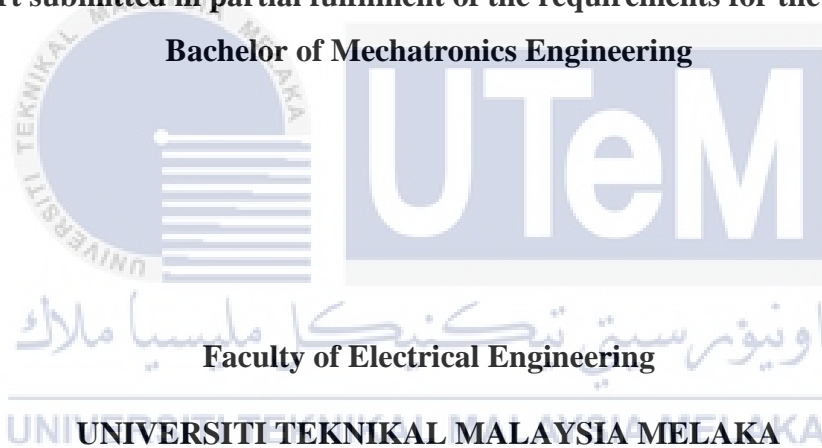


INTEGRATED SENSOR FOR UNMANNED UNDERWATER VEHICLE (UUV)

OOH MAN CHUN

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



2016/2017

“I hereby declare that I have read through this report entitle “**Integrated Sensor for Unmanned Underwater Vehicle (UUV)**” and found that it has comply the partial fulfilment for awarding the degree of Bachelor of Mechatronics Engineering.

Signature

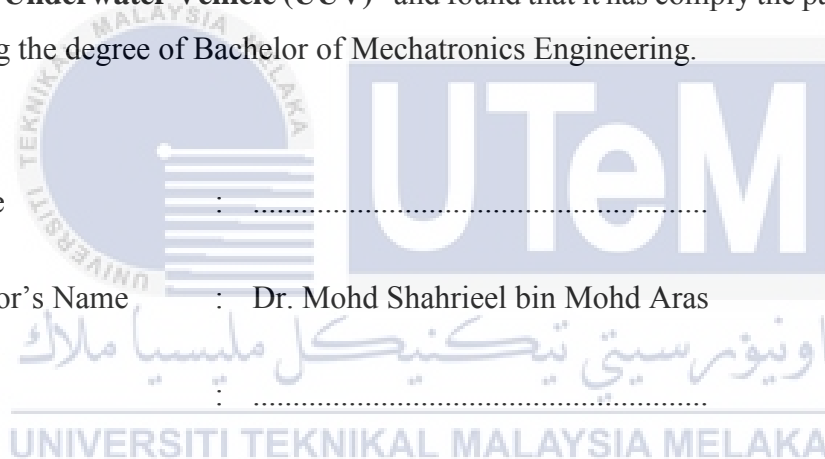
:

Supervisor's Name

: Dr. Mohd Shahrieel bin Mohd Aras

Date

:



I declare that this report entitle “**Integrated Sensor for Unmanned Underwater Vehicle (UUV)**” 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

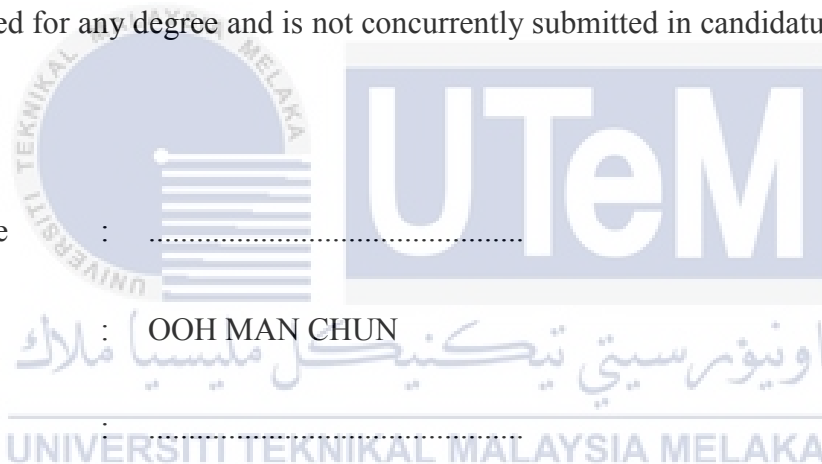
:

Name

: OOH MAN CHUN

Date

:





ACKNOWLEDGEMENT

Firstly, I would like to express my greatest gratitude to the people who have helped and supported me throughout my final year project. A special thanks to my supervisor, Dr. Mohd Shahrieel bin Mohd Aras who accepted me as his student to carry out this project. He gave me a lot of valuable advices and guidance at every stage of my project progress in this whole semester. Without his guidance and persistent help, I would not be able to complete this project.

I would also like to thank all of my friends for sharing useful knowledge and always give support to me on this project. I am also very thankful to everyone who always inspires me in this project. Not forget to give my great appreciation for my beloved parents who support and inspire me to go my own way throughout all these years.

And finally, to God who made everything possible as I believed.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

ABSTRACT

Within the past few decades, Unmanned Underwater Vehicles (UUV) are equipped with different types of sensors depending upon the missions and applications of the underwater vehicles. In underwater industries, sensors is the most important issue for the underwater task because of the cost, realibility and accuracy. However, the sensors that are used in the underwater vehicles are not in the form of integrated sensor and even not waterproof. Integrated sensor is very important for the underwater vehicles to carry out the underwater operations in several different areas. Nowadays, a lot of underwater industries are involved in the integrated sensor's design and development to reduce their production cost in order to increase its efficiency, accuracy, sensitivity and productivity. The aim of this project is to design and develop an integrated sensor which consists of IMU sensor, depth sensor, magnetometer, and leakage sensor for underwater application. Besides that, the performance of the integrated sensor is evaluated and analysed with MATLAB real-time simulation in terms of accuracy and sensitivity. The hardware is tested by the microcontroller board and integrated with MATLAB real-time simulation. To get the real-time data from the integrated sensor, it will be programmed by a microcontroller (Arduino Mega2560). The data from the sensors are used as input for serial monitor and processed in MATLAB real-time simulation to generate the output graph for the performance analysis. The performance of the integrated sensor will be verified based on its sensitivity and accuracy. The result is the integrated sensor able to collect the data for the analysis movement of UUV with low estimation error and high sensitivity.

ABSTRAK

Dalam beberapa dekad yang lalu, kenderaan bawah air tanpa pemandu (UUV) dilengkapi dengan pelbagai jenis sensor yang bergantung kepada misi dan kegunaan kenderaan bawah air. Dalam industri bawah air, sensor adalah isu yang paling penting untuk tugas bawah air kerana kos, realibiliti dan ketepatan. Walau bagaimanapun, sensor yang digunakan dalam kenderaan bawah air adalah tidak dalam bentuk bersepadu sensor dan kalis air. Sensor yang bersepadu adalah sangat penting bagi kenderaan bawah air untuk menjalankan operasi bawah air dalam beberapa bidang yang berbeza. Pada masa kini, banyak industri bawah air melibatkan dalam reka bentuk dan pembangunan sensor bersepadu untuk mengurangkan kos pengeluaran serta meningkatkan kecekapan, ketepatan, sensitiviti dan produktiviti. Projek ini adalah untuk mencipta dan menghasilkan satu sensor bersepadu yang terdiri daripada sensor IMU, sensor kedalaman, magnetometer dan sensor kebocoran untuk kegunaan bawah air. Selain itu, prestasi sensor bersepadu dinilai dan dianalisis dengan simulasi masa nyata. Perkakasan itu akan diuji oleh mikropengawal dan disepadukan dengan perisian MATLAB simulasi masa nyata. Untuk mendapatkan data masa nyata daripada sensing unit, sensor bersepadu akan diprogramkan oleh mikropengawal Atmel (Arduino Mega2560). Data masa nyata daripada sensing unit berkomunikasi dengan carta bersiri dan perisian MATLAB untuk menjana graf output dan analisis prestasi UUV. Prestasi sensor bersepadu akan disahkan berdasarkan sensitiviti dan ketepatan. Hasil ialah sensor bersepadu mampu untuk mengumpul data bagi analisis pergerakan UUV dengan kesilapan anggaran yang rendah.

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE
	ACKNOWLEDGEMENT	v
	ABSTRACT	vi
	ABSTRAK	vii
	TABLE OF CONTENTS	viii
	LISTS OF FIGURES	xi
	LISTS OF TABLE	xiv
	LIST OF ABBREVIATIONS	xvi
1	INTRODUCTION	1
	1.1 Introduction	1
	1.2 Motivation	2
	1.3 Problem Statements	3
	1.4 Objectives	4
	1.5 Scope	5
	1.6 Summary	5
2	LITERATURE REVIEW	6
	2.1 Introduction	6
	2.2 Microcontroller	6
	2.2.1 Arduino Uno	7
	2.2.2 Arduino Mega2560	7
	2.2.3 Arduino Nano	8
	2.2.4 Microbox 2000/2000C	9
	2.3 Sensor	11
	2.3.1 Inertial Measurement Unit (IMU) Sensor	11

2.3.2	Leakage Sensor	15
2.3.3	Pressure Sensor/Depth sensor	16
2.4	Theory	17
2.4.1	Orientation Angle	17
2.4.2	Pressure	19
2.5	Sensors Interface with Microcontroller	20
2.6	The Comparison of Previous Studies	20
2.7	Summary	25
3	METHODOLOGY	26
3.1	Introduction	26
3.2	Hardware Description	28
3.2.1	Microcontroller	28
3.2.2	IMU Sensor	29
3.3	Circuit Design	31
3.4	Design and Assembly of Integrated Sensor	32
3.5	Experiment and Analysis	35
3.5.1	Experiment 1: Orientation Movement Analysis of IMU Sensor	35
3.5.2	Experiment 2: To Determine the Consistency and Accuracy of Depth Sensor	37
3.5.3	Experiment 3: Depth Sensor Performance Analysis in Air	38
3.5.4	Experiment 4: Depth Sensor Performance Analysis in Underwater	40
3.5.5	Experiment 5: Depth Control of UUV	41
3.5.6	Experiment 6: Heading Degree Analysis of Magnetometer	43
3.5.7	Experiment 7: Leakage Sensor Testing	44
3.6	Modelling and Controller	45
3.6.1	System Identification Toolbox of Matlab	45
3.6.2	AUV On-Off Depth Controller	49
3.7	Summary	50
4	RESULTS AND DISCUSSIONS	51
4.1	Introduction	51
4.2	Final Design of Integrated Sensor	51

4.3	Analysis and Discussion of Experiment	52
4.3.1	Experiment 1: Orientation Movement Analysis of IMU Sensor	52
4.3.2	Experiment 2: To Determine the Consistency and Accuracy of Depth Sensor	58
4.3.3	Experiment 3: Depth Sensor Performance Analysis in Air	61
4.3.4	Experiment 4: Depth Sensor Performance Analysis In Underwater	65
4.3.5	Experiment 5: Depth Control of UUV	69
4.3.6	Experiment 6: Heading Degree Analysis of Magnetometer	72
4.3.7	Experiment 7: Leakage Sensor Testing	74
4.4	Summary	75
5	CONCLUSION AND RECOMMENDATION	76
5.1	Conclusion	76
5.2	Future Work and Recommendation	77
	REFERENCES	78
	APPENDIX A	81
	APPENDIX B	82
	APPENDIX C	90
	APPENDIX D	91
	APPENDIX E	91

LISTS OF FIGURES

FIGURE	TITLE	PAGE
1.1	The searching of MH370 plane in underwater	3
2.1	Arduino Uno	7
2.2	Arduino Mega2560	8
2.3	Arduino Nano	8
2.4	Microbox 2000/2000C	9
2.5	An IMU sensor determine an vehicle's orientation	11
2.6	5 DOF IMU sensor	13
2.7	10 DOF IMU sensor	14
2.8	Leakage sensor	15
2.10	x , y and z axis from the IMU sensor in an UUV	17
2.11	The orientation output angle of IMU sensor at X, Y and Z axes	18
2.12	The relationship between the depth and water pressure	19
2.13	LabView real-time output graph of gyroscope	22
2.14	Real time output result for pressure sensor and IMU sensor	21
2.15	The ROV's electronic system published by Pakpong Jantapremjit, 2011	21
2.16	The output graph results by using LibViso2 software	23
3.1	The overall process flow of the methodology	27
3.2	Illustration of the system	28
3.3	Schematic circuit diagram of depth sensor, MPX5700AP	31

3.4	Complete soldered depth sensor with decoupling circuit	31
3.5	Integrated sensor casing	32
3.6	All the components in the casing	33
3.7	5V and ground pin configuration on one strip line of breadboard	33
3.8	Set up of experiment 1	35
3.9	The electronic components in the casing	36
3.10	Set up of experiment 2	37
3.11	Set up of experiment 3	38
3.12	Air compressor tube slightly contact to the depth sensor tube	39
3.13	Set up of experiment 4	40
3.14	Set up of experiment 5	42
3.15	Set up of experiment 6	43
3.16	Set up of experiment 7	44
3.17	MATLAB system identification toolbox	46
3.18	Import data	47
3.19	Time plot for input and output data	47
3.20	Transfer function model	47
3.21	3 poles and 2 zeros implemented	48
3.22	Transfer function model	48
3.23	Step response of the transfer function	48
3.24	Graph of depth input and output against time	50
4.1	Completed design of integrated sensor	51
4.2	Interior part of integrated sensor	52
4.3	Initial output graph of accelerometer and gyroscope	55
4.4	Output graph of acceleromter	56
4.5	Output graph of acceleromter in UUV	56

4.6	Output graph of gyroscope	57
4.7	Graph of pressure reading in MATLAB real time simulation	58
4.8	Graph of analog reading in MATLAB real time simulation	58
4.9	Graph of measured output signal voltage versus input pressure	61
4.10	Characteristics plotted graph of depth sensor, MPX5700AP	62
4.11	Graph of actual and desired pressure versus input pressure	64
4.12	Graph of output signal voltage versus water depth	65
4.13	Pressure converter	66
4.14	Theoretical and actual pressure versus depth	68
4.15	Graph of analog calculation value of depth sensor versus depth	69
4.16	Output graph of magnetometer	72



LISTS OF TABLE

TABLE	TITLE	PAGE
2.1	Comparison of the microcontrollers	10
2.2	The advantages and disadvantages of each type sensor in IMU	12
2.3	Sensor descriptions of 10 DOF IMU	14
2.4	Comparison between IMU sensors	14
2.5	Pins configuration of leakage sensor	15
2.6	Pin configuration of MPX series	16
2.7	Orientation angle at X, Y and Z axis	18
2.8	The comparison of previous studies to current studies	24
3.1	Comparison between the microcontrollers	29
3.2	Comparison between the IMU sensors	30
3.3	Interface description of 5 DOF IMU	30
3.4	Characteristics design of integrated sensor casing	32
3.5	Pins configuration connection between sensors and microcontroller	34
3.6	Depth data of AUV	46
3.7	Depth data of AUV with on-off depth controller	49
4.1	The accelerometer readings at x-axis	53
4.2	The accelerometer readings at y-axis	53
4.3	The readings of accelerometer at z-axis	53
4.4	The readings of accelerometer at x, y and z axis	54

4.5	The readings of gyroscope at x, y and z axis	54
4.6	The output results of experiment 3	61
4.7	Percentage error of depth sensor in experiment 3	63
4.8	The output results of experiment 4	65
4.9	Percentage error of depth sensor in experiment 4	67
4.10	Analog reading of depth sensor at different depth	69
4.11	Percentage error of depth control in UUV	71
4.12	Output heading degree and percentage error of magnetometer	72
4.13	Summary of error percentage of sensors	75



LIST OF ABBREVIATIONS

UUV – Unmanned Underwater Vehicle

AUV – Autonomous Underwater Vehicle

ROV – Remotely Operated Vehicle

UG – Unmanned Glider

IMU – Inertial Measurement Unit

IDE – Integrated Development Environment

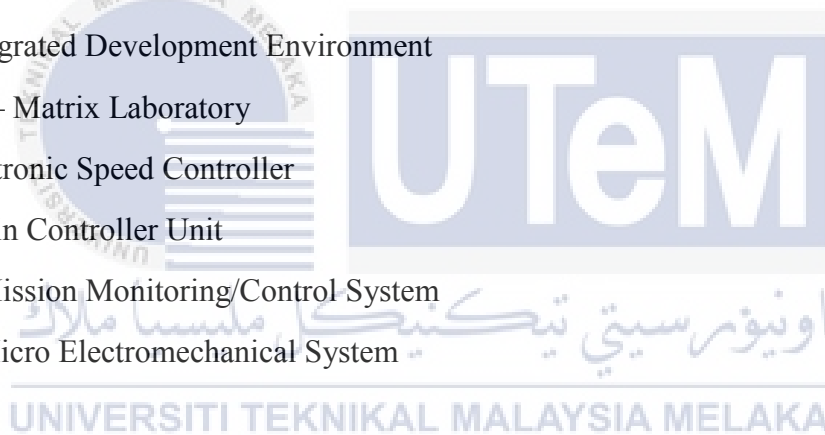
MATLAB – Matrix Laboratory

ESC – Electronic Speed Controller

MCU – Main Controller Unit

MMCS – Mission Monitoring/Control System

MEMS – Micro Electromechanical System



CHAPTER 1

INTRODUCTION

1.1 Introduction

For the Unmanned Underwater Vehicles (UUVs), sensors have become the most important issue in the underwater application. However, the sensors that are applied in the underwater vehicles are not in the form of integrated sensor. Besides that, most of the sensors that are applied in underwater application are quite expensive, sensitive and even not overall waterproof. Therefore, it is not an easy task to get an accurate measurement by using the UUV [1, 2].

An integrated sensor sometimes known as smart sensor is a sensor in a small size that are designed and developed to gather data for analysis. Basically, it consists of different types of sensors which are combined or integrated into one compact device with signal processing hardware. This means that it can be sends the signal without any additional processing hardware or amplifier. Not only this, the space and weight can be reduced to put other components. Furthermore, it can also save the time taken to design a new sensor. It is very important for the UUV to carry out the submerged operations in several different areas such as communications, localization, motion planning and hydrodynamics. For example, the tragedy MH370 which a MH370 plane crashed and sank into the Indian Ocean. The integrated sensor can provide the facilities for UUV to find objects which are submerged beneath the sea and navigation or localization of UUV.

In this project, an integrated sensor system is designed to apply in the underwater application which consist of four major sensors: 5 degree of freedom (DOF) Inertial Measurement Unit (IMU) sensor, depth sensor, magnetometer and leakage sensor. To get

the real-time data from the integrated sensor, it will be programmed by a microcontroller (Arduino Mega2560). The data from the sensors are used as input for serial monitor and processed in MATLAB real-time simulation to generate the output graph for the performance analysis. The performance of the integrated sensor will be verified based on its sensitivity and accuracy.

The aim of this project is to design and develop the ability of each sensor into an integrated sensor and analyse its performance by using MATLAB real-time simulation. The goal performance of this integrated sensor is improved based on low cost, small space, low estimation error, waterproof and multifunctional. However, the performance of the different types of sensors in underwater are more emphasized in this project. The experimental on each sensor are performed to test their functionality and performance.

1.2 Motivation

In Malaysia, the sensor and vehicle in underwater industries are not widely studied and researched. The underwater technology in Malaysia is not much advanced if compared to other foreign countries, such as United States of America (USA), Russia, Australia and etc. They have advanced underwater technology to perform the more challenging underwater mission. It can be proven by missing incidence of MH370 plane on 8th March 2014, Bluefin-21 from Australia Navy performed black box's signal searching mission in the Southern Indian Ocean as shown in Figure 1.1. Malaysia need to seek underwater technology assistance from Australia to carry out the searching mission. In this searching mission, Malaysia need to sponsor in term of financial for some foreign countries. However, they took this chance to research and develop their underwater industries. So, Malaysia need to expand the underwater technology in order to compete with the foreign countries.

Therefore, a greater performance of integrated sensor for UUV is needed to be developed in order to apply in underwater tasks. The integrated sensor consists of four sensors: 5 DOF IMU sensor, depth sensor, magnetometer and leakage sensor. The 5 DOF IMU sensor integrates accelerometer and gyroscope which provides 5 DOF. The depth sensor is used to measure the underwater pressure and control the water depth of UUV. Magnetometer is used to measure the heading degree of UUV and leakage sensor is used to detect water leakage inside the integrated sensor casing.

In the development of integrated sensor, the space and weight of the UUV are reduced. The integrated sensor can help the UUV to collect different types of data information in underwater.

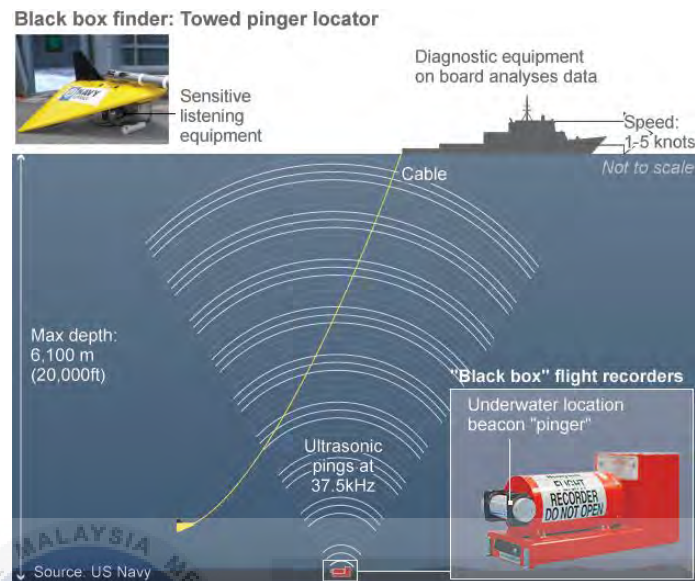


Figure 1.1: The searching of MH370 plane in underwater

1.3 Problem Statements

In underwater industries, UUV is divided into Remotely Operated Underwater Vehicle (ROV), Autonomous Underwater Vehicle (AUV) and Underwater Glider (UG) [3, 4, 5]. These vehicles are the good platform in the underwater industries which cover different kinds of application such as monitoring, inspection, maintenance, communication, and construction [1, 2, 6].

Within the past few years, different types of sensors are installed into UUV to determine the depths of the ocean, monitor, maintain, search and rescue operation. Within the growing of underwater technology, an integrated sensor is designed and developed to perform more challenging underwater operations and even used in military mission.

However, the number of integrated sensor for UUV is limited because most of the sensors utilized in this field are not in the integrated sensor form. Every task performed by UUV is quite expensive because the sensors used in underwater industries are different with the others in terms of waterproof, long-lasting, high durability and other requirement factors

[1, 2]. Furthermore, underwater industries of Malaysia are still much left behind if compared with the other foreign countries. For example, crashing and missing of MH370 plane on 8th March 2014, Australia Navy supplied the advanced underwater technology to perform the searching operation in the Southern Indian Ocean. Therefore, it is necessary to improve the performance of integrated sensor for UUV in terms of the factors requirements.

Furthermore, the underwater sensor for the vehicle is quite expensive if compare to the ground sensor due to the many factors requirements of underwater application. Not only this, most of the sensors available in the market are not suitable to be applied in the underwater industries. For example, altimeter is an instrument used to determine the altitude of an object above the fixed level. However, it is quite expensive and even not suitable applied in underwater vehicles. The performance of the sensor is also one of the problem statement of the integrated sensor. For example, the sensitivity of accelerometer ADXL345 is between $\pm 2g$ to $\pm 16g$. Therefore, it is very important to analyse the performance of the sensor based on its sensitivity and accuracy.

In this project, the UUVs must be able to interpret its environment by gathering data information from its sensors. Four suitable sensors are chosen to design and develop an integrated sensor for UUVs to collect the underwater data information. Besides that, a suitable microcontroller platform is required in order to interface with all of the sensors chosen. Therefore, the mechatronics knowledge is required to develop an integrated sensor circuit from all of the sensors and microcontroller. Competency in programming is necessary to program the sensing unit in order to communicate with the processing software. All these are important parameters that need to be considered in order to make the integrated sensor able to collect the accurate measurement for underwater application.

1.4 Objectives

In this project, there are two objectives going to achieve:

- 1) To design and develop an integrated sensor which consists of IMU sensor, depth sensor, magnetometer and leakage sensor for underwater application.
- 2) To analyse the performance of integrated sensor with MATLAB real-time simulation in terms of sensitivity and accuracy.

1.5 Scope

The main scope in this project is to design and develop an integrated sensor with the combination of four sensors: 5 DOF IMU sensor, depth sensor, magnetometer and leakage sensor to get data in rotational speed, acceleration, underwater pressure, heading degree and also detects water leakage inside the integrated sensor casing. The overall integrated sensing system should be in a waterproof casing that is purposely used to gather underwater data.

The program is wrote and load into a microcontroller board (Arduino Mega2560). The integrated sensor is interfaced with a microcontroller board and then installed into an UUV. To reduce the delay time and get the more accurate output signal values, all the outputs of the sensors are connected to USB cable with length of 10m. All hardware design works are held in laboratory, the experiments are carried out at swimming pool or underwater laboratory.

The hardware is tested by the microcontroller board and integrated with MATLAB real-time simulation. To get the real-time data from the sensing unit, the measurement unit will be programmed by an Atmel microcontroller. The data from the sensors are used as input for serial monitor and processed in MATLAB real-time simulation to generate the output graph for the performance analysis. The performance of the integrated sensor will be verified based on its sensitivity and accuracy.

1.6 Summary

As the conclusion of this chapter, it explains the importance of integrated sensor utilized in underwater industries. Integrated sensor is limited used in underwater industries because it is designed and developed in terms of many criteria, such as waterproof, long lasting, high durability, accuracy and sensitivity. The aim of this project is to design and develop the ability of each sensor into an integrated sensor and install it into an UUV for data analysis. This project also need to interface the integrated sensor with MATLAB real-time simulation in order to get the accurate real-time data from the sensing unit.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

In this chapter, literature review is discussed from journals or conference papers that are related to this project title in order to achieve the objectives of this project. There are many papers have been studied in order to develop an integrated sensor for UUV. The important hardware and software methodology are extracted from these papers. In hardware development, the microcontrollers and sensors are studied, compared and selected. The suitable microcontroller and sensors are discussed to develop an integrated sensor and interface between them. Therefore, it is very important to study the basic working operation of each type of the sensor before the sensor is selected for the integrated sensor development. In software development, the integrated sensor is communicated with the MATLAB real-time simulation and analysed to get the real-time output graph.

2.2 Microcontroller

Microcontroller is a small integrated circuit that consists of a microprocessor core as a brain of the controller, program memory such as random-access memory (RAM) and programmable input/output peripherals. Microcontrollers have been widely used in the devices and system such as robot system, security system and automatically controlled devices. In this part, there are four types of microcontrollers discussed which are Arduino Uno, Arduino Mega2560, Arduino Nano and Microbox 2000/2000C.

2.2.1 Arduino Uno

According to the journal of M.S.M Aras, M.F. Basar, S.S. Abdullah, F.A. Azis and F. A. Ali [6], it presented the Arduino Uno as shown in Figure 2.1 which is used as a microcontroller of the system. The Arduino Uno is a microcontroller board based on the ATmega 328 by Atmel. It has 14 digital input/output pins and 6 analog inputs. It features 32Kb flash memory, 2Kb SRAM and 16 MHz of clock frequency. It also consists of a USB interface and a power jack for 9V to 12V AC to DC adapter connection. Its operating voltage is 5V. An ICSP header of the Arduino is used to upload programming language from an Integrated Development Environment (IDE). In this journal, the Arduino Uno is used as a microcontroller of the system and interfaced with sensing unit (IMU sensor). The sensing unit is programmed via serial communication of Arduino Uno to obtain UUV's navigation data.



Figure 2.1: Arduino Uno

2.2.2 Arduino Mega2560

According to J. Busquets, J. V. Busquets, A. Perles, R. Mercado, R. Saez, J. J. Serrano, F. Albetosa and J. Gilabert [7], the Arduino Mega2560 is chosen as a main controller unit (MCU) of the sensors as shown in Figure 2.2. Arduino Mega is an 8 bits 100 pins microcontroller board based on the ATmega 2560 by Atmel. It has 8Kb SRAM, 4Kb EEPROM, 256Kb flash memory and 16 MHz clock frequency. It has 54 digital input/output pins and 15 of them can be used as PWM outputs, 16 analogue inputs of 10 bits resolution, 4 UARTs (hardware serial port), I2C and SPI. This platform is the most reliable and high performance 8 bits microcontroller board available today by Arduino. All the components, devices or programming can easily interface with Arduino microcontroller boards because most of the header and connector are compatible. It can be used to connect to a computer with a USB cable or powered with an adapter or battery.



Figure 2.2: Arduino Mega2560

2.2.3 Arduino Nano

According to C. R. Rocha, R. M. Brancoy, L. A. D. Cruzz, M. V. Schollx, M. M. Cezarx and Felipe D [16], the Arduino Uno as shown in Figure 2.3 is chosen as a microcontroller for the Mission Monitoring/Control System (MMCS) to supervise te underwater vehicle. Arduino Nano board is a small and complete board based on the ATmega328. It has 14 digital input/output pins (6 provide PWM output) and 8 analog inputs. It features 32KB flash memory, 2KB SRAM and 16 MHz of clock frequency. It consists of a mini USB interface but no DC power jack. Its operating voltage is 5V. The mini USB header of the board is used to upload programming language from an IDE and supply power.

In this journal, there are three Arduino Nano are used to interface H-bridge, sensors and joystick. One of the Arduino Nano board is used to connect with the IMU and pressure sensor to process the sensor information from IMU and pressure sensor in order to determine UUV's attitude, heading and depth.



Figure 2.3: Arduino Nano

2.2.4 Microbox 2000/2000C

Microbox 2000/2000C is a high performance and low power consumption PC as shown in Figure 2.4. According to Mohd Shahrieel Mohd Aras, Fadilah Abdul Azis, Shahrum Shah Abdullah, Lee Dai Cong, Lim Wee Teck, Fara Ashikin Alic and Muhammad Nur Othman [22], Microbox 2000/2000C is used to interface between the ROV hardware and MATLAB software. It is also called as XPC target machine and can act as a microcontroller for hardware interfacing. It is a excellent device to integrate with MATLAB/Simulink and other control modules such as real time modelling, control system simulation, prototyping and hardware loop testing without any manual code or debugging. Therefore, the time and costs are saved because the control system design and testing can be easily done.

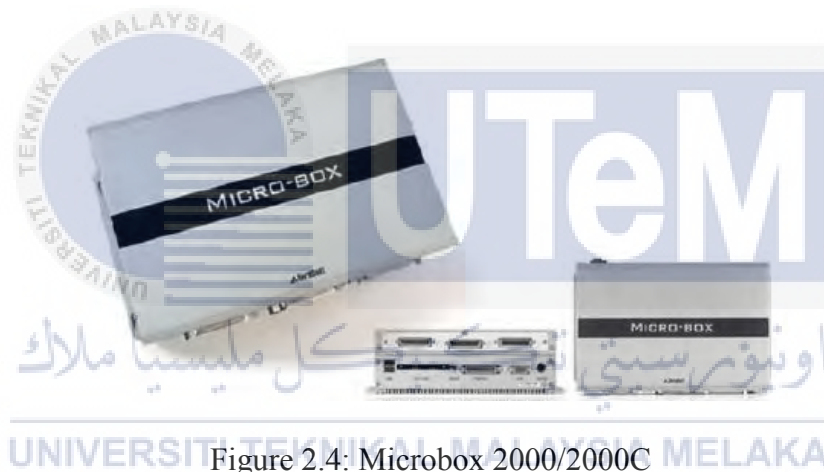


Figure 2.4: Microbox 2000/2000C

Table 2.1: Comparison of the microcontrollers

Journal No.	Micro-controller	Specifications	Advantages
[6]	Arduino Uno	<ul style="list-style-type: none"> • 14 digital input/output pins • 6 analog inputs • 32Kb flash memory • 2Kb SRAM • 16 MHz clock frequency • 5V operating voltage 	<ul style="list-style-type: none"> • Simple programming language • Able to read the analog value from analog sensor
[7]	Arduino Mega2560	<ul style="list-style-type: none"> • 54 digital input/output pins • 16 analogue inputs • 256Kb flash memory • 8Kb SRAM • 16 MHz clock frequency • 5V operating voltage 	<ul style="list-style-type: none"> • Simple programming language • Able to read the analog value from analog sensor • More input/output pins available • High flash memory
[16]	Arduino Nano	<ul style="list-style-type: none"> • 14 digital input/output pins • 8 analog inputs • 32KB flash memory • 2KB SRAM • 16 MHz of clock frequency • 5V operating voltage 	<ul style="list-style-type: none"> • Simple programming language • Able to read the analog value from analog sensor • Small in size • Low cost
[22]	Microbox 2000/2000C	<ul style="list-style-type: none"> • 4 encoder channel • 16 digital I/O channels • 4 D/A channels • 8 A/D channels 	<ul style="list-style-type: none"> • Low power consumption • Able integrate with MATLAB/Simulink and other control modules • Able to run without any manual code or debugging. • the time and costs are saved • Simple to design control system and testing • Save time and cost

2.3 Sensor

Sensor is an electronic component used to detect the changes in its environment and sends its information to another device. The inputs of the sensor can be speed, light, pressure and so on. The input of sensor can be converted into data. Sensor is widely used in the field of control system, measurement, monitoring and automation. In this section, there are different kinds of sensors such as IMU sensor, depth sensor, magnetometer and leakage sensor are discussed and compared to develop an integrated sensor.

2.3.1 Inertial Measurement Unit (IMU) Sensor

Inertial measurement unit sensor, or IMU sensor is the main component of inertial guidance system that are used in watercraft or air space. An IMU sensor works by sensing motion and determining the acceleration rate, rotational speed, including roll, pitch and yaw as shown in Figure 2.5. The data from the IMU sensor is then transferred into a computer to calculate the current position and speed, also known as initial speed and position according to D. Hazry, M. Sofian and A. Z. Azfar [8].

In order to know the orientation and heading of the underwater vehicle in 3 axis, the IMU sensor is utilized which consists of accelerometer, gyroscope and magnetometer. According to T. Mau and J. Mahoney [9], the advantages and disadvantages of each type of sensor in IMU are explained as shown in the Table 2.2. For accelerometer, it can determine the vehicle's orientation by outputting acceleration readings on all three axis with respect to the gravitational force. Any external forces, including disturbances or vibrations will affect the measurement since it is used to measure the gravitational acceleration. In order to determine the vehicle's orientation accurately, it is not enough to use the accelerometer exclusively.

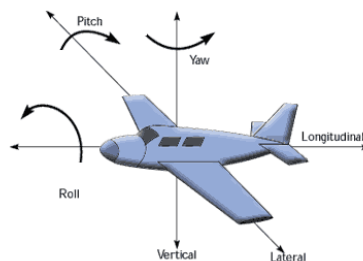


Figure 2.5: An IMU sensor determine an vehicle's orientation

The sensitivity to external forces of the accelerometer can be compensated by utilizing a three-axis gyroscope. The outputs data of the gyroscope are in the angular velocity (degree per second) form on all axis. Since this device measures the angles changes over time, it is not prone to linear disturbances like the accelerometer. Therefore, it is more accurate on a smaller scale. However, gyroscopes have two main disadvantages: gyroscopic bias and drift over time. Therefore, accelerometer often used as a reference to compensate the gyroscopic drift since it will not drift over time.

These two sensors are useful to determine the vehicle's orientation with respect to x and y axis, but they do not provide enough information to measure the heading (rotation about the z -axis) of the vehicle. Therefore, 3 axis magnetometer is used to measure magnetic field strength and heading on all three axis. However, this sensor prone to magnetic distortions, which are divided into hard iron and soft iron biases. In order to compensate the disadvantages of all the three sensors, the data are filtered and fused to combine the advantages of each sensor to obtain accurate positioning data.

Table 2.2: The advantages and disadvantages of each type sensor in IMU

Sensor	Advantages	Disadvantages
Accelerometer	- Stable over time	- Sensitive to vibrations/disturbances
Gyroscope	- Not prone to linear disturbances - Accurate readings over short period	- Values drift over time - Gyroscopic bias
Magnetometer	- Stable readings	- Prone to magnetic distortion - Hard-iron bias - Soft-iron bias

The ADXL345 accelerometer produces three acceleration analogue signal to determine the static and dynamic acceleration from vibration, shock and motion. The basic operation principle of accelerometer is the displacement of a small proof mass etched into the silicon surface of the integrated circuit and suspended by small beams. By applying the Newton's second motion law, $F = ma$, where force is developed to displace the mass. From the external accelerations influence, the proof mass is deflected from its neutral position. The deflection of the proof mass is measured in analogue or digital values.

The L3G4200D gyroscope generates three analogue signals of angular rate during rotation movements. Its working principle likes a disc or spinning wheel which the axis of rotation is free to assume any orientation. Its axis orientation is unaffected by tilting or mounting rotation and according to the conservation of angular momentum.

The journal [1, 2, 6] discussed the 5 DOF IMU sensor as shown in Figure 2.6 is used for the analysis movement of UUV. IMU sensor is a complete inertial system with the combination of gyroscopes and accelerometer sometimes also magnetometers. The IMU circuitry board is now available with cheaper, smaller and faster components by Cytron Company. This IMU sensor is commonly used in the control system, robotics, vehicle navigation, motion sensing and etc. This IMU sensor is the combination of gyroscope IDG500 and accelerometer ADXL335 sensors which enables researchers to measure roll, pitch, and tilt measurements at their projects easily according to G. Antonelli [10]. This IMU board uses a standard 0.1" footprint and consists of all the outputs of gyroscope IDG500 and accelerometer ADXL335. The board is fully assembled, tested and calibrated. The IDG500 is an integrated two axis gyroscope and patented with MEMS technology according to A. M. Plotnik and S. M. Rock [11].

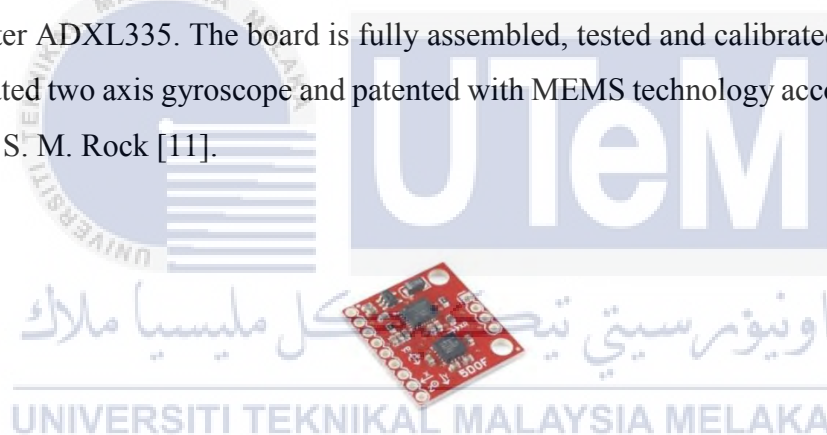


Figure 2.6: 5 DOF IMU sensor

Another journal from J. Busquets, J. V. Busquets, A. Perles, R. Mercado, R. Saez, J. J. Serrano, F. Albentosa and J. Gilabert [7] presents an IMU sensor, GY-80 based on MEMS technology as shown in Figure 2.7, which is the combination of gyroscope, accelerometer and magnetometer. All of them integrated into one to provide 9 DOF plus 1 DOF of barometer. This small IMU board provides three DOF to ADXL345 accelerometer, L3G4200D gyroscope and HMC5883 magnetic compass. By combining all of the sensors, the IMU board can be used to measure rotational speed, acceleration, magnetic field and atmospheric pressure. The features for each sensor are L3G4200D gyroscope 300°/s, ADXL345 accelerometer 13 bits and +/- 16g, and finally HMC5883 magnetic compass 7 mG and +/- 4 Gauss. Table 2.3 shows the sensor descriptions of 10 DOF IMU.

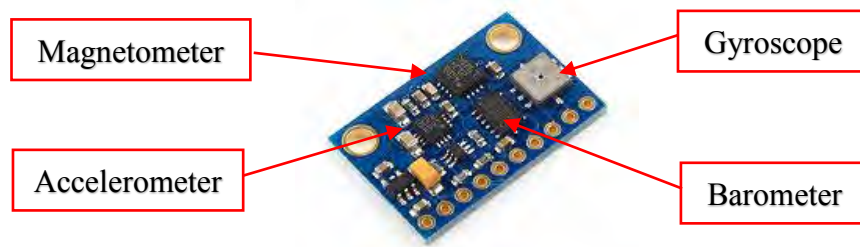


Figure 2.7: 10 DOF IMU sensor

Table 2.3: Sensor descriptions of 10 DOF IMU

Sensors of GY-80	Function
Accelerometer	Measures acceleration. Can be used to sense linear motion, vibration, and infer orientation and force.
Gyroscope	Measures angular rotation which can be used to infer orientation.
Magnetometer	Measures magnetic fields strength and heading degree. Can be used as a compass to determine bearing.
Barometer	Measures atmospheric pressure.
Thermometer	Measures temperature.

اونيور سیتی تکنیکل ملیسیا ملاک

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

Table 2.4: Comparison between IMU sensors

Journal No.	Type of IMU	Number of DOF	Type of sensors
[1, 2, 6]	5 DOF IMU	5	ADXL335 accelerometer and IDG500 gyroscope
[7]	10 DOF IMU, GY-80	10	ADXL345 accelerometer, L3G4200D gyroscope, HMC5883 magnetic compass, BMP180 barometer and thermometer

2.3.2 Leakage Sensor

Leakage sensor also called moisture sensor as shown in Figure 2.8, usually used to detect the humidity of the soil.

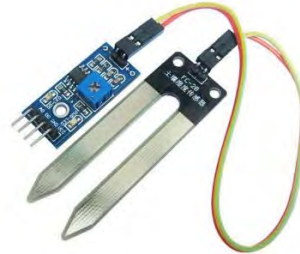


Figure 2.8: Leakage sensor

According to J. Busquets, J. V. Busquets, A. Perles, R. Mercado, R. Saez, J. J. Serrano, F. Albentosa and J. Gilabert [7], leakage sensor YL-69 with pin configuration as shown in Table 2.5 is used to detect water leakage inside the integrated sensor casing. If there is a water leakage inside the casing, a signal will be given. This sensor has a potentiometer for sensitivity adjustment of the digital output (D0), a power LED and a digital output LED. The output can be a digital signal (D0) LOW or HIGH, depending on the presence of water. If the presence of water exceeds a certain predefined threshold value, the outputs voltage becomes LOW, otherwise it outputs HIGH. The threshold value of the digital signal can be adjusted by using the potentiometer.

Table 2.5: Pins configuration of leakage sensor

Pin	Description
Vcc	5V
GND	Ground
A0	Analog pins
D0	Digital pins

2.3.3 Pressure Sensor/Depth sensor

Pressure sensor works as a transducer to generate a signal which is usually used to determine the pressure and also monitoring purpose in daily life. It also can be used to indirectly measure different kinds of parameter such as altitude, water level, gas flow and speed. In underwater industry, pressure sensor also known as depth sensor.

According to M.S.M Aras, S.S Abdullah, S.S Shafei [23], barometer and pressure sensor MPX series are used to investigate and evaluate low depth sensor system for underwater vehicle. There are two types of low cost pressure sensor system are designed and proposed for underwater vehicle. The underwater vehicle need depth sensor system to determine the pressure in underwater in order to determine the depth of the vehicle and prevent the vehicle from buckling.

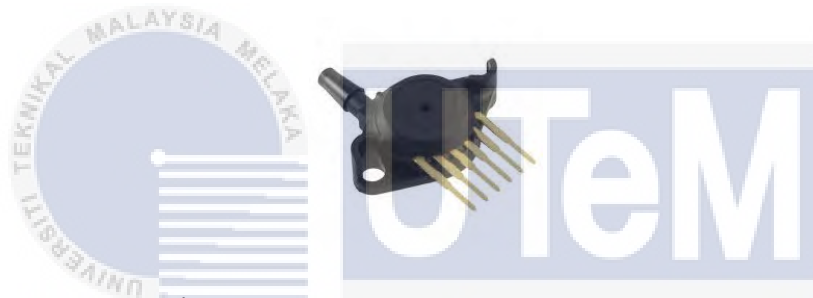


Figure 2.9: Pressure sensor MPX5700AP

According to journal [1, 2], an integrated sensor which consists of pressure sensor MPX5700 series as shown in Figure 2.9 is designed to sense the absolute air pressure in order to control the ROV engine. This pressure sensor is also used to maintain ROV stationary position at the desired depth. For this project, MPX5700 are used as to measure the depth. MPX5700 series is piezoresistive transducer that state of monolithic silicon pressure sensor. This single element transducer combines by micromachining techniques, thin-film metallization and bipolar processing to provide an accurate and high level analog output signal that is proportional to the applied pressure.

Table 2.6: Pin configuration of MPX series

Unibody package pin numbers			
1	V_{out}	4	N/C
2	GND	5	N/C
3	V_s	6	N/C

2.4 Theory

2.4.1 Orientation Angle

According to M.S.M Aras, M.F. Basar, S.S. Abdullah, F.A. Azis and F. A. Ali [6], the raw data unit from the IMU sensor can be read in microcontroller. In order to plot the output graph, these raw data are converted into angular velocity and acceleration. There have a few equations as shown in Equations (2.1), (2.2) and (2.3) to obtain the desired output, where x , y and z are the raw data from the IMU sensor in an UUV as shown in Figure 2.10.

$$X_1 = x^2 \quad (2.1)$$

$$Y_1 = y^2 \quad (2.2)$$

$$Z_1 = z^2 \quad (2.3)$$

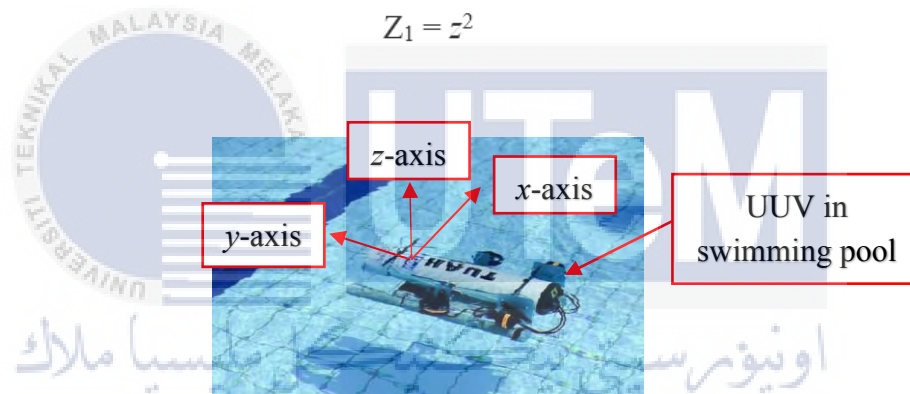


Figure 2.10: x , y and z axis from the IMU sensor in an UUV

From the Equations (2.1), (2.2) and (2.3), the distance of x , y and z axis can be calculated by applying the Pythagoras's Theorem to get the Equations (2.4), (2.5) and (2.6).

$$X_2 = \sqrt{Y_1^2 + Z_1^2} \quad (2.4)$$

$$Y_2 = \sqrt{X_1^2 + Z_1^2} \quad (2.5)$$

$$Z_2 = \sqrt{X_1^2 + Y_1^2} \quad (2.6)$$

By applying trigonometry formula, angle of θ_x , θ_y and θ_z with respect to the x , y and z axis can be obtained as shown in Equations (2.7), (2.8) and (2.9).

$$\Theta_x = \text{atan} (X_1/X_2) \quad (2.7)$$

$$\Theta_y = \text{atan} (Y_1/Y_2) \quad (2.8)$$

$$\Theta_z = \text{atan} (Z_1/Z_2) \quad (2.9)$$

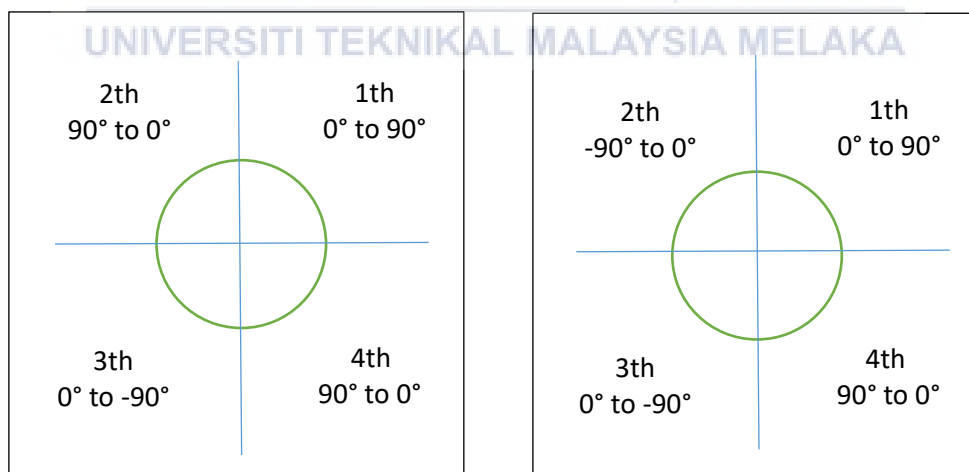
The theta values from Equations (2.7), (2.8) and (2.9) are in radian unit and can be converted into a degree value as shown in Equation (2.10).

$$x^{\circ} = \frac{\theta x \times 180}{\pi} \quad (2.10)$$

For orientation angle of IMU, the output degree is not from 0 to 360 degrees per circle. The orientation angle at x and y axis is different with the z -axis. The orientation angle at x , y and z axis are shown as Table 2.7 and Figure 2.11.

Table 2.7: Orientation angle at x , y and z axis

Orientation angle Quadrant	Orientation angle at X and Y-axis	Orientation angle at Z-axis
First quadrant	0 to 90°	0 to 90°
Second quadrant	90 to 0°	-90 to 0°
Third quadrant	0 to -90°	0 to -90°
Fourth quadrant	90 to 0°	90 to 0°



(a) Orientation angle at x and y axis

(b) Orientation angle at z -axis

Figure 2.11: The orientation output angle of IMU sensor at x , y and z axis

2.4.2 Pressure

The water pressure is difference with the definition of pressure. This is because water pressure need consider water depth and its density. The Equation (2.11) for pressure measurement is shown as following:

$$P = pgh \quad (2.11)$$

where:

P = pressure,
 p = water density,
 g = gravity,
 h = depth of the fluid

Since p and g are constants, the fluid pressure is directly proportional to the depth of the fluid. The relationship between the depth and pressure as shown in Figure 2.12. The water pressure is increases as the water depth increases.

The density of lake water is higher than the density of pool water due to the presence of impurities in lake water. The pool water is free of impurities and clean. Therefore, it gives a lower density and low pressure if compared to lake water. The density for pure water and sea water are 1000 kg/m^3 and 1027 kg/m^3 respectively.

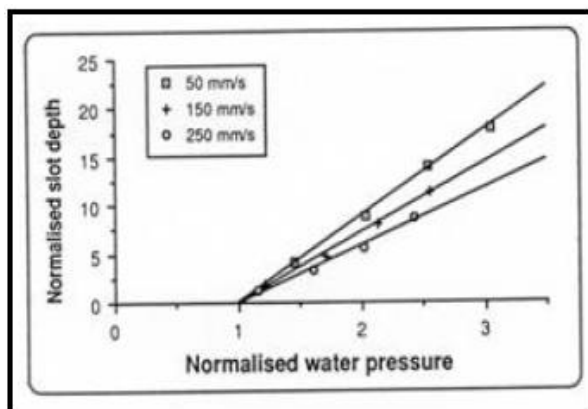


Figure 2.12: The relationship between the depth and water pressure

2.5 Sensors Interface with Microcontroller

The microcontroller of Arduino Mega2560 is used to interface with sensors and also used to integrate with MATLAB real-time simulation. To get the real-time data from sensor, each of the sensing unit will be programmed by a microcontroller. The real-time data from the integrated sensor are communicated with serial chart and processing software to generate the output graph and UUV's real-time 3D animation. From the output graph, the value of rotational speed, translational acceleration, atmospheric pressure, magnetic field and also water intrusion in the pressure hull can be obtained. One clear benefit in this project is more cost and time saving if compared the method of sensing unit interfacing with the Wi-Fi module, GPS + IMU navigation system, etc. Besides that, the flexibility of the system can be improved when dealing with complex control system [6, 15, 16].

2.6 The Comparison of Previous Studies

There are many studies and researches related to the integrated sensor from the various countries. The researchers carried out their integrated sensor research to improve the performance and technology of sensors in underwater application. Therefore, the previous studies are discussed to compare with the current studies.

The following paper that was related to the development and analysis of integrated sensor was "Analysis of Movement for Unmanned Underwater using a Low Cost Integrated Sensor," (M. S. M. Aras, S. S. Abdullah, A. F. N. A. Rahman, M. F. Basar, A. M. Kassim, H. I. Jaafar, 2015) [1]. The integrated sensor in this project is the combination of 5-DOF IMU sensor, pressure sensor MPX5700, temperature sensor, digital compass, and leakage sensor. This integrated sensor is developed to control the ROV movement and maintain its underwater position, known as station keeping. The integrated sensor interfaced with NI DAQ card and all the data can be obtained from LabView software. Figure 2.13 shows the LabView real-time output result of gyroscope by using NI DAQ card interfacing.

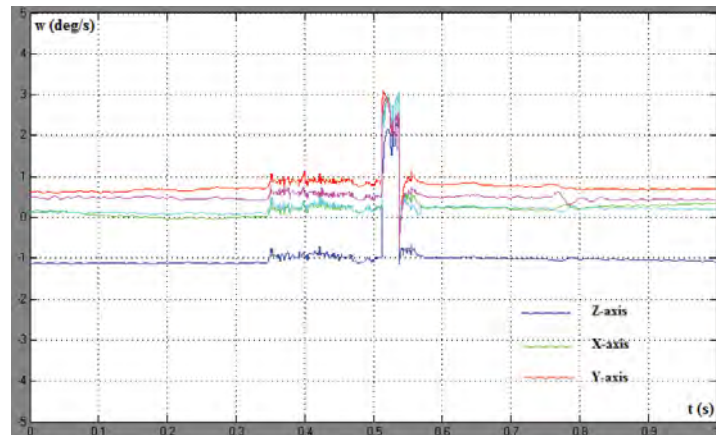


Figure 2.13: LabView real-time output graph of gyroscope

The previous paper that regarded to integrated sensor was “A Low Cost 4 DOF Remotely Operated Underwater Vehicle Integrated with IMU and Pressure Sensor,” (M. S. M. Aras, F. A. Azis, M. N. Othman and S. S. Abdullah, 2012) [2]. In this paper, the integrated sensor is the integration of pressure sensor, IMU, and digital compass. The intelligent controller (IC) is developed for real time simulation and data input by using Visual Basic (VB) Platform or MATLAB. A Graphical User Interface (GUI) of MATLAB is designed for data input, simulation and data storage. Figure 2.14 shows the LabView real-time output result of gyroscope by using Microbox interfacing.

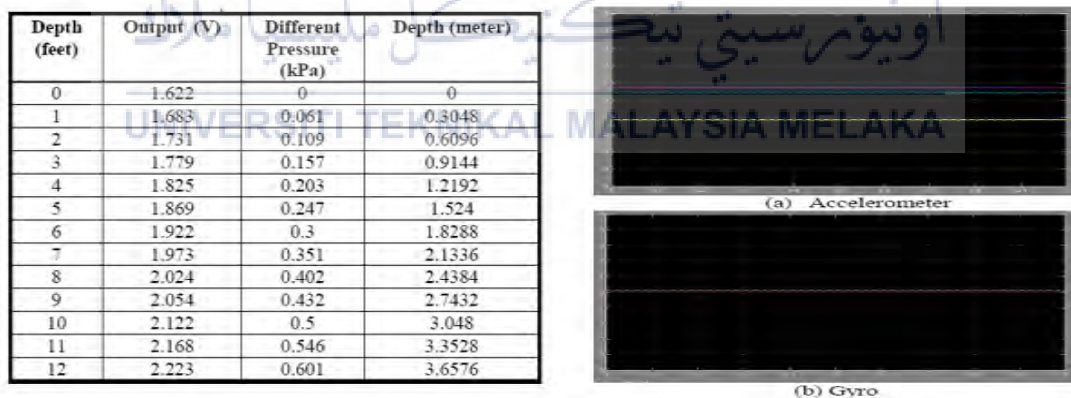


Figure 2.14: Real time output result for pressure sensor and IMU sensor

According to (J. Busquets, J. V. Busquets, A. Perles, R. Mercado, R. Saez, J. J. Serrano, F. Albentosa and J. Gilabert, 2014) [7], the paper with the title of “Communication Challenges for Dual Configuration of ASV and AUV in Twinned Coordinated Navigation,” used the microcontroller board (Arduino Mega2560) as MCU to interface with the sensors and actuators. The sensors that are used to interface are IMU sensor, pressure sensor and moisture sensor.

According to (G. A. Ramadass, N. Vedachalam, V. Balanagajyothi, R. Ramesh, M. A. Atmanand, 2013) [15], a paper with the title of “A Modelling Tool for Sensor Selection for Inertial Navigation System used in Underwater Vehicles,” presented the development of mathematical model of accelerometer and gyroscope used in inertial measurement system and position estimation by using MATLAB software to select the required aiding sensors for the navigation system.

There was a paper with the title of “Design Aspects of An Open Platform for Underwater Robotics Experimental Research,” written by (C. R. Rocha, R. M. Brancoy, L. A. D. Cruz, M. V. Schollx, M. M. Cezarx and Felipe D, 2014) [16]. In this paper, the sensors and actuator are interfaced with Arduino Nano board. Pressure sensor, IMU sensor, temperature sensor and leakage sensor are used to monitor the electronics operational condition and MMCS [16].

According to (Pakpong Jantapremjit, 2011) [17] with the title of “Design and Development of ROV,” the pressure hull of the ROV contains the navigation system with IMU sensor, depth sensor and compass interfacing with microcontroller as shown in Figure 2.15. The microcontroller used in this project is Arduino Mega 1280 board. The navigation sensors comprise an IMU sensor, a pressure sensor and an electronic compass. The rotational rates and accelerations are measured by using a 6 DOF IMU sensor. The ROV depth can be measured from the pressure sensor MPX4250. It provides a pressure range from 20 kPa to 250 kPa and measure depths up to 15 m. For the electronic compass, it determines heading angle of vehicle with precision of 0.5 degrees.

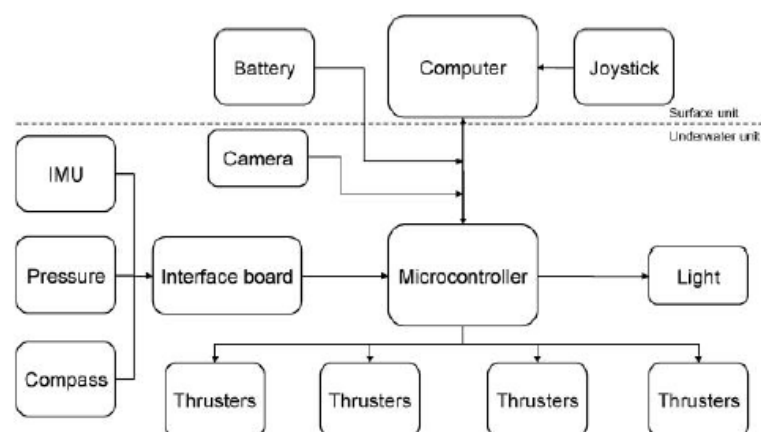


Figure 2.15: The ROV's electronic system published by Pakpong Jantapremjit, 2011

According to (I. N. Jleta and O. A. Taleb, 2015) [18], a study with title of “Land Vehicle Navigation System Based Low Cost Inertial Sensor,” presented the development of a simple land vehicle inertial navigation system by using MEMS sensor. The system used 6 DOF of IMU sensor (3 axis accelerometer and 3 gyroscope) to detect the orientation, velocity and position. Arduino Mega2560 is selected as data acquisition board to interface with the IMU sensor an MATLAB software. INS (Inertial Navigation System) is created in MATLAB/Simulink and the system run in the real-time mode.

Besides that, the research with title of “Inertial Sensor Self-Calibration in a Visually-Aided Navigation Approach for a Micro-AUV,” (F. Bonin-Font, M. Massot-Campos, P. L. Negre-Carrasco, G. Oliver-Codina and J. P. Beltran, 2015) [19] used 6-DOF IMU sensor and pressure sensor. The sensing data are adjusted and corrected by using error state Kalman filter or MESKF. LibViso2 software is used to get the output graph results. Figure 2.16 shows the output graph results from LibViso2 software.

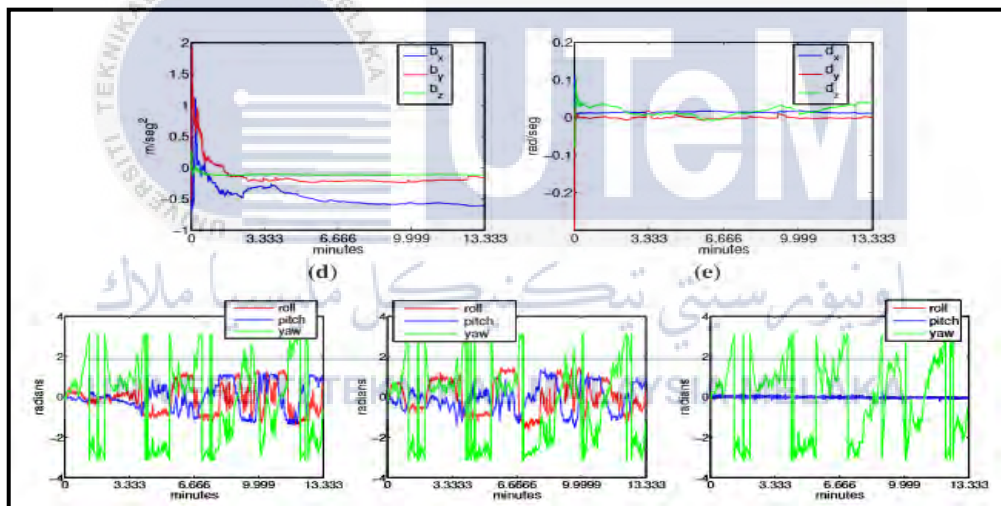


Figure 2.16: The output graph results by using LibViso2 software

According to (C. Stal, G. Deruyter and M. Paelinck, 2015) [20], the orientation parameters of the platform are calculated based on the Inertial Navigation System (INS). This system consists of an IMU sensor and a processor for the data manipulation obtained from the IMU sensor. The data include measurements from accelerometer, gyroscopes and magnetometers. The underwater measurements are performed on a small MEMS based motion sensor with integrated pressure (depth) and temperature sensor.

Table 2.8: The comparison of previous studies to current studies

Journal No.	Type of sensors used	Micro-controller	Software	Remarks
[2]	IMU and pressure sensor	Microbox 2000	MATLAB	The intelligent controller (IC) is developed for real time simulation and data input.
[1]	5 DOF IMU sensor, pressure sensor MPX 5700, temperature sensor, digital compass and leakage sensor	-	LabView	The integrated sensor is used to control the ROV movement, maintain its underwater position and interface with NI DAQ card to obtain the real time output graph.
[7]	10 DOF IMU sensor, pressure sensor moisture sensor	Arduino Mega 2560	C++, Arduino proprietary and Robot Operating System (ROS)	Low power acoustic modem is applied on the low cost vehicle.
[15]	Acclerometer and gyroscope	-	MATLAB	The integrated sensor used in inertial measurement system and position estimation
[16]	IMU sensor, pressure sensor, temperature sensor and leakage sensor	Arduino Nano	Python	The integrated sensor is used to monitor the electronics operational condition and MMCS (Mission Monitoring/Control System).
[17]	6 DOF IMU sensor, pressure sensor MPX 4250 and electronic compass	Arduino Mega 1280	Arduino programming language in JAVA	The integrated sensor is used to determine the depth and heading angle of ROV.
[18]	6 DOF IMU sensor	Arduino Mega 2560	MATLAB	The sensor is used to detect the orientation, velocity and position. INS (Inertial Navigation System) is created in MATLAB and run the system in real time mode.
[19]	6 DOF IMU sensor and pressure sensor	PC104	LabViso2	The sensing data are corrected and adjusted by using Kalman filter.
[20]	IMU sensor, pressure sensor and temperature sensor	Arduino	-	The underwater measurements are performed based on integrated sensor.

2.7 Summary

Based on all of the paper studies in this chapter, the main differences between these papers are the types of sensors used to develop integrated sensor and methodology. The main similarity from all of the previous studies is the IMU sensor used as the main sensor of the system. This sensor always used to develop an integrated sensor because this board is a complete inertial system with the combination of different kinds of sensors. Therefore, it is widely used in the precision instrumentation, vehicle navigation system, robotics, platform stabilization and etc. There are many methodologies used for simulation, but the LabView or MATLAB/Simulink are commonly used for real-time simulation, data input and data storage.



CHAPTER 3

METHODOLOGY

3.1 Introduction

In this chapter, there is some methodology to be discussed in order to test the functionality and performance of IMU sensor, depth sensor, magnetometer and leakage sensor. The overall process flow of the methodology of this project as shown in Figure 3.1. Basically, the methodology of this project consists of three main parts: hardware development, software development and hardware testing. In hardware development, the suitable microcontroller, sensors and casing are selected. The circuit is designed and the functionality each of the sensor is tested with microcontroller before all the sensors are assembled into the casing to make an integrated sensor. After that, all the sensors are assembled into a casing to make an integrated sensor. In software development, the coding of each sensor is written and loaded into the microcontroller of integrated sensor. Lastly, for the hardware testing and data collection, the integrated sensor is installed into an UUV to get the real-time data in the pool and lab tank. The output data from the sensor will be processed with the written programming and transferred into the computer by using USB communication. The data from the sensors are collected to use as input for serial monitor and processed in MATLAB real-time simulation to generate the real-time output graph. As a result, the sensors continuously provide real time data to generate the output data graph. From the output data and graph, the performance of the sensor can be analysed and also can be analytically the movement of UUV.

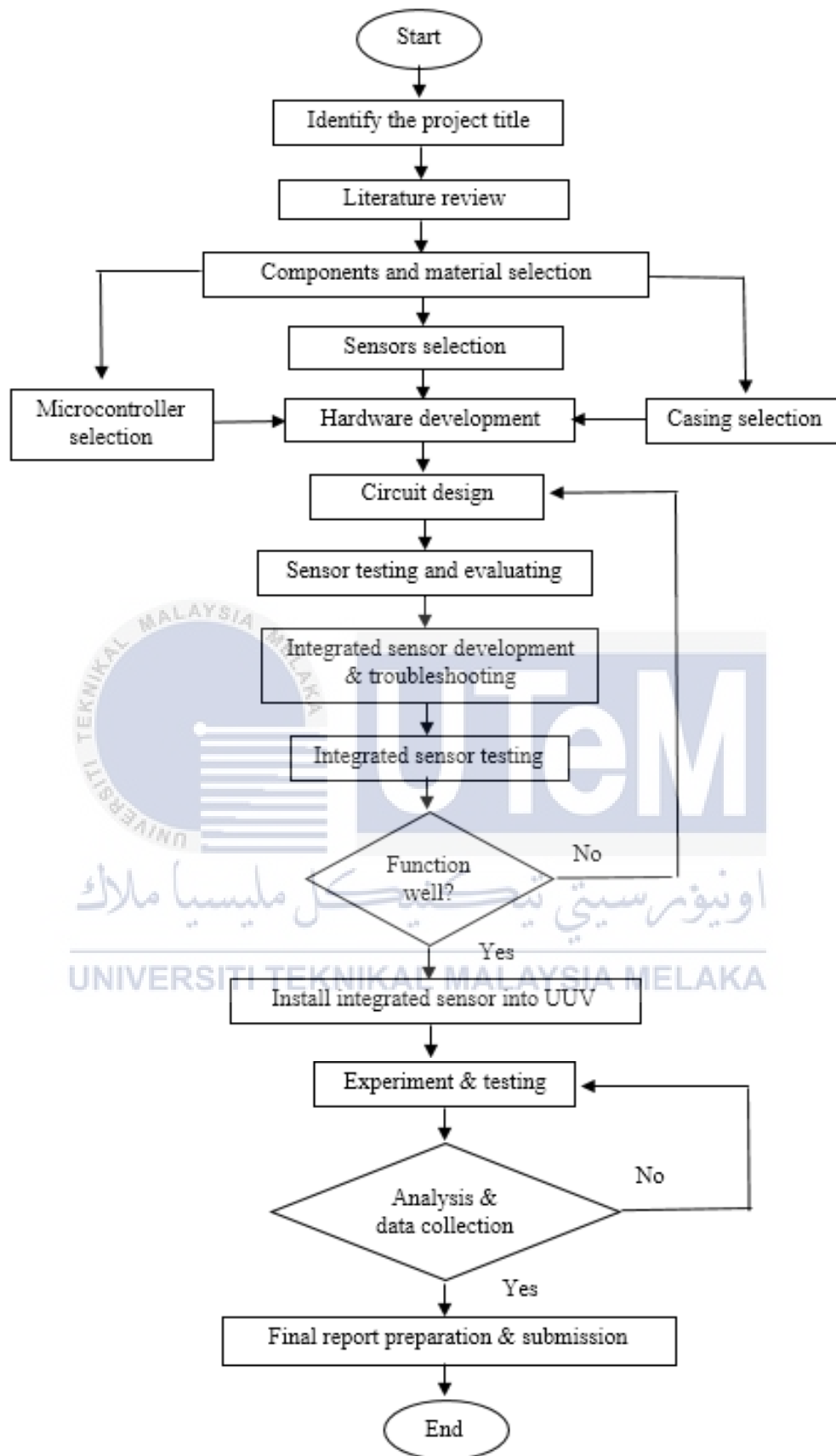


Figure 3.1: The overall process flow of the methodology

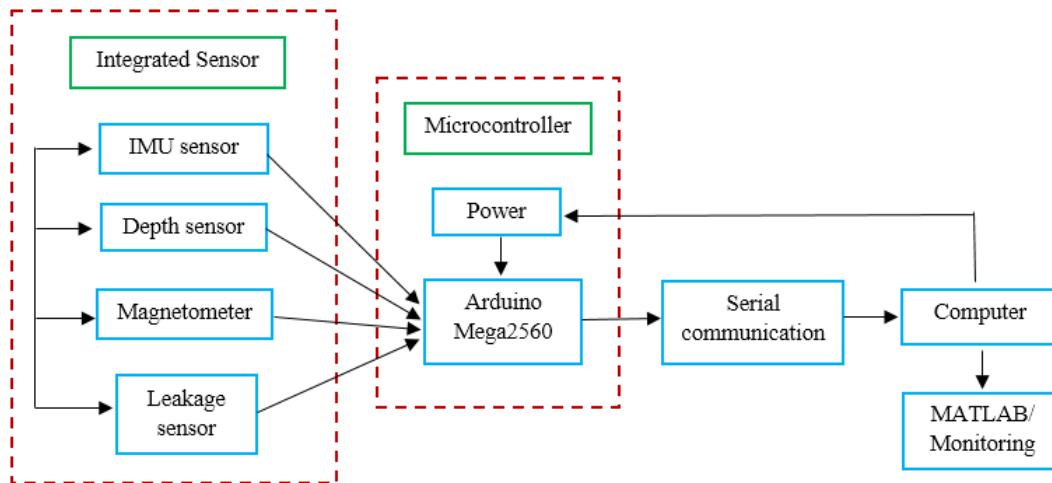


Figure 3.2: Illustration of the system

The whole system has been constructed as shown in Figure 3.2. The integrated sensor consists of four different kinds of sensors: IMU sensor, depth sensor, magnetometer and leakage sensor which interfaced with microcontroller, Arduino Mega2560. The data from the sensors are collected and processed with the help of the written microcontroller program and as inputs for serial monitor via USB communication. The MATLAB program gets the data from serial port and provides real-time output graph for performance analysis. The power for the microcontroller and sensors is provided by computer via USB cable. The performances and functionality of the sensors are monitored via serial communication.

3.2 Hardware Description



3.2.1 Microcontroller

Arduino is an open-source computing platform that created microcontroller board for building digital devices and interactive objects to control the physical devices.

This platform has been chosen in this project because there are many sensor libraries available and provides an IDE based on a simple programming language. This 8-bit microcontroller board is sufficient in this project because there is no image data or sonar image processing. Furthermore, Atmel architecture is a reliable platform which is able to provide the required reliability in UUV's operation. In addition, the cost of Arduino microcontroller is much lower than the other microcontroller.

In choosing the microcontroller of this project, there are a few aspects that need to be considered as listed on Table 3.1. Table 3.1 listed out the key differences and similarities between Arduino Uno and Arduino Mega2560 for further understanding. From the Table 3.1, the main differences between these two microcontrollers are the number of the input/output pins and flash memory. The objective of this project is to design an integrated sensor with various sensors and low estimation error. In order to achieve that, the number of the input/output pins must be sufficient for all the sensors and the flash memory must be higher for better performance. Therefore, the final decision of the microcontroller used in this project is Arduino Mega2560 which is more suitable for the development of the integrated sensor and has better specifications than Arduino UNO.




Table 3.1: Comparison between the microcontrollers

Type of microcontroller	Arduino UNO	Arduino Mega2560
Aspects		
Operating voltage	5V	5V
Digital I/O pins	14	54
Analog input pins	6	16
Flash memory	32 KB	256 KB
SRAM	2KB	2KB
Clock speed	16 MHz	16 MHz

3.2.2 IMU Sensor

In this project, IMU sensor is an important sensor for the integrated sensor development and movement analysis of UUV. There are many types of IMU sensor available in the market with different number of DOF and sensors. Basically, there are two aspects that need to be considered in choosing the IMU sensor as listed in Table 3.2 which are the number of DOF, number of sensors and type of signal. DOF is the number of independent parameters that used to define the configuration of UUV. It is important to determine the state of physical system and movement analysis of the UUV's body.

Table 3.2: Comparison between the IMU sensors

Type of IMU sensor	5 DOF IMU	9 DOF IMU	10 DOF IMU
Aspects			
Number of DOF	5	9	10
Number of sensor	2	3	5
Type of signal	Analog	Digital	Digital
Type of sensor	Accelerometer ADXL345 and gyroscope IDG500	Accelerometer ADXL345, gyroscope ITG3200 and magnetic compass HMC5883	Accelerometer ADXL345, gyroscope L3G4200D, magnetic compass HMC5883, barometer BMP180 and thermometer

After considering in every aspect, the 5 DOF IMU sensor is chosen in this project although it has the lowest number of DOF and sensor. It has only 5 DOF and integrates only 2 different types of sensors into a single small package. However, this sensor is very easy to use and programmed because the type of signals received are in analog. Therefore, it is easy to study its performance and implemented in UUV's movement analysis. The interface description of 5 DOF IMU as shown in Table 3.3.

Table 3.3: Interface description of 5 DOF IMU

Pin No.	Symbol	Descriptions
1	Vcc	5V power supply
2	GND	Supply ground
3	X-ACC	X-axis of accelerometer
4	Y-ACC	Y-axis of accelerometer
5	Z-ACC	Z-axis of accelerometer
8	X-RATE	X-axis of gyroscope
9	Y-RATE	Y-axis of gyroscope

3.3 Circuit Design

There are four sensors used in this project which are 5 DOF IMU sensor, MPX5700 depth sensor, magnetometer and leakage sensor to design and develop an integrated sensor for UUV. All the sensors are interfaced with microcontroller, Arduino Mega2560 to get the output data and results for performance analysis. There are many procedures to develop an integrated sensor from these different kinds of sensors. To ensure that the hardware is successfully implemented, all the procedures are required to be followed.

Figure 3.3 shows the recommended decoupling circuit to interface with the configuration pins of the depth sensor. Firstly, the schematic circuit diagram of MPX5700 depth sensor is designed and soldered based on the schematic circuit diagram in Figure 3.3. Proper decoupling of the power supply of depth sensor is recommended.

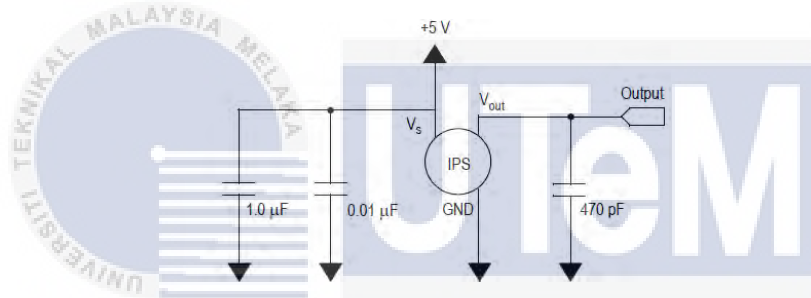


Figure 3.3: Schematic circuit diagram of depth sensor, MPX5700AP

After the circuit board is soldered, the circuit connection is tested by using multimeter to check its continuity before it is interfaced with microcontroller. Make sure the circuit is soldered properly and connection is connected properly to avoid short circuit. Figure 3.4 shows the complete soldered depth sensor with decoupling circuit board. If the circuit is in the good condition, the depth sensor with its decoupling circuit board is placed into the casing and sealed.

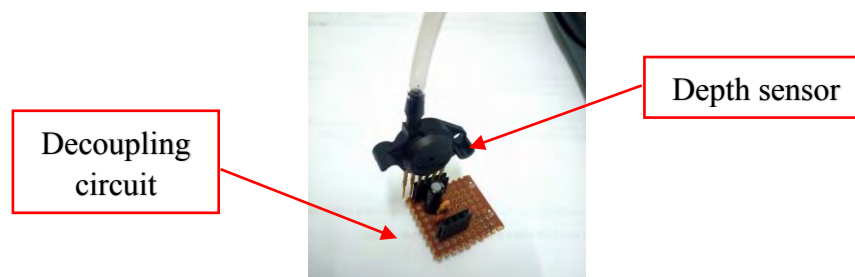


Figure 3.4: Complete soldered depth sensor with decoupling circuit

3.4 Design and Assembly of Integrated Sensor

The integrated sensor casing is used to store all the sensors inside to form an integrated sensor. Figure 3.5 shows the integrated sensor casing used in this project. The casing is made of transparent polycarbonate to allow the user to observe through inside the casing. It is waterproof and sealed to prevent water entering into the casing. The polycarbonate plastic is high impact strength, dimensional stability and mechanical performance to withstand the high underwater pressure. After all the sensors are assembled in this casing, the casing will horizontally put inside the pressure hull of UUV. Table 3.4 shows the characteristic design of integrated sensor casing.

Table 3.4: Characteristics design of integrated sensor casing

Material	Polycarbonate plastic
Dimensions	180 x 110 x 70 mm
Weight	± 500 g
Shape	Rectangle



Figure 3.5: Integrated sensor casing

The casing is drilled to make 2 holes for the depth sensor and USB cable. All the components are placed inside the casing in a fixed position inside the casing as shown in Figure 3.6. In this integrated sensor, make sure the 5 DOF IMU sensor and magnetometer are horizontally placed inside the casing in a fixed position to prevent any inaccurate data taken during experiment. After that, every hole of the casing are sealed to avoid the water intrusion inside the casing that will brings damage to the components.

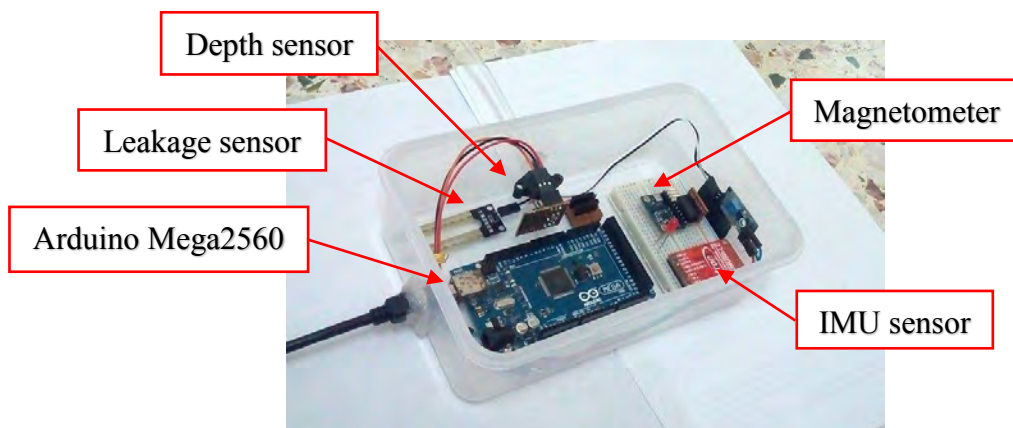


Figure 3.6: All the components in the casing

Each of the sensor has its own 5V and ground pin configuration respectively. Therefore, the ground pin configuration of each sensor is soldered on one line of strip board as shown in Figure 3.7. The same step is also performed to the 5V pin configuration. Make sure its continuity is checked by using multimeter.



Figure 3.7: 5V and ground pin configuration on one strip line of stripboard

Table 3.5 shows the pin configuration connection between the sensors and microcontroller, Arduino Mega2560. The sensors are connected with microcontroller according to the Table 3.5. The output signal pin of the depth sensor is connected to an analog pin (A0) of Arduino Mega2560. For 5 DOF IMU sensor, the 5 output signals (3 for accelerometer and 2 for gyroscope) are connected to 5 analog pins of Arduino Mega2560. For magnetometer, the 2 digital output signals, SDL (same data line) and SDA (same data address) are connected to digital pin 21 and 20 respectively. For the leakage sensor, its signal pin is connected to digital pin (D0).

Table 3.5: Pins configuration connection between electronic components and microcontroller

Sensors	Microcontroller, Arduino Mega2560
Depth sensor	
Vcc (5V)	5V
Ground	Ground
Output (signal)	A0
5 DOF IMU	
Vcc (5V)	5V
Ground	Ground
X – ACC (accelerometer)	A1
Y – ACC (accelerometer)	A2
Z – ACC (accelerometer)	A3
X – RATE (gyroscope)	A4
Y – RATE (gyroscope)	A5
Magnetometer	
Vcc (5V)	5V
Ground	Ground
SCL	Pin 21
SDA	Pin 20
Leakage sensor	
Vcc (5V)	5V
Ground	Ground
Output (signal)	A8
Blinking LED	
Anode	D6
Cathode	Ground

3.5 Experiment and Analysis

Before assembly all the sensors into the casing to make an integrated sensor, each of the sensor is tested with a microcontroller. The coding of each sensor is written and loaded into the microcontroller. The functionality of each sensor is tested in air and also in underwater to collect the data. The data from serial monitor and MATLAB real-time simulation of the sensors are collected and their performance are analysed.

3.5.1 Experiment 1: Orientation Movement Analysis of IMU Sensor

Objective:

To analyse the orientation movement performance of IMU sensor in UUV by using MATLAB real-time simulation.

Parameter:

Manipulated variable: Movement and position of the IMU sensor

Responding variable: Output graphs of IMU sensor

Equipment:

1. IMU sensor
2. Arduino Mega2560
3. USB cable
4. Station (PC)

Procedure:

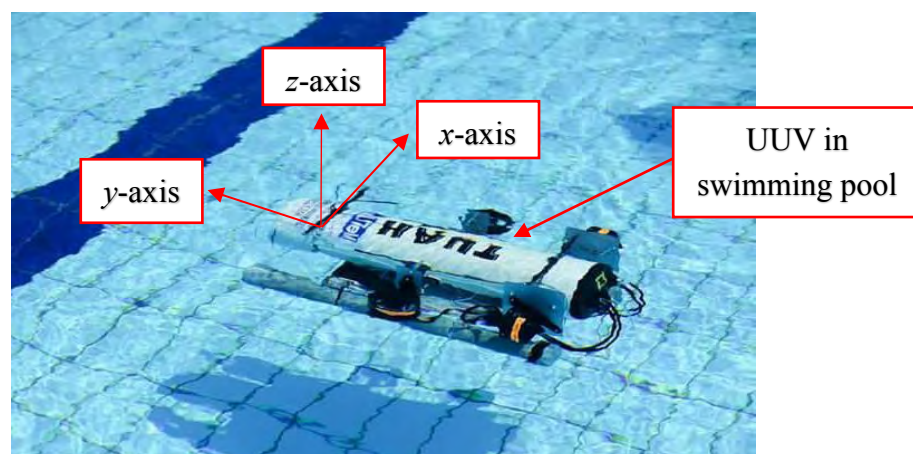


Figure 3.8: Set up of experiment 1

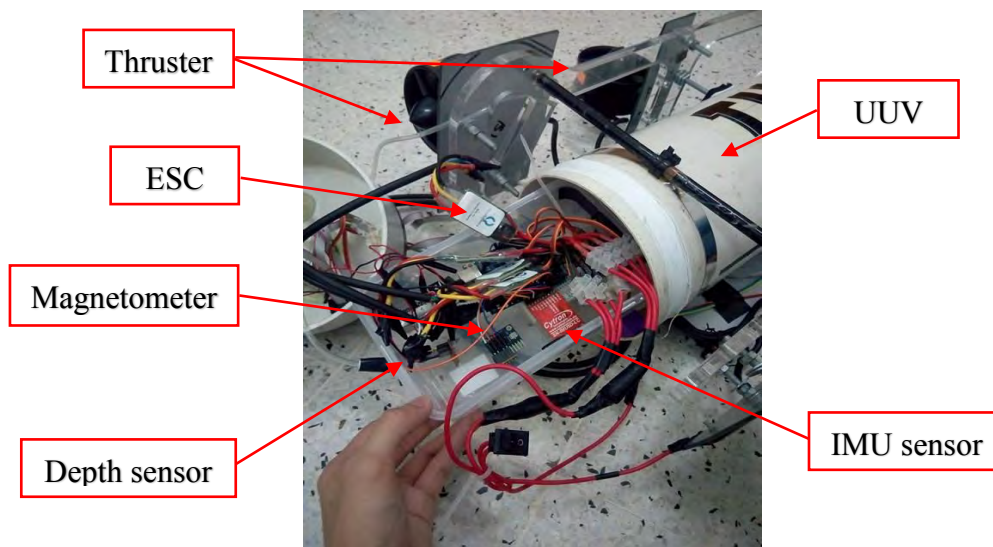


Figure 3.9: Electronic components in the integrated sensor casing

1. The experiment 1 is set up as shown in Figure 3.8.
2. The Arduino coding of accelerometer is prepared as shown in Appendix B and uploaded into microcontroller, Arduino Mega2560.
3. The IMU sensor is interfaced with microcontroller and then the microcontroller is interfaced with station (PC) via USB communication.
4. The integrated sensor casing with electronic components are installed into an UUV as shown in Figure 3.9. Make sure the IMU sensor is placed in a fixed position to avoid any inaccurate data.
5. The MATLAB real time simulation of IMU sensor is opened and run in 100 s to obtain the initial real-time output graph of accelerometer.
6. After the real-time simulation, the initial real-time output graph of accelerometer is saved and analysed.
7. The UUV is put on the water pool surface and move forward as shown in Figure 3.8.
8. The MATLAB real-time simulation of accelerometer is opened and run in 100 s. The movement of the UUV is observed during the real-time simulation.
9. The real-time output graph of accelerometer is saved and analysed.
10. The steps above are repeated to generate the output graph of gyroscope by changing the Arduino coding and MATLAB real-time simulation coding.

3.5.2 Experiment 2: To Determine the Consistency and Accuracy of Depth Sensor

Objective:

To determine the consistency and accuracy of depth sensor by determining the initial pressure and analog reading in MATLAB real-time simulation.

Parameter:

Manipulated variable: Time

Responding variable: Pressure or analog reading

Equipment:

1. Depth sensor with decoupling circuit
2. Arduino Uno
3. USB cable
4. Station (PC)

Procedure:

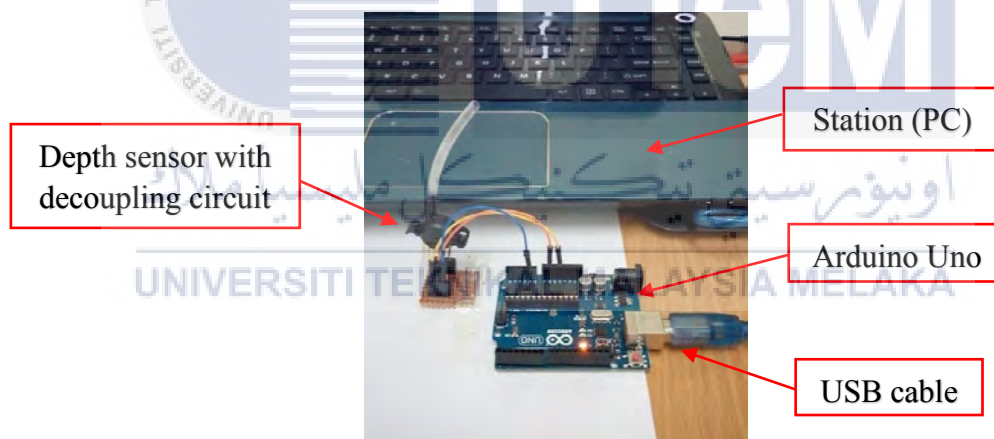


Figure 3.10: Set up of experiment 2

1. The experiment 2 is set up as shown in Figure 3.10.
2. Microcontroller, Arduino Mega2560 is uploaded with depth sensor coding as shown in Appendix B and then interfaced with depth sensor and station (PC).
3. The MATLAB real-time simulation coding of depth sensor as shown in Appendix B is opened and run.
4. After that, the real-time output graph of depth sensor is saved and analysed.
5. The steps above are repeated to generate real-time output graph of analog reading by changing the Arduino coding and MATLAB real time coding.

3.5.3 Experiment 3: Depth Sensor Performance Analysis in Air

Objective:

To analyse the performance of depth sensor by measuring the output signal voltage in air.

Parameter:

Manipulated variable: Pressure of air compressor

Responding variable: Output signal voltage of depth sensor

Equipment:

1. Depth sensor, MPX5700AP
2. Arduino Uno with 9V battery voltage supply
3. Station (PC)
4. Multimeter
5. 1 HP (horsepower) air compressor
6. Pressure regulator
7. Distributor

Procedure:

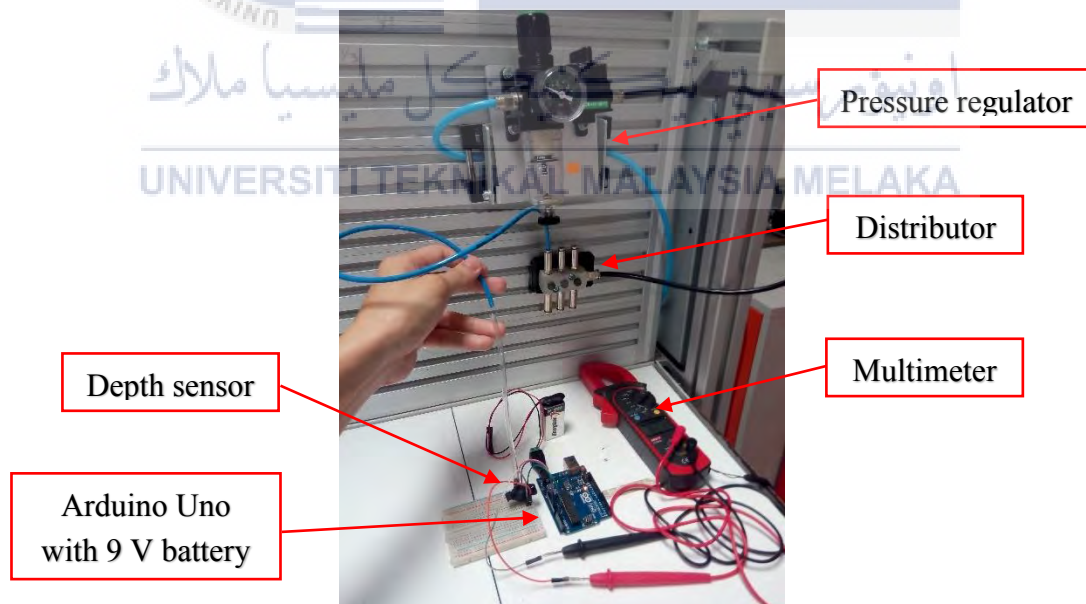


Figure 3.11: Set up of experiment 2

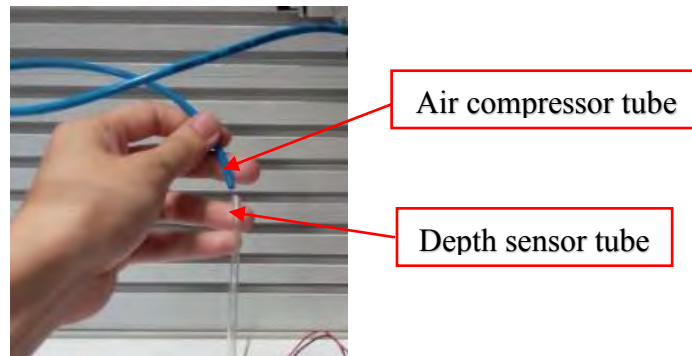


Figure 3.12: Air compressor tube slightly contact to the depth sensor tube

1. The experiment 3 is set up as shown in Figure 3.11.
2. The microcontroller, Arduino Uno is uploaded with depth sensor coding as shown in Appendix B and interfaced with depth sensor.
3. The 1 HP air compressor is set up and its pressure is adjusted to 1 bar via pressure regulator.
4. The pressure regulator is switched on.
5. The tube of the air compressor is slightly contact to the tube of the depth sensor as shown in Figure 3.12 to avoid the depth sensor is damaged by the high pressure from the air compressor.
6. The output signal voltage of the depth sensor is measured by using multimeter. The readings are taken three times to get more accurate data.
7. The steps above are repeated from the pressure range of 1 bar to 6 bar with each increment of 0.5 bar.
8. The results is plotted in the graph of output signal voltage (V) versus pressure (kPa).

3.5.4 Experiment 4: Depth Sensor Performance Analysis in Underwater

Objective:

To analyse the performance of depth sensor by measuring output signal voltage in underwater.

Parameter:

Manipulated variable: Water depth

Responding variable: Output signal voltage and pressure measured of depth sensor

Equipment:

1. Depth sensor with decoupling circuit
2. Arduino Mega2560
3. USB cable
4. Station (PC)
5. Integrated sensor casing
6. Multimeter
7. Measuring tape
8. Adjustable water pipe
9. Rectangular water tank

Procedures:

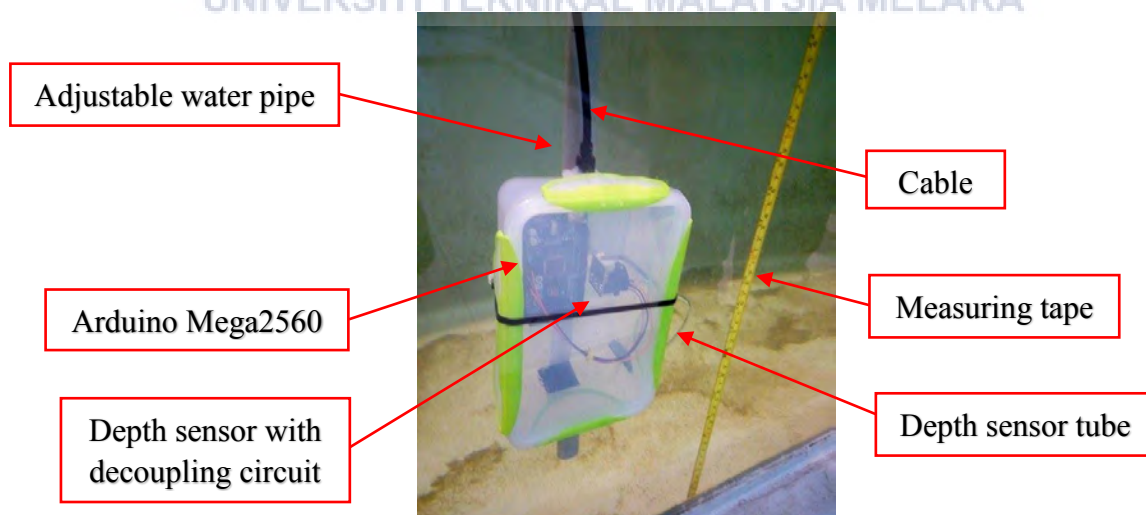


Figure 3.13: Set up of experiment 4

1. The experiment 4 is set up for underwater environment as shown in Figure 3.13.
2. The microcontroller, Arduino Mega2560 is uploaded with depth sensor coding as shown in Appendix B and interfaced with depth sensor.
3. The integrated sensor is put on the water surface of rectangular water pool.
4. The output signal voltage of depth sensor is measured by using multimeter. The readings are taken three times to get more accurate results. These readings are considered as the 0 cm of water depth since the casing is placed on the water surface.
5. The step 3 above is repeated from the water depth range from 10 cm to 100 cm with each increment of 10 cm.
6. The long and adjustable pipe is used to submerge the integrated sensor until 100 cm depth of water.
7. The results are plotted in the graph of output signal voltage (V) versus depth (cm) and pressure (kPa) versus water depth (cm).



3.5.5 Experiment 5: Depth Control of UUV

Objective:

To analyse the performance of depth sensor in depth control application of UUV in underwater.

Parameter:

Manipulated variable: Desired water depth of UUV

Responding variable: Actual water depth of UUV

Equipment:

1. Unmanned underwater vehicle (UUV) with depth sensor
2. Station (PC)
3. Measuring tape
4. Rectangular tap water tank

Procedures:

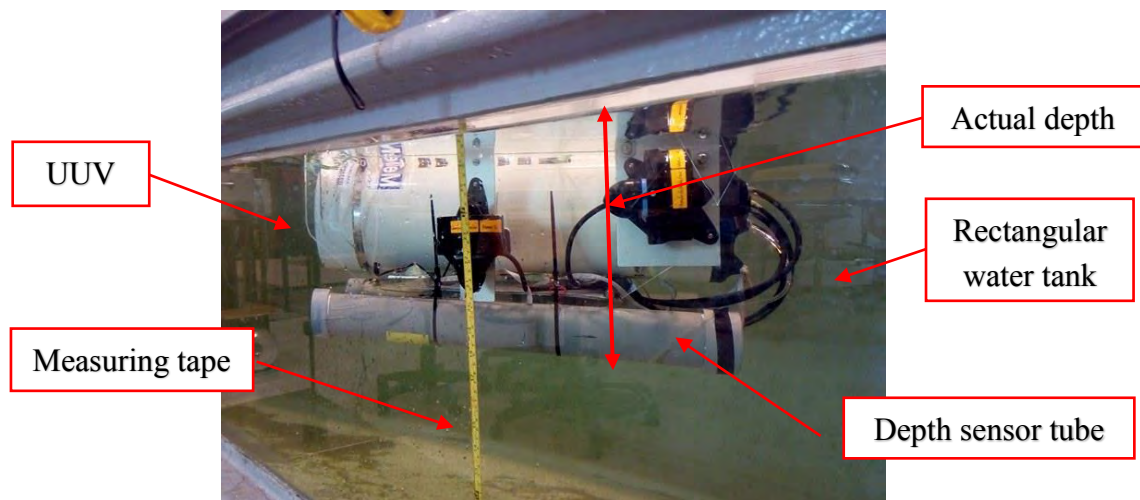


Figure 3.14: Set up of experiment 4

1. The experiment 4 is set up as shown in Figure 3.14.
2. The depth control coding is created as shown in Appendix B to allow the UUV to submerge until 10 cm of water depth and then stops.
3. The integrated sensor of UUV is uploaded with depth control coding.
4. The measuring tape is set up as shown in Figure 3.14 to measure the water depth.
5. The UUV is put on the water surface and the power supply of UUV is switched on. The UUV starts to submerge into the water.
6. The actual depth of UUV submerges in underwater is measured and recorded when the UUV starts to stop at a certain of depth. The readings are taken three times to get more accurate results. The actual depth of UUV is the distance between the water surface and the depth sensor tube as shown in Figure 3.14.
7. The steps above are repeated until the UUV submerge to 100 cm of water depth with each increment of 10 cm by changing the analog reading of the depth control coding.

3.5.6 Experiment 6: Heading Degree Analysis of Magnetometer

Objective:

To analyse the position angle measurement of magnetometer.

Parameter:

Manipulated variable: Actual heading Degree

Responding variable: Measured output heading degree

Equipment:

1. Magnetometer
2. Arduino Mega2560
3. USB cable
4. Station (PC)
5. Protractor

Procedure:

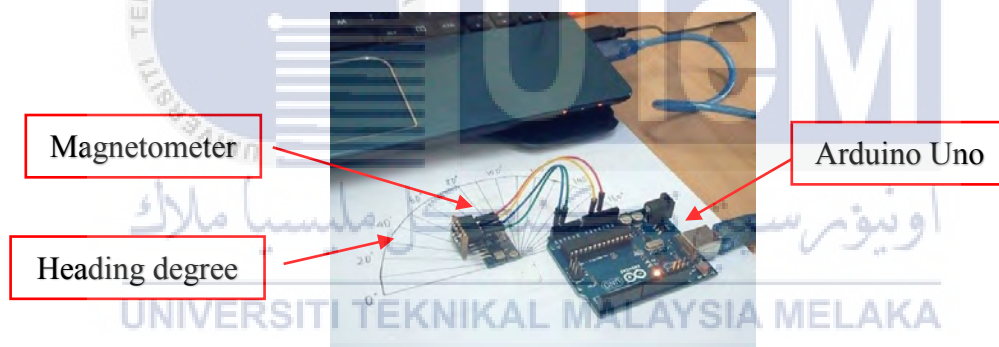


Figure 3.15: Set up of experiment 6

1. The experiment is set up as shown in Figure 3.15.
2. The microcontroller, Arduino Mega2560 is uploaded with magnetometer coding as shown in Appendix B and interfaced with magnetometer.
3. The heading degree of the magnetometer is monitored in Arduino serial monitor.
4. The magnetometer is rotated until the serial monitor shows zero heading degree. This position is set as a reference angle position.
5. After that, the magnetometer is rotated to 20° of heading degree. The output heading degree in the serial monitor is measured and recorded.
6. The step 6 is repeated with heading degree from the range of 20° to 180° with each increment of 20° .
7. The graph of the actual heading degree versus measured heading degree is plotted.

3.5.7 Experiment 7: Leakage Sensor Testing

Objective:

To test the functionality of the leakage sensor in the presence of water.

Parameter:

Manipulated variable: Presence of water

Responding variable: Condition of LED and buzzer

Equipment:

1. Microcontroller, Arduino Uno
2. Leakage sensor
3. LED
4. A glass of water

Procedure:

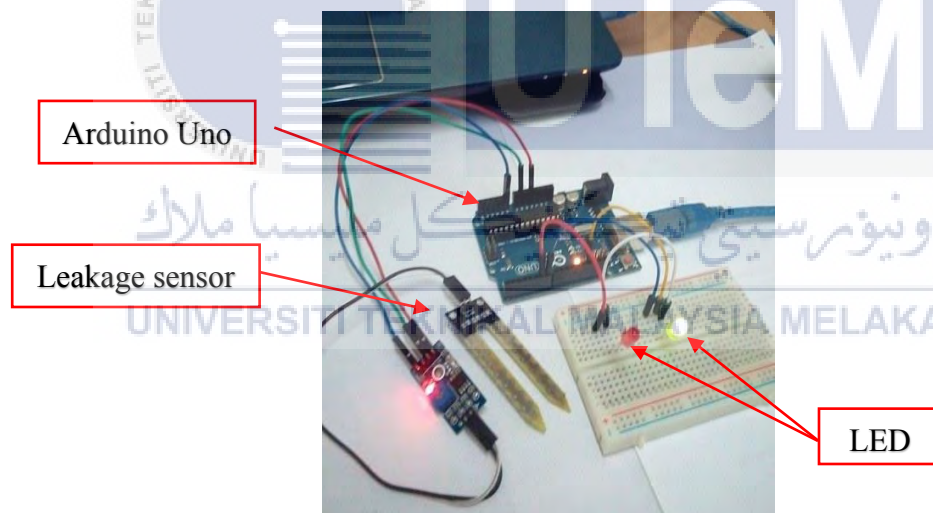


Figure 3.16: Set up of experiment 7

1. The experiment is set up as Figure 3.16.
2. The microcontroller, Arduino Uno is uploaded with leakage sensor coding as shown in Appendix B and interfaced with leakage sensor.
3. The condition of the LED and buzzer are observed.
4. The leakage sensor is immersed into a glass of water. The condition of the LED and buzzer are observed again.

3.6 Modelling and Controller

3.6.1 System Identification Toolbox of Matlab

In the dynamic control system design, mathematical model is required to design and applied. In underwater industries, a dynamic model of the AUV is established based on its general motion of six degrees of freedom which includes roll, pitch, yaw, heave, sway and surge. In this project, only three axis of x , y and z are considered for depth control of AUV.

For modelling the AUV of this project, the system identification toolbox of MATLAB is used to generate the transfer function for depth control. This toolbox is used to generate the depth control transfer function modelling for underwater vehicles. The parameters for the modelling are collected when the AUV submerge in pool for 1m of water depth. The collected data from the Arduino serial monitor is exported to a Microsoft Excel file as shown in Table 3.6.

In Table 3.6, the readings are chosen up to 25 seconds to analyse the model response. Depth input is the depth of AUV submerges by predicted and assumed while depth output is the actual depth of AUV submerges without depth controller. It submerges freely and will stops stationary at 120 cm depth due to AUV's weight is almost equal to buoyancy force. Thus, it requires a depth controller to maintain it at 100 cm depth.

The data from the Microsoft Excel are exported to the MATLAB workspace. The command "systemIdentification" is inserted in the MATLAB command window to open the system identification tool. The collected data are imported from the MATLAB workspace to the MATLAB system identification tool as shown in Figure 3.17

Table 3.6: Depth data of AUV

Time (s)	Depth input (cm)	Depth output (cm)
0	0	0
1	100	4
2	100	15
3	100	27
4	100	38
5	100	49
6	100	58
7	100	69
8	100	78
9	100	87
10	100	95
11	100	100
12	100	105
13	100	110
14	100	114
15	100	116
16	100	120
17	100	120
18	100	120
19	100	120
20	100	120
21	100	120
22	100	120
23	100	120
24	100	120
25	100	120

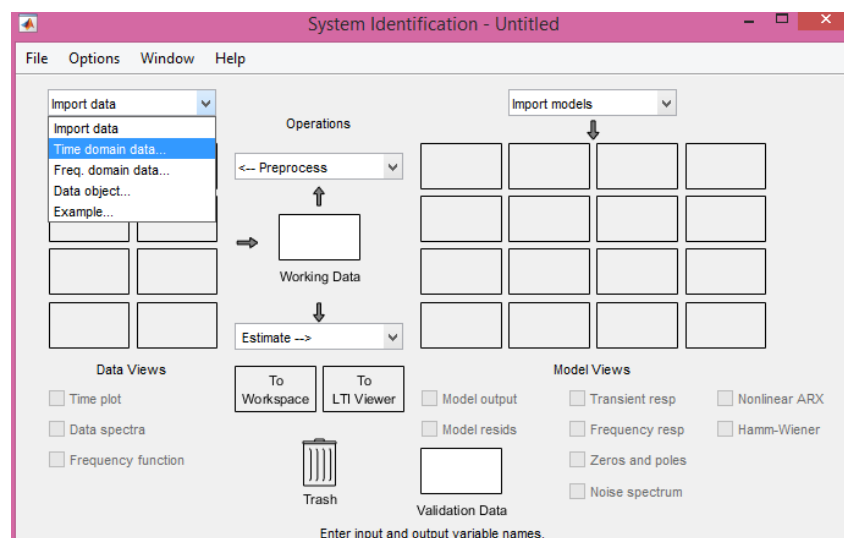


Figure 3.17: MATLAB system identification toolbox

The variables in the MATLAB workspace are inserted into the import data window as shown in Figure 3.18. The starting time and sampling interval are set to 0 s and 0.02 s respectively. The time plot of input and output data can be viewed as shown in Figure 3.19.

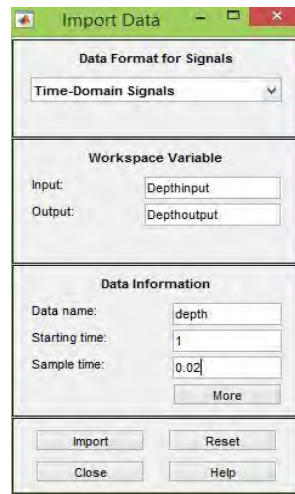


Figure 3.18: Import data

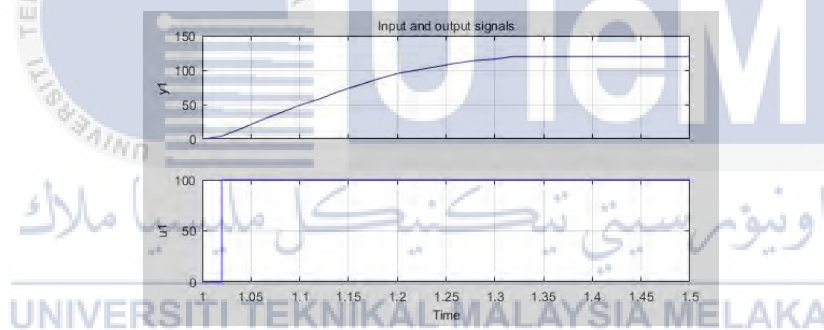


Figure 3.19: Time plot for input and output data

As shown in Figure 3.20, the "Transfer Function Model" is selected from the "Estimate" drop down list.

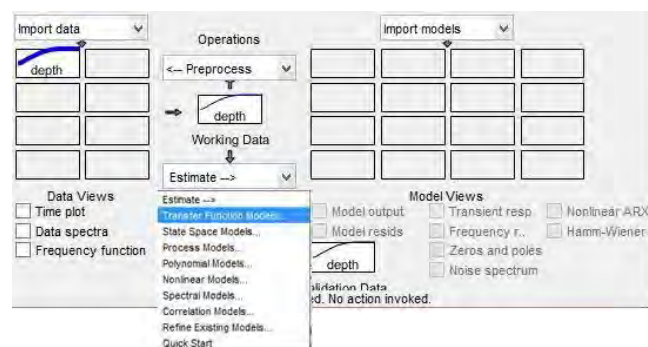


Figure 3.20: Transfer function model

The number of poles and zeroes are assigned to get the model response as shown in Figure 3.21. In this AUV depth control system, three poles and two zeros are used to implement the three axes of AUV depth motion. Next, the plant identification progress estimates the transfer function of AUV.

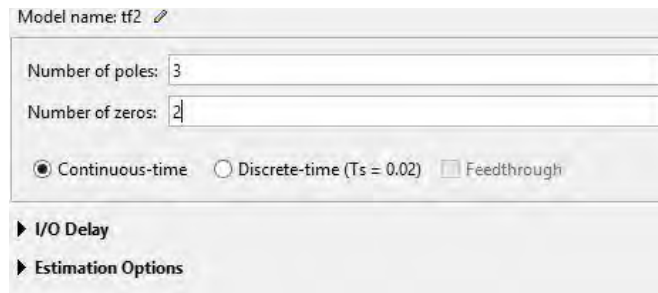


Figure 3.21: 3 Poles and 2 zeros implemented

After that, the model is exported to the MATLAB workspace. When the name of model which is "tf2" is inserted to the command window, the transfer function of the model is displayed as shown in Figure 3.22. Figure 3.23 shows the step response of AUV depth control. The amplitude of the step response is maintained at 1.2 and there is no overshoot.

From input "u1" to output "y1":

$$\frac{5.932 s^2 + 128.3 s + 2109}{s^3 + 16.91 s^2 + 290.5 s + 1694}$$
 Name: tf2
 Continuous-time identified transfer function.

Figure 3.22: Transfer function model

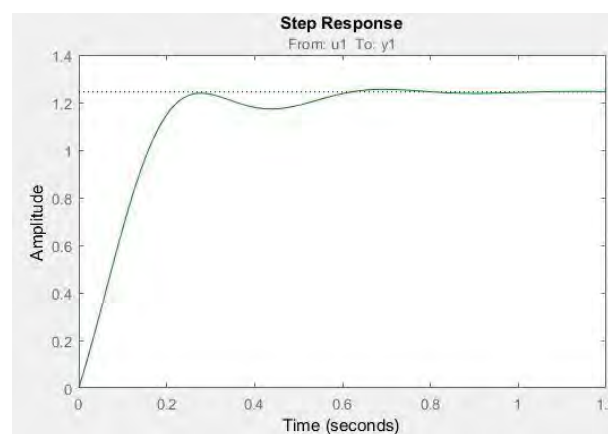


Figure 3.23: Step response of the transfer function

3.6.2 AUV On-Off Depth Controller

On-off controller is the simplest controller implemented with the microcontroller coding. Depth sensor is used to determine actual depth of AUV submerge. It sends the signal to microcontroller for processing and send back the signal to ESC (electronic speed controller) to actuated thrusters to move downward (submerge). Arduino coding is written in the form of “if ...else if...” to give command for AUV to response in different movement.

In this project, Arduino depth control coding for on-off controller is designed as shown in Appendix B. There are three conditions for AUV to response which include AUV moves downward if its depth less than 100 cm, moves upward if its depth more than 100 cm and stops if its depth equal to 100 cm.

Table 3.7: Depth data of AUV with on-off depth controller

Time (s)	Depth input (cm)	Depth output (cm)
0	0	0
1	100	4
2	100	15
3	100	27
4	100	38
5	100	49
6	100	58
7	100	69
8	100	78
9	100	87
10	100	95
11	100	104
12	100	110
13	100	117
14	100	110
15	100	102
16	100	92
17	100	97
18	100	103
19	100	104
20	100	106
21	100	110
22	100	117
23	100	114
24	100	109
25	100	105

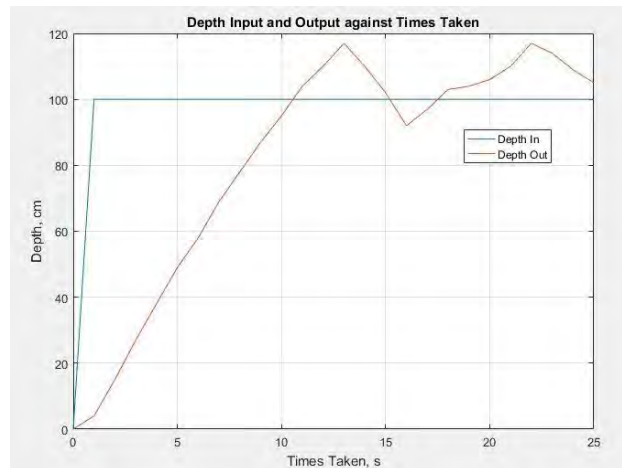


Figure 3.24: Graph of depth input and output against time

Table 3.7 shows the depth data of AUV with on-off depth controller. Depth input is the desired depth of AUV submerges while depth output is the actual depth of AUV submerges with on-off depth controller.

In Figure 3.24, the depth output value is not stable against time after the water depth is more than 100 cm. This means that the AUV is moving upward and downward continuously after it submerge until 100 cm. It moves downward when less than 100 cm of depth and moves upward when more than 100 cm of depth, but it will not stop instantaneous when it reaches 100 cm due to its weight. The AUV cannot keep its underwater depth (input depth) to 100 cm as desired against time but it able to keep its underwater depth within ± 10 cm of 100 cm of input depth. As a conclusion, on-off depth controller is not a perfect depth controller for UUV but it is one of the lowest cost and easy method in depth control application of UUV.

3.7 Summary

As a summary, based on the methodology that has been discussed in this chapter, the stages to complete this project has been done accordingly. The steps to complete this project are followed as planned so that the project can be completed on time and achieve the objectives of the project. In this chapter, the information about the implement process of this project, equipments and components used as well as the clear experiment set up have been discussed. The functionality of each sensor is tested and a complete data are collected for the performance analysis. A complete integrated sensor is developed and ready to install into an UUV for the real-time simualtion and data analysis.

CHAPTER 4

RESULTS AND DISCUSSIONS

4.1 Introduction

This chapter discuss how to analyse the performance of the sensors from the data obtained. The integrated sensor is tested to collect the output data and generate the output graph for the performance analysis. The integrated sensor also installed to the UUV to get the real time data.

4.2 Final Design of Integrated Sensor

Figure 4.1 shows the completed design of the integrated sensor with USB cable. The IMU sensor, depth sensor, magnetometer and leakage sensor are assembled to develop an integrated sensor. There are two drilled holes for the depth sensor and USB cable. The drilled holes are sealed to prevent water intrusion into the casing. The signals of IMU and depth sensor are analog input wheares magnetometer and leakage sensor are digital. Figure 4.2 shows the interior part of the integrated sensor.

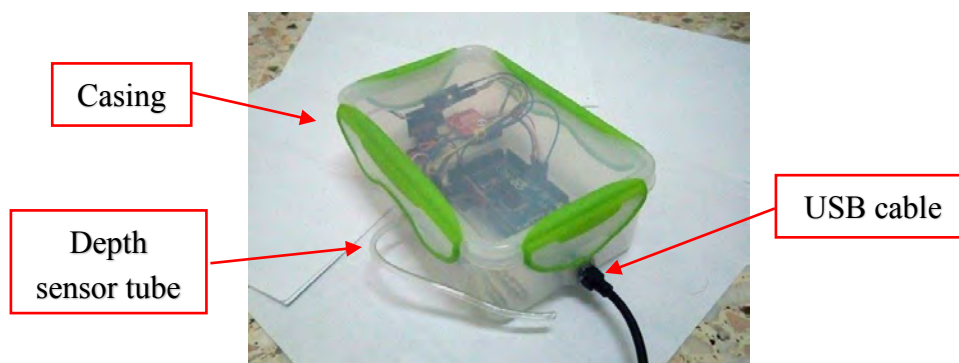


Figure 4.1: Completed design of integrated sensor

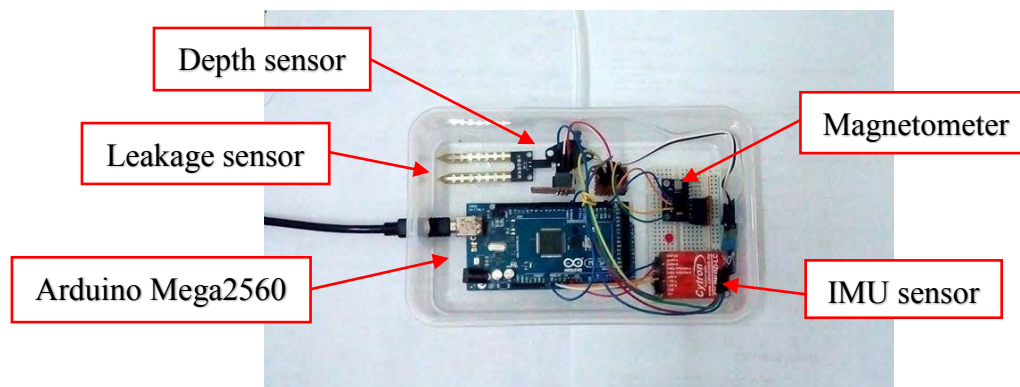


Figure 4.2: Interior part of integrated sensor

4.3 Analysis and discussion of experiment

Each of the sensor is interfaced and programmed with the microcontroller, Arduino Mega2560 to get the data for the performance analysis.

4.3.1 Experiment 1: Orientation Movement Analysis of IMU sensor

For IMU sensor, there is methods to estimate its error and accuracy. There are two factors that may affected the performance of the IMU sensor: gravitational force and offset errors. The gravitational force is acted with the z -axis that aligns with it. In normal condition, there is measurement error because z -axis aligns with the gravitational force. The other one is the offset error where if there is reading in static condition, an offset error is occurred. Offset error is the difference between the nominal and actual offset points. The reading will be inaccurate and affect the actual result. By comparing the split results of the output graphs, it is easier to find an offset error and gravitational force because the graphs of accelerometer and gyroscope are separated. Therefore, the readings will be more accurate by calculating the offset error and gravitational force.

Table 4.1, Table 4.2 and Table 4.3 show the reading of accelerometer sensor at x , y and z -axis respectively. Table 4.4 shows the data obtained from the sensor after adjusting the Arduino coding by referring to the offset error. The raw data of IMU sensor are taken from the movement of accelerometer and rotation of gyroscopes. To improve the accuracy of the results, there are three times readings taken from the accelerometer at x , y and z -axis respectively by referring the offset error.

Table 4.1: The accelerometer readings at x-axis

Axis	Readings of Accelerometer sensor (ms ⁻²)			
	Reading 1	Reading 2	Reading 3	Average values
<i>x</i>	9.41	9.49	9.92	9.61
<i>y</i>	0	-0.24	-0.43	-0.22
<i>z</i>	-2.31	-2.12	-2.24	-2.22

Table 4.2: The accelerometer readings at y-axis

Axis	Readings of Accelerometer sensor (ms ⁻²)			
	Reading 1	Reading 2	Reading 3	Average values
<i>x</i>	-0.59	-0.59	-0.51	-0.56
<i>y</i>	9.92	9.96	9.89	9.92
<i>z</i>	-0.63	-0.98	-0.75	-0.79

Table 4.3: The readings of accelerometer at z-axis

Axis	Readings of Accelerometer sensor (ms ⁻²)			
	Reading 1	Reading 2	Reading 3	Average values
<i>x</i>	-0.59	-0.59	-0.59	-0.59
<i>y</i>	0.04	0.08	0.04	0.05
<i>z</i>	9.38	9.41	9.38	9.39

The gravitational force is measured by comparing the highest average values at z-axis in Table 4.1, Table 4.2 and Table 4.3. From Table 4.3, the average readings at z-axis is 9.39 m/s². Based on the gravitational acceleration, - 9.81m/s², the offset error as shown in Equation (4.1).

$$-9.81 \text{ m/s}^2 + 9.39 \text{ m/s}^2 = - 0.42 \text{ m/s}^2 \quad (4.1)$$

In Equation (4.1), the gravitational force for IMU sensor is - 0.42 m/s². Then, the IMU sensor is tested again and the readings are collected as shown in Table 4.4 to find the offset error percentages.

Table 4.4: The readings of accelerometer at x, y and z axis

Axis	Readings of Accelerometer sensor (ms ⁻²)			
	x	y	z	Highest values
x	9.78	-0.39	-0.46	9.78
y	-0.38	9.82	-0.87	9.82
z	-0.29	-0.32	9.84	9.84

Based on the measurement values of accelerometer readings at x, y and z-axis, the highest values of each axis are used to find an offset error.

$$x\text{-axis: } [-9.81 + 9.78 \text{ m/s}^2] / [-9.81 \text{ m/s}^2 * 100\%] = + 0.3\% \quad (4.2)$$

$$y\text{-axis: } [-9.81 + 9.82 \text{ m/s}^2] / [-9.81 \text{ m/s}^2 * 100\%] = - 0.1\% \quad (4.3)$$

$$z\text{-axis: } [-9.81 + 9.84 \text{ m/s}^2] / [-9.81 \text{ m/s}^2 * 100\%] = - 0.3\% \quad (4.4)$$

The results from Equations (4.2), (4.3) and (4.4) show that x, y and z-axis error are in the range between - 0.3% and 0.3%. Therefore, the result is acceptable because the percentage error is very small. It can be ignored because it does not affect the actual results significantly. For gyroscope in IMU, the rotational speed reading are taken as shown in Table 4.5.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

Table 4.5: The readings of gyroscope at x, y and z axis

Axis	Readings of Gyroscope (°s ⁻¹)			
	Reading 1	Reading 2	Reading 3	Average values
x	14	21	-6	9.67
y	-4	-28	18	-14.00
z	-8	-1	-7	-5.33

Figure 4.3 shows the initial output graph of accelerometer and gyroscope. The IMU sensor possesses its own offset value for all the axis. Therefore, an offset calibration is required to be performed before the data are collected from the sensor. The offset values can be obtained from the data when it is at stationary. The offset values are subtracted from the data so that all the axis give zero values when it is stationary.

In Figure 4.3, their initial values are approximate to zero when the IMU sensor is interfacing with MATLAB real-time simulation. This means that all the axis give the value of zero when the top of IMU sensor faces upward and stationary. However, there are disturbances before the end of the simulation.

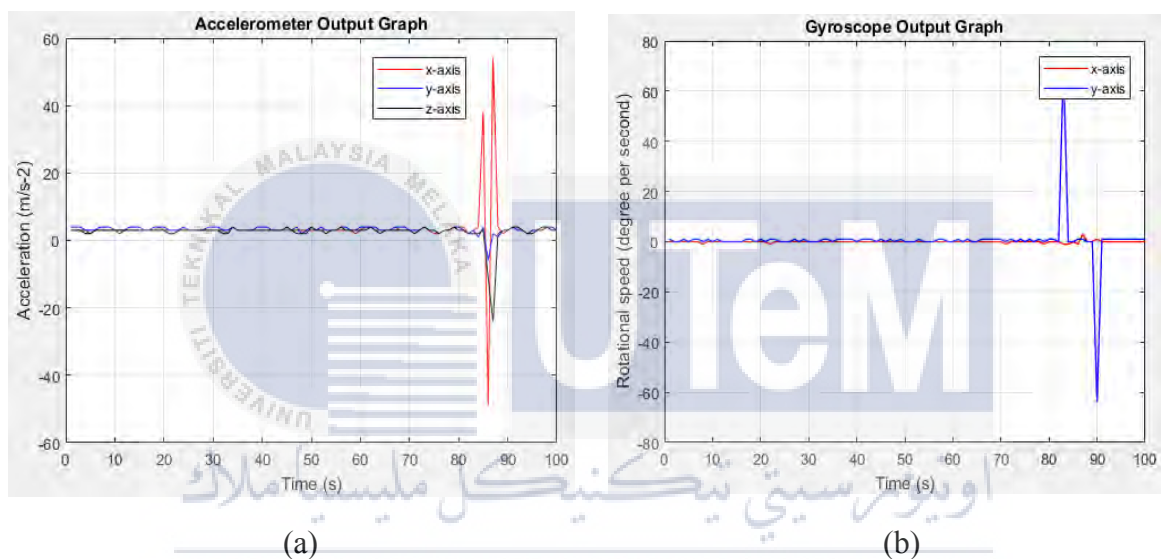


Figure 4.3: Initial output graph of accelerometer and gyroscope

Figure 4.4 and Figure 4.5 shows the output movement graph of accelerometer. The x -axis of the graph is time in second and the y -axis is acceleration (m/s^2). The x , y and z -axis are red, blue and black respectively. The sample rate of this accelerometer is 1 Hz which corresponding to 1 data per second. In Figure 4.4 (a) and (b), there are 3 maximum positive value and 3 minimum negative value respectively. In Figure 4.4 (a), the y -axis (blue line) and z -axis (black line) have significant vibration at 10 s, 20 s and 30 s, but there is no significant vibration at x -axis (red line). This is because the IMU sensor is rotated at x -axis which can bring disturbances to the others two axis. The same condition also happens in Figure 4.4 (b), but the differences are the x -axis (red line) and z -axis (black line) have significant vibration and there is no significant vibration at y -axis (blue line). This condition occurred because the y -axis (blue line) of the IMU sensor is rotated.

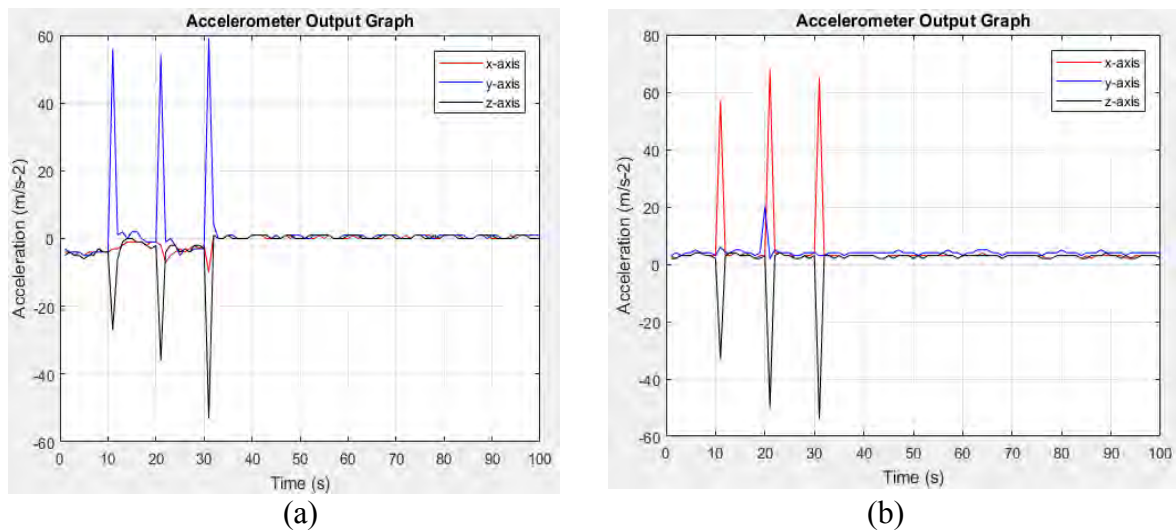


Figure 4.4: Output graph of accelerometer

Figure 4.5 shows the graph of the forward movement of UUV using accelerometer. The theoretical speed of this UUV is 0.4 ms^{-1} . In this graph, the x -axis (red line) is vibrated, followed by vibration of the z -axis (black line) and the last is y -axis (blue line). The y -axis (blue line) is more stable than the others two axes. This shows that the UUV does not have too much changing at y -axis which means that y -axis has less frequency vibration but x -axis and z -axis have more vibration in the output graph. As observed, the UUV moves forward in depth control. The UUV is submerging when moving forward due to the high weight of the body. When the UUV submerges until to the certain of depth, the UUV will moves upward. Therefore, in this up and down sequence movement of UUV, the vibration of x -axis and z -axis are much higher than y -axis.

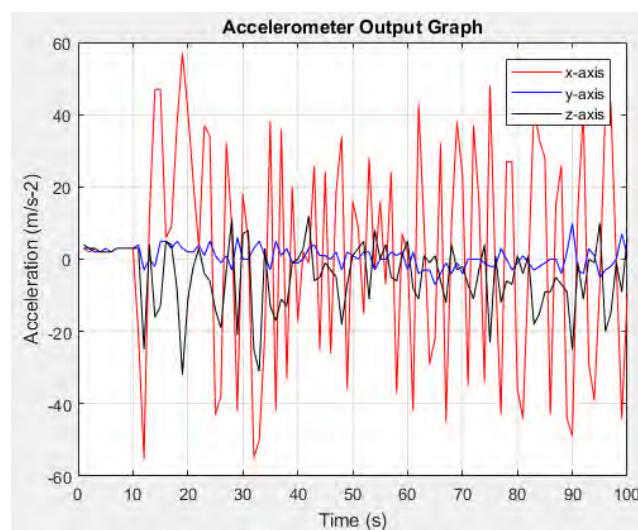


Figure 4.5: Output graph of accelerometer in UUV

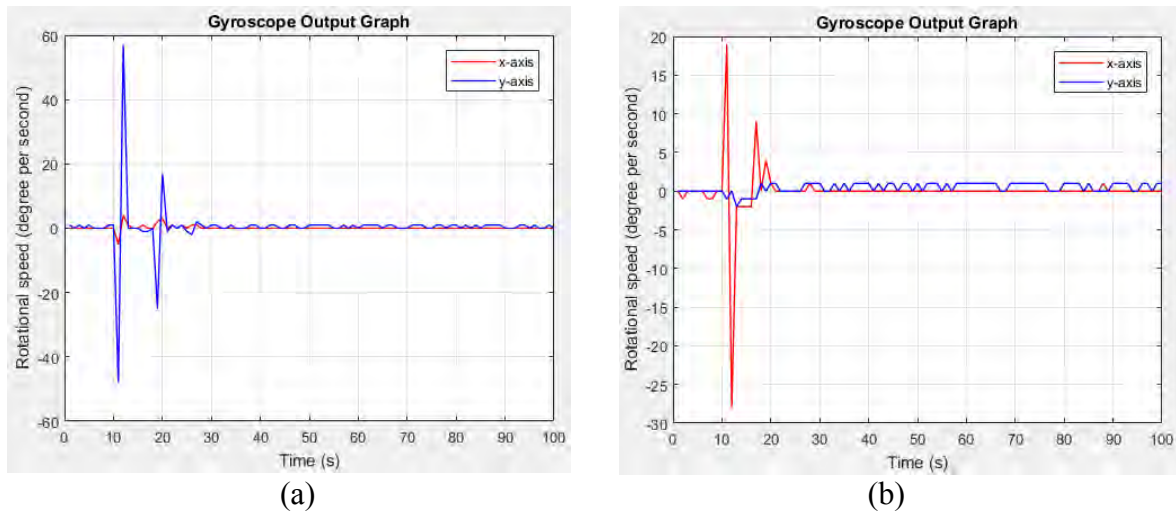


Figure 4.6: Output graph of gyroscope

Figure 4.6 shows the output movement graph of gyroscope. The x -axis of the graph is time in second and the y -axis is rotational speed ($^{\circ} \text{s}^{-1}$). The x -axis and y -axis are red and blue respectively. The sample rate of this gyroscope is 1 Hz which corresponding to 1 data per second. In Figure 4.6 (a) and (b), it shows that there are maximum positive value and minimum negative value respectively. In Figure 4.6 (a), the y -axis (blue line) has significant vibration at 10 s and 20 s, but there is no significant vibration at x -axis (red line). This is because the IMU sensor is rotated at x -axis which can bring disturbances to the y -axis. The same condition also happens in Figure 4.6 (b), but the differences is the x -axis (red line) has significant vibration and there is no significant vibration at y -axis (blue line). This condition occurred because the y -axis (blue line) of the IMU sensor is rotated.

As a conclusion of this experiment, MATLAB real-time simulation is successfully communicated with Arduino by using serial communication port, the output graph of accelerometer and gyroscope are matched with the movement sequences of the UUV.

4.3.2 Experiment 2: To Determine the Consistency and Accuracy of Depth Sensor

In experiment 1, the results are shown as Figure 4.7 and Figure 4.8. Figure 4.7 shows the graph of pressure reading of depth sensor in real time while Figure 4.8 shows the graph of analog reading of depth sensor in real time. In Figure 4.7, the pressure reading of the depth sensor is at the range from 96.5 kPa to 98.0 kPa in 100 s. In Figure 4.8, the analog reading of the depth sensor is at the range from 168 to 169 kPa in 100 s.

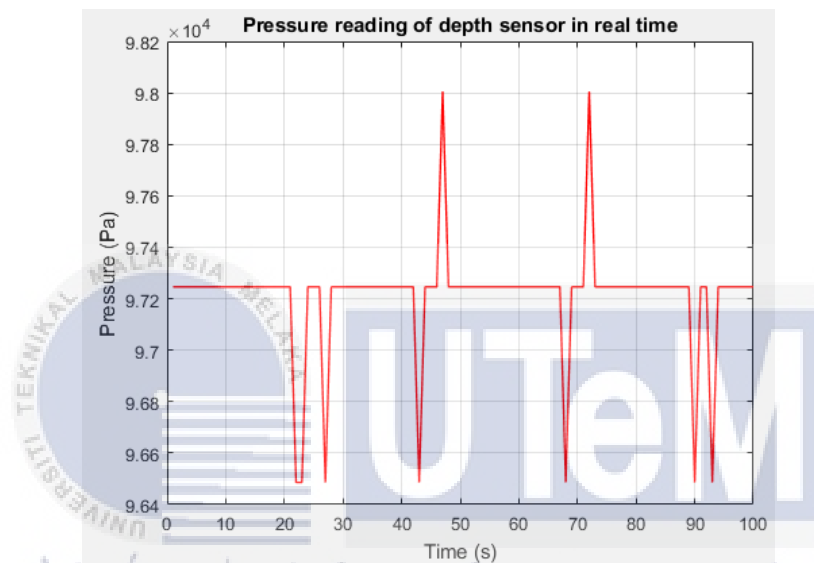


Figure 4.7: Graph of pressure reading in MATLAB real time simulation

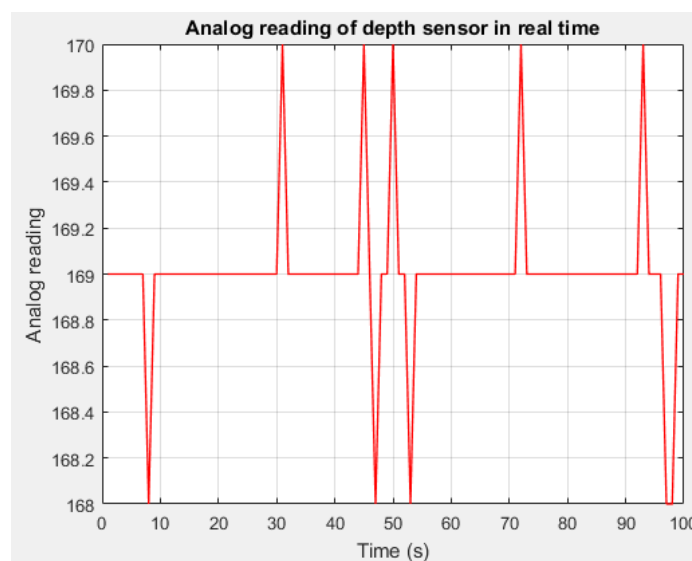


Figure 4.8: Graph of analog reading in MATLAB real time simulation

Calculation and Analysis Finding

To determine the consistency of depth sensor, the mean and standard deviation are calculated in Figure 4.7 and Figure 4.8. In Figure 4.7, the mean and standard deviation of the depth sensor are 97.167 and 0.2145 respectively. In Figure 4.8, the mean and standard deviation of the depth sensor are 169 and 0.3162 respectively. Since the standard deviation of these two graphs are 0.2145 and 0.3162 respectively which are very small and near to 0. Therefore, it can be concluded that all the data that are collected in the experiment are very consistent.

The Mean and Standard Deviation of Figure 4.7

$$\text{Mean, } \mu = \frac{P1 + P2 + P3 \dots P100}{\text{Number of reading}} = \frac{96.5 \times 7 + 97.2 \times 91 + 98.0 \times 2}{100} = 97.167$$

Standard deviation, σ

$$= \sqrt{\frac{(P1 - 97.167)^2 + (P2 - 97.167)^2 + (P3 - 97.167)^2 \dots (P100 - 97.167)^2}{\text{Number of reading}}}$$

$$= \sqrt{\frac{(96.5 - 97.167)^2 \times 7 + (97.2 - 97.167)^2 \times 91 + (98.0 - 97.167)^2 \times 2}{100}}$$

$$= 0.2145$$

The Mean and Standard Deviation of Figure 4.8

$$\text{Mean, } \mu = \frac{R1 + R2 + R3 \dots R100}{100 \text{ Number of reading}} = \frac{168 \times 5 + 169 \times 90 + 170 \times 5}{100} = 169$$

Standard deviation, σ

$$= \sqrt{\frac{(P1 - 169)^2 + (P2 - 169)^2 + (P3 - 169)^2 \dots (P100 - 169)^2}{\text{Number of reading}}}$$

$$= \sqrt{\frac{(168 - 169)^2 \times 5 + (169 - 169)^2 \times 90 + (170 - 169)^2 \times 5}{100}}$$

$$= 0.3162$$

To determine the accuracy of depth sensor, root-mean-square (RMS) error is calculated in Figure 4.7. The desired pressure in this RMS calculation is 100 kPa (1 bar) because the atmosphere pressure of earth is approximately 100 kPa (1 bar). From the calculation, the RMS error of this depth sensor is 2.8411. Based on this low RMS error, it can be concluded that the depth sensor gives quite accurate output pressure.

RMS Error

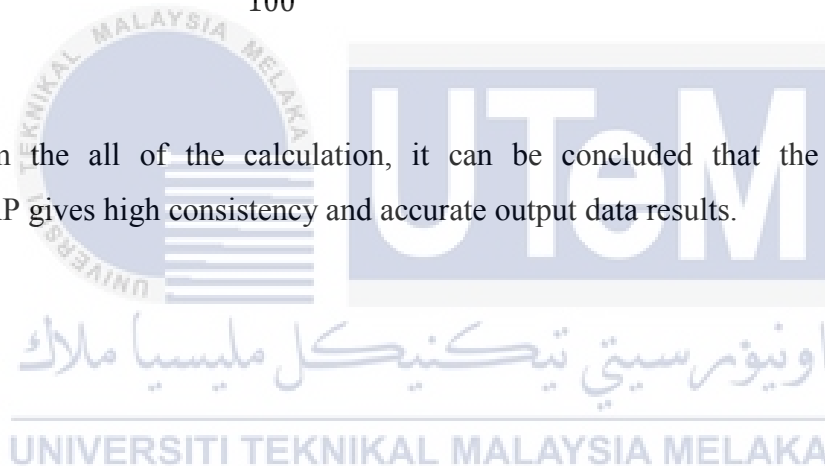
$$= \sqrt{\frac{(P1 - P0)^2 + (P2 - P0)^2 + (P3 - P0)^2 \dots (P100 - P0)^2}{\text{Number of reading}}}$$

Let desired pressure, P0 = 100 kPa

$$= \sqrt{\frac{(96.5 - 100)^2 \times 7 + (97.2 - 100)^2 \times 91 + (98.0 - 100)^2 \times 2}{100}}$$

$$= 2.8411$$

From the all of the calculation, it can be concluded that the depth sensor, MPX5700AP gives high consistency and accurate output data results.



4.3.3 Experiment 3: Depth Sensor Performance Analysis in Air

In experiment 3, the output signal voltage is measured three times to get an accurate average output value and recorded in Table 4.6. The graph of measured output signal voltage (V) versus input pressure of air compressor (kPa) is plotted as shown in Figure 4.9. In Figure 4.9, the output voltage is linearly proportional to the input pressure of air compressor. The three output reading almost have the same linearly pattern. The multimeter is used to measure the output signal voltage of depth sensor because the output signal voltage is low. Therefore, this manual measurement is applied in this experiment.

Table 4.6: The output results of experiment 3

Input Pressure of Air Compressor (bar)	Input Pressure of Air Compressor (kPa)	Measured Output Signal Voltage (V)			
		1 st	2 nd	3 rd	Average
1.0	100	0.86	0.87	0.85	0.86
1.5	150	1.16	1.2	1.24	1.2
2.0	200	1.54	1.52	1.56	1.54
2.5	250	1.83	1.89	1.86	1.86
3.0	300	2.22	2.18	2.14	2.18
3.5	350	2.55	2.56	2.54	2.55
4.0	400	2.90	2.88	2.86	2.88
4.5	450	3.19	3.25	3.22	3.22
5.0	500	3.47	3.45	3.49	3.47
5.5	550	3.79	3.83	3.82	3.82
6.0	600	4.14	4.17	4.11	4.14

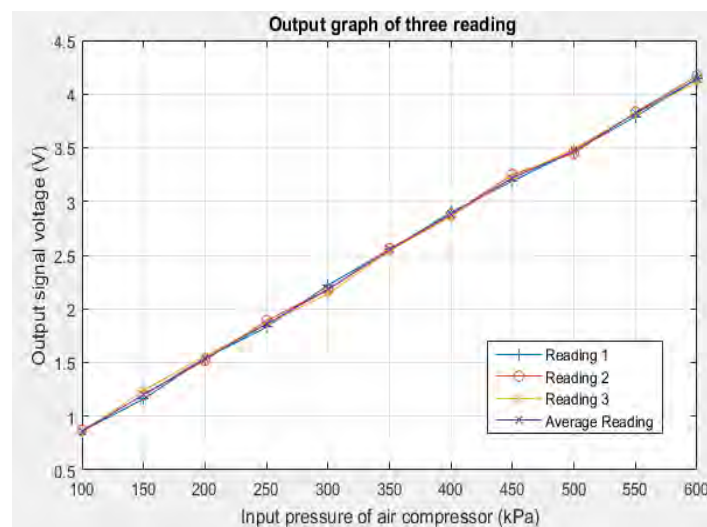


Figure 4.9: Graph of measured output signal voltage versus input pressure of air compressor

Figure 4.10 shows the characteristic plotted graph of depth sensor output signal voltage (V) versus to pressure (kPa). This graph can be found in datasheet of pressure sensor, MPX5700AP. In Figure 4.10, the output signal voltage is directly proportional to pressure. However, the output signal voltage is saturated at outside of the specified pressure range, that is maximum pressure of 700 kPa. Therefore, in this experiment 3, the output signal voltage of depth sensor is measured until 6 bar (600 kPa) to prevent any damage to the depth sensor. There are three lines shown in this figure: minimum, typical and maximum for operation in a temperature range of 0°C to 85°C. The typical line is referred to determine the actual pressure of the depth sensor. Therefore, based on the average voltage value in Table 4.6, actual pressure in this experiment can be calculated by referring the characteristic plotted graph of Figure 4.10. The actual pressure of the experiment is calculated by using Equation (4.5) and tabulated in Table 4.7.

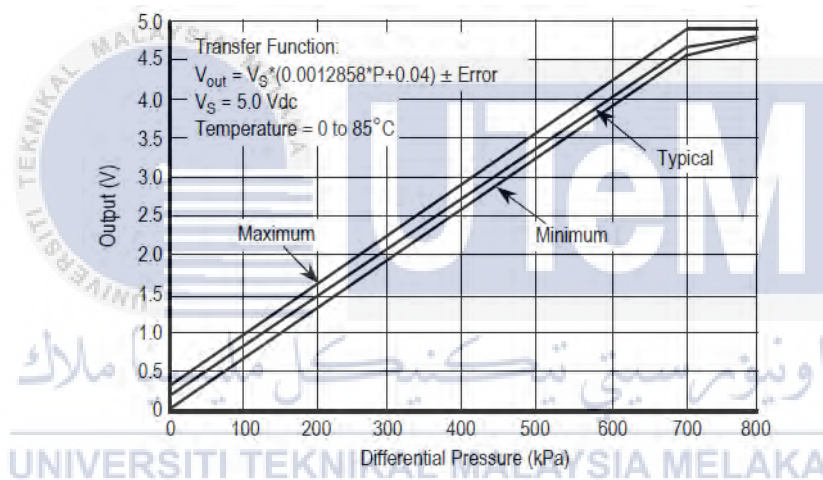


Figure 4.10: The characteristic plotted graph of depth sensor, MPX5700AP

Calculation for Actual Pressure

In Figure 4.10, the equation formula is given as shown in the following:

$$V_{out} = V_s * (0.0012858 * P + 0.04) \quad (4.5)$$

$$P = \frac{\frac{V_{out}}{V_s} - 0.04}{0.0012858}$$

Example actual pressure calculation at 1 bar pressure of air compressor:

$$P = \frac{\frac{V_{out}}{V_s} - 0.04}{0.0012858} = \frac{\frac{0.86}{5} - 0.04}{0.0012858} = 103.13 \text{ kPa}$$

Performance of Pressure System

To evaluate the performance of the depth sensor MPX 5700, each of the error percentage calculation at different input pressure of air compressor is calculated and continued by using the Equation (4.6) and tabulated in Table 4.7. The graph of actual pressure (kPa) and desired pressure versus input pressure of air compressor (kPa) is plotted as shown in Figure 4.11. In Figure 4.11, actual pressure is approximate to the desired pressure. The readings almost have the same linearly output pattern.

Percentage of error

$$= \left| \frac{\text{Input pressure of air compressor (kPa)} - \text{Actual pressure (kPa)}}{\text{Actual pressure (kPa)}} \right| \times 100\% \quad (4.6)$$

Table 4.7: Percentage error of depth sensor in experiment 3

Input pressure of air compressor (kPa)	Average output signal voltage (V)	Actual pressure (kPa)	Percentage of error (%)
100	0.86	102.66	2.66
150	1.2	155.55	3.70
200	1.54	208.43	4.22
250	1.86	258.21	3.28
300	2.18	307.98	2.66
350	2.55	365.53	4.44
400	2.88	416.86	4.22
450	3.22	469.75	4.39
500	3.47	508.63	1.73
550	3.82	563.07	2.38
600	4.14	612.85	2.14

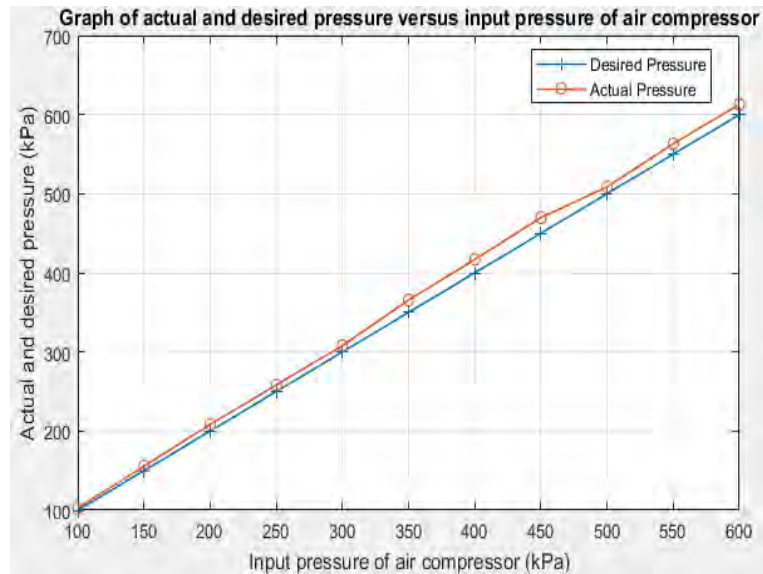


Figure 4.11: Graph of actual and desired pressure versus input pressure of air compressor

The depth sensor performance is analysed by determining the average percentage of error. The average percentage of error is calculated as following:

$$\text{Average percentage of error} = \frac{\text{Total percentage of error}}{\text{Number of reading}} = \frac{35.80}{11} = 3.25\%$$

From the calculation, the average percentage of error is 3.25%. The minimum percentage error is 1.73 %, while the maximum percentage error is 4.44 %. It shows that this sensor has a low percentage of error to measure the pressure in air. The performance analysis in this experiment shows that the depth sensor can accurately measure the pressure value in air since the actual pressure value of the experiment is approximate to the input pressure from air compressor.

4.3.4 Experiment 4: Depth Sensor Performance Analysis in Underwater

In experiment 4, the output signal voltage is measured and recorded in Table 4.8. The output signal voltage is measured three times to get an accurate average output value. The multimeter is used to measure the output signal voltage of depth sensor because the output signal voltage is very low. Therefore, this manual measurement is applied in this experiment. The setting depth between 0.863V and 0.922V is 1m. From the results obtained, the graph of average output signal voltage (V) versus water depth (cm) is plotted as shown in Figure 4.12. The measured output signal voltage of the depth sensor is increases as the water depth increases until it saturated when the depth reaches to pressure sensor limit up to 70 m. The three output readings almost have the same linearly pattern.

Table 4.8: The output results of experiment 4

Depth (cm)	Measured Output Signal Voltage (V)			
	1 st	2 nd	3 rd	Average
0	0.863	0.864	0.862	0.863
10	0.874	0.871	0.868	0.871
20	0.875	0.879	0.877	0.877
30	0.881	0.883	0.885	0.883
40	0.888	0.889	0.887	0.888
50	0.891	0.893	0.895	0.893
60	0.896	0.902	0.899	0.899
70	0.907	0.905	0.903	0.905
80	0.910	0.912	0.908	0.910
90	0.917	0.916	0.915	0.916
100	0.924	0.920	0.922	0.922

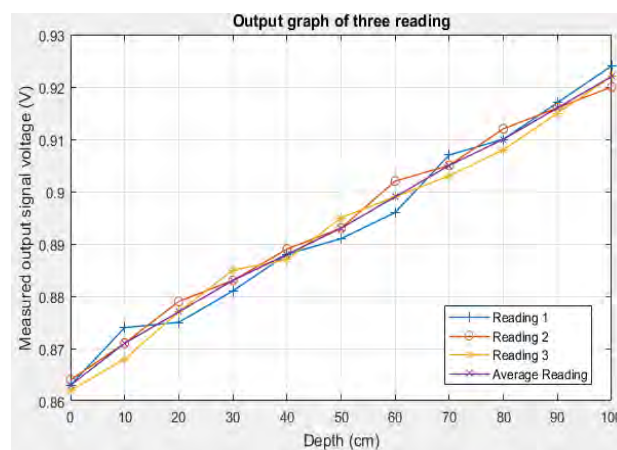
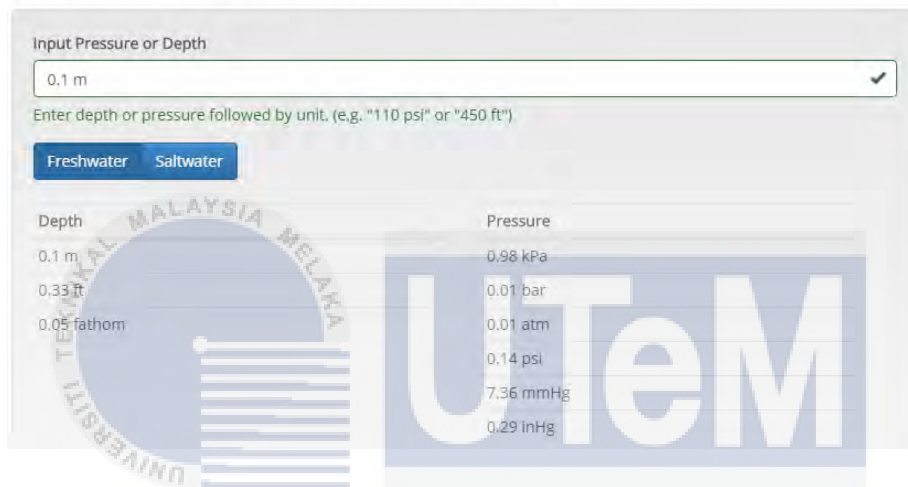


Figure 4.12: Graph of output signal voltage versus water depth

The theoretical pressure of this experiment can be obtained from the software converter that can be used to convert the water depth into pressure value. Figure 4.13 shows the software converter used to calculate the theoretical pressure at the given depth. Besides that, the theoretical pressure also can be calculated by using Equation (2.11). The theoretical pressure at different water depth are obtained and tabulated in Table 4.9.

Calculator

This calculator determines the water depth to reach a given pressure or the pressure at a given depth. You can enter the depth or pressure input with most common units and it will be converted automatically. This calculator *does not* add air pressure at the surface - the results are relative to the surface pressure.



Depth	Pressure
0.1 m	0.98 kPa
0.33 ft	0.01 bar
0.05 fathom	0.01 atm
	0.14 psi
	7.36 mmHg
	0.29 inHg

Figure 4.13: Pressure converter

Calculation for Theoretical Pressure

Example of theoretical pressure calculation is shown as following:

For depth, $h = 0.1\text{m}$, by using Equation (2.11) yield

$$g = 9.81\text{m/s}^2$$

$$p = 1000\text{ kg/m}^3$$

$$\begin{aligned}
 p &= pgh \\
 &= \left(1000 \frac{\text{kg}}{\text{m}^3}\right) \left(9.81 \frac{\text{m}}{\text{s}^2}\right) (0.1\text{m}) \\
 &= 0.981\text{kPa}
 \end{aligned}
 \tag{2.11}$$

The actual pressure value in this experiment 4 can be obtained as shown in experiment 3. Therefore, based on the average voltage value in Table 4.8, actual pressure in this experiment 4 can be calculated by referring the characteristic plotted graph of Figure 4.10. The actual pressure of the experiment is calculated by using Equation (4.7) and tabulated in Table 4.9. The graph of theoretical and actual pressure (kPa) versus depth (cm) is plotted as Figure 4.14. In Figure 4.14, theoretical and actual pressure is linearly proportional to the water depth. The readings almost got the same linearly pattern.

Calculation for Actual Pressure

In Figure 4.10, the equation formula is given as shown in the following:

$$V_{out} = V_s * (0.0012858 * P + 0.04) \quad (4.7)$$

$$P = \frac{\frac{V_{out}}{V_s} - 0.04}{0.0012858}$$

Example actual pressure calculation at 0 cm of water depth:

Let $V_s = 5 V$,

$$P = \frac{\frac{0.863}{5} - 0.04}{0.0012858} = 103.13 \text{ kPa}$$

Table 4.9: Percentage error of depth sensor in experiment 4

Depth (cm)	Theoretical pressure (kPa)	Actual pressure (kPa)	Percentage of error (%)
0	101.33	103.13	1.78
10	102.31	104.37	2.01
20	103.29	105.30	1.95
30	104.27	106.24	1.89
40	105.25	107.02	1.68
50	106.23	107.79	1.47
60	107.22	108.73	1.41
70	108.20	109.66	1.35
80	109.18	110.44	1.15
90	110.16	111.37	1.10
100	111.14	112.30	1.04

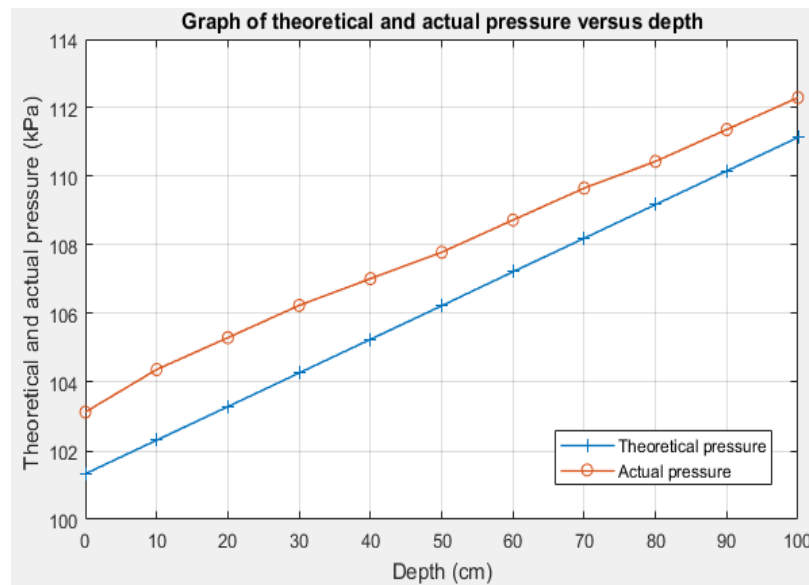


Figure 4.14: Theoretical and actual pressure (kPa) versus depth (m)

Performance of Pressure System

To evaluate the performance of the depth sensor MPX 5700, the percentage of error is calculated as following Equation (4.8):

$$\text{Percentage of error} = \left| \frac{\text{Theoretical pressure (kPa)} - \text{Actual pressure (kPa)}}{\text{Actual pressure (kPa)}} \right| \times 100\% \quad (4.8)$$

The first value (0 cm of water depth) obtained in the experiment is set as reference value. All the theory and actual pressure of the experiment are recorded in the Table 4.9. Each of the error percentage calculation at different depth is continued by using the Equation (4.8) and tabulated in Table 4.9. The depth sensor performance is analysed by determining the average percentage of error as show in the following:

$$\text{Average percentage of error} = \frac{\text{Total percentage of error}}{\text{Number of reading}} = \frac{16.83}{11} = 1.53\%$$

From the calculation above, the average percentage of error is 1.53%. The minimum percentage error is 1.04 %, up to 1 kPa while the maximum percentage error is 2.01%, up to 2 kPa. The depth sensor performance is accpetable since it has a low percentage of error to measure the pressure in underwater. The analysis shows that it can accurately measure the pressure value in underwater since the actual water pressure value from the experiment is approximate to the theoretical water pressure value.

4.3.5 Experiment 5: Depth Control of UUV

The output signal voltage of depth sensor obtained in experiment 4 can be used to calculate the analog reading value of the depth sensor. The analog reading of the depth sensor is calculated by using the Equation (4.9). From the calculated analog value as shown in Table 4.10 and Figure 4.15, it can be concluded that the analog calculation value approximately increases 1 value in every 10 cm of water depth increases.

Table 4.10: Analog reading of depth sensor at different depth

Depth (cm)	Output voltage (V)	Analog calculation value of depth sensor
0	0.863	176.82 \approx 177
10	0.871	177.97 \approx 178
20	0.877	179.20 \approx 179
30	0.883	180.43 \approx 180
40	0.888	181.66 \approx 182
50	0.893	182.89 \approx 183
60	0.899	184.12 \approx 184
70	0.905	185.34 \approx 185
80	0.910	186.37 \approx 186
90	0.916	187.60 \approx 188
100	0.922	188.83 \approx 189

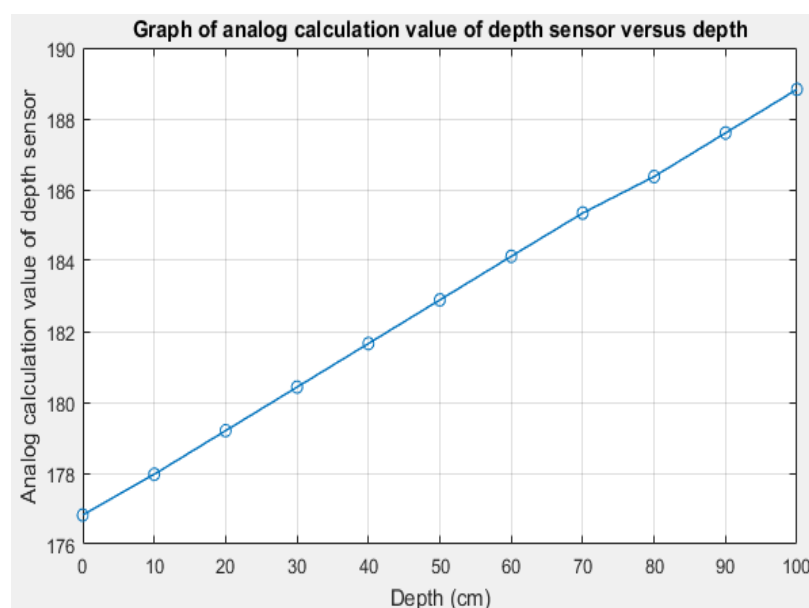


Figure 4.15: Graph of analog calculation value of depth sensor versus depth

Analog Reading Calculation of Depth Sensor

The equation formula is shown as the following:

$$\frac{\text{Analog reading}}{1024} = \frac{\text{Output signal voltage (V)}}{5V}$$

$$\text{Analog reading} = \frac{\text{Output signal voltage (V)}}{5V} \times 1024 \quad (4.9)$$

Example analog reading calculation at 0 cm of water depth,

$$\text{Analog reading} = \frac{0.863 V}{5 V} \times 1024$$

$$\text{Analog reading} = 176.7424 \approx 177$$

Depth Control Performance of UUV

In experiment 5, the actual depth of UUV is measured and tabulated in Table 4.11. To evaluate the depth control performance of the depth sensor MPX 5700 in UUV, the percentage of error is calculated as following Equation (4.10),

$$\text{Percentage of error} = \left| \frac{\text{Desired depth of UUV (cm)} - \text{Actual depth of UUV in experiment (cm)}}{\text{Desired depth of UUV (cm)}} \right| \times 100\% \quad (4.10)$$

Each of the error percentage calculation at different depth of UUV is continued by using the Equation (4.10) and tabulated in Table 4.11. The depth control performance of depth sensor in UUV is analysed by determining the average percentage of error. The average percentage of error is calculated as following:

$$\text{Average percentage of error} = \frac{\text{Total percentage of error}}{\text{Number of reading}} = \frac{38.52}{10} = 3.85 \%$$

From the calculation above, the average percentage of error is 3.85 %. . The minimum percentage error is 1.25 %, 1 cm while the maximum percentage error is 8.33 %, 2.50 cm. It shows that this sensor has a low percentage of error to control the water depth of UUV in underwater. The analysis shows that it can accurately measure the water depth in underwater since the actual depth of UUV in the experiment is approximate to the desired depth of UUV.

Table 4.11: Percentage error of depth control in UUV

Desired depth of UUV (cm)	Actual depth of UUV in experiment (cm)				Percentage of error (%)
	1 st	2 nd	3 rd	Average	
10	10.0	10.5	11.0	10.5	5.00
20	21.0	21.5	20.5	21.0	5.00
30	26.0	27.5	29.0	27.5	8.33
40	36.5	40.5	38.5	38.5	3.75
50	55.0	53.0	51.0	53.0	6.00
60	62.0	59.5	64.5	62.0	3.33
70	66.5	68.5	70.5	68.5	2.14
80	78.0	80.0	79.0	79.0	1.25
90	89.5	92.0	95.5	92.0	2.22
100	98.5	99.5	97.5	98.5	1.50

Therefore, by applying the calculated analog reading in Arduino coding, the underwater depth of the UUV can be controlled as desired. This means that if we want an UUV submerges to the water depth of 50 cm from the water surface, we can just changing the analog reading in Arduino coding to 183. The UUV will sinks until to the water depth of 50 cm. Besides that, the UUV also can keep its depth while moving in underwater.

As a conclusion in experiment 2, 3, 4 and 5, MPX5700AP pressure sensor shows a good performance to measure the pressure value in air and underwater. Besides that, it also can be used as a depth sensor for underwater vehicle in underwater depth control application since it has a wide range of measurements.

4.3.6 Experiment 6: Heading Degree Analysis of Magnetometer

In experiment 6, the measured output heading degree is measured three times to get an more accurate average output heading degree and recorded in Table 4.12. The output reading graph of magnetometer is plotted as shown in Figure 4.16. The average measured output heading degree almost has the same pattern as the actual heading degree of magnetometer.

Table 4.12: Output heading degree and percentage error of magnetometer

Reading	Actual heading degree (°)	Measured output heading degree (°)				Percentage of error (%)
		1 st	2 nd	3 rd	Average	
1	20	18.5	21.5	24.5	21.5	7.50
2	40	36.0	38.0	40.0	38.0	5.00
3	60	64.5	67.5	61.5	64.5	7.50
4	80	83.5	82.0	80.5	82.0	2.50
5	100	102.5	107.5	105.0	105.0	5.00
6	120	134.0	128.0	122.0	128.0	6.67
7	140	133.0	130.0	136.0	133.0	5.00
8	160	168.0	172.0	176.0	172.0	7.50
9	180	187.0	184.0	181.0	184.0	2.22
10	200	211.0	209.0	207.0	209.0	4.50

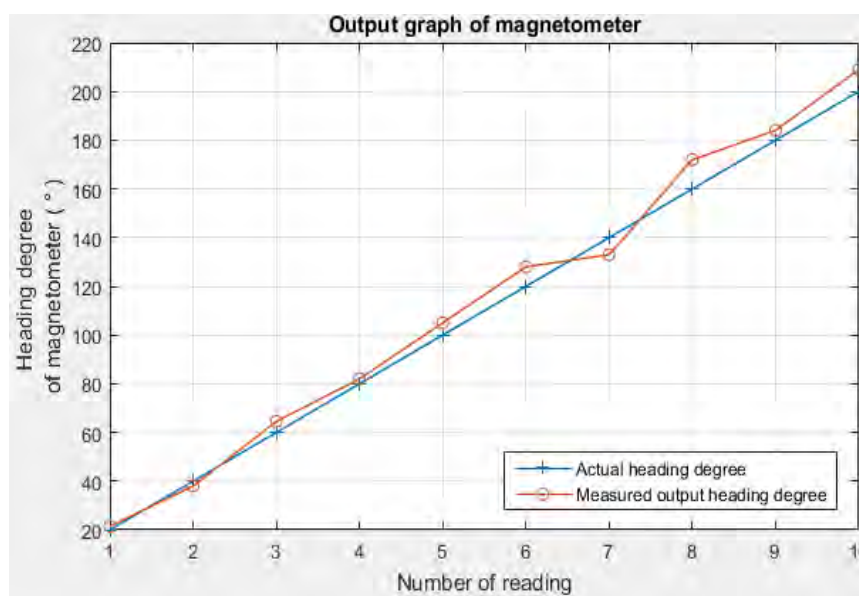


Figure 4.16: Output graph of magnetometer

Heading Degree Measurement Performance of Magnetometer

To evaluate the heading degree measurement performance of the magnetometer, the percentage of error is calculated as following Equation (4.11),

Percentage of error

$$= \left| \frac{\text{Actual heading degree } (^{\circ}) - \text{Average measured output heading degree } (^{\circ})}{\text{Actual heading degree } (^{\circ})} \right| \times 100\% \quad (4.11)$$

The first value (0° of actual heading degree) obtained in the experiment is set as reference value. Each of the error percentage calculation at different heading degree is continued by using the Equation (4.11) and tabulated in Table 4.12. The heading degree measurement performance of magnetometer is analysed by determining the average percentage of error. The average percentage of error is calculated as following:

$$\text{Average percentage of error} = \frac{\text{Total percentage of error}}{\text{Number of reading}} = \frac{53.39}{10} = 5.34 \%$$

From the calculation, the average percentage of error is 5.34 %. The minimum percentage error is 2.22 % while the maximum percentage error is 7.50 %. It shows that this sensor has a low percentage of error to measure the heading degree. From this experiment, the analysis shows that it can be used in underwater to measure the heading degree of UUV accurately since the measured output heading degree in the experiment is approximate to the actual heading degree.

4.3.7 Experiment 7: Leakage Sensor Testing

As shown in the Figure 4.18, the green LED is turned on and the red LED is kept off when there is no water is detected by the leakage sensor. However, in Figure 4.19, the green LED is turned off and the red LED is turned on when the leakage sensor is immersed into a glass of water. This means that there is a signal when there is water detected by the leakage sensor. Therefore, it can be used to detect the water intrusion of the integrated sensor casing to prevent further damage to the components inside the casing.

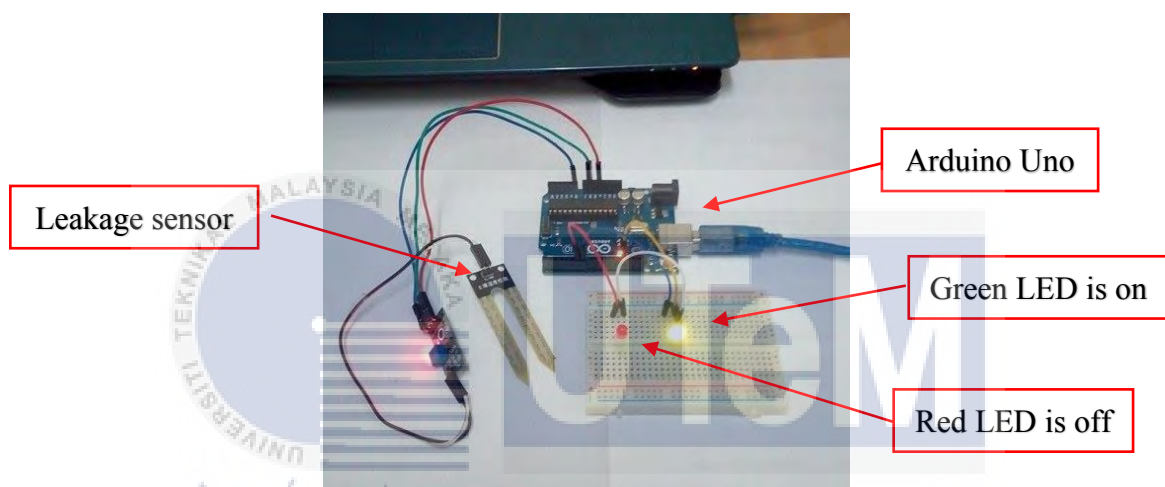


Figure 4.17: The green LED is on when no water is detected

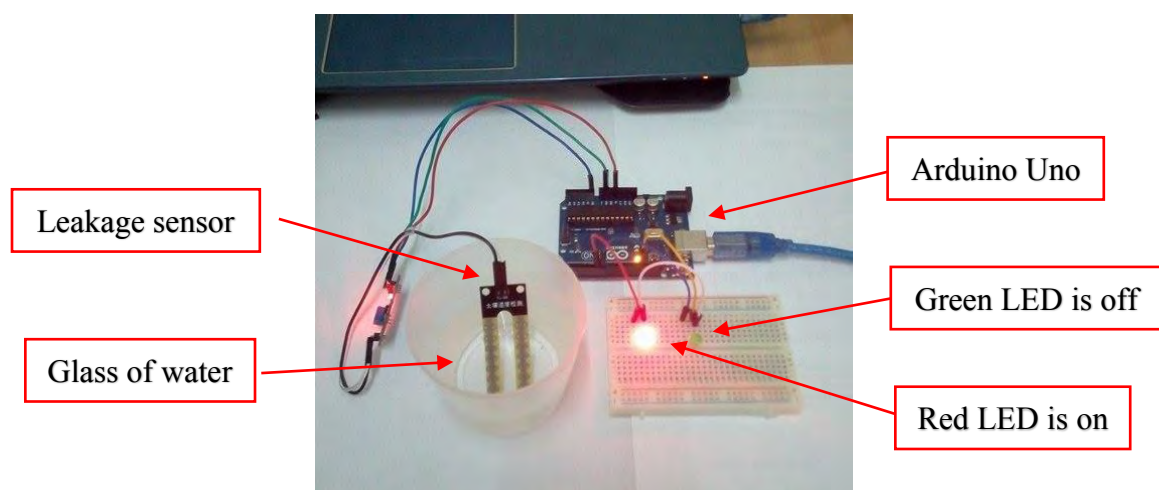


Figure 4.18: The red LED is ON when leakage sensor is immersed into the water

4.4 Summary

There are differences between the measured value (actual value) from the experiment and theoretical value. This is because the environment factor affect the reading during the experiment. Based on the calculation, IMU sensor, depth sensor and magnetometer have low percentage of error as shown in Table 4.13. Their error and results are accepted because the error percentage of each sensor is low and still in accepted range. In conclusion, all of the sensors show good performance in experiments in terms of accuracy and sensitivity.

Table 4.13: Summary of error percentage of sensors

Sensors	Experiment	Percentage of error (%)
IMU	1	0.60
	2	2.84
Depth	3	3.25
	4	1.53
	5	3.85
Magnetometer	6	5.34
Leakage	7	-

اونيورسيتي تيكنيكل مليسيا ملاك

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

CHAPTER 5

CONCLUSION AND RECOMMENDATION

5.1 Conclusion

As a conclusion, the objectives of this Final Year Project (FYP) are achieved. The integrated sensor for UUV is successfully developed and ready to collect the data in underwater with low estimation error. The progress to complete this project is smooth because the problems faced during the project progress can be solved in an easy way.

Before the project is started, many information and ideas related to the project title are studied and reviewed. More than 20 journals or papers are reviewed and the useful information are extracted for the literature review with citation. The sensors and microcontrollers are studied and selected to develop an integrated sensor for underwater application. The sensors selected to develop an integrated sensor are IMU sensor, depth sensor, magnetometer and leakage sensor. Each of the sensor is tested with microcontroller to make sure that sensors are function well and operate properly. The serial monitor data and MATLAB real-time data from the sensors are collected and analysed. The results came out as expected before the sensor testing.

For 5 DOF IMU sensor, MATLAB real-time simulation is successfully communicated with Arduino by using serial communication port, the output graph of accelerometer and gyroscope are matched with movement sequences of UUV. For pressure sensor MPX5700, it gives high consistency and accuracy output data results in MATLAB real-time simulation. It can measure the pressure in air or underwater with percentage error of 3.25 % and 1.53 % respectively. It even can be used as a depth sensor for UUV in depth control application with percentage error of 3.85 %. For magnetometer, it can be used to measure the heading degree of UUV with percentage error of 5.34 %. Lastly, leakage sensor

can be used to detect the water intrusion of the integrated sensor casing to prevent further damage to the components inside the casing. The result is the integrated sensor able to collect the data for the analysis movement of UUV with low estimation error and high sensitivity.

In this FYP, the specifications and function of different types of sensors for underwater application are reviewed and studied. The methodology to interface the microcontroller with the sensors and MATLAB real-time simulation are applied and improved. The skill to write the Arduino coding and MATLAB coding for the sensor are also improved as well.

5.2 Future Work and Recommendation

For the recommendation of this project, others types of the sensors or devices such as, sonar sensor, laser sensor and underwater imaging devices are highly recommended in the development of integrated sensor. Their functionality and performance are required to be studied and tested in order to apply them in underwater industry. Besides that, wireless integrated sensor are also highly recommended and developed. In this way, we can collect the data in underwater without any wire connection between the integrated sensor and computer. It is very important in the underwater industry because wireless integrated sensor able to carry out the more challenging tasks in underwater.

In conclusion, this project will give benefit to the underwater researchers and can apply in underwater industries with minimum cost implementation and percentage error. In the future, the performance of integrated sensor for underwater industry is highly improved based on three aspects: accuracy, sensitivity and efficiency.

REFERENCES

- [1] M. S. M. Aras, S. S. Abdullah, A. F. N. A. Rahman, M. F. Basar, A. M. Kassim, H. I. Jaafar, "Analysis of Movement for Unmanned Underwater Vehicle Using a Low Cost Integrated Sensor," in AIP Conference Proceedings, 2015.
- [2] M. S. M. Aras, F. A. Azis, M. N. Othman, S. S. Abdullah, "A Low Cost 4 DOF Remotely Operated Underwater Vehicle Integrated With IMU and Pressure Sensor," in 4th International Conference on Underwater System Technology: Theory and Applications 2012, 2012, pp. 18-23, 2012.
- [3] R. D. Christ and R. L. Wernli Sr, "The ROV Manual: A User Guide for Observation-Class Remotely Operated Vehicles," 1st Ed., Oxford UK: Elsevier Ltd., 2007.
- [4] M. S. M. Aras, H. A. Kasdirin, M. H. Jamaluddin and M. F. Basar, "Design and Development of an Autonomous Underwater Vehicle (AUV-FKEUTE_M)", in Proceedings of Malaysian Technical Universities Conference on Engineering and Technology (MUCEET2009), 2009, pp 1-5.
- [5] F. A. Azis, M. S. M. Aras, S. S. Abdullah, Rashid, M. Z. A. and M. N. Othman, *Procedia Engineering* 41, pp 554-560, 2102
- [6] M.S.M Aras, M.F. Basar, S.S. Abdullah, F.A. Azis, F. A. Ali, "Analysis Movement of Unmanned Underwater Vehicle Using the Inertial Measurement Unit;" 2013
- [7] J. Busquets, J. V. Busquets, A. Perles, R. Mercado, R. Saez, J. J. Serrano, F. Albentosa, J. Gilabert, "Communication Challenges for Dual Configuration of ASV and AUV in Twinned Coordinated Navigation," 2014.
- [8] D. Hazry, M. Sofian, A. Z. Azfar, "Study of Inertial Measurement Unit Sensor," in Proceedings of the International Conference on Man-Machine Systems, 2009.

- [9] T. Mau and J. Mahoney, "Control System for an Underwater Remotely Operated Vehicle," 2007.
- [10] G. Antonelli, "Underwater Robots: Motion and Force Control of Vehicle-Manipulator Systems," 2nd Ed., Cassino, Italy: Springer, 2006.
- [11] A. M. Plotnik and S. M. Rock, "A Multi-sensor Approach to Automatic Tracking of Midwater Targets by an ROV," in Proceedings of the AIAA, 2007, pp. 1 -5.
- [12] B. Sithole, S. Rimer, K. Ouahada, C. Mikeka, J. Pinifolo, "Smart Water Leakage Detection and Metering Device," 2016.
- [13] M. Jeunnette, "Flow Sensor Integration for AUV and ROV Applications," 2001.
- [14] J. Kennedy, "Decoupled Modelling and Controller Design for the Hybrid Autonomous Underwater Vehicle: MACO," 2002.
- [15] G. A. Ramadass, N. Vedachalam, V. Balanagajyothi, R. Ramesh, M. A. Atmanand, "A Modelling Tool for Sensor Selection for Inertial Navigation System used in Underwater Vehicles," 2013.
- [16] C. R. Rocha, R. M. Brancoy, L. A. D. Cruz, M. V. Schollx, M. M. Cezarx and Felipe D., "Design Aspects of An Open Platform For Underwater Robotics Experimental Research," 2014.
- [17] P. Jantapremjit, "Design and Development of a Remotely Operated Vehicle," in the Second TSME International Conference on Mechanical Engineering, 2011.
- [18] I. N. Jleta, O. A. Taleb. "Land Vehicle Navigation System Based Low Cost Inertial Sensor," 2015.
- [19] F. Bonin-Font, M. Massot-Campos, P. L. Negre-Carrasco, G. Oliver-Codina and J. P. Beltran, "Inertial Sensor Self-Calibration in a Visually-Aided Navigation Approach for a Micro-AUV," 2015.
- [20] C. Stal, G. Deruyter, M. Paelinck, "Towards Cost-Efficient Prospection and 3D Visualization of Underwater Structures using Compact ROVs," 2015.
- [21] S.K. Choi and O. Y. Easterday, "An Underwater Vehicle Monitoring System and its Sensors," 2014.

- [22] M. S. M. Aras, F. A. Azis, S. S. Abdullah, D. C. Lee, W. T. Lim, F. A. Ali and M. N. Othman, "Study on the Effect of Shifting 'zero' in Output Membership Function on Fuzzy Logic Controller of the ROV using Micro-box," 2015.
- [23] M.S.M Aras, S.S Abdullah, S.S Shafei, "Investigation and Evaluation of Low cost Depth Sensor System Using Pressure Sensor for Unmanned Underwater Vehicle," in journal of electrical engineering, 2012.



APPENDIX B

SENSORS CODING

Accelerometer

```

#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_ADXL345_U.h>

Adafruit_ADXL345_Unified accel = Adafruit_ADXL345_Unified(12345);

void displaySensorDetails(void)
{
  sensor_t sensor;
  accel.getSensor(&sensor);
  Serial.println("-----");
  Serial.print ("Sensor:   "); Serial.println(sensor.name);
  Serial.print ("Driver Ver: "); Serial.println(sensor.version);
  Serial.print ("Unique ID:  "); Serial.println(sensor.sensor_id);
  Serial.print ("Max Value:  "); Serial.print(sensor.max_value); Serial.println(" m/s^2");
  Serial.print ("Min Value:  "); Serial.print(sensor.min_value); Serial.println(" m/s^2");
  Serial.print ("Resolution: "); Serial.print(sensor.resolution); Serial.println(" m/s^2");
  Serial.println("-----");
  Serial.println("");
  delay(500);
}

void displayDataRate(void)
{
  Serial.print ("Data Rate:  ");

  switch(accel.getDataRate())
  {
    case ADXL345_DATARATE_3200_HZ:
      Serial.print ("3200 ");

```

```

break;
case ADXL345_DATARATE_1600_HZ:
  Serial.print ("1600 ");
  break;
case ADXL345_DATARATE_800_HZ:
  Serial.print ("800 ");
  break;
case ADXL345_DATARATE_400_HZ:
  Serial.print ("400 ");
  break;
case ADXL345_DATARATE_200_HZ:
  Serial.print ("200 ");
  break;
case ADXL345_DATARATE_100_HZ:
  Serial.print ("100 ");
  break;
case ADXL345_DATARATE_50_HZ:
  Serial.print ("50 ");
  break;
case ADXL345_DATARATE_25_HZ:
  Serial.print ("25 ");
  break;
case ADXL345_DATARATE_12_5_HZ:
  Serial.print ("12.5 ");
  break;
case ADXL345_DATARATE_6_25HZ:
  Serial.print ("6.25 ");
  break;
case ADXL345_DATARATE_3_13_HZ:
  Serial.print ("3.13 ");
  break;
case ADXL345_DATARATE_1_56_HZ:
  Serial.print ("1.56 ");
  break;
case ADXL345_DATARATE_0_78_HZ:
  Serial.print ("0.78 ");
  break;
case ADXL345_DATARATE_0_39_HZ:
  Serial.print ("0.39 ");
  break;
case ADXL345_DATARATE_0_20_HZ:
  Serial.print ("0.20 ");
  break;
case ADXL345_DATARATE_0_10_HZ:
  Serial.print ("0.10 ");
  break;
default:
  Serial.print ("???? ");
  break;
}

```

```

Serial.println(" Hz");
}

void displayRange(void)
{
  Serial.print ("Range:   +/- ");

  switch(accel.getRange())
  {
    case ADXL345_RANGE_16_G:
      Serial.print ("16 ");
      break;
    case ADXL345_RANGE_8_G:
      Serial.print ("8 ");
      break;
    case ADXL345_RANGE_4_G:
      Serial.print ("4 ");
      break;
    case ADXL345_RANGE_2_G:
      Serial.print ("2 ");
      break;
    default:
      Serial.print ("?? ");
      break;
  }
  Serial.println(" g");
}

void setup(void)
{
  Serial.begin(9600);
  Serial.println("Accelerometer Test"); Serial.println("");

  if(!accel.begin())
  {
    Serial.println("Ooops, no ADXL345 detected ... Check your wiring!");
    while(1);
  }

  accel.setRange(ADXL345_RANGE_16_G);
  // displaySetRange(ADXL345_RANGE_8_G);
  // displaySetRange(ADXL345_RANGE_4_G);
  // displaySetRange(ADXL345_RANGE_2_G);

  displaySensorDetails();
  displayDataRate();
  displayRange();
  Serial.println("");
}

```

```

void loop(void)
{
  sensors_event_t event;
  accel.getEvent(&event);
  Serial.print("X: "); Serial.print(event.acceleration.x); Serial.print(" ");
  Serial.print("Y: "); Serial.print(event.acceleration.y); Serial.print(" ");
  Serial.print("Z: "); Serial.print(event.acceleration.z); Serial.print(" ");
  Serial.println("m/s^2 ");
  delay(500);
}

```

Gyroscope

```

#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_L3GD20_U.h>

Adafruit_L3GD20_Unified gyro = Adafruit_L3GD20_Unified(20);

void displaySensorDetails(void)
{
  sensor_t sensor;
  gyro.getSensor(&sensor);
  Serial.println("-----");
  Serial.print ("Sensor: "); Serial.println(sensor.name);
  Serial.print ("Driver Ver: "); Serial.println(sensor.version);
  Serial.print ("Unique ID: "); Serial.println(sensor.sensor_id);
  Serial.print ("Max Value: "); Serial.print(sensor.max_value); Serial.println(" rad/s");
  Serial.print ("Min Value: "); Serial.print(sensor.min_value); Serial.println(" rad/s");
  Serial.print ("Resolution: "); Serial.print(sensor.resolution); Serial.println(" rad/s");
  Serial.println("-----");
  Serial.println("");
  delay(500);
}

void setup(void)
{
  Serial.begin(9600);
  Serial.println("Gyroscope Test"); Serial.println("");

  gyro.enableAutoRange(true);

  if(!gyro.begin())
  {
    Serial.println("Ooops, no L3GD20 detected ... Check your wiring!");
    while(1);
  }
}

```

```

    }
    displaySensorDetails();
}
void loop(void)
{
    sensors_event_t event;
    gyro.getEvent(&event);
    Serial.print("X: "); Serial.print(event.gyro.x); Serial.print(" ");
    Serial.print("Y: "); Serial.print(event.gyro.y); Serial.print(" ");
    Serial.print("Z: "); Serial.print(event.gyro.z); Serial.print(" ");
    Serial.println("rad/s ");
    delay(500);
}

```

Magnetometer

```

#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_HMC5883_U.h>

Adafruit_HMC5883_Unified mag = Adafruit_HMC5883_Unified(12345);

void displaySensorDetails(void)
{
    sensor_t sensor;
    mag.getSensor(&sensor);
    Serial.println("-----");
    Serial.print ("Sensor:   "); Serial.println(sensor.name);
    Serial.print ("Driver Ver: "); Serial.println(sensor.version);
    Serial.print ("Unique ID:  "); Serial.println(sensor.sensor_id);
    Serial.print ("Max Value:  "); Serial.print(sensor.max_value); Serial.println(" uT");
    Serial.print ("Min Value:  "); Serial.print(sensor.min_value); Serial.println(" uT");
    Serial.print ("Resolution: "); Serial.print(sensor.resolution); Serial.println(" uT");
    Serial.println("-----");
    Serial.println("");
    delay(500);
}

void setup(void)
{
    Serial.begin(9600);
    Serial.println("HMC5883 Magnetometer Test"); Serial.println("");

    if(!mag.begin())
    {
        Serial.println("Oops, no HMC5883 detected ... Check your wiring!");
    }
}

```

```

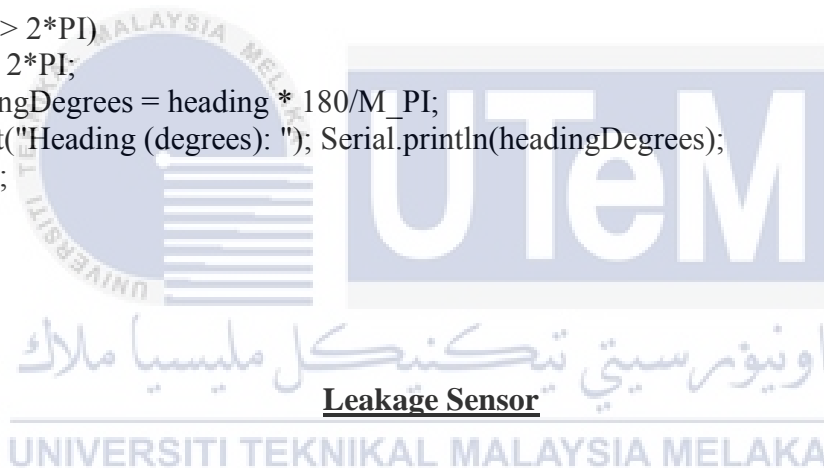
    while(1);
  }
  displaySensorDetails();
}

void loop(void)
{
  sensors_event_t event;
  mag_getEvent(&event);
  Serial.print("X: "); Serial.print(event.magnetic.x); Serial.print(" ");
  Serial.print("Y: "); Serial.print(event.magnetic.y); Serial.print(" ");
  Serial.print("Z: "); Serial.print(event.magnetic.z); Serial.print(" "); Serial.println("uT");
  float heading = atan2(event.magnetic.y, event.magnetic.x);
  float declinationAngle = 0.22;
  heading += declinationAngle;

  if(heading < 0)
    heading += 2*PI;

  if(heading > 2*PI)
    heading -= 2*PI;
  float headingDegrees = heading * 180/M_PI;
  Serial.print("Heading (degrees): "); Serial.println(headingDegrees);
  delay(500);
}

```



```

int Pin = A0;
int greenLED = 6;
int redLED = 7;
int thresholdValue = 800;

void setup()
{
  pinMode(Pin, INPUT);
  pinMode(greenLED, OUTPUT);
  pinMode(redLED, OUTPUT);
  digitalWrite(greenLED, LOW);
  digitalWrite(redLED, LOW);
  Serial.begin(9600);
}

void loop()
{
  int sensorValue = analogRead(Pin);

```

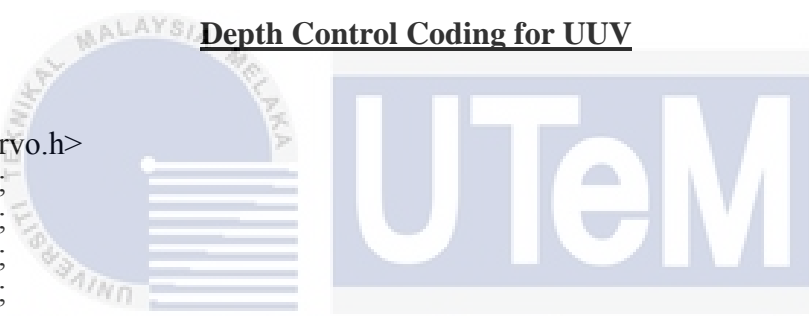
```

Serial.print(sensorValue);

if(sensorValue < thresholdValue)
{
  Serial.println("No water leakage");
  digitalWrite(redLED, LOW);
  digitalWrite(greenLED, HIGH);
}
else {
  Serial.println("Water leakage!!!");
  digitalWrite(redLED, HIGH);
  digitalWrite(greenLED, LOW);
}
delay(500);
}

```

Depth Control Coding for UUV



```

#include <Servo.h>
Servo servo1;
Servo servo2;
Servo servo3;
Servo servo4;

void setup() {
  Serial.begin(115200);
  servo1.attach(9);
  servo2.attach(10);
  servo3.attach(11);
  servo4.attach(12);
  delay(1000);
}

void loop() {
  float voltage = analogRead(A1);
  float pressure=((voltage/1024.0)-0.04)/0.0012858;
  Serial.println(pressure);
  delay(10);
  Serial.println(voltage);
  delay(10);

  if (voltage <= 177) {
    servo1.writeMicroseconds(1500); // Send signal to ESC.
    servo2.writeMicroseconds(1500); // Send signal to ESC.
    servo3.writeMicroseconds(1600); // Send signal to ESC.
    servo4.writeMicroseconds(1600); // Send signal to ESC.

```



```

delay(1000);
servo1.writeMicroseconds(1500); // Send signal to ESC.
servo2.writeMicroseconds(1500); // Send signal to ESC.
servo3.writeMicroseconds(1500); // Send signal to ESC.
servo4.writeMicroseconds(1500); // Send signal to ESC
delay(50);
Serial.println("DOWN");
}

else if (voltage > 177) {
servo1.writeMicroseconds(1500); // Send signal to ESC.
servo2.writeMicroseconds(1500); // Send signal to ESC.
servo3.writeMicroseconds(1500); // Send signal to ESC.
servo4.writeMicroseconds(1500); // Send signal to ESC
delay(50);
Serial.println("STOP");
}
}
}

```

MATLAB Real-Time Simulation Coding

One of the real-time simulation coding for accelerometer is shown as following:

```

clc; clear all;
comport = serial('COM5', 'BaudRate', 115200); % setup comport
fopen(comport); % Open comport
x=0;
while(x<100)
x=x+1;
y1(x)=fscanf(comport, '%d'); % receive ADC1
y2(x)=fscanf(comport, '%d'); % receive ADC1
y3(x)=fscanf(comport, '%d'); % receive ADC1
drawnow;

plot(y1,'r','linewidth',0.5)
grid on; hold on;
plot(y2,'b','linewidth',0.5)
grid on; hold on;
plot(y3,'k','linewidth',0.5)

title('Accelerometer Output Graph');
xlabel('Time (s)');
ylabel('Acceleration (m/s-2)');
legend('x-axis','y-axis','z-axis')
pause(0.1);
end
fclose(comport); % Close comport
delete(comport); % Clear comport

```

APPENDIX C

ARDUINO SERIAL MONITOR OF ACCELEROMETER

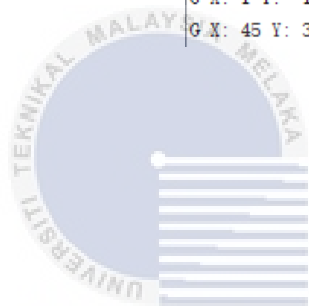
COM4 (Arduino/Genuino Mega or Nano)	COM4 (Arduino/Genuino Mega or Nano)	COM4 (Arduino/Genuino Mega or Nano)
<pre>X: 9.98 Z: -2.35 m/s^2 X: 9.57 Y: -1.06 Z: -2.24 m/s^2 X: 9.49 Y: -1.02 Z: -2.16 m/s^2 Accelerometer Test Sensor: ADXL345 Driver Ver: 1 Unique ID: 12345 Max Value: -156.91 m/s^2 Min Value: 156.91 m/s^2 Resolution: 0.04 m/s^2</pre>	<pre>X: -1 m/s^2 X: -0.55 Y: 9.89 Z: -0.78 m/s^2 X: -0.55 Y: 9.92 Z: -0.67 m/s^2 Accelerometer Test Sensor: ADXL345 Driver Ver: 1 Unique ID: 12345 Max Value: -156.91 m/s^2 Min Value: 156.91 m/s^2 Resolution: 0.04 m/s^2</pre>	<pre>X: X: -0.55 Y: 0.04 Z: 9.45 m/s^2 O Z: 8.98 m/s^2 X: -0.51 Y: 0.04 Z: 9.38 m/s^2 X: -0.55 Y: 0.00 Z: 9.49 m/s^2 X: -0.55 Y: 0.04 Z: 9.45 m/s^2 Accelerometer Test Sensor: ADXL345 Driver Ver: 1 Unique ID: 12345 Max Value: -156.91 m/s^2 Min Value: 156.91 m/s^2 Resolution: 0.04 m/s^2</pre>
<pre>Data Rate: 100 Hz Range: +/- 16 g X: 9.41 Y: 0.00 Z: -2.31 m/s^2 X: 9.49 Y: -0.24 Z: -2.12 m/s^2 X: 9.92 Y: -0.43 Z: -2.24 m/s^2 X: 9.85 Y: -0.47 Z: -2.16 m/s^2 X: 9.61 Y: -0.24 Z: -2.47 m/s^2</pre>	<pre>Data Rate: 100 Hz Range: +/- 16 g X: -0.59 Y: 9.92 Z: -0.63 m/s^2 X: -0.59 Y: 9.96 Z: -0.98 m/s^2 X: -0.51 Y: 9.89 Z: -0.75 m/s^2 X: -0.51 Y: 9.89 Z: -0.86 m/s^2 X: -0.67 Y: 9.89 Z: -0.75 m/s^2</pre>	<pre>Data Rate: 100 Hz Range: +/- 16 g X: -0.59 Y: 0.04 Z: 9.38 m/s^2 X: -0.59 Y: 0.08 Z: 9.41 m/s^2 X: -0.59 Y: 0.04 Z: 9.38 m/s^2 X: -0.67 Y: 0.00 Z: 9.41 m/s^2 X: -0.59 Y: 0.04 Z: 9.41 m/s^2</pre>

APPENDIX D

ARDUINO SERIAL MONITOR OF GYROSCOPE

COM4 (Arduino/Genuino Mega or Mega 2560)

```
G X: 14 Y: -4 Z: -8
G X: 21 Y: -28 Z: -1
G X: -6 Y: 18 Z: -7
G X: 1 Y: -1 Z: -29
G X: 45 Y: 32 Z: -25
```



APPENDIX E



اونيورسيتي تيكنيكل مليسيا ملاك

ARDUINO SERIAL MONITOR OF MAGNETOMETER

HMC5883 Magnetometer Test

```
Sensor:      HMC5883
Driver Ver:   1
Unique ID:    12345
Max Value:    800.00 uT
Min Value:    -800.00 uT
Resolution:   0.20 uT
```

```
X: -29.00 Y: 20.64 Z: 1.22 uT
Heading (degrees): 157.17
X: -29.18 Y: 20.64 Z: 1.22 uT
Heading (degrees): 157.34
X: -29.09 Y: 20.64 Z: 1.22 uT
Heading (degrees): 157.25
```