data is eliminating out. In stage 3, the there is no movement detected by sensor because it is stop and remain at the current position, the gyroscope reading starts to decrease

to zero while accelerometer and complementary filter reading is remain at the same value. For the following stages is explained in table 4.14.

The fluctuation from the result shown is cause by the high sensitivity of gyroscope, which is 2000deg/sec. even a slight movement it will cause the reading far different from others. The complementary filter is filtering out those unwanted result which consider as the disturbance and create a smooth and more reliable result. It also can be concluded that each movement can be identified clearly by IMU sensor.

4.4 Quadrotor Mathematical Modeling Validation

Open-loop test is carried out on quadrotor testbed and a set of reference data is obtained. On the other hand, block diagram of the system is designed using Matlab Simulink and simulation test is ran. The data from real time implementation of quadrotor testbed and result from simulation test is analyzed.

4.4.1 Simulation Result

The result of mathematical modeling in section 4.1 is used to represent the quadrotor in simulation. The model is built and simulate under open loop condition, with the irregularities input block to provide the input value and the output value is monitored using scope block. Figure 4.17 shows the construction of the Simulink diagram of the quadrotor system in open loop condition. For figure 4.18, the mathematical modeling block is built with all six differential equations is inserted into the function block, the equation are $\ddot{x},\ddot{y},\ddot{z},\ddot{\phi},\ddot{\theta},\ddot{\psi}$ which represent the translational and rotational movement of quadrotor.

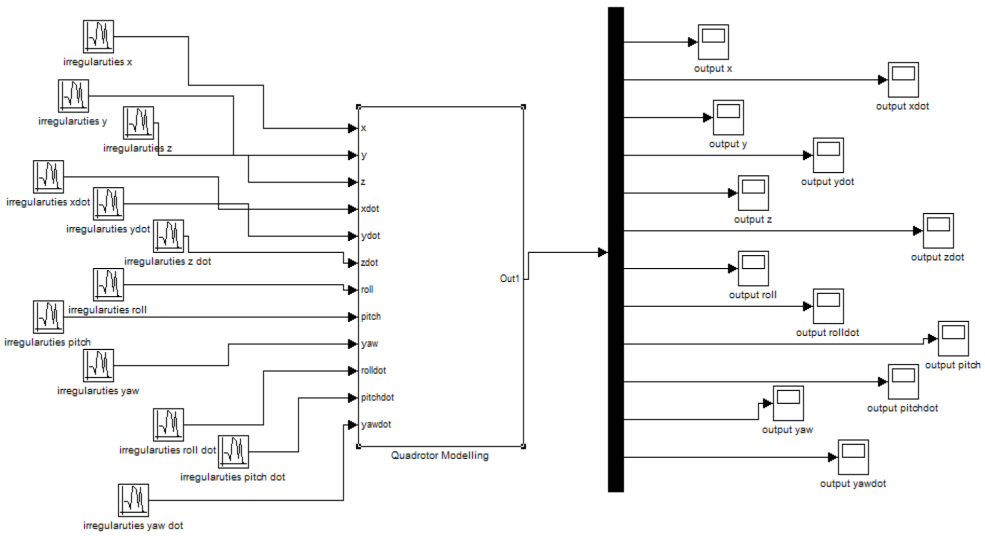

Figure 4.17: Simulink diagram of quadrotor system

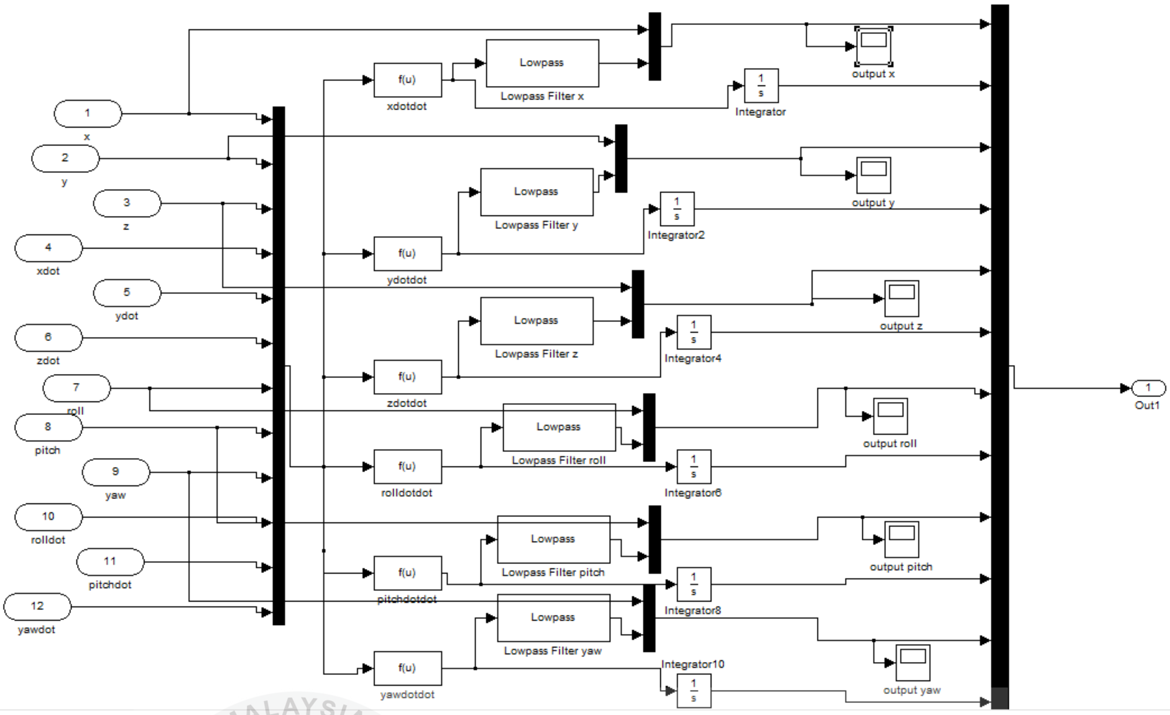

Figure 4.18: Subsystem of quadrotor system's modeling

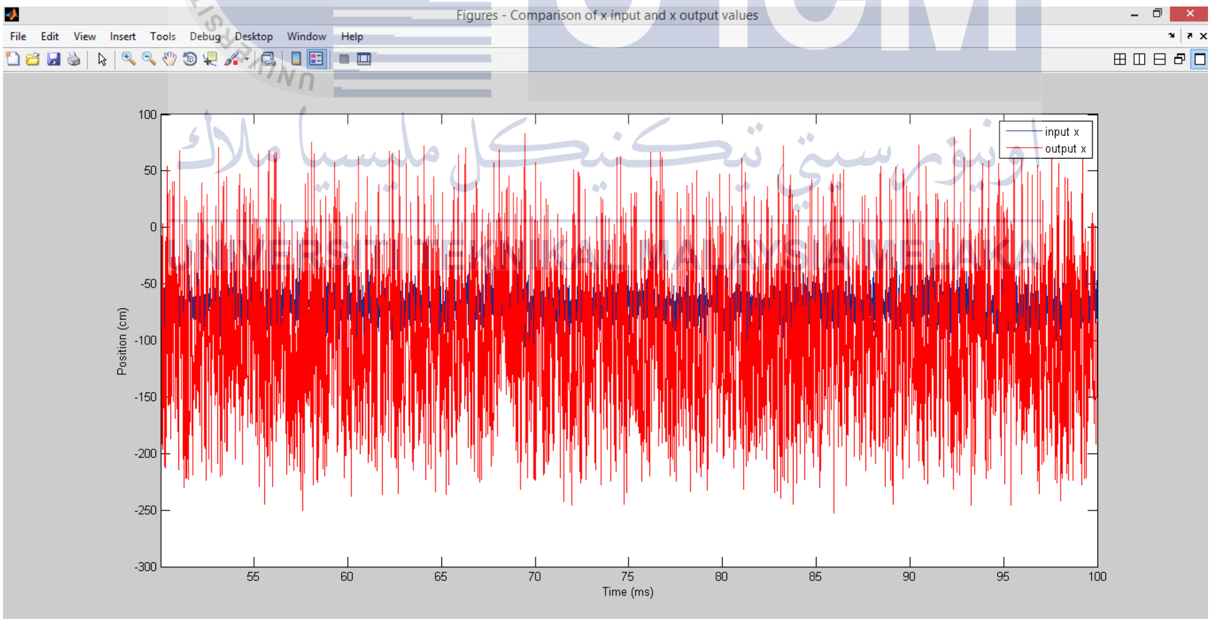

Figure 4.19: Simulation result of open loop test for x-axis

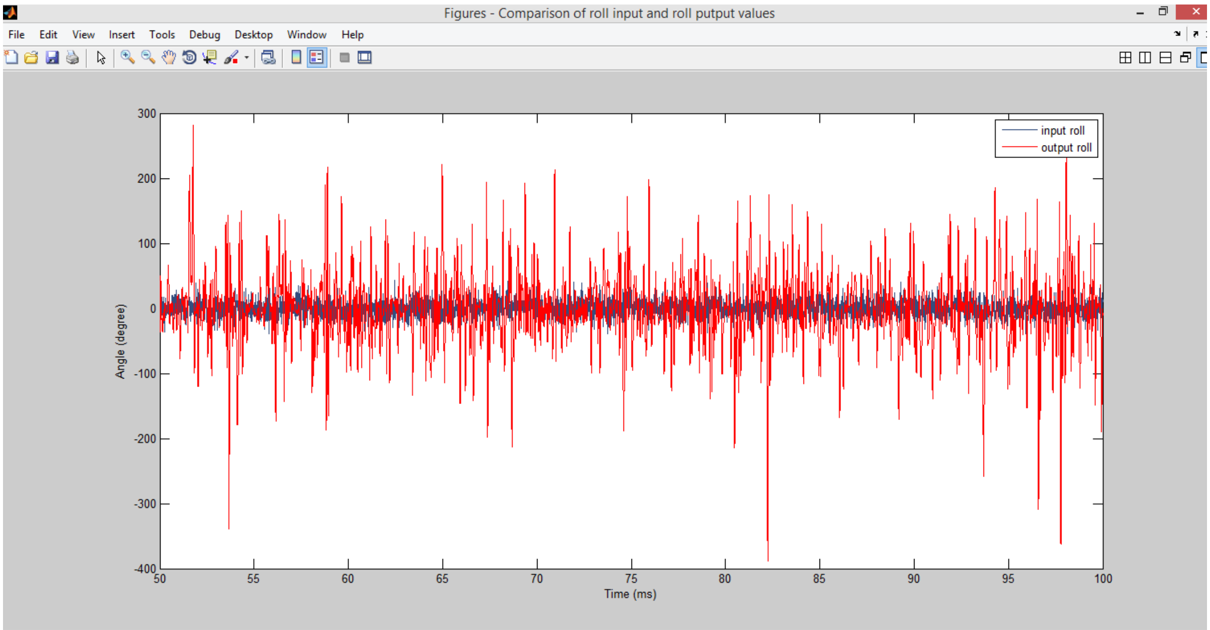

Figure 4.20: Simulation result of open loop test for roll movement

Figure 4.19 and figure 4.20 shows the simulation result of open loop test for x axis and roll movement. As mentioned before, to imitate the real time input for quadrotor, the input reference in this simulation is using irregularities block. Based on the output generated, it shows the rapid oscillation within the greater range compared to the input. This is due to the sustem built is open loop and no controller is involved, therefore the output range is uncontrollable. It can be concluded that the mathematical modeling derived is correct as this is an open loop system which means the input is highly non-linear, the output result shows non-linear as well where the pattern is similar to the irregular input to system. Kindly refer to Appendix D for overall results of y-axis, z-axis, pitch and yaw movement.

4.4.2 Real time Implementation Result

The open loop test on quadrotor system is carried out with 3 different program speeds which is 60 (low), 100 (medium) and 130 (high). This is to observe the difference of quadrotor's response effects the position and angle of rotation when the speed is low or high. Although during this open loop test the input is command from Arduino board, there are external disturbance such as drag factor and wind disturbance that will cause the output result become highly non-linear.

a. Result of open loop test for x-axis at different speed

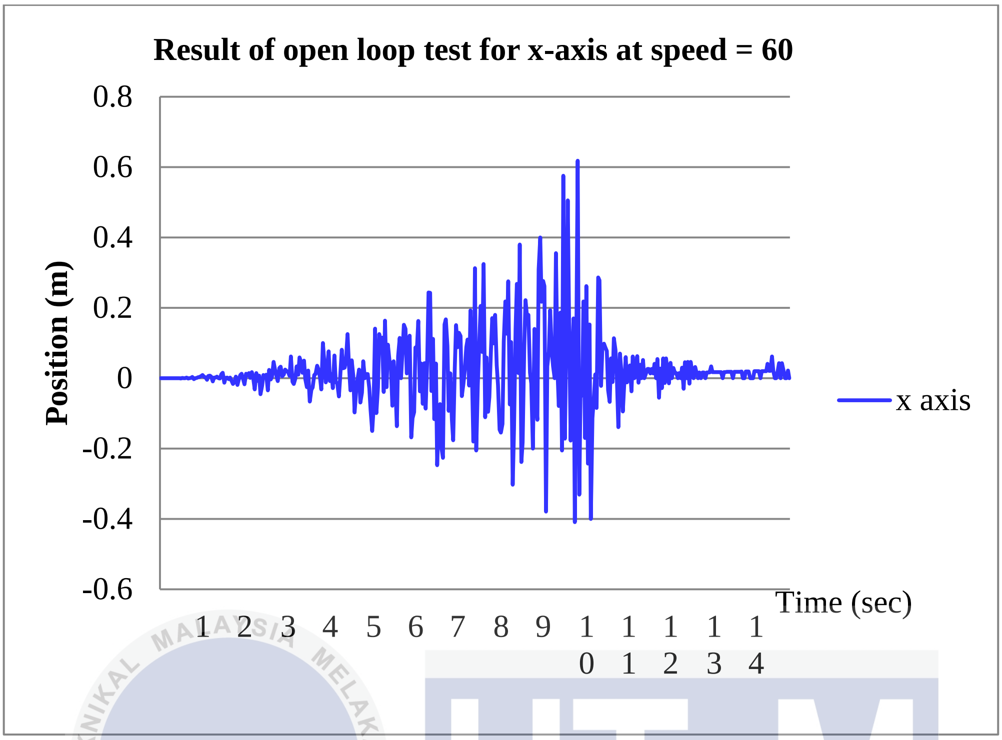

Figure 4.21: Result of open loop test for x-axis at speed = 60

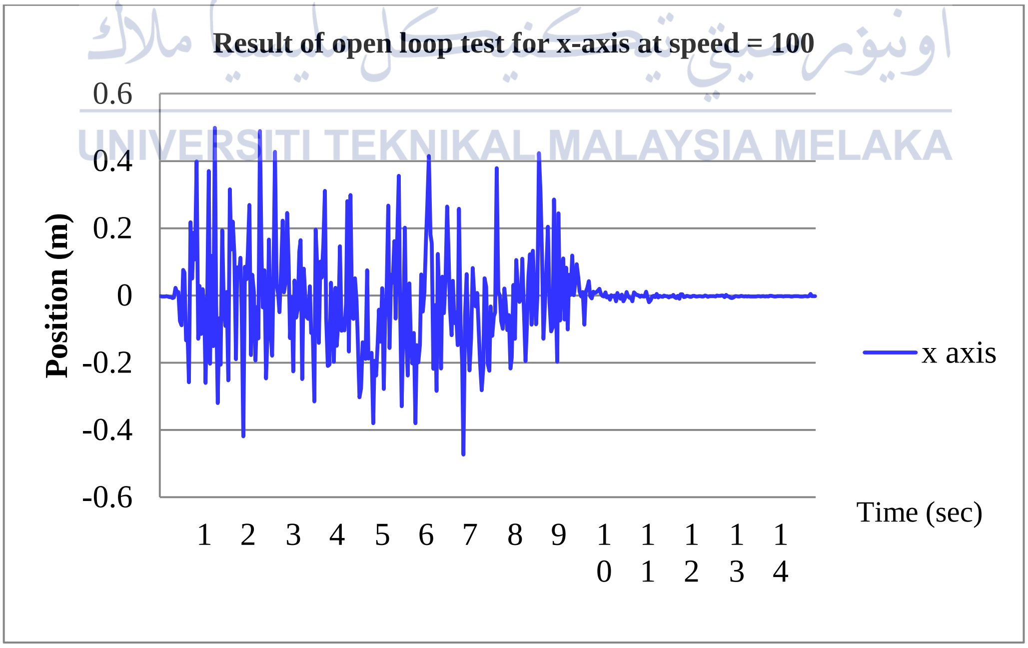

Figure 4.22: Result of open loop test for x-axis at speed = 100

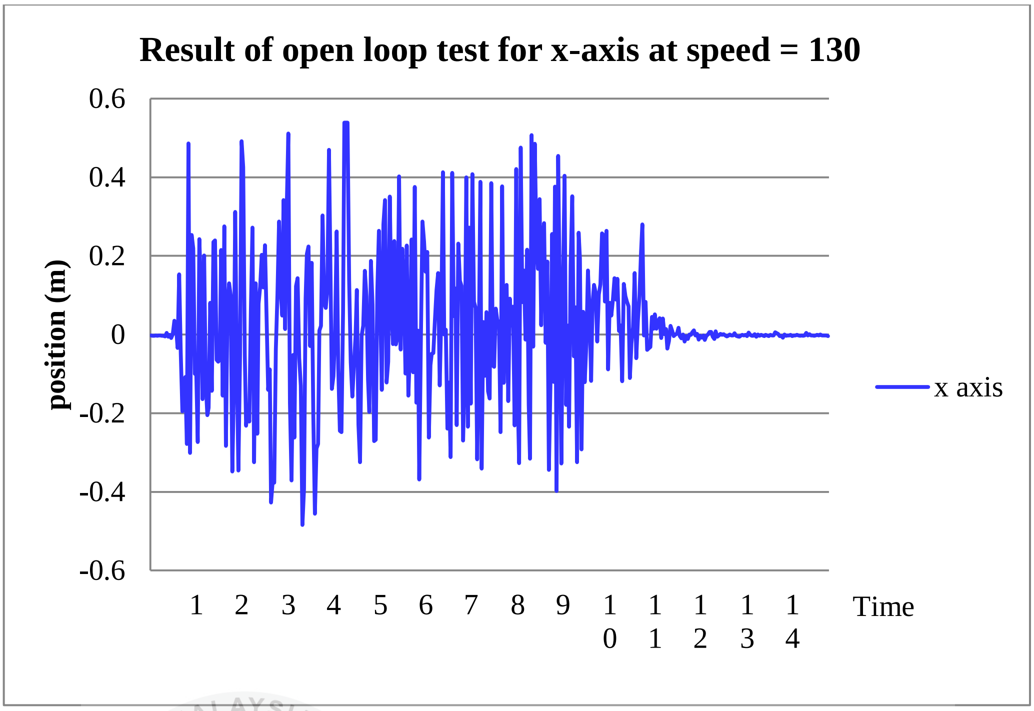

Figure 4.23: Result of open loop test for x-axis at speed = 130

b. Result of open loop test for roll movement at different speed

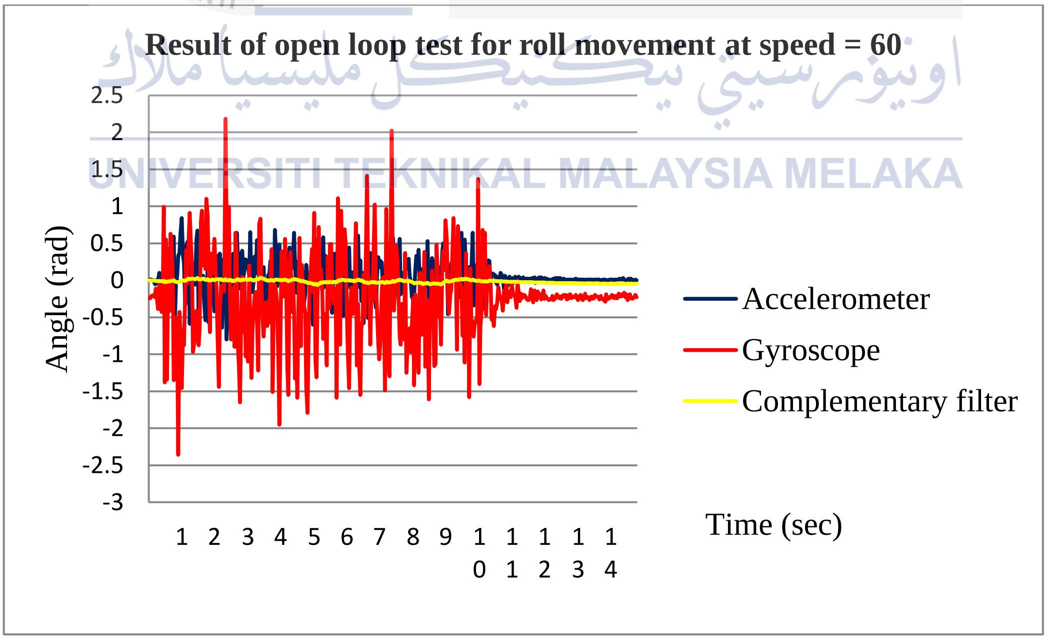

Figure 4.24: Result of open loop test for roll movement at speed = 60

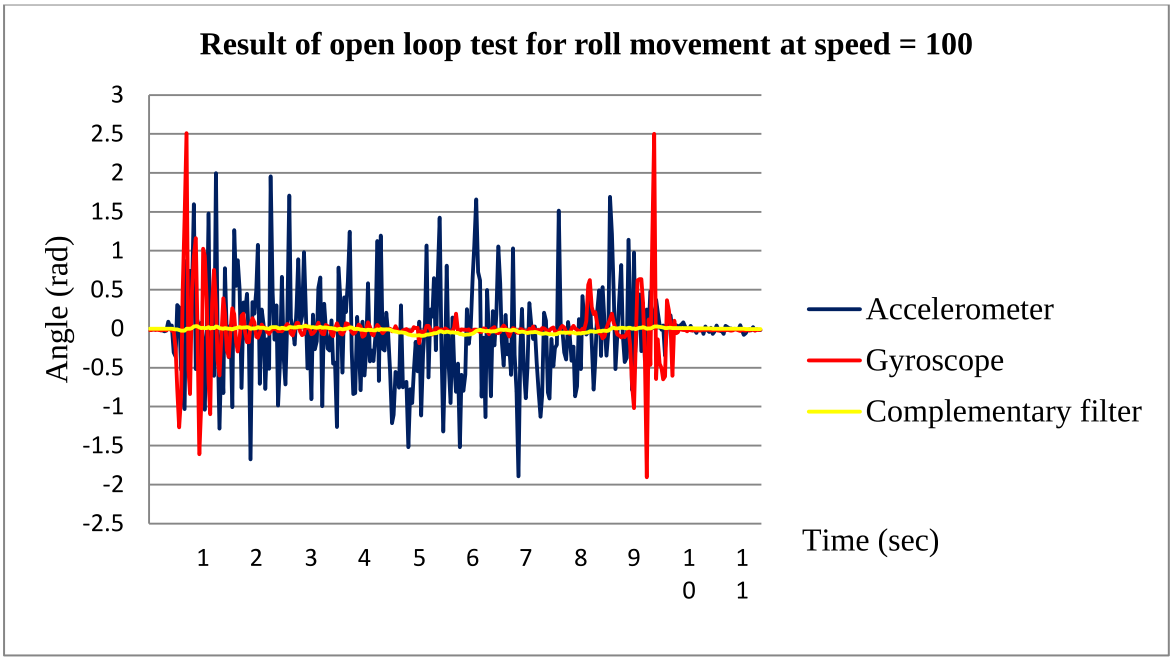

Figure 4.25: Result of open loop test for roll movement at speed = 100

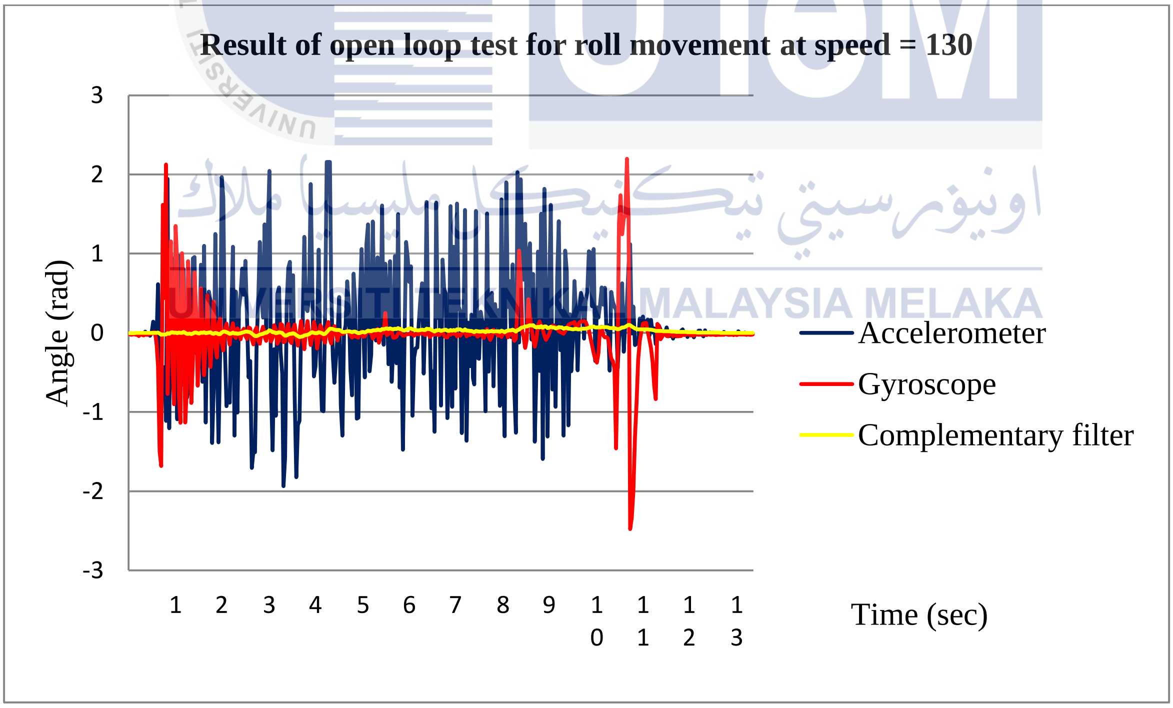

Figure 4.26: Result of open loop test for roll movement at speed = 130

Figure 4.21 to figure 4.23 shows the result of open loop test for x-axis, while figure 4.24 to figure 4.26 shows the result if open loop test for roll movement. The outputs are oscillated rapidly throughout the experiment. This is because there is no controller in the quadrotor system to manipulate the output. For the x-axis and roll movement output result, the range value is similar for all three speeds. This is because during the open loop test; a precaution step is taken by tied all 4 edges of the quadrotor testbed. The movement of the testbed is limited by the length of the rope. However it is clearly showed in the results that the time taken for the quadrotor to reach the maximum x value is vary to the program speed. For low speed, it reaches at 10s. For medium and high speed, it reach maximum x value at early stage of the test. When comparing the x value result of medium speed and high speed, it is noticeable that at high speed quadrotor testbed is reach maximum x value throughout the experiment. Overall the pattern of these output data is identical to each other where it is meet the characteristic of open loop system. For the result of y-axis, z-axis, pitch movement and yaw movement at different speeds, which have similar data pattern as x-axis and roll movement, kindly refer to Appendix E.

4.4.3 Comparison on signals generated

The characteristic of time response for simulation test and real time open loop test is different and incomparable. During the open loop test using quadrotor testbed, the system is started by the command from Arduino board, depends on the program speed that is given to it. However the input will not remain the same all the time due to the external disturbance that would effect on the testbed. Besides that, this type of input is difficult to model, therefore for the simulation test, the irregularities block is used to imitate the unpredictable external disturbance condition. This is also the reason for the different value range of output value from both tests.

The simulation results for translational movement of quadrotor system yield satisfactory signals. This can be observed in figure 4.19, which illustrates the output signal of x-axis. The patterns of signal are alike for y-axis and z-axis which the graph is in Appendix D. Similarly, the open loop test yield same pattern signals for x axis at different speed which is 60, 100 and 130. The results are illustrated in figure 4.21 to figure 4.23. Besides, the output result for x, y and z axes at same speed also have a same output shape form.

For the rotational movement, result of roll from simulation test and real time open loop test is analyzed. Both results show the same pattern, regardless different speed in open loop test. The roll movement result for simulation is illustrated in figure 4.20, while figure 4.24 to figure 4.26 shows real time open loop test result of roll movement at different speeds.

It can be concluded that the pattern of the result shape obtained from simulation test and the real time implementation is the same which is in highly nonlinear form. This meets the characteristic of the tests because these tests are performed on a quadrotor system in open loop condition; therefore the outcome is unpredictable and non-linear.

CHAPTER 5

CONCLUSION AND RECOMMENDATION

5.1 Conclusion

As a conclusion, the complementary filter that applied for system estimation in quadrotor system is useful to obtain more reliable result because the gyroscopic drift from gyroscope and the external disturbance value read by accelerometer is eliminated. The modeling of quadrotor system derived using Newton-Euler method is proven identical with the quadrotor testbed as the simulation test result pattern is similar to real time implementation result. These research outcomes can be used for further development of the controller of UAV quadrotor system as it is a highly nonlinear system and multivariable with six degree of freedom which is $x, y, z, \phi, \theta, \psi$ but only four inputs from rotors.

5.2 Recommendation

Since the quadrotor system is nonlinear, it is recommended to include controller such as PID, backstepping PID or fuzzy logic in the system to improve and control the system performance. Besides that, by improvising the sensory system that is currently available, the performance of the system will definitely improve. Ultrasonic sensor can be used in the system to monitor the height of the quadrotor, when the quadrotor system is able to VTOL and hovering. Barometer in IMU sensor can be used to measure the atmospheric pressure when it is hovering. Laser range finder can be used to measure the angle and range of the obstacle object during VTOL and hovering.

REFERENCE

[1] Carlos Espadas & Michael Thoss, Geometric Professionals Photogrammetry and Remote Sensing. *Case Study: Microdrones in Geomatics – Remote Sensing*, [online]. Available at http://microdrones.com/references/case-study/geomatic-professionals-photogrammetry-and-remote-sensing-with-microdrones-md4-1000.pdf,2012 [Accessed 1 October 2013]

[2] Sqn Ldr Rajesh Kumar, *Tactical Reconnaissance: UAVS Versus Manned Aircraft*,[online]. Available at http://www.fas.org/irp/program/collect/docs/97-0349.pdf [Accessed 11 October 2013]

[3] Professor J.Gordon Leishman, *The Breguet-Richet Quad-Rotor Helicopter of 1907*, [online]. Available at http://aero.umd.edu/~leishman/Aero/Breguet.pdf [Accessed 12 October 2013]

[4] Mark Dupuis, Jonathan Gibbons, Maximillian Hobson-Dupont, Alex Knight, Artem Lepilov, Michael Monfreda, and George Mungai, Design Optimization of a Quad-Rotor Capable of Autonomous Flight, M.S. thesis, Worcester Polytechnic Institue, United States, 2008.

[5] Josh Villbrandt, *The Quadrotor's Coming of Age*, [online]. Available at http://illumin.usc.edu/162/the-quadrotors-coming-of-age/ [Accessed 12October 2013]

[6] Bouabdallah, S., Murrieri, P., & Siegwart, R., Design and control of an indoor micro quadrotor, *IEEE transections on Robotics and Automation*, vol.5, pp.4393,4398, 2004.

[7] Yu yali, Jiamg changhong, & Wu haiwei, Backstepping Control of each Channel for a Quaadrotor Aerial Robot, *IEEE transactions on Computer, Mechatronics, Control and Electronic Engineering (CMCE)*, vol.3, pp.403, 407, 2010.

[8] Anonymous, Cutting-Edge Flare Inspection Technology, *Flares & Stacks Inspection*, [online].Available at http://www.flares-stacks.com/ flare_inspection_stack_ inspection.html [Accessed12 November 2013]

[9] Anonymous, UAV Inspection Flight, *Cammotion Aerial Video & Photography*, [online]. Available at http://www.cammotion.be/services/industrial-inspections/ [Accessed 15 October 2013]

[10] Anonymous, *Centrica's "The Energy" Newspaper Features Cyberhawk's Offshore Live Flare Inspection*, [online]. Available at http://www.thecyberhawk. com/2013/ 04/centrias-the-energy-newspaper-features-cyberhawks-offshore-live-flare-inspection/ [Accessed12 November 2013]

[11] Hui Bai, Shihuang Shao & Hongyu Wang, VTOL Quadrotor Platform for Multi-UAV Path Planning, *Electronic and Mechanical Engineering and Information Technology (EMEIT), 2011 International Conference*, pp. 3079-3081, Vol. 6, 2011.

[12] Mustafa Ilarslan, M. Kemal Bayrakceken, & Aydemlr Arboy. Avionics System Design of a Mini VTOL UAV, *Aerospace and Electronic Systems Magazine, IEEE*, Vol. 26, Issue 10, pp. 35-40, 2011.

[13] Wang Qi, Yuan Mingxin, Zhang Peng, Cheng Fei & Cao Yuqiang, System Design and Control Research for New Quadrotor Searching and Saving Aircraft, *The 2nd International Conference on Computer Application and System Modeling*, pp. 0731-0734, 2012.

[144] James F. Roberts, Timothy S. Stirling, Jean-Christophe Zufferey & Dario Floreano, Quadrotor Using Minimal Sensing For Autonomous Indoor Flight, *European Micro Air Vehicle Conference and Flight Competition (EMAV2007)*, pp. 1-8, 2007.

[15] Zul Azfar & D. Hazry. A Simple Approach on Implementing IMU Sensor Fusion in PID Controller for Stabilizing Quadrotor Flight Control, *IEEE 7th International Colloquium on Signal Processing and its Applications*, pp. 28-32, 2011.

[16] G. Angeletti, J. R. Pereira Valente, L. Iocchi, & D. Nardi, Autonomous Indoor Hovering with a Quadrotor, *Intl.Conf. On Simulation, Modeling and Programming For Autonomous Robots Venice, Italy*, pp. 472-481, 2008.

[17] Nick Chamnong, *3D Visual Odometry in Real Time VTOL UAV Navigation*, [online]. Available at *http://isl2.cp.eng.chula.ac.th/research-file/nick/nick-proposal.pdf*. [Accessed 19 October 2013]

[18] Norafizah Abas, Ari Legowo, & Rini Akmeliawati. Parameter Identification of an Autonomous Quadrotor, International Conference on Mechatronics (ICOM), 2011

[19] Kong Wai Weng & Mohamad Shukri b. Zainal Abidin. Design and Control of a Quad-Rotor Flying Robot for Aerial Surveillance, *Student Conference on Research and Development (SCOReD 2006),* pp. 173-177, 2006

[20] Randy W. Beard. Quadrotor Dynamic and Control, M.S. thesis, Brigham Young University, United States, 2008.

[21] Sergiu Baluta, *A Guide to Using IMU (Accelerometer and Gyroscope Devices) in Embedded Applications*, [online]. Available at *http://www.starlino.com/imu_guide.html* [Accessed 17 November 2013]

[22] Shane Colton, *The Balance Filter*, [online]. Available at *http://web.mit.edu/scolton/www/filter.pdf*, [Accessed 17 October 2013]

[23] Pieter-Jan Van de Maele, *Reading a IMU without Kalman: The Complementary Filter*, [online]. Available at *http://www.pieter-jan.com/node/11* [Accessed 8 November 2013]

APPENDIX

Appendix A: Gantt Chart

	2013				2014					
Project Activities	**SEPT**	**OCT**	**NOV**	**DEC**	**JAN**	**FEB**	**MAC**	**APR**	**MAY**	**JUNE**
Title selection										
Understanding the Background and Study of Journal										
Objective and Scope										
Problem Statement										
Literature Review										
Methodology										
Expected Result and Discussion, Conclusion										
PSM1 Report Writing										
Presentation										
Submission of Report										
Derivation of mathematical modeling										
Design of On-board System										
Simulation and Testing										
Analysis on the Performance										
PSM2 Report Writing										

Appendix B: Coding for Single Rotor Run

```
#include <Servo.h>
Servo p1F;
int p1FSpeed = 0;
int STEP = 10;
void arm()
{
  // arm the speed controller by setting speed to 0 and delay for 1 second.
  // Some speed controllers may need longer delay.
  setSpeed();
  delay(1000);
}
// speed control
void setSpeed()
{
  p1F.write(p1FSpeed);
}
void setAllSpeedTo(int speed)
{
  //set the speed for all four motors to "speed"
  p1FSpeed = speed;
  Serial.print("Set all speed to: ");
  Serial.println(speed);
    //send new values to motors
  setSpeed();
}
void setup()
{
  // initialize serial communication:
  Serial.begin(9600);
   //Attach servo motors to pins
  p1F.attach(12);
    arm();
  setAllSpeedTo(20);
}
void loop()
{
  // read the keyboard:
  if (Serial.available() > 0) {
    int inByte = Serial.read();
    switch (inByte)
    {
```

```
    case 'i':
     if(p1FSpeed < 170)
     {
       p1FSpeed += STEP;
     }
     else if(p1FSpeed = 170)
     {
       p1FSpeed = 170;
     }
     Serial.print("Set F.Motor 1 speed to: ");
     Serial.println(p1FSpeed, DEC);
     break;
      case 'd':
     if(p1FSpeed > 20)
     {
       p1FSpeed -= STEP;
     }
     else if(p1FSpeed = 20)
     {
       p1FSpeed = 20;
     }
     Serial.print("Set F.Motor 1 speed to: ");
     Serial.println(p1FSpeed, DEC);
     break;
      default:
     // turn all the motors off
     setAllSpeedTo(20);
    }
  }
  setSpeed();
}
```

Appendix C: Coding for Complementary Filter Implementation

```
// Cabling for i2c using Sparkfun breakout with an Arduino Uno / Duemilanove:
// Arduino <-> Breakout board
// Gnd      -  GND
// 3.3v     -  VCC
// 3.3v     -  CS
// Analog 4 -  SDA
// Analog 5 -  SLC

#include <Wire.h> // I2C library
#include <Servo.h>
Servo myspeedcont1;
Servo myspeedcont2;
Servo myspeedcont3;
Servo myspeedcont4;

int pos=0;

float speed1;

float speed_gy=0;
float speed_y=0;
float speed_gx=0;
float speed_x=0;

unsigned long lastMicros;
unsigned long intertime=0;
unsigned long pretime=0;
unsigned long interval_millis=0;
unsigned long time_millis=0;
unsigned long interval;

//accelero start define
#define DEVICE (0x53) // Device address as specified in data sheet

byte _buff[6];

char POWER_CTL = 0x2D;  //Power Control Register
char DATA_FORMAT = 0x31;
char DATAX0 = 0x32;         //X-Axis Data 0
char DATAX1 = 0x33;         //X-Axis Data 1
char DATAY0 = 0x34;         //Y-Axis Data 0
```

```
char DATAY1 = 0x35;        //Y-Axis Data 1
char DATAZ0 = 0x36;        //Z-Axis Data 0
char DATAZ1 = 0x37;        //Z-Axis Data 1

int xA;
int yA;
int zA;
//accelero end define

//gyro start define
#define CTRL_REG1 0x20
#define CTRL_REG2 0x21
#define CTRL_REG3 0x22
#define CTRL_REG4 0x23
#define CTRL_REG5 0x24

int L3G4200D_Address = 105; //I2C address of the L3G4200D

int xG;
int yG;
int zG;
//gyro end define

double compAngleX, compAngleY, compAngleZ;

double timer = micros();

void setup()
{
  Wire.begin();        // join i2c bus (address optional for master)
  Serial.begin(9600);  // start serial for output. Make sure you set your Serial Monitor to the same!

  myspeedcont1.attach(9);
  myspeedcont2.attach(10);
  myspeedcont3.attach(11);
  myspeedcont4.attach(12);
  delay(10);
  myspeedcont1.write(180);
  myspeedcont2.write(180);
  myspeedcont3.write(180);
  myspeedcont4.write(180);
  delay(1000);
  myspeedcont1.write(40);
```

```
  myspeedcont2.write(40);
  myspeedcont3.write(40);
  myspeedcont4.write(40);
  delay(2000);
  myspeedcont1.write(41);
  myspeedcont2.write(41);
  myspeedcont3.write(41);
  myspeedcont4.write(41);
  delay(2000);
  myspeedcont1.write(43);
  myspeedcont2.write(43);
  myspeedcont3.write(43);
  myspeedcont4.write(43);
  delay(2000);

  Serial.println("Starting up ADXL345 and L3G4200D. Motor will running shortly");

  //Put the ADXL345 into +/- 4G range by writing the value 0x01 to the DATA_FORMAT register.
  writeTo(DATA_FORMAT, 0x01);
  //Put the ADXL345 into Measurement Mode by writing 0x08 to the POWER_CTL register.
  writeTo(POWER_CTL, 0x08);

  setupL3G4200D(2000); // Configure L3G4200  - 250, 500 or 2000 deg/sec
  delay(1500); //wait for the sensor to be ready
  pretime=millis();
}

void loop()
{
  time_millis=millis();
  interval_millis = time_millis-pretime;
  intertime = intertime+interval_millis;

if(intertime<1000){
  myspeedcont1.write(20);
  myspeedcont2.write(20);
  myspeedcont3.write(20);
  myspeedcont4.write(20);
  }

if(intertime<1000 && intertime<10000){
  myspeedcont1.write(60);
  myspeedcont2.write(60);
```

```
  myspeedcont3.write(60);
  myspeedcont4.write(60);
  }

if(intertime>10000 && intertime<20000){
  myspeedcont1.write(20);
  myspeedcont2.write(20);
  myspeedcont3.write(20);
  myspeedcont4.write(20);
  }

  pretime = time_millis;

  readAccel(); // read the x/y/z tilt accelerometer sensor
  getGyroValues();  // This will update x, y, and z with new values gyroscope sensor

  compAngleX = (0.98*(compAngleX+(xG*0.0055)))+(0.02*xA); // calculate the angle using a complimentary filter, filter the responses
  compAngleY = (0.98*(compAngleY+(yG*0.0055)))+(0.02*yA); // calculate the angle using a complimentary filter, filter the responses
  compAngleZ = (0.98*(compAngleY+(zG*0.0055)))+(0.02*zA); // calculate the angle using a complimentary filter, filter the responses

  Serial.print( xA  );
  Serial.print(",");
  Serial.print(  yA  );
  Serial.print(",");
  Serial.print(  zA  );

  Serial.print(",");
  Serial.print(  xG  );
  Serial.print(",");
  Serial.print(  yG  );
  Serial.print(",");
  Serial.print(  zG  );

  Serial.print(",");
  Serial.print(  compAngleX  );
  Serial.print(",");
  Serial.print(  compAngleY  );
  Serial.print(",");
  Serial.print(  compAngleZ  );

  Serial.println("");
```

```
  delay(0); // only read every t seconds, 500=0.5 sec
}

void readAccel() {
  uint8_t howManyBytesToRead = 6;
  readFrom( DATAX0, howManyBytesToRead, _buff); //read the acceleration data from the
ADXL345

  // each axis reading comes in 10 bit resolution, ie 2 bytes.  Least Significat Byte first!!
  // thus we are converting both bytes in to one int
  xA = (((int)_buff[1]) << 8) | _buff[0];
  yA = (((int)_buff[3]) << 8) | _buff[2];
  zA = (((int)_buff[5]) << 8) | _buff[4];
}

void getGyroValues(){

  byte yMSB = readRegister(L3G4200D_Address, 0x2B); //Y-axis High
  byte yLSB = readRegister(L3G4200D_Address, 0x2A); //Y-axis Low
  xG = ((yMSB << 8) | yLSB);

  byte xMSB = readRegister(L3G4200D_Address, 0x29); //X-axis High
  byte xLSB = readRegister(L3G4200D_Address, 0x28); //X-axis Low
  yG = ((xMSB << 8) | xLSB);
  yG = -yG; //invert yG value

  byte zMSB = readRegister(L3G4200D_Address, 0x2D); //Z-axis High
  byte zLSB = readRegister(L3G4200D_Address, 0x2C); //Z-axis Low
  zG = ((zMSB << 8) | zLSB);
  zG = -zG; //invert zG value, yaw right make +ve value
}

//ACCELERO WIRE
void writeTo(byte addressA, byte valA) {
  Wire.beginTransmission(DEVICE); // start transmission to device
  Wire.write(addressA);           // send register address
  Wire.write(valA);               // send value to write
  Wire.endTransmission();         // end transmission
}

//GYRO WIRE
void writeRegister(int deviceAddress, byte addressG, byte valG) {
    Wire.beginTransmission(deviceAddress); // start transmission to device
    Wire.write(addressG);       // send register address
```

```
  Wire.write(valG);        // send value to write
  Wire.endTransmission();     // end transmission
}

// (ACCELERO)Reads num bytes starting from address register on device in to _buff array
void readFrom(byte addressA, int num, byte _buff[]) {
 Wire.beginTransmission(DEVICE); // start transmission to device
 Wire.write(addressA);          // sends address to read from
 Wire.endTransmission();        // end transmission

 Wire.beginTransmission(DEVICE); // start transmission to device
 Wire.requestFrom(DEVICE, num);    // request 6 bytes from device

 int i = 0;
 while(Wire.available())        // device may send less than requested (abnormal)
 {
  _buff[i] = Wire.read();    // receive a byte
  i++;
 }
 Wire.endTransmission();        // end transmission
}

//GYRO
int setupL3G4200D(int scale){
 //From  Jim Lindblom of Sparkfun's code

 // Enable x, y, z and turn off power down:
 writeRegister(L3G4200D_Address, CTRL_REG1, 0b00001111);

 // If you'd like to adjust/use the HPF, you can edit the line below to configure CTRL_REG2:
 writeRegister(L3G4200D_Address, CTRL_REG2, 0b00000000);

 // Configure CTRL_REG3 to generate data ready interrupt on INT2
 // No interrupts used on INT1, if you'd like to configure INT1
 // or INT2 otherwise, consult the datasheet:
 writeRegister(L3G4200D_Address, CTRL_REG3, 0b00001000);

 // CTRL_REG4 controls the full-scale range, among other things:

 if(scale == 250){
  writeRegister(L3G4200D_Address, CTRL_REG4, 0b00000000);
 }else if(scale == 500){
  writeRegister(L3G4200D_Address, CTRL_REG4, 0b00010000);
 }else{
```

```
    writeRegister(L3G4200D_Address, CTRL_REG4, 0b00110000);
  }

  // CTRL_REG5 controls high-pass filtering of outputs, use it
  // if you'd like:
  writeRegister(L3G4200D_Address, CTRL_REG5, 0b00000000);
}

//GYRO
int readRegister(int deviceAddress, byte addressG)
{

    int v;
    Wire.beginTransmission(deviceAddress);
    Wire.write(addressG); // register to read
    Wire.endTransmission();

    Wire.requestFrom(deviceAddress, 1); // read a byte

    while(!Wire.available()) {
        // waiting
    }

    v = Wire.read();
    return v;
}
```

Appendix D: Simulation Test Result

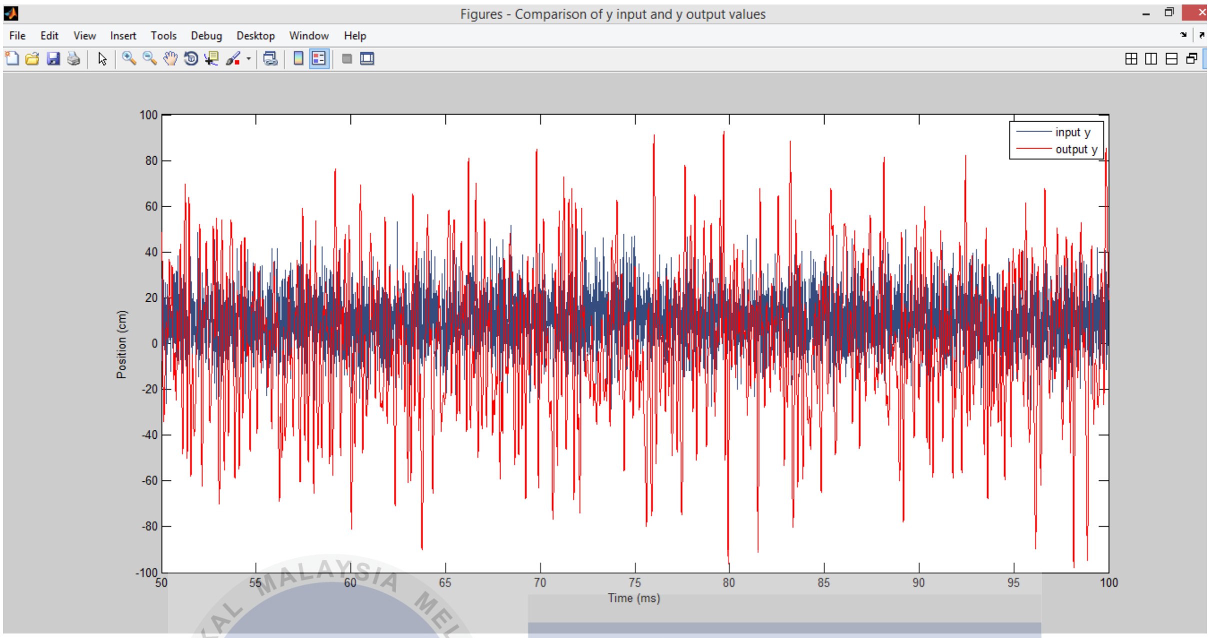

Figure D.1: Simulation result of open loop test for y-axis

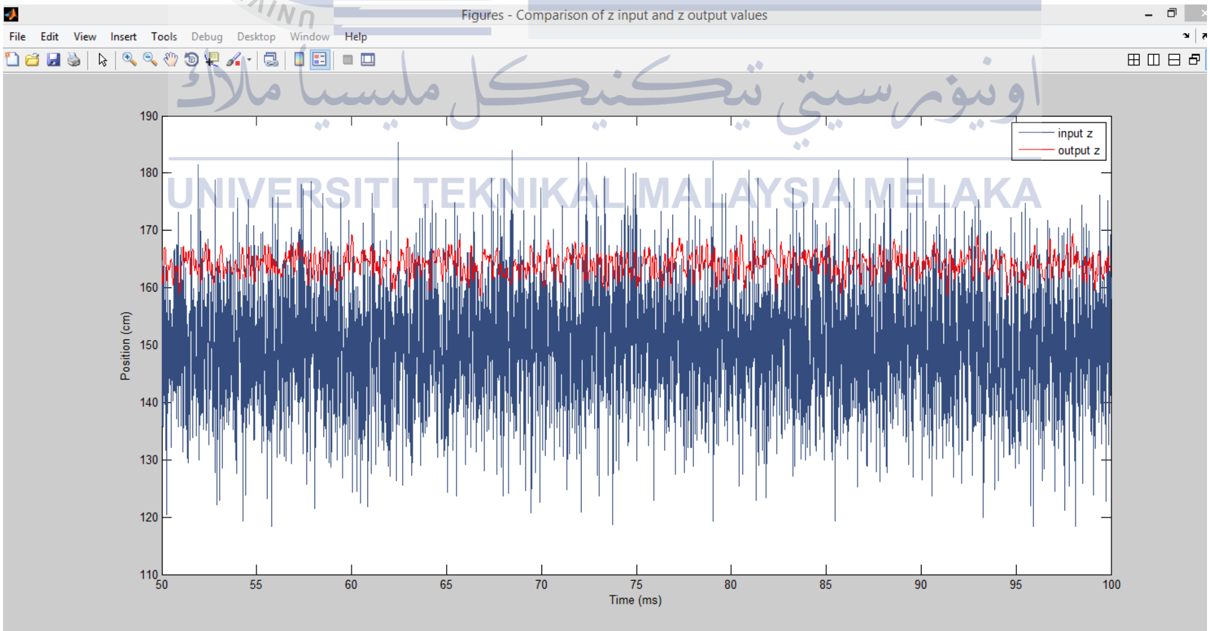

Figure D.2: Simulation result of open loop test for z-axis

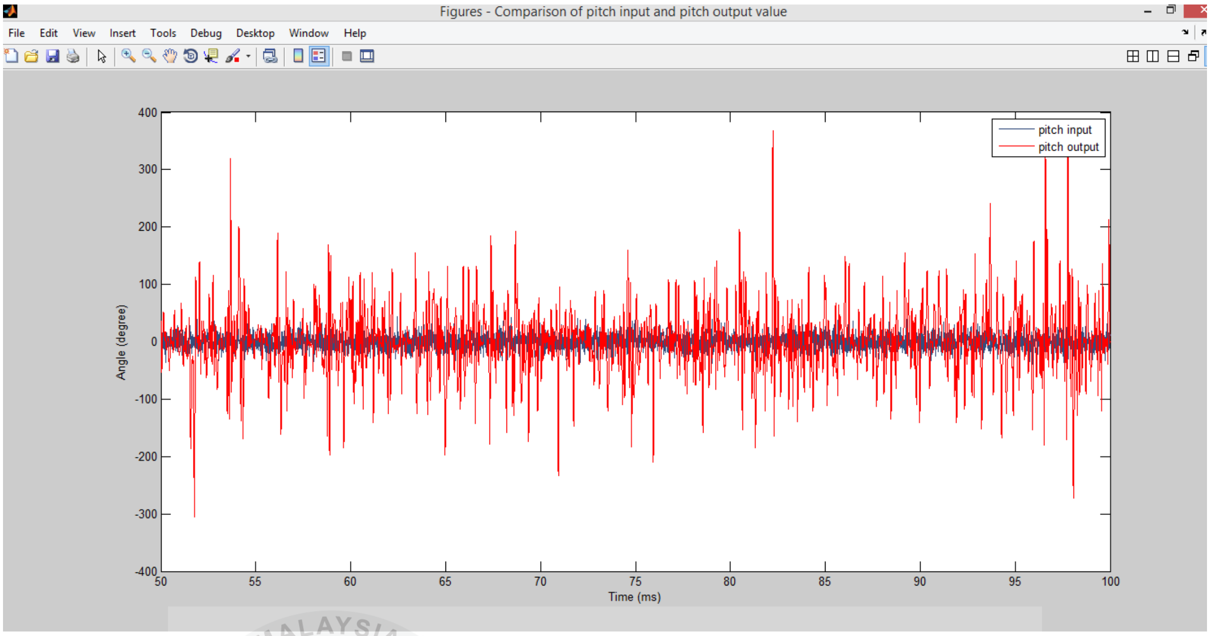

Figure D.3: Simulation result of open loop test for pitch movement

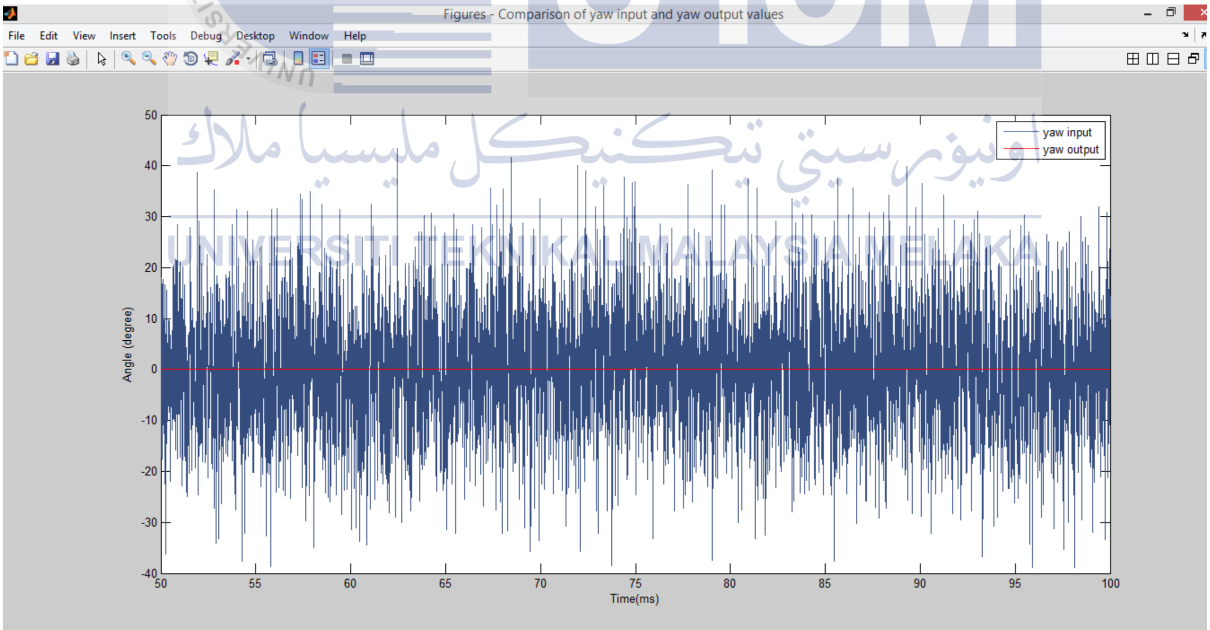

Figure D.4: Simulation result of open loop test for yaw movement

Appendix E: Real time Implementation Result

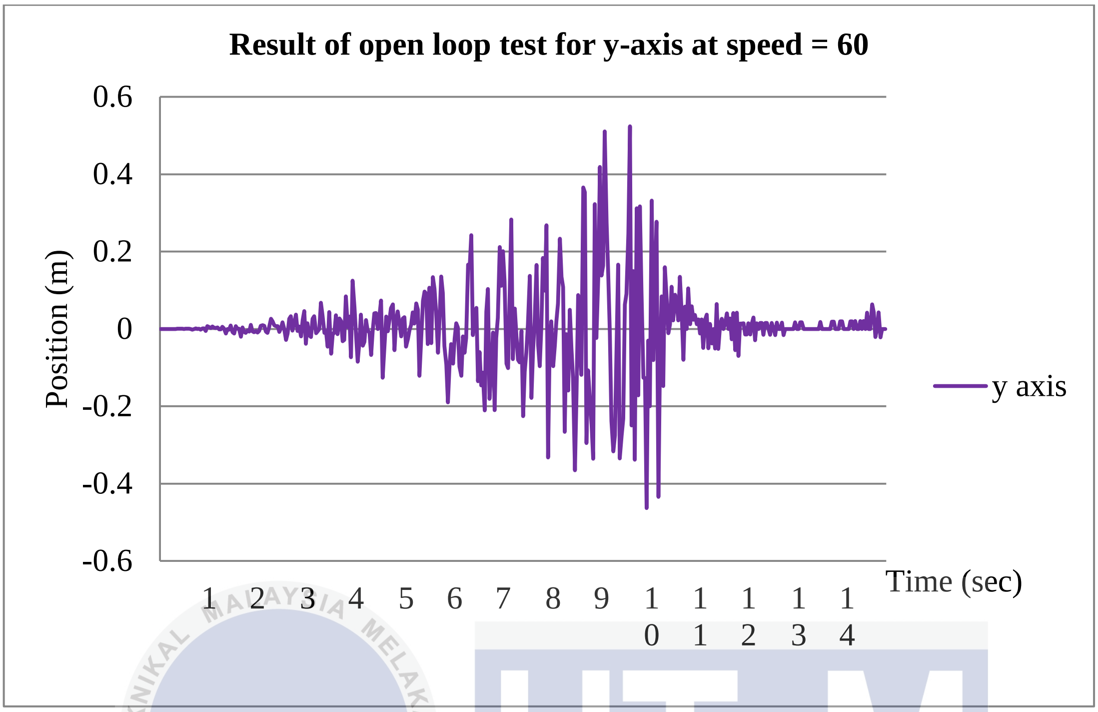

Figure E.1: Result of open loop test for y-axis at speed = 60

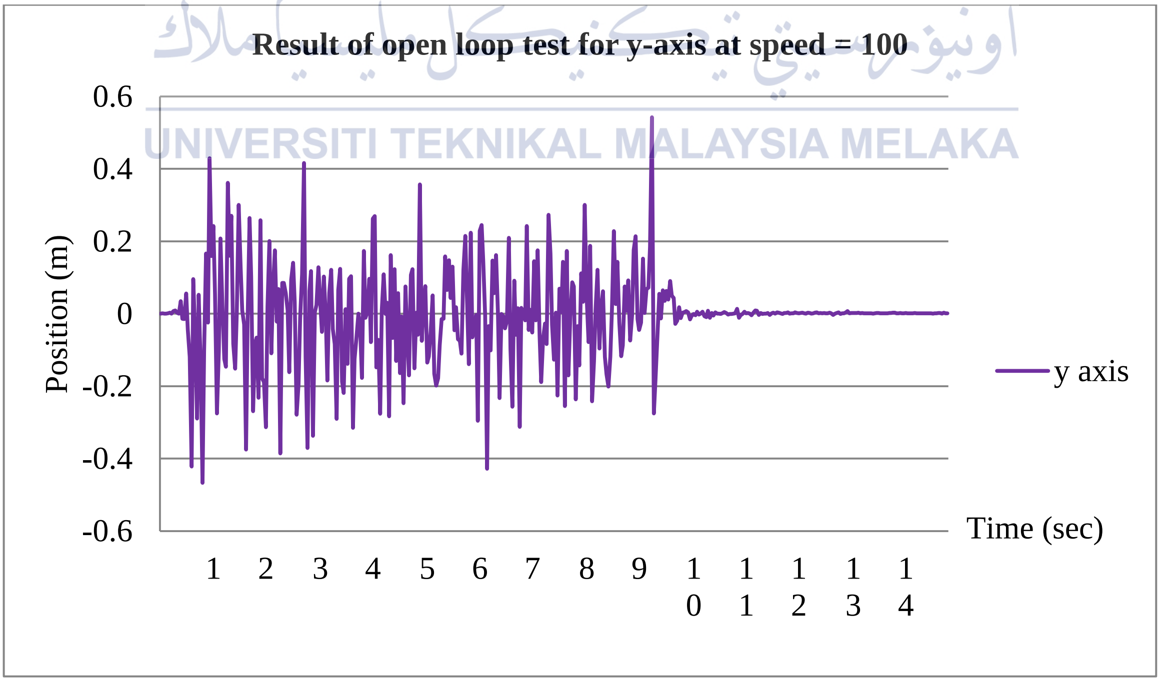

Figure E.2: Result of open loop test for y-axis at speed = 100

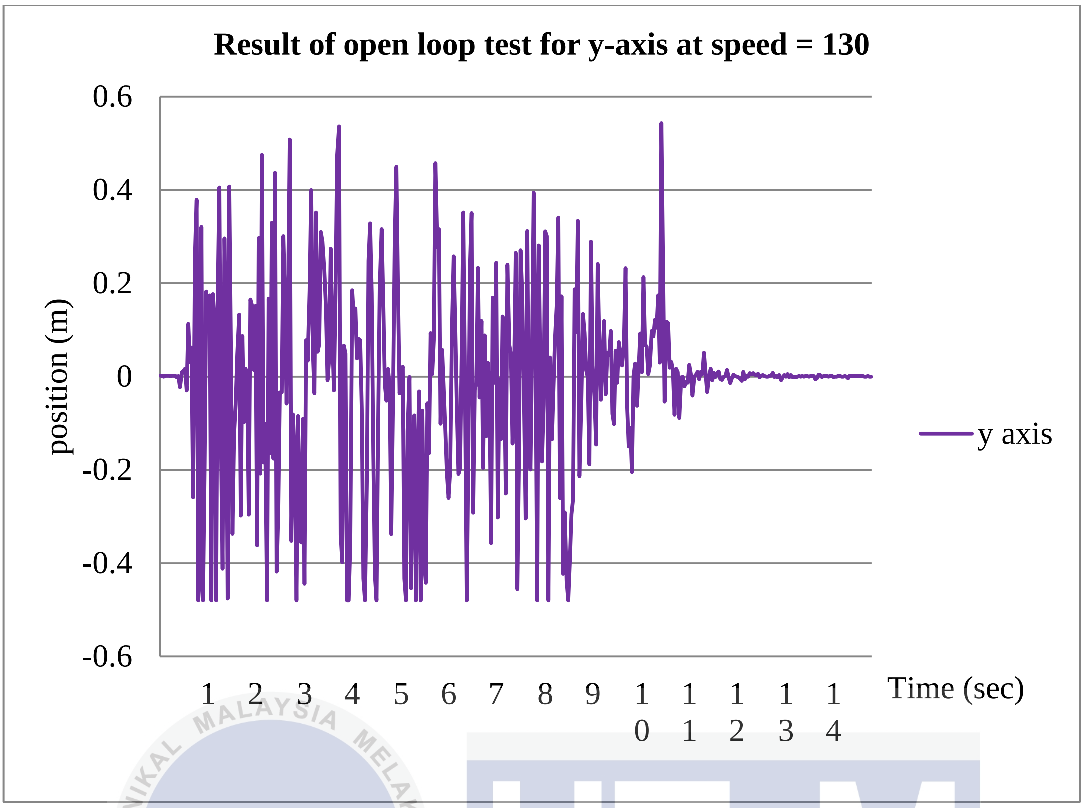

Figure E.3: Result of open loop test for y-axis at speed = 130

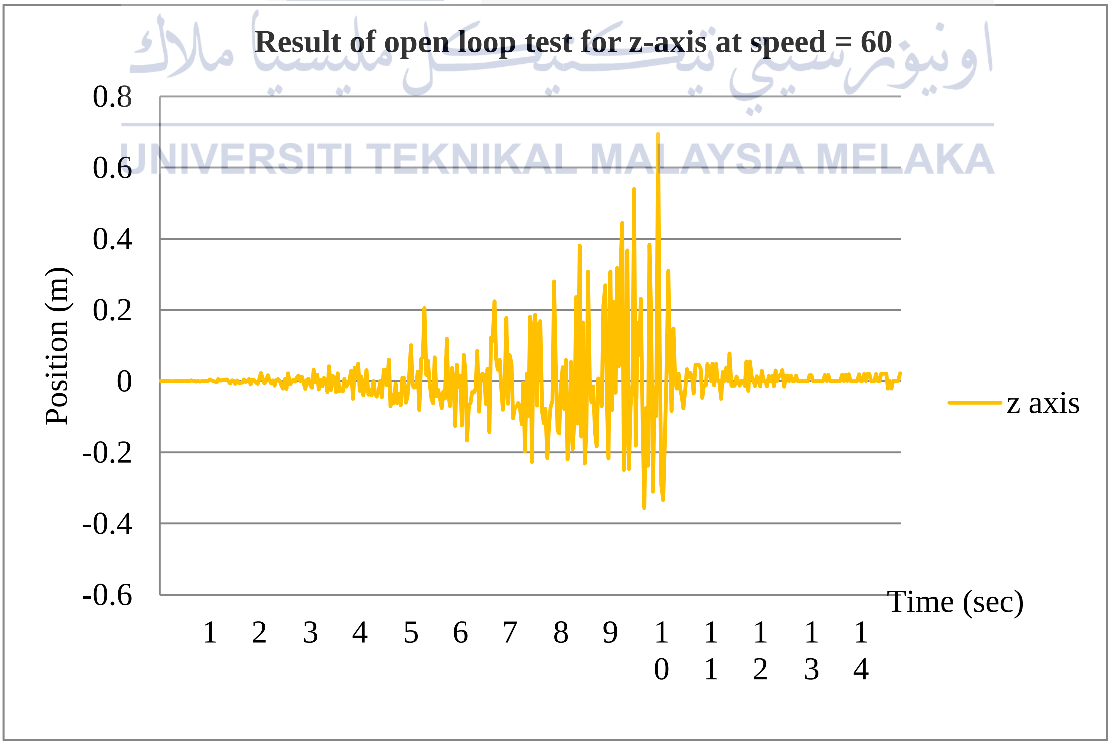

Figure E.4: Result of open loop test for z-axis at speed = 60

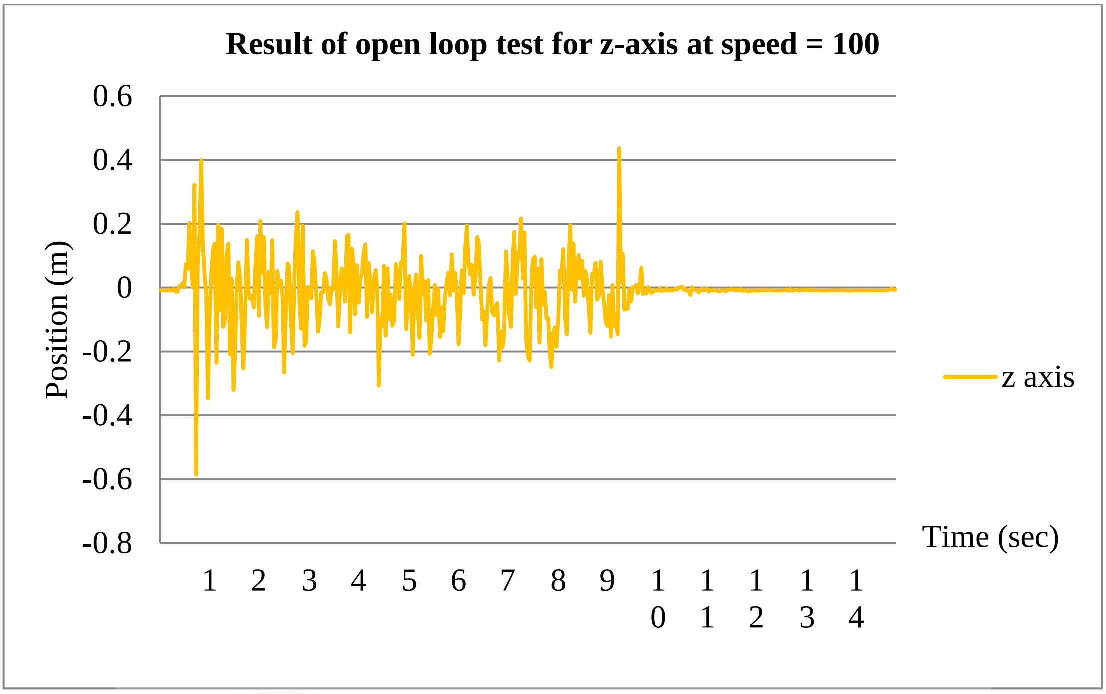

Figure E.5: Result of open loop test for z-axis at speed = 100

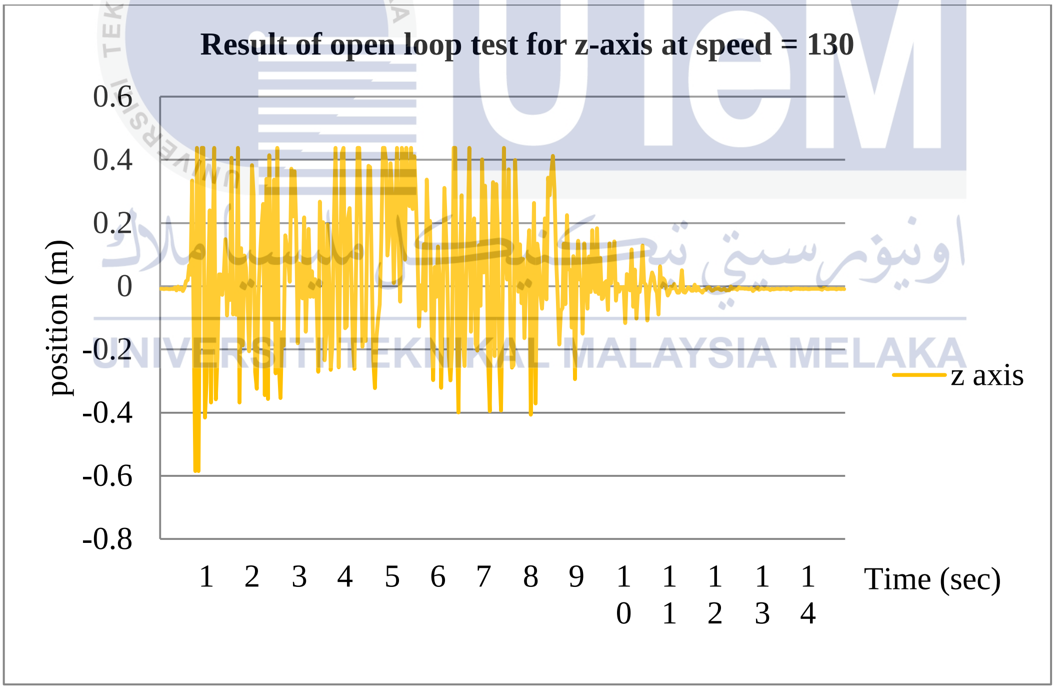

Figure E.6: Result of open loop test for z-axis at speed = 130

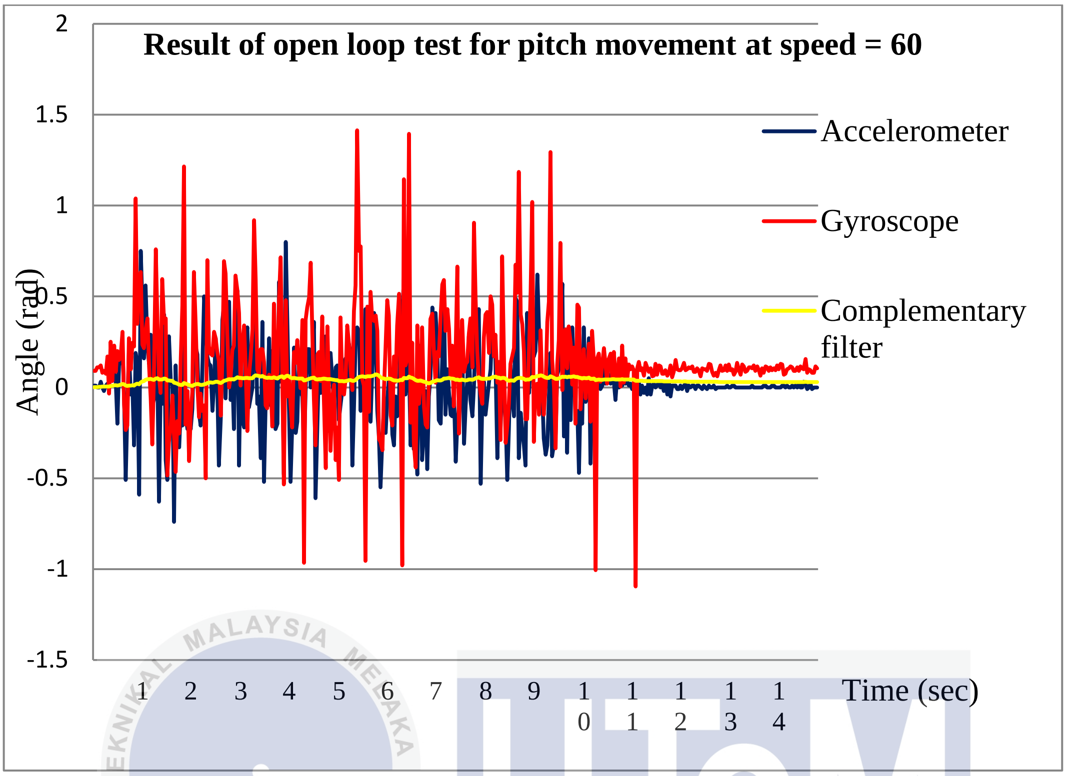

Figure E.7: Result of open loop test for pitch movement at speed = 60

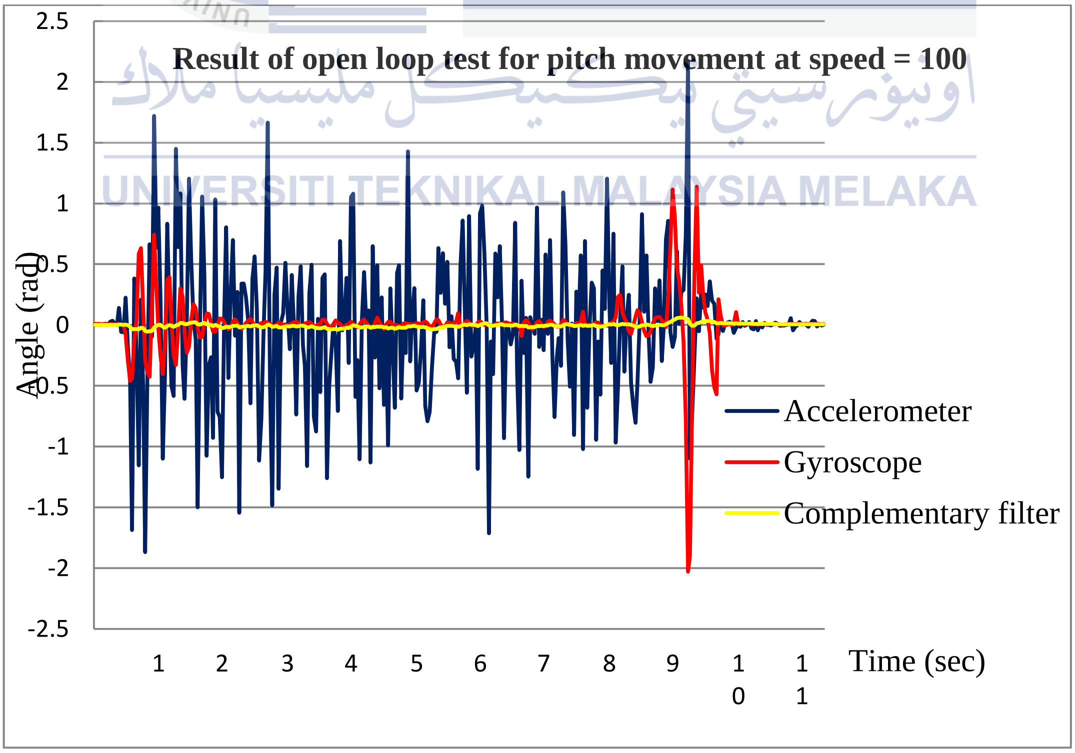

Figure E.8: Result of open loop test for pitch movement at speed = 100

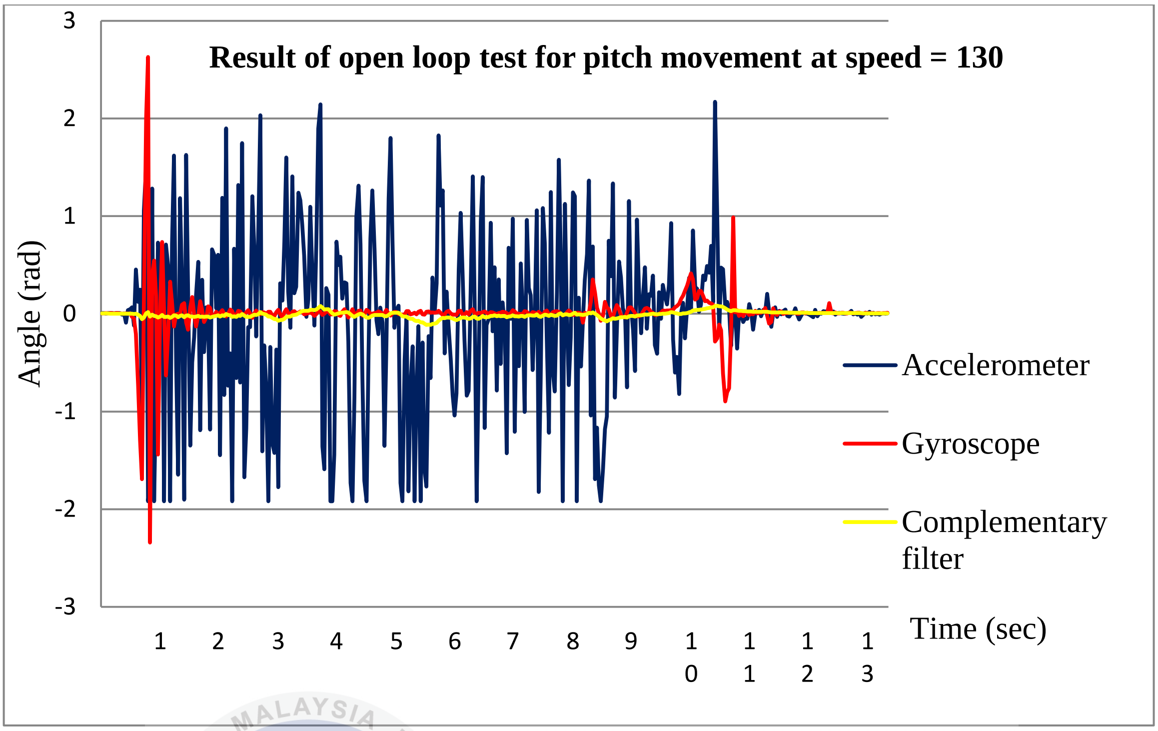

Figure E.9: Result of open loop test for pitch movement at speed = 130

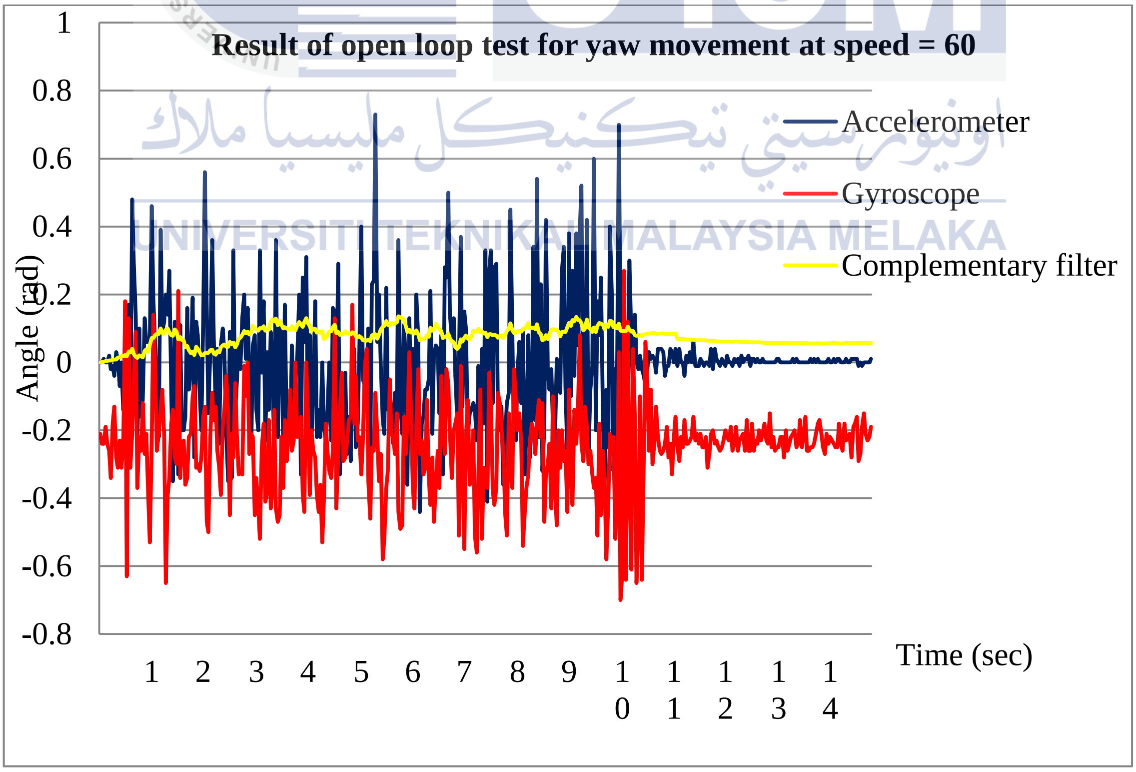

Figure E.10: Result of open loop test for yaw movement at speed = 60

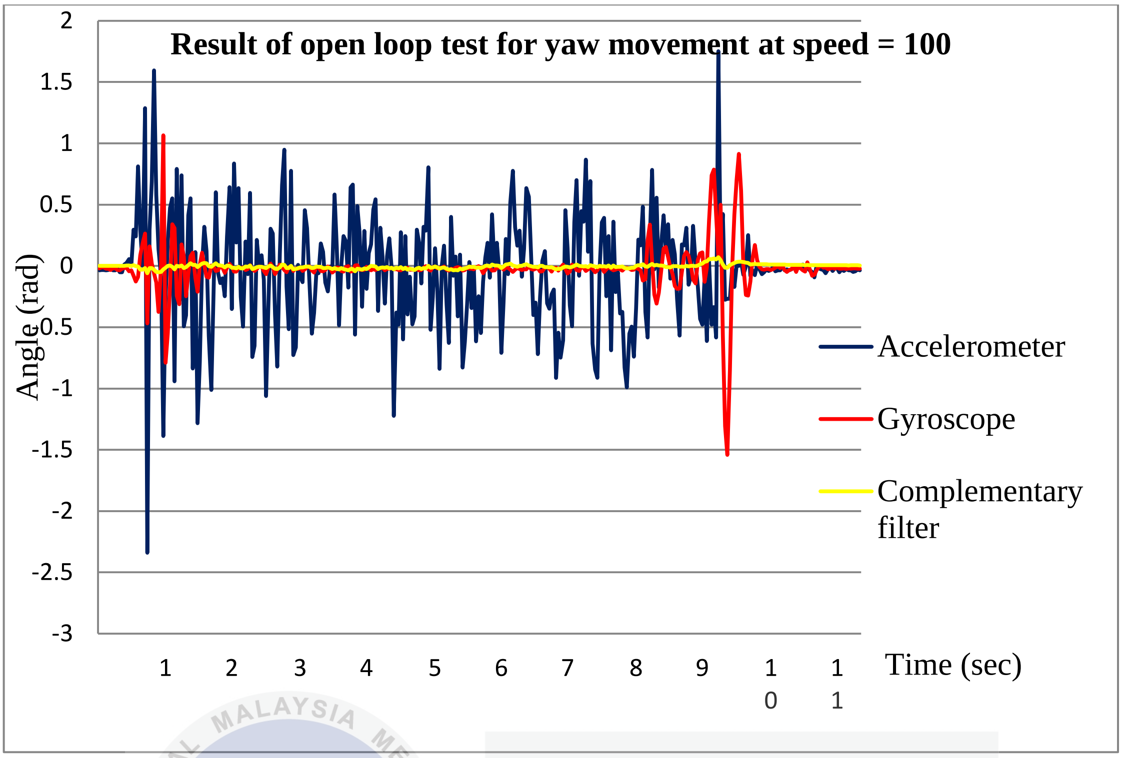

Figure E.11: Open loop test - yawing angle of quadrotor at speed 100

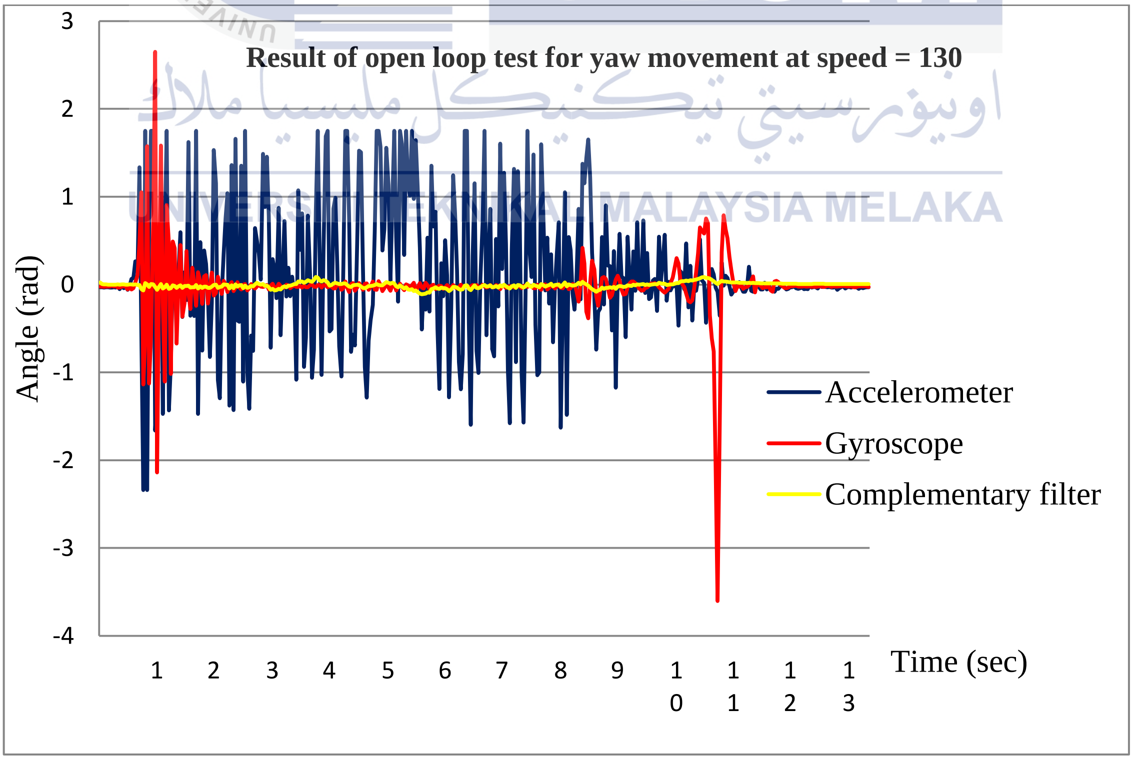

Figure E.12: Result of open loop test for yaw movement at speed = 130