DEVELOPMENT OF ROV FOR ORIENTATION CONTROL

CH'NG SWEE GUAN

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

Faculty of Electrical Engineering

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

2018

"I hereby declare that I have read through this report entitle "**DEVELOPMENT OF ROV FOR ORIENTATION CONTROL**" and found that it has comply the partial fulfilment for awarding the degree of Bachelor of Mechatronics Engineering.

Signature : ....

Supervisor's Name : SYED MOHAMAD SHAZALI BIN SYED ABDUL HAMID

Date : ....

I declare that this report entitles "**DEVELOPMENT OF ROV FOR ORIENTATION CONTROL**" 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 the candidature of any other degree.

Signature : ....

Name : CH'NG SWEE GUAN

Date : ....

To my beloved mother and father

ACKNOWLEDGEMENT

First and foremost, I, Ch'ng Swee Guan as an undergraduate student from Universiti Teknikal Malaysia Melaka (UTeM) would like to express my greatest appreciation and deepest gratitude to my supervisor, Syed Mohamad Shazali Bin Syed Abdul Hamid for his patient guidance, professional training, encouragement and valuable advice during this FYP. I am very thankful for providing me with a valuable comment about my work on this project.

My grateful thanks are also extended to Professor Madya Dr Ahmad Zaki Bin Shukor and Mr Zamani Bin Mohammad Sani also Universiti Teknikal Malaysia Melaka (UTeM) for offering me resources in running the program during the Final Year Project. Besides that, I would also like to thank all of my friends for sharing useful knowledge and always give support and motivation to me to work on this project.

Last but not least, I would take this opportunity to express my deepest gratitude to my parents for their continuous shower of love, unceasing encouragement and support throughout all these years. I derived inspiration from their sacrifice, encouragement from their faith, found happiness in their pride and all my strength from their unconditional love. Without the help of the particulars that I mentioned above, I might face difficulties during my Final Year Project. I thank and sense of gratitude to everybody who directly or indirectly offered their helping hand during the entire period of the Final Year Project.

ABSTRACT

In the field of underwater unmanned vehicle, Remotely Operated Vehicle (ROV) and Autonomous Underwater Vehicle (AUV) are mobile robots that replace human to do dangerous task such as carrying out operation under the deep ocean with the operator on the surface of the ocean. One of the problems faced by ROV is that the new operator cannot run the ROV perfectly because there is absence of direct orientation control of ROV. So, a steering system that uses the sense of motion to control a ROV is discussed in this project. In designing the ROV, SolidWorks software is used and undergoes various simulation tests such as stress and strain test, sustainability test and stability by referring the center of mass. This project was used MPU6050 sensor as a steering system to control the angle of rotation of ROV in yaw, pitch and roll. In this research, the performance of the ROV with steering system was evaluated in terms of manoeuvrability and ease of handling. The experiments are carried out in the lab pool to test the orientation control of the ROV through the communication between the ROV and the steering system using Arduino MEGA 2560. The result obtained from SolidWorks simulation tests was that the design of ROV has good stability in water due to its center of gravity. The drifting angle on pitch and roll was filtered by using Kalman Filter coding. The ROV was able to move in pitch and roll direction but not yaw because the angle of the rotation on yaw was drifted too much. The only solution for the drift angle on z axis is by replacing the MPU 6050 with MPU 9150.

ABSTRAK

Dalam bidang kenderaan tanpa pemandu bawah air, kenderaan kawalan jauh (ROV) dan kenderaan autonomi bawah air (AUV) adalah robot bergerak yang menggantikan manusia untuk melakukan tugas yang berbahaya seperti menjalankan operasi di bawah laut dalam dengan pengendali di permukaan lautan. Salah satu masalah yang dihadapi oleh ROV adalah bahawa pengendali baru tidak dapat menjalankan ROV dengan sempurna kerana tidak ada kawalan orientasi langsung ROV. Jadi, sistem pemanduan yang menggunakan isyarat gerakan untuk mengawal ROV dibincangkan di dalam projek ini. Untuk mereka bentuk ROV, perisian SolidWorks digunakan dan menjalani pelbagai ujian simulasi seperti tekanan dan ujian terikan, ujian kelestarian dan kestabilan dengan merujuk kepada pusat jisim. Projek ini digunakan sensor MPU6050 untuk mengawal sudut putaran ROV dan kayuria. Dalam kajian ini, ROV hanya bergerak dalam yaw, padang, dan roll. Eksperimen dilakukan di kolam lab untuk menguji kawalan orientasi ROV dari komunikasi antara ROV dengan sistem pemanduan menggunakan Arduino MEGA 2560. Hasil yang diperoleh dari ujian simulasi SolidWorks adalah bahwa desain ROV mempunyai kestabilan yang baik di dalam air ke pusat graviti. Sudut drifting di padang dan gulung ditapis dengan menggunakan pengekodan Kalman Filter. ROV dapat bergerak di arah padang dan gulung tetapi tidak mengecil kerana sudut putaran pada lekukan terlalu banyak. Satu-satunya penyelesaian untuk sudut drift pada paksi z adalah dengan menggantikan MPU 6050 dengan MPU 9150.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURE

LIST OF ABBREVIATION

ROV - Remotely Operated Vehicle

AUV - Autonomous Underwater Vehicle

PID- Proportional Integral Derivative

FLC- Fuzzy Logic Controller

FYP- Final Year Project

CG- Center of Gravity

CB- Center of Bouyancy

DOF- Degrees of Freedom

CFD- Computational Fluid Dynamics

PMM- Planar Motion Mechanism

SMC- Sliding Mode Controller

CAD - Computer-Aided Design

FEA- Finite Element Analysis

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

On the Earth's surface, 71 percent covered by the oceans and it still has plenty of resources that not yet been explored and extracted [1]. Underwater vehicles are used to replace diver that works in the hazardous environment such as explore, investigate or recovery of the item under the deep ocean. There are 2 types of underwater vehicles which are unmanned vehicles and manned vehicles. Autonomous Underwater Vehicle (AUV) and Remotely Operated Vehicle (ROV) are categorized under unmanned vehicles. A ROV is a tethered underwater robot that allows the operator to stay above the ocean and give the command to ROV, whereas AUV is controlled by pre-programmed instructions set by the operator on the controller which is equip with sensors of the robot [2]. The overview of underwater vehicles is shown in Figure 1.

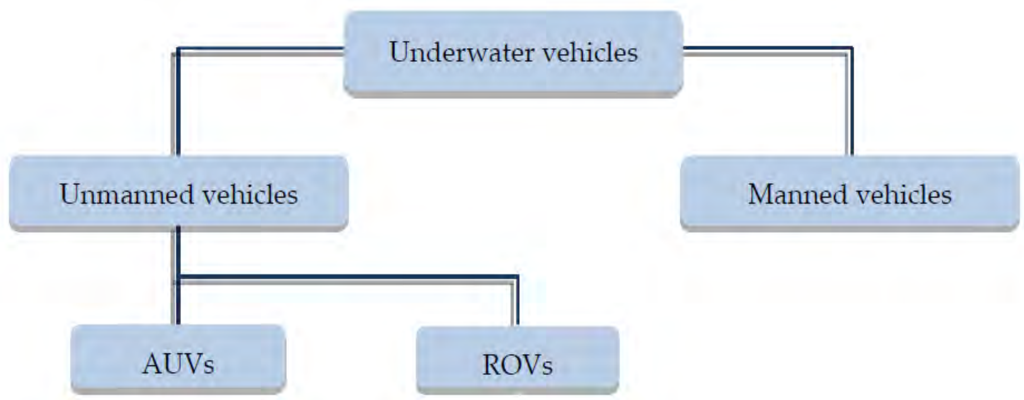


Figure 1.1: Overview of Underwater Vehicles. [3]

1.2 MOTIVATION

The demand for ROV is high in drilling support, construction support, repair, and maintenance. According to World ROV Operations Market Forecast, Douglas-Westwood expected annual expenditure on ROV operations has increased from USD 1.6 billion in 2013 to USD 2.4 billion in 2017. In the year between 2013 and 2017, drilling support of exploration has occupied 75% of the total expenditure while the construction support occupies for 20% and repair and maintenance for 4%. The majority of global ROV demand is from Africa [4].The expenditure in global work-class ROV operations from 2008 to 2017 is shown in Figure 1.2.

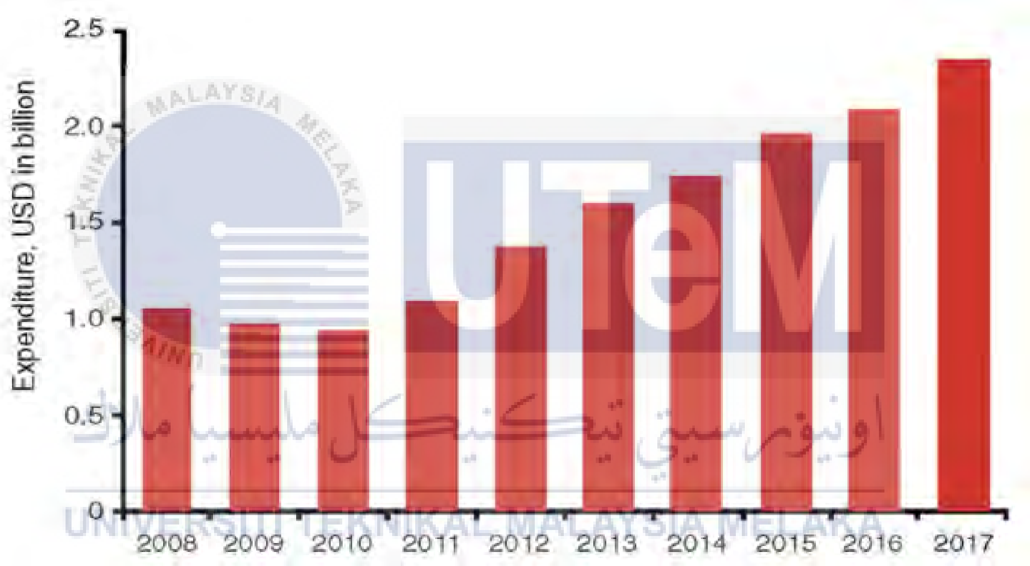


Figure 1.2: The expenditure in global work-class ROV operations from 2008 to 2017. [4]

There are many researchers and engineers in western countries that try to design and develop the unmanned vehicles for underwater exploration. In Malaysia, the research of underwater technology is much lagging behind those western countries such as German, Japan, and the USA. As a result of unable to develop the underwater unmanned vehicle, tragedy of Malaysia Airlines flight MH370 submerge into the Indian Ocean occurred and gave a serious warning to everyone about the vital sign of ROV technology in Malaysia. Malaysia required paying about $20 million to $70 million to Ocean Infinity US Company for the trace of missing Malaysia Airlines flight MH370 [5].

1.3 PROBLEM STATEMENT

Recently, ROV started to grow as an important tool in various operation such as exploring, investigating, cleaning, latching or recovery of the item to protect our ocean resources. With the aid of ROV, human cognition of the ocean become higher. This application requires ROV to be able to navigate to a given point and controllable from the surface of the ocean [6] .Besides that, stability in orientation control is an important issue for ROV. On the ocean, the waves disturb the underwater vehicle severely. The change in direction and speed of currents in an irregular water has caused the operation of ROV become harder. Moreover, the operator of ROV needs to control the orientation of a ROV with the need to control the thrusters directly. The body of the ROV must be waterproof and able to withstand certain deep water pressure. It is not easy to fully seal the electronic components. Once the water entered the component, the circuit will then short circuited. The stability of ROV due to its center of gravity and center of buoyancy will also affect the performance of the ROV. The center of buoyancy should be slightly higher than the center of gravity for stable orientation control of ROV.

This project was developed in order to solve the difficulty in control a ROV using traditional button type joystick. New operator cannot run the ROV perfectly because there is absence of direct orientation control of ROV. The controller which sensing the motion will ease the user in controlling the ROV. Therefore, a steering system for ROV can make the user handle the movement easily without looking at the manual of control. Besides that, the control system is needed for the ROV to move in the desired position. There are various advanced techniques such as Proportional Integral Derivative (PID) controller and Fuzzy Logic controller (FLC) to be used as a controller of ROV. From the controller, a control algorithm can be made. A control algorithm not only built for flexible movement in various direction but also maintain the stability of orientation and move with constant velocity.

1.4 OBJECTIVE

There are three objectives that required to be achieved during this FYP

1. To design and develop a ROV for orientation control in yaw, pitch and roll.
2. To develop a steering system for ROV orientation which are yaw, pitch and roll using MPU 6050 sensors.
3. To evaluate the performance of a steering system developed in terms of maneuverability and ease of handling.

1.5 SCOPE

The scopes and limitations of this project are:

1. The ROV is required in small size, dimension less than 70cm x 50cm x 30cm and lightweight less than 5kg.
2. The orientation of ROV involves only yaw, pitch and roll.
3. Arduino Mega is used as microcontroller due to it has the high processing power and up to 54 General Purpose Input Output pins.
4. 6 DOF MPU 6050 sensors are used on the joystick and the body of the ROV to measure the angle of rotation.
5. The maximum depth for ROV is 2 meters.
6. MPU 6050 sensor is used as input to control the orientation of ROV.
7. Test on working space which are yaw, pitch and roll.
8. Test on time response when using MPU 6050 sensor to control orientation of ROV.

1.6 ORGANIZATION OF REPORT

For the content of chapter 2 on literature review, it involves the theoretical background that needed for this project. It also discussed the factors of choosing the design of ROV for this project. Lastly, it summarises the design proposed by the journals and the conclusion made from the review. Furthermore, chapter 3 methodology includes the block diagram and flowchart of operation and prototype of a ROV. The method of doing the design is also mentioned in this chapter. Moreover, chapter 4 for results and discussion has described several simulations using SolidWorks Simulation. Besides that, the value that the sensor detects is also recorded in this chapter. Last but not least, chapter 5 on conclusion and recommendation, future work reviews the whole information in details for this project. There are some suggestions for future work and recommendations.

1.7 SUMMARY

This chapter concludes that the importance of a ROV in the underwater application. The development of this technology is needed to further exploration of the ocean. This project is focused more on designing a control system to stabilize the orientation of the ROV.

CHAPTER 2

LITERATURE REVIEW

2.1 Theory and basic principles

2.1.1 Centre of gravity

The center of gravity is the average position of the weight of an object whereas the center of buoyancy is the center of mass of the floating or submerged body that caused by a displaced fluid. Most ROVs are designed to be as stable as practical. Most common design of ROV is open frame type. The open frame configuration also aids the stability of the vehicle which is related to the distance where both the center of gravity and the center of buoyancy intersect is known as metacentric height. The longer the distance between the center of buoyancy and the center of gravity, the more stable the vehicle and vice versa but the more maneuverable [7]. In order to prevent the ROV from rotating itself, the center of buoyancy should be located above the center of gravity [3]. Figure 2.1 shows the center of buoyancy located above the center of gravity.

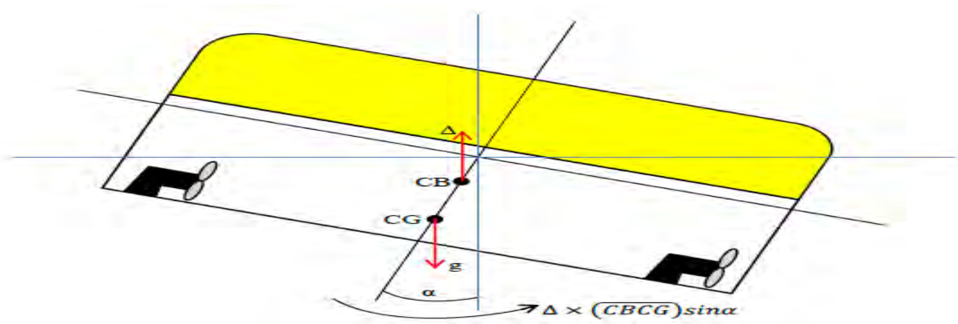


Figure 2.1: Righting moment of stable ROV. [3]

2.1.2 Stability of ROV

Passive Stability

The stability of ROV is the ability of ROV to return to original position when tipped or flipped by a disturbance such as a wave. BG is an important term in stability of underwater vehicles. BG is the distance between the center of gravity (CG) and center of buoyancy (CB). In Figure 2.2, the position of BG is shown [7].

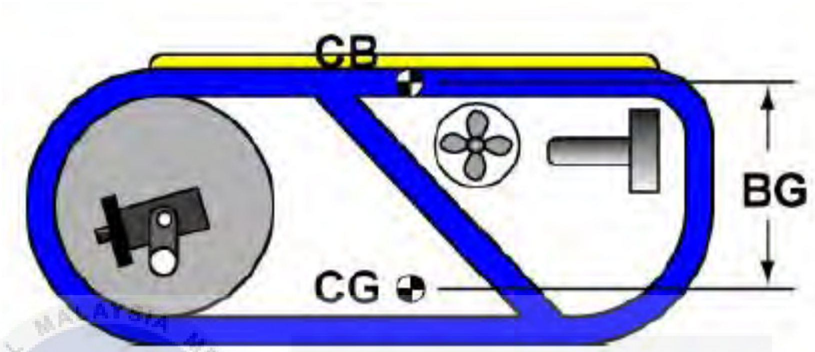


Figure 2.2: The distance between CB and CG is BG.

According to a special case of moments, the couple is consist of two equal but opposite forces that act upon a body [8]. It is about the same that happens on ROV during operation. When the BG increases, d will increase and produce a larger turning torque. From the formula of torque [9]:

$$\tau = F \times d \tag{2.1}$$

$$\tau = F_W \times BG \sin\theta \tag{2.2}$$

Where τ = Torque,

F_w = Force of weight,

BG = distance between the center of buoyancy and center of gravity,

Θ = pitch or roll angle.

In Figure 2.3, the similar of couple force on ROV is shown. The stability of ROV will increase when larger BG produces a larger torque [9].

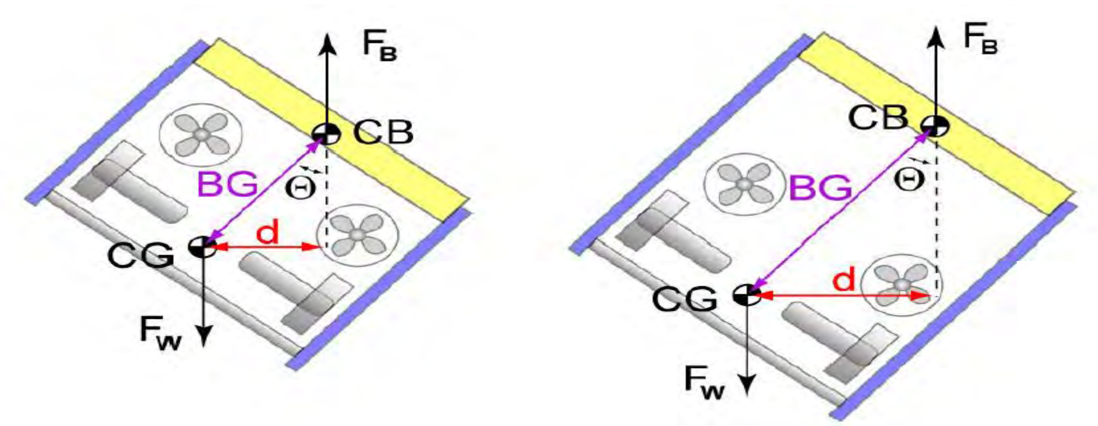


Figure 2.3: Couple force on ROV. [7]

Active Stability

The thruster configuration is also an important factor for having a good stability of ROV. The vectored thruster configuration gives a better performance as mentioned in 2.2.2The equation 2.3 and 2.4, it shows that the thrust force for x and y axis. From these equation, it is known that if any absence of the thruster will make the ROV become unbalance in thrust force. Therefore, it will affect the stability of ROV on motion to oppose the force.

2.1.3 Ballast System

Ballast is the adding or removing of weight on an underwater vehicle to improve its stability [7]. A ROV requires both floatation and weight to carry out the operation for the diving purpose. There are 2 types of ballast systems that can be used by ROV which are the static and dynamic ballast. For static ballast system, the ROV will be pre-set with the desired ballast and remain unchanged throughout the operation. For dynamic ballast system, the ROV will change its ballast during the operation. It allows the ROV to be heavy when diving in the high current situation. However, the disadvantage of this system is the air in the tank changes in volume as the ROV dive to a different depth. Therefore, it is uncommon for most ROV. The most common material used for ballast system is syntactic foam. This is because of its low density and ability to withstand high pressure [9, 10]. In Figure 2.4, it shows the float block and ballast weights added to ROV to increase its stability.

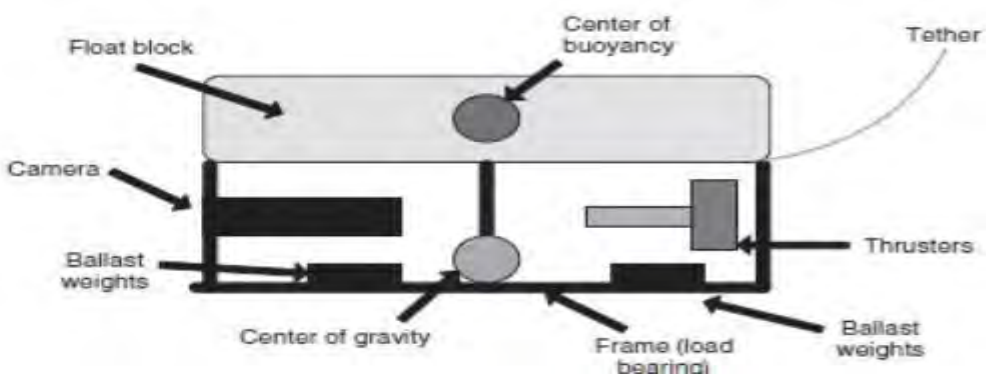


Figure 2.4: Ballast system on ROV. [11]

2.1.4 Buoyancy (Archimedes' Principle)

Buoyancy is a force that exerted by the liquid on an object that is fully or partially submerged on it. Archimedes' principle indicates that the upward buoyant force that exerted on an object is equal to the weight of the fluid displaced by the object [11]. ROV need to neutralize the negative buoyancy effect of heavier than water materials on the submersible for example frame and pressure housing with lighter than water materials. A slightly positive buoyant is the goal of ROV [7]. This is because it allows the ROV to float to the surface when the propeller is damaged. Moreover, it is easier to make any modification when the ROV stay at the surface. Most of the material of frame used by the ROV is polymer type. Metal is not recommended to be used as the material of the frame of ROV due to its increase in weight and degradation after long periods of contact with seawater [3].

2.1.5 Control System

A Proportional-Integral-Derivative controller is a control algorithm that widely used in industrial control system [12]. The most common tuning method of PID used is Ziegler-Nichols rule. The Ziegler-Nichols rule is a method of PID tuning rule that try to produce good values for the three PID gain parameters which are the controller path gain (K_p), controller's integrator time constant (T_i) and controller's derivative time constant (T_d) [13]. The period T_u of the oscillation frequency at the stability limit and the gain margin K_u for loop stability are the two measured feedback loop parameters derived from the measurements [13]. The advantage of this tuning method is that it only need to change the P controller. Moreover, it gives a better illustration on how the system is behaving. By using the values of K_u and T_u, the values of PID gain setting can be determined as shown in Table 2.1 [13].

Table 2.1: Tuning Rule

Rule Name	Tuning Parameters
Classic Ziegler-Nichols	Kp = 0.6 Ku Ti = 0.5 Tu Td = 0.125 Tu
Pessen Integral Rule	Kp = 0.7 Ku Ti = 0.4 Tu Td = 0.15 Tu
Some Overshoot	Kp = 0.33 Ku Ti = 0.5 Tu Td = 0.33 Tu
No Overshoot	Kp = 0.2 Ku Ti = 0.5 Tu Td = 0.33 Tu

2.1.6 Propulsion and Thruster Configuration

ROV propulsion system is divided into 3 types which are electrical, hydraulic, and ducted jet propulsion. The size of the vehicle, type of operation and location of work can affect the type of propulsion system used. The electrical type propulsion system is used when there is required to change energy from electrical to mechanical. For hydraulic type, it is needed when the vehicle requires huge work tooling for intervention. Moreover, the vehicle that operated under the condition that can cause waste to pull into the thrusters needs to use ducted jet propulsion system. The main target of the design of ROV is to obtain a high thrust to physical size and power input ratios [11].

The propeller is divided into 2 main groups which are fixed pitch propeller and controllable pitch propeller. For the fixed pitch propeller, the position of the blades is fixed. For controllable pitch propeller, there is a large hub for a mechanism to control the pitch of the blades when hydraulically activated. Figure 2.5 shows the types of propeller [14]. A number of propeller blades are different for certain operation. It can be made of 2 to 6 blades. However, the lesser the propeller blade, the greater the propulsion efficiency.

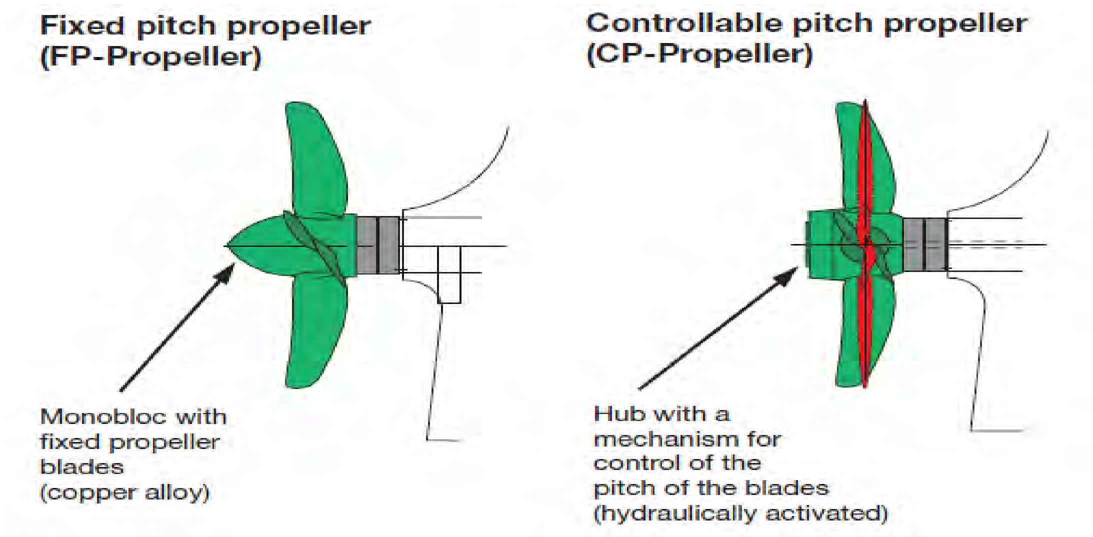


Figure 2.5: Propeller types. [14]

The motion of ROV is dependent on the thruster configuration used within the vehicle. In sequence to get a better maneuverability and controllability of a ROV, the thruster must be well positioned on the vehicle so that the moment arm of their thrust force, relative to the central mass of the vehicle is equal [11]. In order to have 6 DOF for ROV, at least 6 thrusters must be utilized where two vertical and four horizontal thrusters. With this thruster configuration, the ROV can move in heave, surge, sway, heading, pitch and roll. However, this thruster configuration has limitations in the surge and heading control. In order to have a more accurate surge, heading control and sway, 4 thrusters in vectored configuration is needed. It is one of the most popular methods of actuating an ROV due to increases in control in all horizontal direction [11]. If the vectored thrusters are orientated at 45°, it will give greater thrust capabilities and a higher level of fault tolerance. The comparison of vectored and non-vectored thruster configuration is shown in Figure 2.6. The equation of thrust is shown in 2.3 and 2.4 [3].

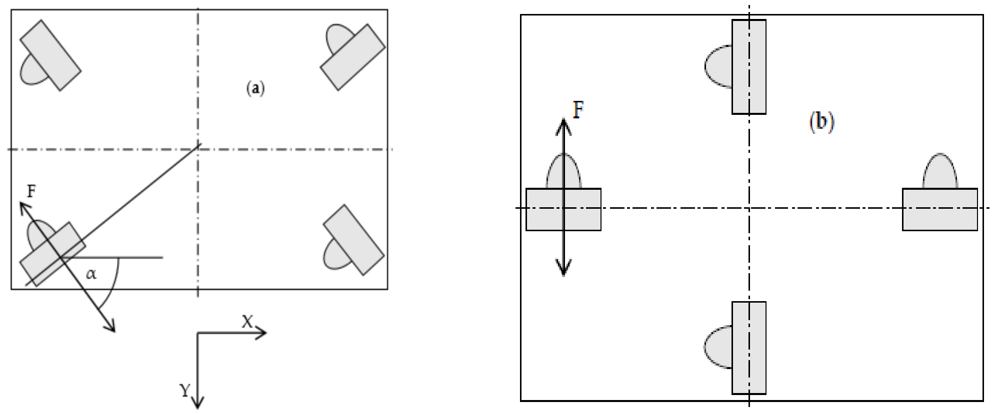


Figure 2.6: Comparison of thruster configuration. (a) Vectored, (b) Non-vectored [3]

Overall thrust,

$$T_X = 4F\cos\alpha \quad (2.3)$$

$$T_Y = 4F\sin\alpha \quad (2.4)$$

If 45°, then $T_X = T_Y = 2\sqrt{2}\,F$

2.1.7 Hydrodynamic

Hydrodynamics is the physics that deals with the dynamic motion of fluid and force exerted on the object that immersed in fluid [15]. Due to the design of ROV is not concerns about the speed, there is little effort put on hydrodynamics. Most of the ROV carry out the heavy work using its high power. However, when the size of ROV is reduced, the hydrodynamic coefficient become significant as the vehicle requires more energy to oppose the forces [3].

There are 2 methods to measure the hydrodynamics coefficient on small ROV which is the experimental method and computational fluid dynamics method. Using the experimental method such as tow tank that equipped a planar motion mechanism (PMM) is very expensive [16]. However, for CFD method, it is inexpensive as it used simulation to get the engineering data and the data can be obtained in a short period of time [17].

2.1.8 Joystick

The joystick is an input device that helps in control the direction and angle. There are few types of the joystick which are a digital joystick, paddle joystick, analog joystick, accelerometer type joystick and potentiometer joystick [18]. Potentiometer based joystick obey the Ohm's Law which it uses resistance as working principle. It has physical components that moved allow the increase or decrease of voltage or current through the change in resistor value. This can be done by applying with the very small amount of force. However, it has limitations in terms of durability and reliability because of its wearing of moving parts [19]. For joystick accelerometer, the accelerometer sensor is put inside the joystick and control by sensing the angle of rotation.

2.2 Review of Previous Related Work

2.2.1 Type of Controller

There are many types of the controller can be used in ROV for feedback purpose. Based on the literature findings, a controller such as Proportional Integral Derivative (PID) controller, sliding mode controller, Fuzzy Logic Controller (FLC) and the combination of PID with adaptive control. Different type of controller will give the different efficiency of work.

From the journals, most of them used Proportional Integral Derivative controller (PID) to control the movement of ROV. The method used is such as trial and error [21], Ziegler Nichols [24] and placing poles in the left half plane of complex plane to achieve stability. There is one paper combined the PID with an adaptive control which is neuro-network [23]. Besides that, there are few papers used fuzzy logic as their controller for research. Usually, the fuzzy logic controller uses MAMDANI fuzzy inference because there is the library at MATLAB that ease the work [22]. MATLAB is the most common simulation software used to check the controller and vary the gain using the SIMULINK. Only one article does research on higher order sliding mode controller. The reason the researcher chose higher order other than standard sliding mode controller is to prevent the high-frequency signal sent into the system which called as chattering effect [20]. The type of controller used by the paper summarises on Table 2.2.

Therefore, there are many controllers that can be used on ROV. PID controller is the most common type of controller used on ROV to eliminate the error.

Table 2.2: Comparison of the type of controller used for ROV.

Type of Controller Used.	Error detection
Second Order Sliding Mode (SMC) [20].	Position tracking error.
PID. It uses the trial and error method to get the value of gain [21].	Depth control and heading control.

Fuzzy Logic using rule matrix of the fuzzy controller [22].	Precision tracking of ROV.
PID with neural network by tuning the PID gain [23].	Learning rate.
PID using Ziegler Nichols tuning method [24].	Overshoot and steady state error.
PD controller for station keeping is developed by constructed using MATLAB Simulink [26].	Damping value.
Proportional control by placing all poles in the left half of the complex plane [27].	Velocity control.
Fuzzy Logic using Mamdani fuzzy inference method [28].	Pitch, roll and depth correction

2.2.2 Thruster Configuration and Number of Thrusters Used

There are different kinds of configuration for thrusters which are x-shaped [20], cross-shaped [20] and vectored [25] which are shown in Figure 2.6 to Figure 2.8. Based on the vectors of forces and moments calculated by Anh Duy Nguyen [22], the ROV can move smoothly in various direction when using the x-shaped configuration with 45° to each other compared to the cross-shaped configuration. Due to the difference in configuration, the number of thrusters used also not the same. The most common number of thrusters used are six thrusters. The journals were suggested to use six thrusters because it can control the orientation of ROV in six degrees of freedom (DOF) easily [3]. The orientation of ROV is a surge, heave, sway, roll, pitch, and yaw. For a system that used lesser thruster to control more DOF is called as an underactuated system. Different orientation will need a different type of configuration in order to achieve stability. For three thruster used with 1 on a vertical plane and 2 on a horizontal plane, the stability of roll and pitch is low when there is significant disturbance [25].

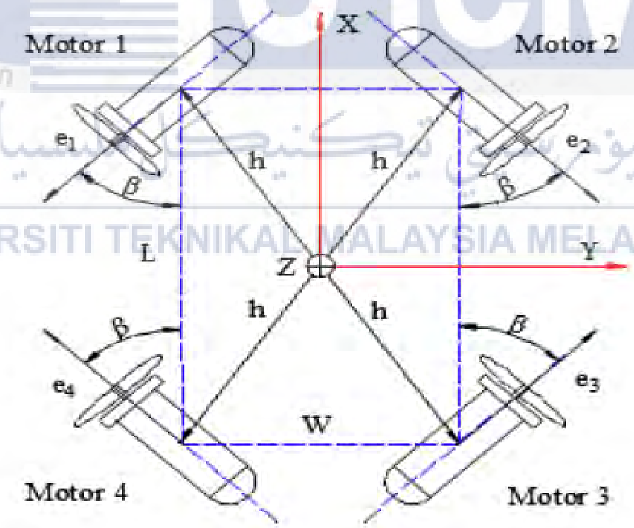


Figure 2.7: Thruster configuration in x-shape.

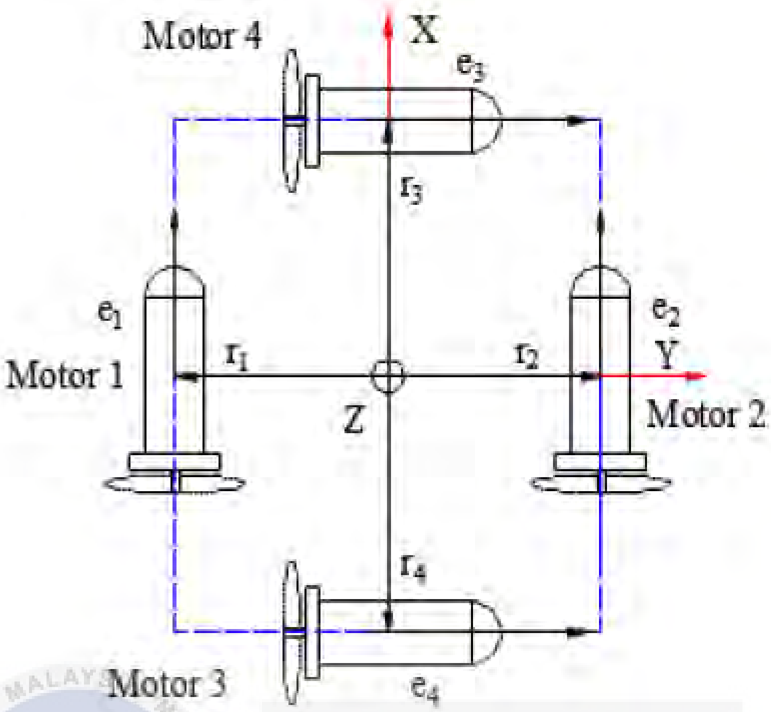


Figure 2.8: Thruster configuration in cross-shape.

From the papers, a different type of thruster configuration will have a different orientation. The most stable orientation is using x-shaped thruster configuration with 45° and six thrusters are used.

Table 2.3: Comparison of thruster configuration

Thruster Configuration And Number Of Thruster Used	Orientation/ Degree of freedom
4 thrusters are used where 1 is located vertical, 1 lateral and 2 rear [20].	Surge, sway, heave, pitch and yaw.
Comparison between X-shaped and cross-shaped thruster configuration. Total of 6 thrusters are used where 2 are located vertical and 4 horizontal [21].	Surge, sway, heave, pitch, yaw and heading.
Total of 3 thrusters are used where 1 located vertical and 2 horizontal [23].	Surge, yaw and heave.

Total of 4 thrusters are used where 2 are located vertical and 2 horizontal [24]	Heave.
Thrusters are in vectored. Total of 6 thrusters are located 2 at x-axis and 2 at y-axis and 2 at z-axis [25].	Heave, surge, sway, heading, pitch and roll.
Total of 4 thrusters are used [26].	Surge, sway, heave and yaw.
Total of 2 thrusters are used [27].	Yaw.
Total of 5 are used where 2 at horizontal and 3 at vertical [28].	Surge, heave, sway, roll and pitch.
One–norm algorithm and modified singular values. Total of 4 are used [29].	Surge, sway, heave and yaw.

2.2.3 SENSOR

There are many sensors can be used to determine the location and orientation of an object. The most common sensors used are gyroscope and accelerometer. A gyroscope sensor can be used determine the orientation of ROV. It is a device that uses the principle of Earth's gravity to help determine the orientation [31]. An accelerometer is a compact device designed to measure non-gravitational acceleration [31]. For this project, a gyroscope is used as it can measure the angle of rotation around a particular axis. From the sensor, we are able to know that how much the ROV has rotated from its original position. It is needed to control the ROV directly without the need to control the thruster directly. Observation can be made using the software that obtains value from the sensor.

Based on the findings in the literature review, MPU 6050 sensor is the most common sensor used for ROV. It consists of 3 axes gyroscope sensor and 3 axes accelerometer. It is needed to sense the angle of rotation of ROV using MPU 6050 to give the feedback to PID controller to check any angle of deviation from the input. The pressure sensor is needed when the ROV has to heave for certain operation.

2.3 SUMMARY

As a conclusion, most of the papers use PID controller in the control system of ROV. For this paper, the control system chosen will be PID as it is simpler. The ROV can steadily move in the surge, sway, heave, pitch, roll and yaw when using the x-shaped thruster configuration with 4 located 45° to each other on horizontal and 2 located on vertical [22]. Therefore, x-shaped thruster configuration was used in this project. Besides that, MPU 6050 sensor is quite popular to use for ROV to detect the angle of rotation of the ROV [22], [24]. From the value obtained by MPU 6050, the PID controller will deal with the error occurred by ROV due to disturbance. When there is heave movement for ROV, there is pressure sensor needed on the ROV body [24], [25], [26]. This is because the ROV can only withstand up to a certain pressure. With the aid of pressure sensor, the ROV will not go beyond its limit. However, this project was only involve the orientation in yaw, pitch and roll, therefore there is no need to have pressure sensor. Moreover, the softwares that often used for simulation in the journals are MATLAB and SolidWorks. MATLAB is used for the control system purpose whereas the SolidWorks is used for drawing and analysis of the drawing. Summary of all articles is made in the Table 2.4.

Table 2.4: Summary of literature review.

REFERENCE	TYPE OF CONTROLLER	THRUSTER CONFIGURATION AND NUMBER OF Thrusters USED	ORIENTATION	SIMULATION SOFTWARE	SENSOR USED	CONCLUSION	RECOMMENDATION
Research, Design and Control of a Remotely Operated Underwater Vehicle [22]	PID using trial and error method to get the gain.	X-shaped and cross-shaped. Total of 6. 2 vertical and 4 horizontal	Surge, sway, heave, pitch and yaw	Visual Studio C#2013, Computational Fluid Dynamics (CFD).	MPU6050 consists of 3 axis gyroscope and 3 axes accelerometer.	- move to desire depth and response of the heading control within ±2°	- Add camera to observe - Reducing ROV weight - Operating depth
Modelling, Design and Robust Control of a Remotely Operated Underwater Vehicle [20]	Second order sliding mode (SMC)	Total of 4. 1 vertical, 1 lateral and 2 rear.	Heave, surge, sway, heading, pitch and roll	MATLAB-SIMULINK, SOLIDWORKS, CAD.	Depth sensor, gyros, inclinometer and compass	- Position tracking with little error and without overshoot.	- Model-Free High Order Sliding Mode Control (MF-HOSMC) - To prevent high-frequency signals into the system.
Fuzzy Controller for Underwater Remotely Operated Vehicle Which is Moving In Conditions of Environment Disturbance Occurrence [22]	Fuzzy Logic using rule matrix of the fuzzy controller.		Heave, surge, sway, heading, pitch and roll	MATLAB-SIMULINK		- Easy project regulators for the nonlinear object.	- Neuro-controller.

A Self-tuning Nonlinear PID Controller for a Three-Thruster Remotely Operated Underwater Vehicle [23]	PID with neuro network by tuning the PID gain	Total of 3. 1 vertical and 2 horizontal	Surge, yaw, and heave.	MATLAB-SIMULINK, CFD	Gyroscope.	- Improve the performance of a highly nonlinear ROV. - Carefully in select the learning rate value.	- Identify suitable adaptive control rates for ROV controller.
Design of Thruster Configuration and Thrust Allocation Control for a Remotely Operated Vehicle [29]		One–norm algorithm and modified singular values. Total of 4.	Surge, sway, heave and yaw.	MATLAB-SIMULINK		- Each thruster achieve a lower voltage input and lower thruster utilization.	- Roll and pitch can self-stabilizable.
Implementation of PID Controller for Hold Altitude Control in Underwater Remotely Operated Vehicle [24]	PID using Ziegler Nichols tuning method.	Total of 4. 2 vertical and 2 horizontal	Heave	Visual Basic, MATLAB and Graphical User Interface (GUI)	A pressure sensor, MPU6050.	- Ziegler Nichols tuned PID controller is efficiently better than classical PID controller	- Best result when Kp=23.4, Ki=15 and Kd= 3.75 for depth control.
Design, Construction and Control of a Remotely Operated Vehicle (ROV) [25]	PID by modified the gain.	Vectored. Total of 6. 2 at x-axis and 2 at y-axis and 2 at z-axis	Heave, surge, sway, heading, pitch and roll		Pressure sensor, 2 axis inclinometer, and 3 axes accelerometer	- Successful in yaw angle regulation	- Sliding mode controller to track different paths is considered. - Tracking roll and pitch angles are also taken into account.

Preliminary Studies of the Modelling and Control of Twin-Barrel Under actuated Underwater Robotic Vehicle [26]	PD controller for station keeping is developed by constructed using MATLAB Simulink	Total of 4.	Surge, sway, heave and yaw.	SIMULINK		- Able to move the vehicles to the desired coordinated as instructed.	- Sensor and advanced thrusters. - Nonlinear controller using Lyapunov method.
The Eyeball ROV: Design and Control of a Spherical Underwater Vehicle Steered by an Internal Eccentric Mass [27]	Proportional control by placing all poles in the left half of the complex plane.	Total of 2.	Yaw.		Gyro sensor and accelerometer	- Successful in implementing stability augmentation on the system.	- Future prototypes with internal thrusters are predicted to behave similarly to this research. -Wireless communication will remove disturbances from the tether.
Underwater ROV with Fuzzy Logic Motion Control [28]	Fuzzy Logic using Mamdani fuzzy inference method	Total of 5. 2 at horizontal and 3 at vertical.	Surge, heave, sway, roll and pitch		10 DOF IMU (accelerometer, magnetometer, gyroscope and pressure sensor)	- Able to adopt a shallow water environment up to 10m depth.	- Every movement including the self-correction was very smooth.

CHAPTER 3

METHODOLOGY

3.1 INTRODUCTION

This chapter discussed the project methodology that was carried out in this research. Project methodology is a method or technique used to achieve the aim of this research. From an introduction and background of this project until the project design, this chapter will provide a guideline for the reader so that they can understand the project well.

Firstly, a block diagram was included in this chapter as shown in Figure 3.1. This block diagram explained the control system of ROV. Besides that, the methodology of research, a flowchart of ROV operation, a flowchart of building prototype and Gantt chart were shown in Figure 3.2, Figure 3.3, Figure 3.4 and Table 3.3 and 3.4. This flowchart explained the overview of this research.

Moreover, there are some simulations and experiments that had carried out to test the performance of the ROV. The mechanical parts used in this research drawn using SolidWorks. Type of analysis done in SolidWorks software is discussed.

3.2 CONTROL SYSTEM

When the user rotates the joystick to a certain angle, the MPU 6050 sensor on the joystick will sense the angle of rotation made by the user. There is another MPU 6050 sensor located on the body of ROV to sense the angle of rotation of ROV. When there is error or deviation of angle of rotation between the joystick and ROV, the PID controller will correct it and send a signal to the motor driver. Then, the motor driver will control which motor to turn clockwise and anti-clockwise.

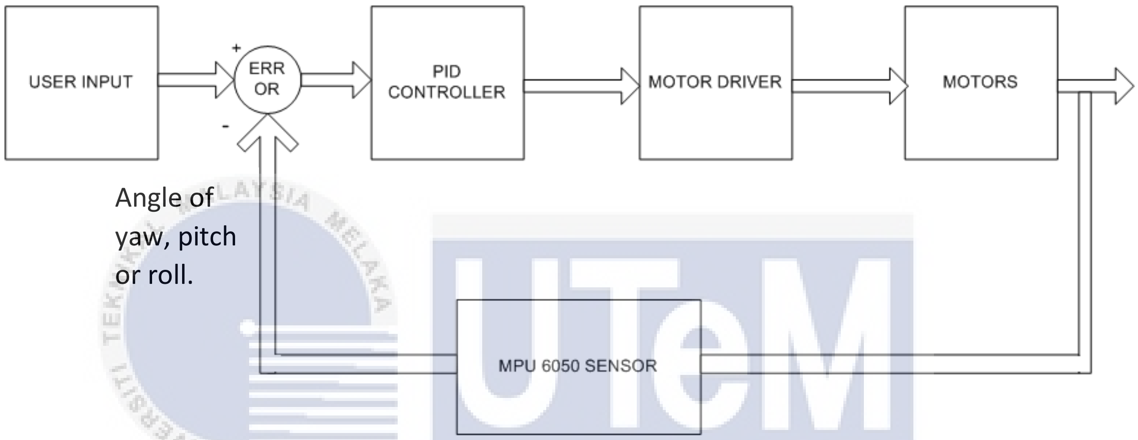


Figure 3.1: Block diagram of a control system for ROV.

3.3 METHODOLOGY OF RESEARCH

In order to start this project, understanding of project can be done by defining the problem faced by the project. After noticed the problem, research on the ROV design and orientation control is needed to solve the problem. There are several designs proposed in the literature review. Based on the proposed design in the literature review, a new design that can meet the objectives of this project was created. Beside the body frame of the ROV, electrical and software design are also required so that the ROV can operation for its function. If there is any error during the simulation, modification will be made to remove the error. When there is no more error, the hardware of ROV will be constructed so that ROV can undergo the real-time full test. The full test includes waterproof test, balancing test and buoyancy test. If there is an error occurred, the modification is needed. After that, performance analysis of ROV will be carried out. Writing is the last step to complete the thesis. In Figure 3.2 shows the flowchart of research.

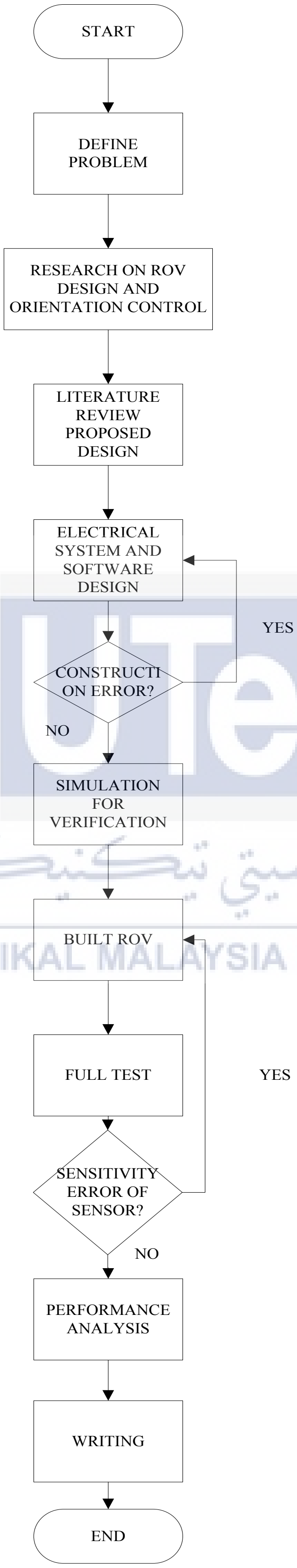


Figure 3.2: Methodology of research.

3.4 PROTOTYPE OPERATION

When the user rotates the remote control to a certain angle, the sensor on the remote will send a signal to PID controller. At the same time, the ROV will rotate according to the desired angle of the user. Sensor value of the angle of rotation in 3 axes on ROV will be compared with the signal received from the sensor on the remote. If there is a deviation in the angle between 2 sensors, the PID controller will make the correction by tuning the gain. Then the PID controller sends a signal to the motor driver for controlling the motor to turn clockwise and anti-clockwise. The flowchart of prototype operation is shown in Figure 3.3

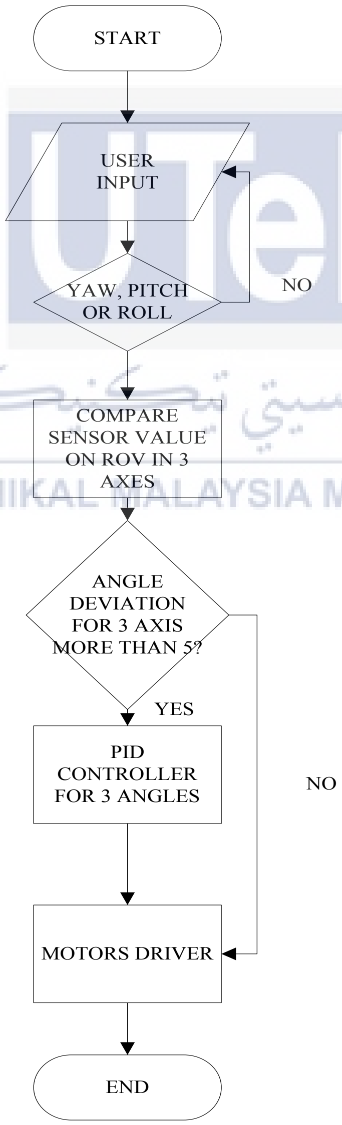


Figure 3.3: Flowchart of prototype operation.

3.5 BUILDING PROTOTYPE

For creating an ROV, research on the ROV design is the initial step. From the design proposed on the literature review, a new design of ROV was created. The design is drawn using SolidWorks software. After done assembly all the parts of ROV, analysis on ROV was done using SolidWorks simulation. Those analysis are the center of gravity, stress and strain test, and sustainability test. After getting all the value from SolidWorks, fabrication of the body frame was carried out. Installation and construction of electrical, mechanical and software system can be done simultaneously. Verification and validation were started after the system was constructed. Testing was carried out to test the performance of ROV. The flowchart of building prototype is shown in Figure 3.4.

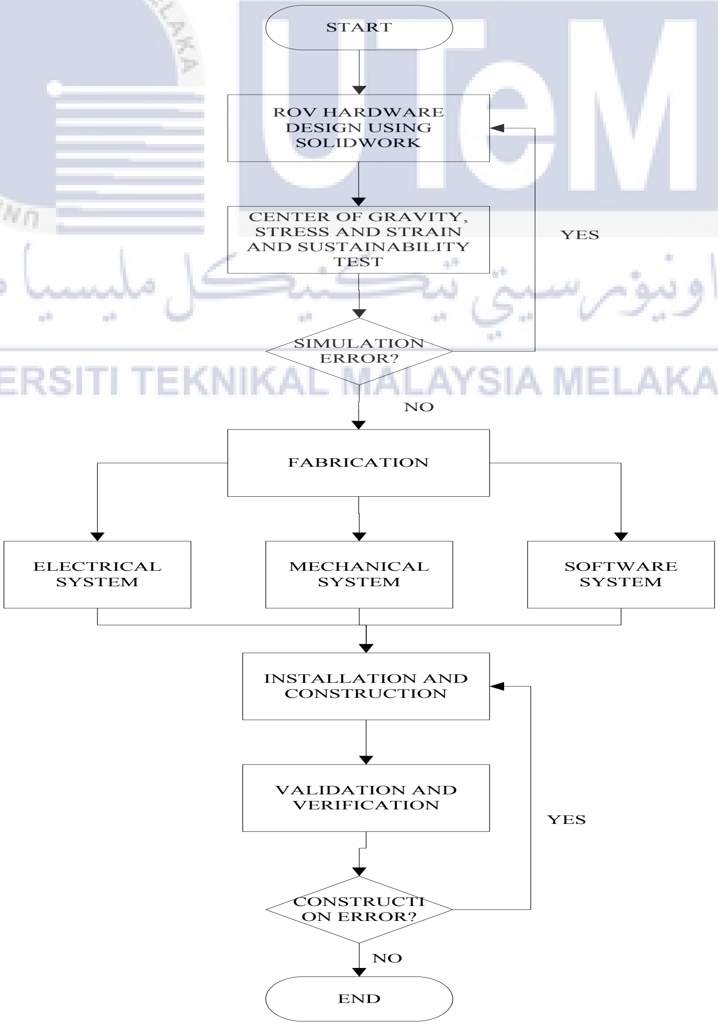


Figure 3.4: Flowchart of building prototype.

3.5.1 ELECTRICAL DESIGN

The electrical part design of a ROV is consists of an Arduino Mega 2560, MPU 6050 sensor, 4 channels relay module, Electronic Speed Controller (ESC) and brushless motor. The connection is shown in Figure 3.5.

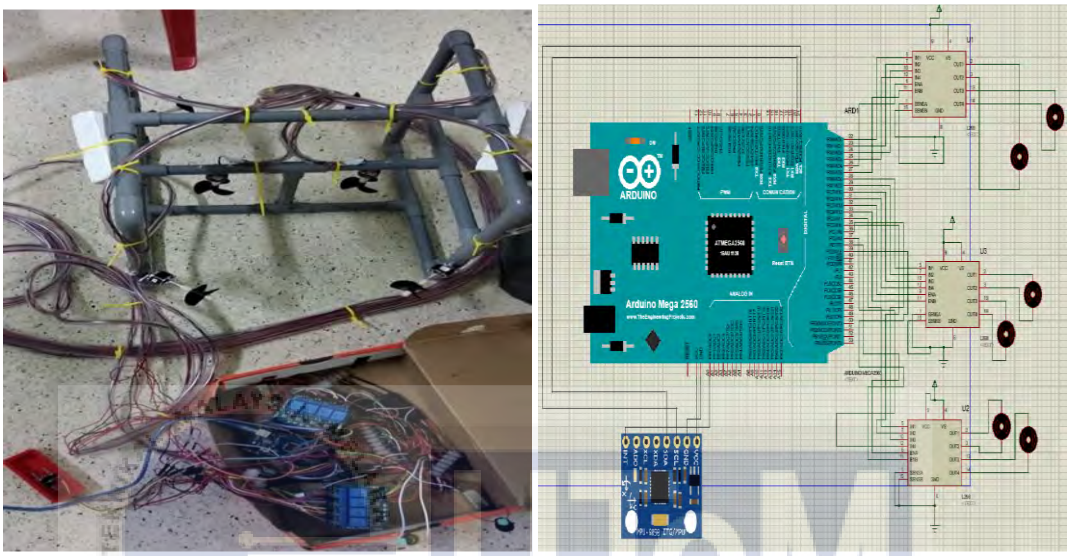


Figure 3.5: Circuit and schematic diagram for a ROV.

3.5.1.1 ARDUINO MEGA 2560

The ROV is equipped with Arduino Mega 2560 as the main board to control the ROV is shown in Figure 3.6. Arduino Mega was chosen because it has many input/output pins which is suitable for this project that required many pins. The microcontroller board of Arduino Mega operates using the ATmega2560. It consists of 54 digital input/output pins which 16 analog inputs,15 can be used as PWM outputs, 4 UARTs (hardware serial ports), a USB connection, a power jack, an ICSP header, a reset button and a 16 MHz crystal oscillator.

Figure 3.6: Arduino Mega 2560.

3.5.1.2 MPU 6050 SENSOR

MPU 6050 sensor is a device that mixed a 3-axis gyroscope and a 3-axis accelerometer which processes complex 6-axis MotionFusion algorithm [32].Figure 3.7 shows the MPU 6050 sensor board and Table 3.3 shows the product details of MPU 6050. This sensor is used because it is simpler to obtain the coding to read the angle.

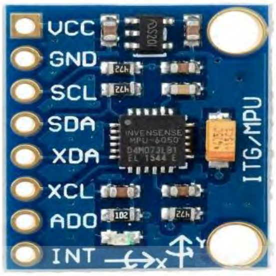


Figure 3.7: MPU 6050.

Table 3.1: Product details of MPU 6050. [32]

Part #	Gyro Full Scale Range	Gyro Sensitivity	Gyro Rate Noise	Accel Full Scale Range	Accel Sensitivity	Digital Output	Logic Supply Voltage	Operating Voltage Supply	Package Size
UNITS:	(°/sec)	(LSB/°/sec)	dps/√Hz	(g)	LSB/g		(V)	(V +/-5%)	(mm)
 MPU-6050	±250 ±500 ±1000 ±2000	131 65.5 32.8 16.4	0.005 0.005 0.005 0.005	±2 ±4 ±8 ±16	16384 8192 4096 2048	I^2C	1.8V±5% or VDD	2.375V–3.46V	4x4x0.9

3.5.1.3 BRUSHLESS MOTOR

Brushless motor is chosen for this project because it has higher efficiency, longer lifespan and more reliable than a brushed motor. This motor is able to operate with less noise and electromagnetic interference because it is fully enclosed for internal parts. The brushless motor used is SunnySky A2212 980KV outrunner as shown in Figure 3.8. It is cheap and waterproof so that it is suitable for this project. Figure 3.9 shows the operation of brushless motor for yaw, pitch and roll.

Table 3.2: Specification of SunnySky A2212 980KV

KV	980
Stator Diameter	22mm
Stator Length	12mm
Shaft Diameter	3mm
Motor Dimension (Diameter * Length)	27 x 30mm
Weight	50g
Maximum Continuous Current	14.5A
Maximum Continuous Power	160W
Maximum Efficiency Current	(4-9A) > 80%

Figure 3.8: SunnySky A2212 980KV [33]

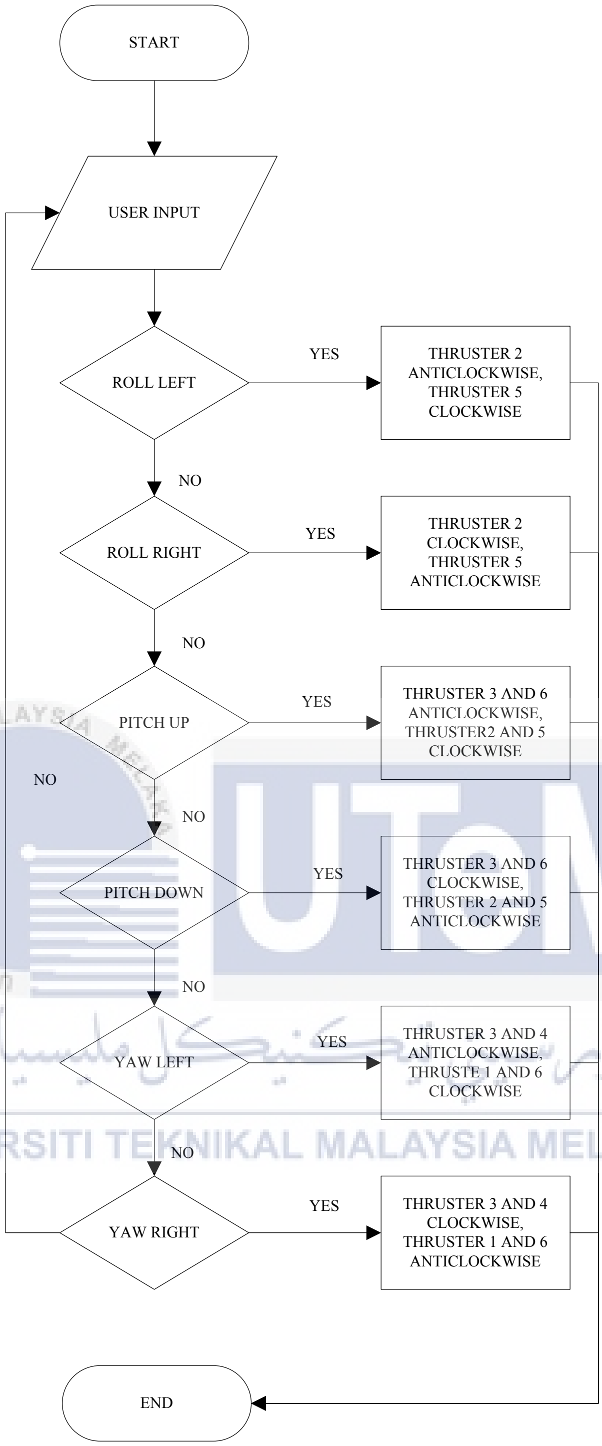


Figure 3.9 Flowchart of motor operation.

3.5.1.4 Electronic Speed Controller (ESC)

ESC is a device used to control the speed of servo motor and direction of motor. It is used to match with the brushless motor as the brushless motor acts like a servo motor. ESC are often used to supply an electronically generated three phase electric power low voltage source of energy to motor. The ESC chosen for this project is Hobbywing Skywalker 30A ESC (Figure 3.9) because it is cheaper compared to other ESC. However, it can only control the motor to run in one direction. Only through manually changing the wire connection between ESC and brushless motor for change of direction of motor turn. In order to solve the problem, a 4 channels relay module was used for switching of wire automatically through coding using Arduino. The ESC is required to get a signal from Arduino for activation of brushless motor. The range of signal speed is from 1000μs to 2000μs. 1000μs is for the motor to stop and 2000μs is 100% motor speed. For safety precaution, only 1400μs is used in the coding which is equals to 40% of motor speed.

Table 3.3: Specification of Hobbywing Skywalker 30A.

Continuous Current	30A
Burst Current (less than equal 10s)	40A
BEC Mode	Linear
BEC Output	5V/2A
Weight	37g

Figure 3.10: Hobbywing Skywalker 30A ESC.

3.5.2 MECHANICAL DESIGN

3.5.2.1 FRAME

Basically, the frame is divided into 2 types which are open frame and closed frame as shown in Figure 3.11. Most of the ROV manufacturers use open frame as their design. This is because open frame configuration can provide higher stability and easier to add equipment such as a sensor on the body frame. Besides that, the open frame can give the water flow directly to thruster without any obstacle.

Figure 3.11: Open frame ROV (left) and closed frame ROV (right). [34]

The body frame design of ROV is an important part for this research. This is because the frame of an ROV is needed to provide support and protection for ROV during operation. The material selection will also affect the performance of ROV. Material that can withstand the strength and stress exerted by the water pressure must be chosen. For this project, PVC pipe is used for building the body frame because it is cheaper compared to aluminum. The Figure 3.12 shows the body frame design of ROV for this project.

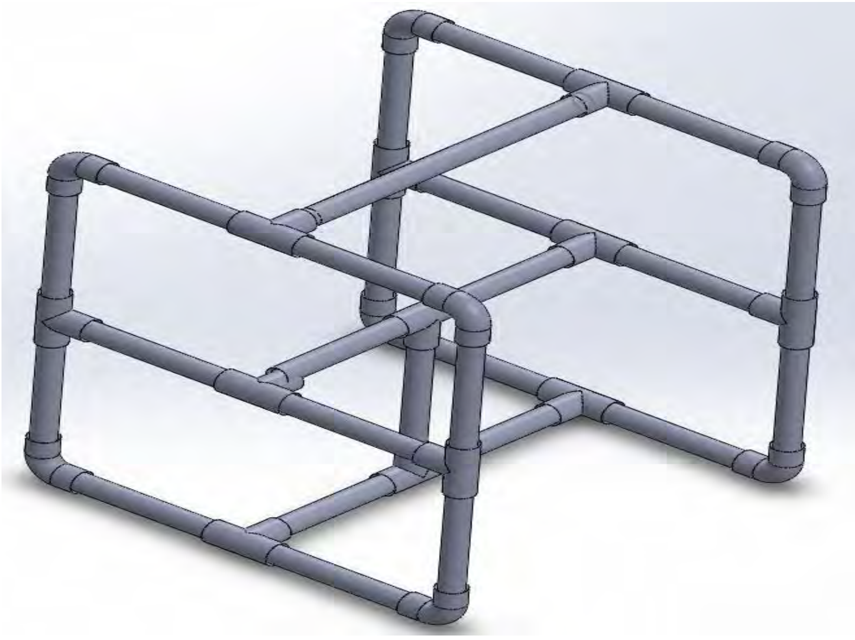


Figure 3.12: ROV frame design.

3.5.2.2 THRUSTER

A thruster is consists of a motor that connects to the propeller. Different thruster configuration will have different control level of motion of ROV. From the journal in chapter 2, 6 thrusters are used for this project to run in 3 degrees of freedom which are a roll, pitch and yaw. There are 4 horizontal thrusters arranged 45° and 2 vertical thruster.

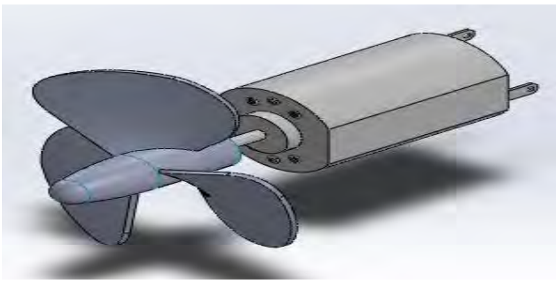


Figure 3.13: Thruster drawn in SolidWorks.

3.5.2.3 ROV ASSEMBLY DRAWING

Finally, a complete ROV design is done by combining the all mechanical parts. A complete ROV design is shown in Figure 3.14 with 30cm length, 30cm width and 30cm height approximate 1kg. The shape is symmetrical. It consists of PVC pipe frame and 6 thrusters arranged in vectored configuration. The horizontal thrusters are 45° to each other.

Figure 3.14: Complete design of ROV.

3.5.2.4 JOYSTICK

The joystick is used to control the movement of ROV by the user. The joystick is designed about the same as the WII controller. There is a sensor inside the joystick to communicate with the control system. The drawing of the joystick is shown in Figure 3.15. The fabricated design is shown in Figure 3.16. This design has 10cm long, 4cm wide, and 2.5cm thick. This dimension was set in order to comfort the user to use it.

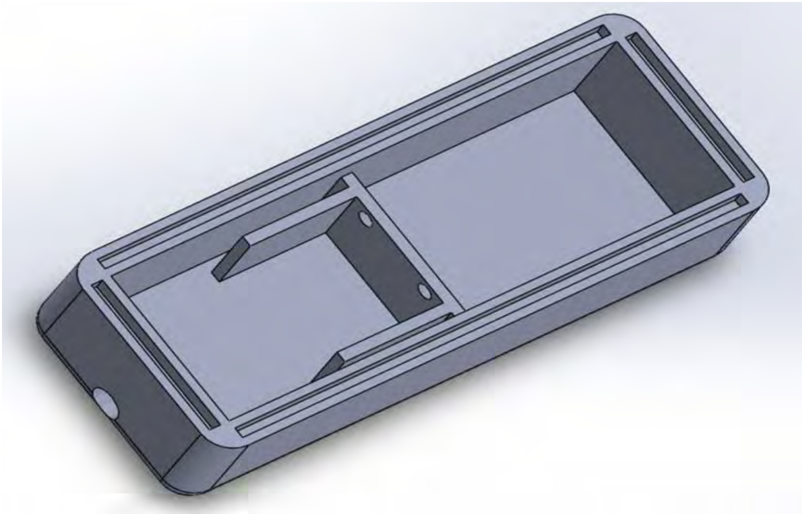


Figure 3.15: Joystick design.

Figure 3.16: Fabricated design.

3.5.3 SIMULATION SOFTWARE

3.5.3.1 SolidWorks, MATLAB

SolidWorks is a computer-aided design (CAD) software that used to draw the engineering design. It is easy to use and lower total cost while giving high efficiency. Before fabrication of ROV, the parts that drawn using SolidWorks can be analyzed using the simulation on SolidWorks software. Simulation such as the center of gravity, stress-strain test, and sustainability test can be done using SolidWorks. The user can choose any type of materials and undergo the analysis.

MATLAB is a software that used to solve engineering problem with high-performance language. It consists of many libraries tools that can be used for many application such as hardware interface with Arduino. The movement control of ROV can be done using the toolbox in the MATLAB such as PID, FLC and Neural Network. For PID, the proportional controller (K_P) is used to reduce overshoot and steady state error while the integral controller (K_i) is used to remove the steady-state error and derivative controller (K_D) is used to reduce the rise time and settling time hence increase the stability of the system. The effect of increasing K_P, K_I, K_D parameters is shown in Table 2.4. FLC is a control algorithm that uses experienced based rules to produce an output. FLC is easier to design by giving certain membership value to function but the tuning process requires trial and error that will take a longer time to obtain the result.

Table 3.4: Effect of increasing KP, KI, KD parameters *[30]*

CL RESPONSE	RISE TIME	OVERSHOOT	SETTLING TIME	S-S ERROR
Kp	Decrease	Increase	Small Change	Decrease
Ki	Decrease	Increase	Increase	Eliminate
Kd	Small Change	Decrease	Decrease	Small Change

3.6 GANTT CHART

Gantt chart is the overview of schedule of this research. It is divided into 2 which are Final Year Project 1 (FYP 1) and Final Year Project 2 (FYP 2). This chart is important because it gives the dateline for each task so that the project will not be a delay. The Gantt chart for FYP 1 and FYP 2 are shown in Table 3.6 and Table 3.7 respectively.

Table 3.5: Gantt chart for FYP 1

ACTIVITIES	W1	W2	W3	W4	W5	W6	W7	W8	W9	W10	W11	W12	W13	W14
Define Problem														
Research on ROV design														
Literature review														
Proposed design														
Perform analysis in SolidWorks														
Design of experiment procedures														
Data collection and analysis														
Presentation														
Report writing														

Table 3.6: Gantt chart for FYP 2.

ACTIVITIES	W1	W2	W3	W4	W5	W6	W7	W8	W9	W10	W11	W12	W13	W14
Fabrication														
Installation and construction of ROV														
Validation														
Performance Analysis														
Presentation														
Report writing														

3.7 SIMULATION AND EXPERIMENT FOR ANALYSIS

In order to achieve three objectives discussed on chapter 1, few experiments were carried out. The simulation tests to fulfil first objective are centre of mass, sustainability test and stress and strain test whereas experiment done was balancing test and buoyancy test. These tests were carried out to make sure that this design is environmental friendly and it is stable when operate underwater. Besides, second objective was achieved by experiment such as MPU 6050 sensor and motor speed test. Lastly, third objective was solved by using controller to control the orientation of ROV and observe the changes.

3.7.1 Simulation 1: Centre of Mass

Objective: To determine the centre of mass and mass properties of the ROV.

Procedure

1) The assemble part is opened in SolidWorks.
2) In the drawing, click 'Insert > Model Items Property Manager'.
3) Under Reference Geometry, click 'Centre of Mass'.
4) The position of the center of mass appears in the drawing.
5) The image of the center of mass is captured.
6) Click the mass properties tab and change the units to any suitable units.
7) The center of mass refers to 3 axes is recorded.

3.7.2 Simulation 2: Sustainability test

Objective: To determine the sustainability of the ROV.

Procedure:

1) The assemble part is opened in SolidWorks.
2) Click 'Sustainability icon'.
3) Click 'continue', fill in the required parameters.
4) Run the simulation.
5) Click 'Save as'. Change the 'file type, file name and location to save' and proceed.
6) Result obtained is saved and recorded.

3.7.3 Simulation 3: Stress and Strain test

Objective: To determine the deformation and stress of the ROV.

Procedure:

1) The assemble part is opened in SolidWorks.
2) Click 'Simulation Xpress" and add a fixture by select the face model.
3) Add load by applying a force or pressure to the particular area on the model.
4) Calculate the stress and displacement by select a material to the part.
5) The mesh setting can be changed between coarse and fine.
6) Run the simulation.
7) Click 'continue', the 'stress, displacement and factor of safety' will show out.
8) The result is obtained in Microsoft Word.

3.7.4 Experiment 1: Balancing Test

Objective: To study and determine the position of weight to balance ROV.

Parameters: Manipulated Variable: Size of polystyrene foam

Responding Variable: Movement of ROV

Apparatus: ROV and polystyrene foam 2.7×10^{-5} m^3 (3cm wide, 3cm height and 3cm long).

Procedure:

1. The ROV is put into a lab pool without any weight attached and then balance it when floating.
2. A polystyrene foam is put inside both PVC pipes respectively.
3. The ROV is then submerged until it reached stability, neither float nor submerge as shown in Figure 3.17.
4. Step 2 to 3 is repeated with increasing the weight for 5.4×10^{-5} m^3, 8.1×10^{-5} m^3 and 1.08×10^{-4} m^3 long polystyrene.
5. The observation is then recorded and tabulated in table form.

For this experiment, the possible error to be happen is parallax error when observing the result. The eyes should place perpendicular to the top surface of the ROV.

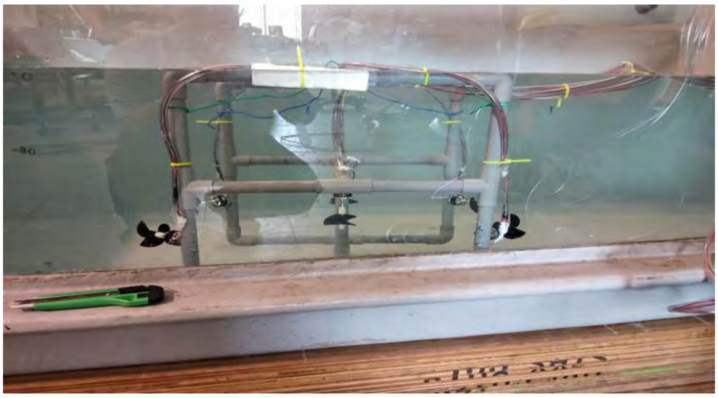


Figure 3.17: ROV is neither float or submerge.

3.7.5 Experiment 2: Buoyancy Test

Objective: To study and measure the buoyancy force acting on the ROV.

Parameters: Manipulated Variable: Weight/payload

Responding Variable: Buoyancy force (upward force)

Apparatus: ROV and weight measuring device.

Procedure:

1. The ROV is measured weight by hanging it from the string of weight measuring device without any weight attached as shown in Figure 3.18.
2. While the ROV is still hanging from the device, submerge it in the underwater pool but it is not touching the sides or bottom of the pool.
3. Step 1 and 2 is repeated by attached the weight to ROV with a different mass.
4. The observation is then recorded and tabulated in table form.
5. Calculate the buoyant force by taking the difference between the mass (or weight) in the air and the mass (or weight) in water.

The possible errors for this experiment are random error and systematic error. The sensitivity of the electronic scale is need to be consider for measuring the weight of ROV. The experiment should be repeated several times to get the average value.

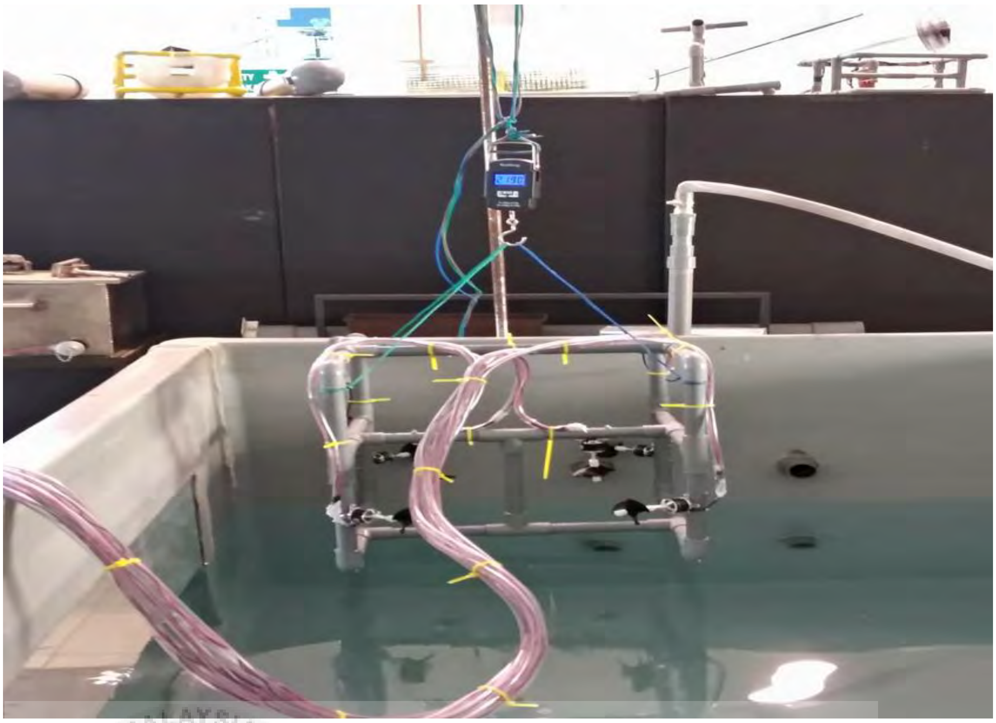


Figure 3.18: ROV is measured in air using electronic scale.

3.7.6 Experiment 3: MPU 6050 Sensor.

Objective: To calibrate and measure the angle of rotation of ROV in 3 axes.

Procedure:

1) The calibration coding is opened using Arduino software.
2) The calibration coding is uploaded to Arduino Mega and sensor.
3) The offset value is calculated by the sensor.
4) The offset value is edited in the original coding.
5) The coding is uploaded to Arduino Mega and sensor.
6) MATLAB software is opened with the coding.
7) The sensor is rotated and the values changes is observed.
8) The result obtained is captured and recorded.

The default sensitivity of the MPU 6050 is 131°/s. The sensitivity can be change inside the coding for MPU 6050.

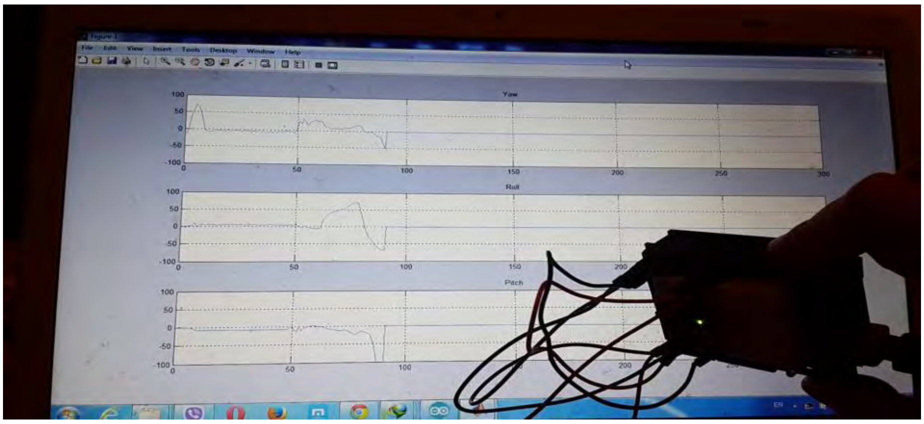


Figure 3.19: Measure angle using MPU 6050.

3.7.7 Experiment 4: Comparison test on MPU 6050 of joystick and ROV

Objective: To determine the changes in the angle of orientation on ROV when user giving input.

Parameters: Manipulated Variable: Angle of orientation on joystick

Responding Variable: Angle of rotation on ROV

Apparatus: ROV and joystick.

Procedure:

1) The angle of rotation of joystick is set by the user.
2) The ROV will move according to the user input.
3) The angle of rotation of joystick and ROV in degrees are recorded and tabulated.

The possible error for this experiment is the sensitivity of MPU 6050. The sensor is very sensitive during measuring the angle. It will cause the control of the orientation of ROV to be have latency.

3.7.8 Experiment 5: Motor Speed test on the motor.

Objective: To determine the speed of rotation of the motor against thrust.

Parameters: Manipulated Variable: Speed of rotation of motor

Responding Variable: Thrust

Apparatus: ROV and electronic scale.

Procedure:

1) The motor is hanged in air using string for measuring the weight using electronic scale.
2) The thrust in air is recorded.
3) The motor is hanged in water using string for measuring the weight using electronic scale.
4) The thrust in water is recorded.
5) The step 3 to 4 is repeated by changing the motor speed from 0% to 100%.
6) The result obtained is recorded and tabulated in table.

Figure 3.20: Measuring motor thrust.

The possible errors for this experiment are random error and systematic error. The sensitivity of the electronic scale used is in 3 decimals for measuring the thrust of ROV. The experiment should be repeated several times to get the average value.

CHAPTER 4

RESULT AND DISCUSSION

4.1 INTRODUCTION

This chapter discusses the simulation results obtained from SolidWorks and the sensor value obtained using Arduino. All the results obtained are tabulated in table or figure to provide a better understanding of the result obtained.

For the first part, the results taken from the analysis using SolidWorks is analyzed and discussed. The analysis included the center for mass, sustainability and stress and strain test.

The experiment on the sensor is also included in this chapter. The graph shown by the sensor is taken as the result from the sensor. The experiments on ROV such as waterproof test, balancing test and buoyancy test were tested in the lab pool. The result obtained was tabulated in table form.

4.2 SolidWorks Analysis

4.2.1 CENTER FOR MASS

The purpose of this analysis is to determine the center of gravity or also known as the center of mass of the ROV. The center of mass will appear in SolidWorks by followed the procedures stated in the previous chapter. This function not only can obtain the center of mass but also properties such as mass, density and volume. By knowing the center of gravity, the stability of ROV during orientation will be increased by placing ballast weight to the bottom of ROV or float material to the top of ROV. The overview of centre of mass of ROV is shown in Figure 4.1.

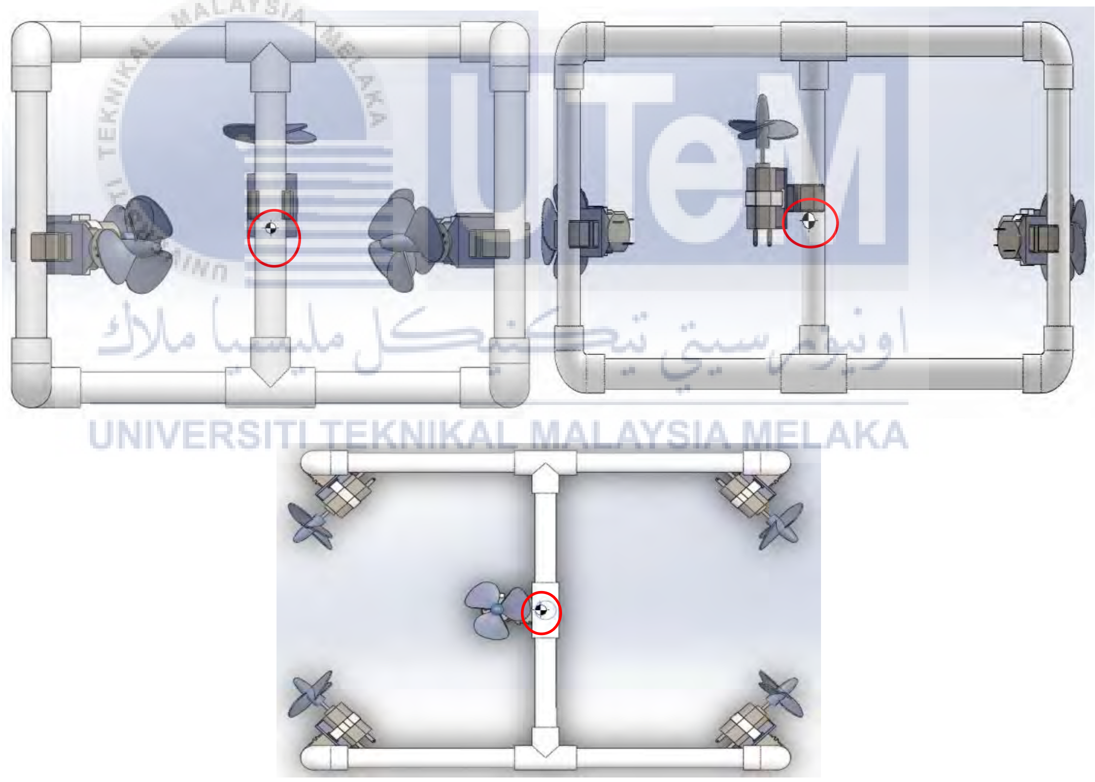


Figure 4.1: Different view of ROV of the center of mass.

Table 4.1: Centre of mass in SolidWorks Mass Properties

Axis	Coordinates (cm)
X	14.42
Y	14.87
Z	16.49

From the mass properties shown in Figure 4.2, it shows that the center of mass of ROV is placed at 14.42cm on X-axis, 14.87cm on Y-axis, and 16.49cm on Z-axis. From the result obtained through simulation, the center of mass is located at the center of the body of ROV. It means that it is symmetrical shape and the ROV is more stable when the center of mass is located at center of body as stated in chapter 2.1.1 on center of gravity. However, it has some error on drawing the body as the result obtained was not exactly located at the center. This maybe due to the diamater of the pipe, tee and elbow are different.

```
Mass properties of ROV
    Configuration: Default
    Coordinate system: -- default --

Mass = 1020.22 grams

Volume = 373100.97 cubic millimeters

Surface area = 349402.01 square millimeters

Center of mass: (millimeters)
    X = 144.29
    Y = 148.74
    Z = 164.91

Principal axes of inertia and principal moments of inertia: (grams * square millimeters)
Taken at the center of mass.
     Ix = (1.00, -0.01, 0.00)          Px = 12359754.23
     Iy = (0.00, 0.01, -1.00)          Py = 19719976.98
     Iz = (0.01, 1.00, 0.01)           Pz = 27335973.41

Moments of inertia: (grams * square millimeters)
Taken at the center of mass and aligned with the output coordinate system.
    Lxx = 12361564.20   Lxy = -164593.18    Lxz = 1741.47
    Lyx = -164593.18    Lyy = 27333718.74   Lyz = -58268.21
    Lzx = 1741.47       Lzy = -58268.21     Lzz = 19720421.68

Moments of inertia: (grams * square millimeters)
Taken at the output coordinate system.
    Ixx = 17406538.37   Ixy = -1799765.98   Ixz = -2140219.98
    Iyx = -1799765.98   Iyy = 31960544.38   Iyz = 2375003.62
    Izx = -2140219.98   Izy = 2375003.62    Izz = 23017391.27
```

Figure 4.2: Mass properties.

4.2.2 SUSTAINABILITY TEST

The purpose of this analysis is to evaluate the environmental impacts of ROV design. The results include the end of life, total energy consumed, carbon footprint, water eutrophication and air acidification.

Table 4.2: End of life of ROV

Weight	1083.48g
Duration of use	1 year
Built to last	1 year
Recycled	15%
Incinerated	2%
Landfill	83%

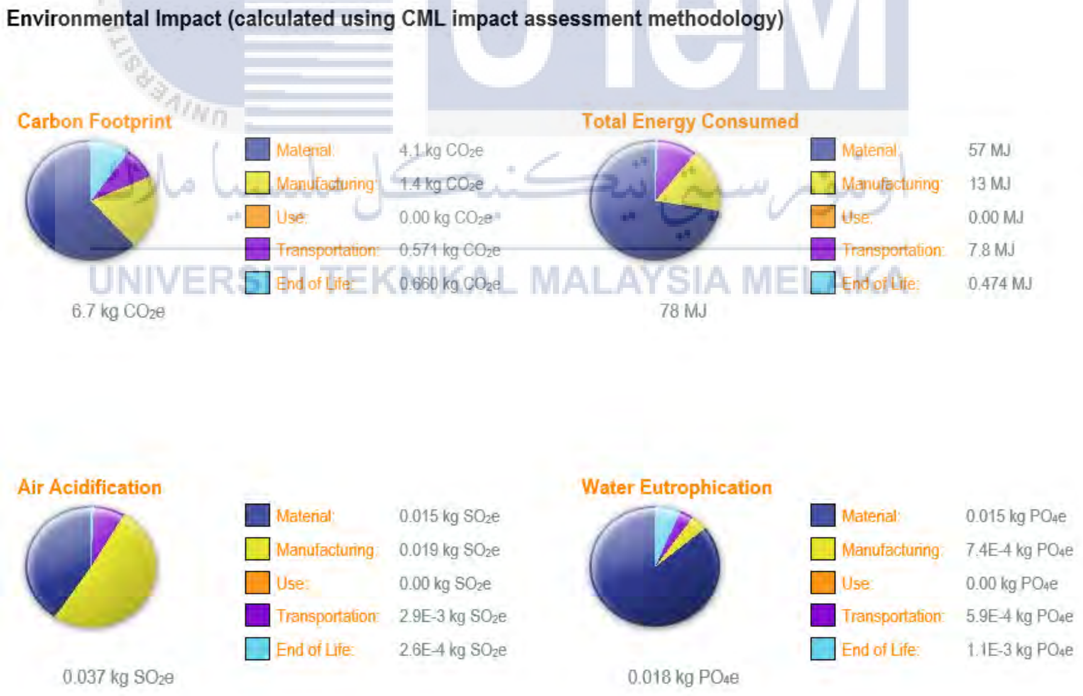


Figure 4.3: Environmental Impact of ROV.

Table 4.3: Component environmental impact of ROV.

Component	Carbon	Water	Air	Energy
Motor	0.606	2×10^{-3}	3.1×10^{-3}	6.5
Pipe Clamp	0.243	8.0×10^{-4}	1.3×10^{-3}	2.3
Wire Tie Holder	0.096	3.2×10^{-4}	5.0×10^{-4}	1.0
30cm pipe	0.061	3.4×10^{-5}	5.0×10^{-4}	1.0
10cm pipe	0.041	2.6×10^{-5}	3.3×10^{-4}	0.688
Tee	0.052	2.9×10^{-5}	4.3×10^{-4}	0.890
Elbow	0.040	2.2×10^{-5}	3.2×10^{-4}	0.670

Based on the simulation result in table 4.2, ROV will go to a landfill after the end of the life. Besides that, on Figure 4.3, there are 6.7kg of carbon dioxide produced and the highest percentage is on material which consists of 61%, followed by manufacturing (21%) and the least percent of producing carbon dioxide is during transportation with 8.5%. For total energy consumed, there are 78MJ consumed by the ROV. Out of 78MJ, material took 57MJ, manufacturing consumed 13MJ and end of life use up 0.474MJ of energy. Next, air acidification, manufacturing parameter emitted 0.019kg out of 0.037kg, 0.015kg for material and end of life released least sulphur dioxide which is 0.000026kg. Lastly, water eutrophication, the highest value is of material which is 0.015kg from a total of 0.018kg and transportation release the least phosphate which is 0.000059kg.

From table 4.3, there are four areas of environmental impact which are carbon, water, air, and energy. Based on the top ten components contributing the most to the four areas of environmental impact, the motor is on the top of the list as it produced most carbon (0.606) to the surrounding and the energy consumed is also very high which is 6.5. the component that on the bottom of the list is wire tie. It has all the lowest value for carbon, water, air, and energy.

From the result obtained through this simulation, the design of this body frame of ROV is environmental friendly because it has lesser impact to environment based on the data in Figure 4.3.

4.2.3 STRESS AND STRAIN TEST

The purpose of this analysis is to determine the deformation of the frame of ROV when the pressure exerted on it. The analysis is carried out using the Finite Element Analysis (FEA) simulation Xpress in the SolidWorks. The main material used for this project is PVC. The value of pressure acting on the submerged part is depends on the depth. Different depth will have different pressure and causes the material to have different maximum stress. The body of the ROV is tested with different values of pressure. The results obtained are shown in Figure 4.4 and 4.5.

Tensile strength is the maximum pressure a material can withstand before breaking. For PVC Rigid, it has a density of 1300kg/m^3 and tensile strength of 40700kPa. To calculate the PVC start to break at which depth,

0 meter depth = 1 atmospheric pressure = 101.325kPa

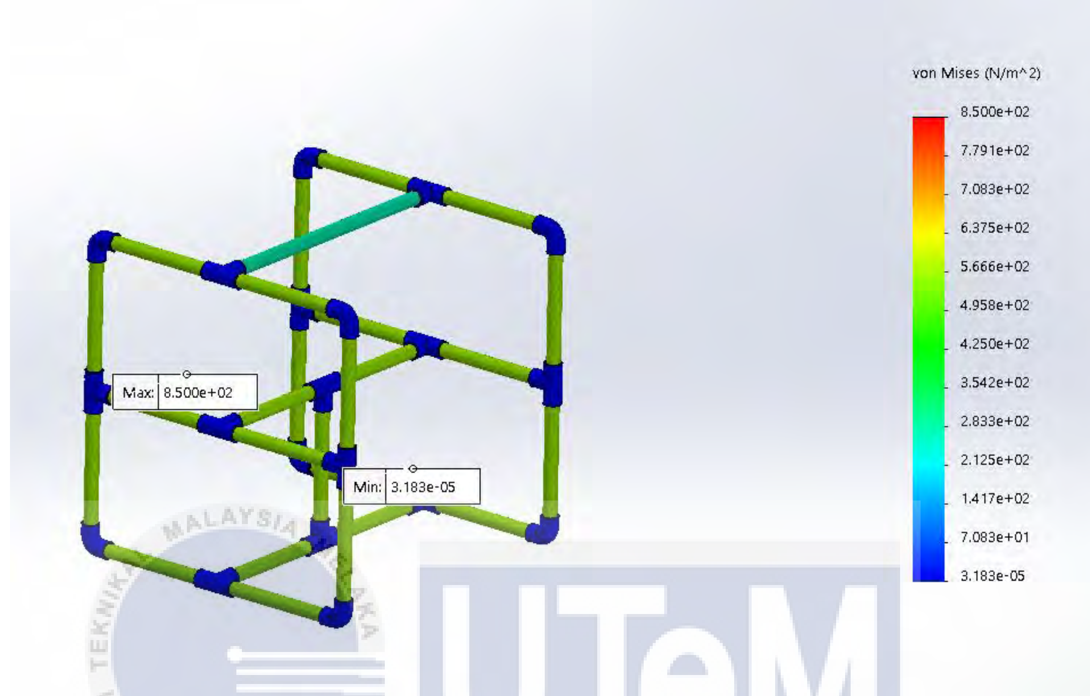
10 meter depth = 2 atmospheric pressure = 202.650kPa.

Given tensile strength = 40700kPa,

$$\frac{x}{10} = \frac{40700}{202.650}$$

X= 2008.39 meters from the surface of the ocean.

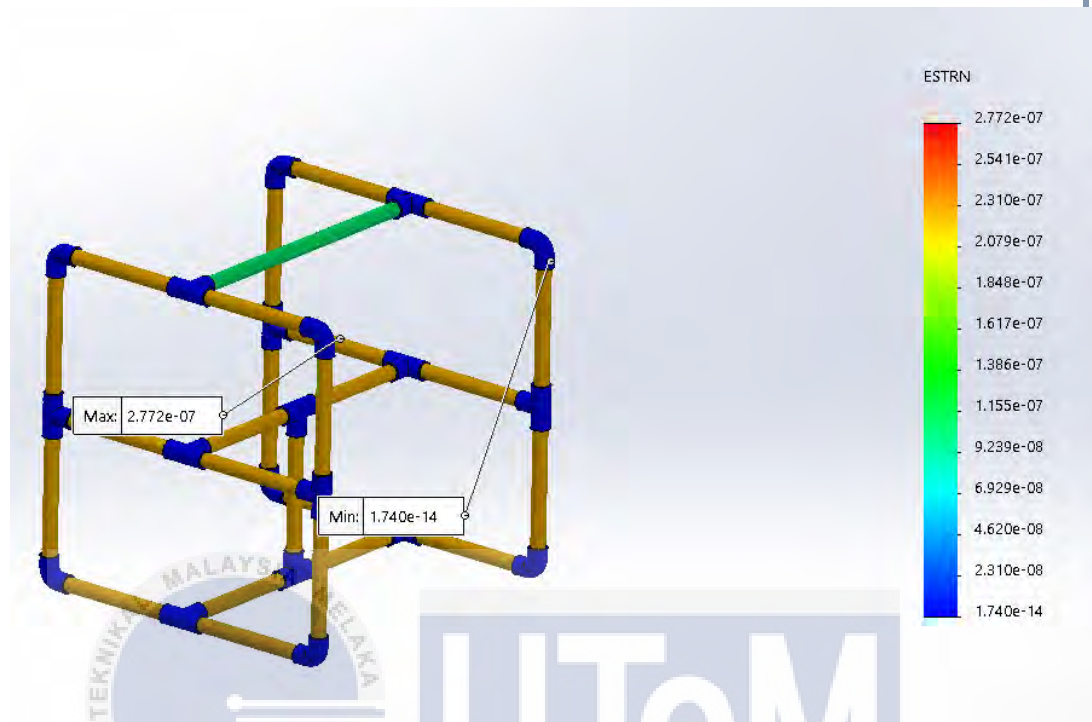
Name	Type	Min	Max
Stress	VON: von Mises Stress	3.183e-05 N/m^2 Node: 6599	8.500e+02 N/m^2 Node: 1810



1p25Str-SimulationXpress Study-Stress-Stress

Figure 4.4: Result of stress for the ROV.

Name	Type	Min	Max
Strain	Equivalent	1.740e-14 Node: 6599	2.772e-07 Node: 1810



1p25Str-SimulationXpress Study-Strain-Strain

Figure 4.5: Result of strain for pipe.

Table 4.4: Relationship between depth and water pressure.

Depth (m)	Pressure (kPa)	Maximum Stress (kPa)
0	101.3250	637.9
1	111.4575	701.7
2	121.5900	765.5
3	131.7225	829.3
4	141.8550	893.1
5	151.9875	956.9
6	162.1200	1021.0
7	172.2525	1084.0
8	182.3850	1148.0
9	192.5175	1212.0
10	202.6500	1276.0
2009 (Start to break)	40700	break

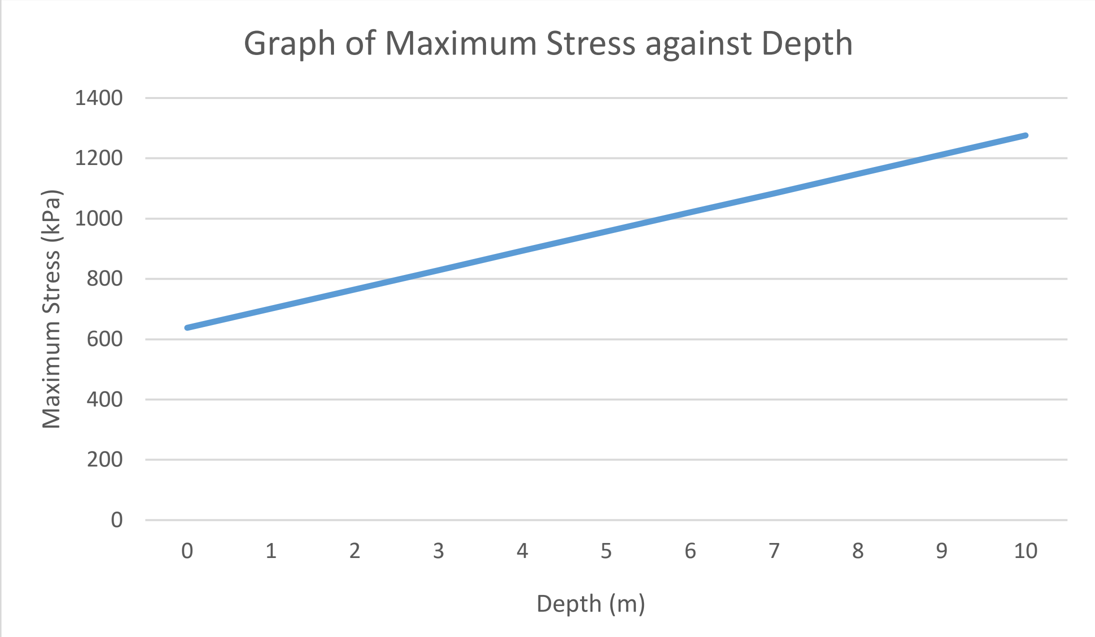


Figure 4.6: Graph of maximum stress against depth.

From Table 4.4 and Figure 4.6, the maximum stress is directly proportional to depth. The material will break at depth of 2009 meters. However, the body of ROV is designed to enter at maximum depth of 2 meters only as stated in scope of project. Therefore, the body frame of ROV is able to withstand the pressure exerted to it.

4.3 EXPERIMENT ON MPU6050

Firstly, the sensor is calibrated using Arduino software. The new offset values for accelerometer X,Y,Z and gyroscope XYZ are 550, -602, 2291, 190, 62, 9. After that, the MATLAB code is used to display the graph and 3D visualization. Comparison between the measured angle using protractor and MATLAB simulation are shown in Figure 4.7, 4.8, 4.9 and 4.10. The values obtained are about the same. The tracking the angle of rotation using MPU 6050 is very accurate and fast as it can read the angle at maximum of 2000 degree per seconds. The sensitivity of MPU 6050 can set using the coding to choose the range of scale it needs to read.

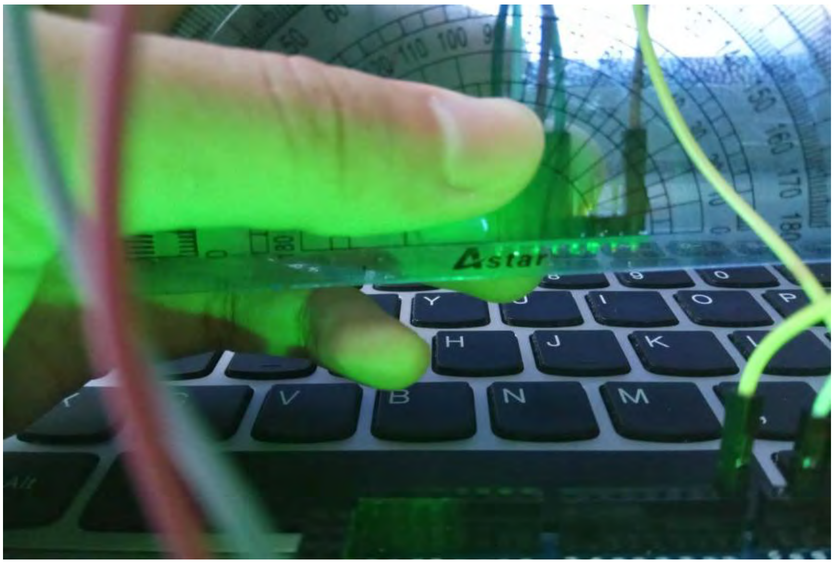


Figure 4.7: Angle of 0.

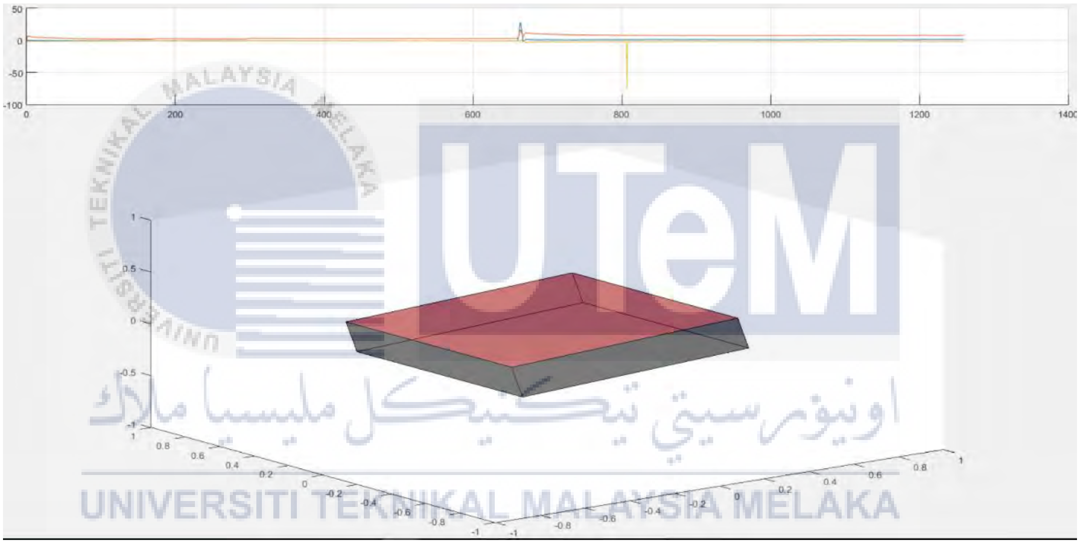


Figure 4.8: Result obtained from MATLAB.

Figure 4.9: Angle of 40.

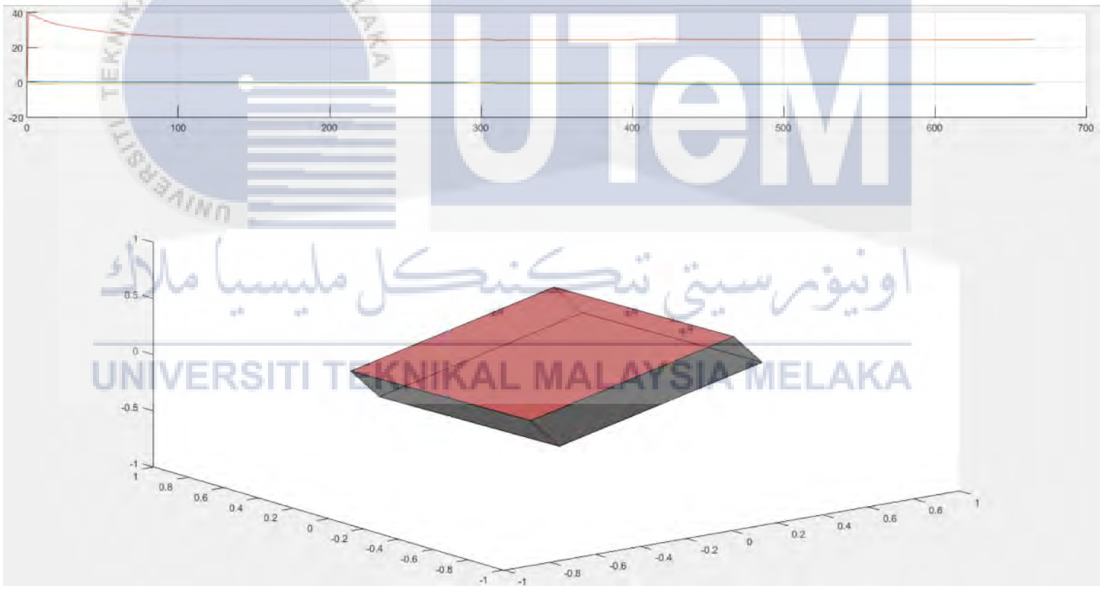


Figure 4.10: Result obtained from MATLAB.

However, gyro values for x axis (roll), y axis (pitch), and z axis (yaw) are drifting over a long period of time. The value tends to vary a lot and no longer accurate. Graph of drifting angles against time for x axis (red colour) and y axis (black colour) are shown in Figure 4.11 and 4.12. In order to solve the drift problem occurred on MPU 6050 sensor, Kalman filter and Complementary filter are used to get a better and accurate value of gyro x and y axis. These filter uses the concept of combining both accelerometer and gyroscope value into a single quaternion value. Accelerometer itself is not stable in a short period of time whereas gyroscope will tends to drift during a long period of time. By combining both devices, the values obtained will become more accurate. These filter are done by minus the angle drift if the angle is exceed 180°.

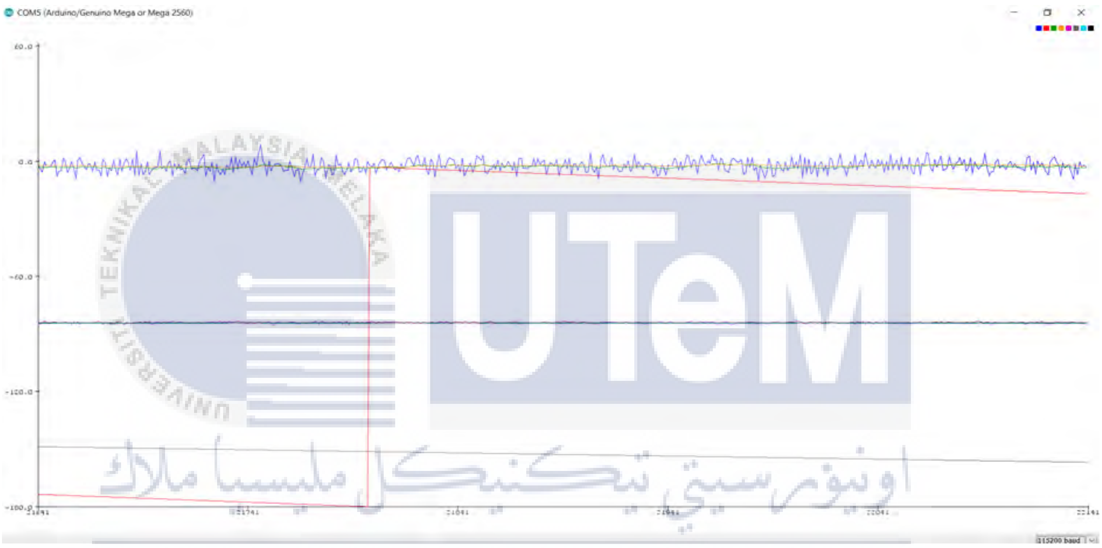


Figure 4.11: Drift angle for x axis (roll).

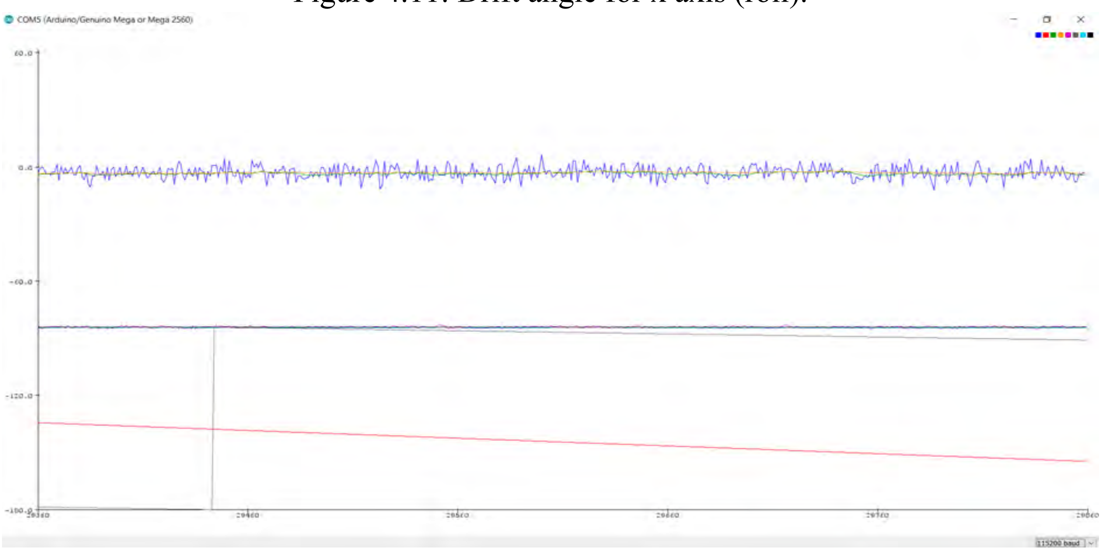


Figure 4.12: Drift angle for y axis (pitch).

For yaw angle which is z axis, there is no solution for the drifting angle. It is parallel to gravity vector, therefor yaw has no fixed reference point for any modification. Roll and pitch can be calibrated by a partial frame of reference which is gravity. The only way for getting an accurate result for yaw is to replace the MPU 6050 to MPU 9150. MPU 9150 is consists of 3 axis accelerometer, 3 axis gyroscope and magnetometer. Magnetometer is an important part to prevent the yaw angle from drift.

4.4 EXPERIMENT ON BALANCING TEST

After the addition of component and cable, the center of gravity for ROV is changed. The reason to carry out balancing test is to balance the ROV back so that it can operate well. Table 4.5 shows the observation for ROV in balancing test.

Table 4.5: Observation of ROV in balancing test.

Volume (m^3)		Balance	Observation	Adjustment
Left	Right			
2.7×10^{-5}	2.7×10^{-5}	No	ROV was submerged.	Replace a longer size.
5.4×10^{-5}	5.4×10^{-5}	No	ROV was submerged.	Replace a longer size.
8.1×10^{-5}	8.1×10^{-5}	Yes	ROV still little submerged.	Replace a longer size.
1.08×10^{-4}	1.08×10^{-4}	Yes	ROV neither float nor submerge.	-

From the observation in Table 4.5, ROV with different size of polystyrene foam will have different results and cause unbalance. The polystyrene was tie to ROV using cable ties. The longer the size of polystyrene foam will make the ROV to float. For the ROV to be operate well, balancing is an important factor. The 1.08×10^{-4} m^3 long polystyrene was able to make the ROV to be balance in water. Figure 4.13 shows the balance of ROV in water. The ROV is at neutral buoyancy which is important for stability of ROV. The polystyrene foam size must be symmetrical to maintain the accuracy of the result obtained. When observing the top surface of the ROV, the eyes is put perpendicular to the surface to prevent parallax error.

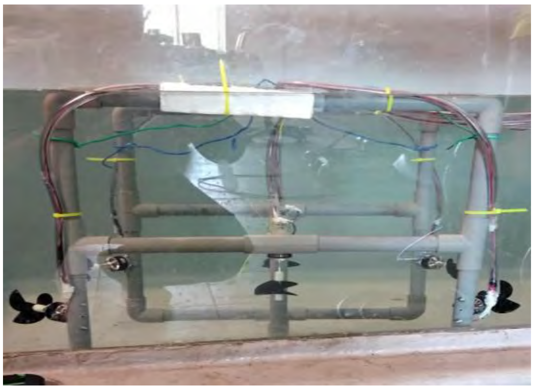


Figure 4.13: Balancing test of ROV.

4.5 EXPERIMENT ON BUOYANCY TEST

The buoyancy of the ROV is important because it helps to support the weight on the ROV. When an object is submerged in a fluid, the upward force which is buoyant force will cause the weight in air and in fluid to be different. Therefore, the buoyant force B is found by using the weight of object minus the tension force. The mass of ROV measured using electronic scale is 2.195kg without weight attached to it. The assumption made was the acceleration due to gravity is equals to 10m/s^2.

Table 4.6: Buoyant force on ROV.

Mass of ROV attached with average weight (kg)		F = m * a (N)		
In air, m_1	In water, m_2	$W = m_1 * a$	$T = m_2 * a$	B = W- T
2.195	0.000	21.95	0.00	21.95
2.445	0.200	24.45	2.00	22.45
2.695	0.415	26.95	4.15	22.80
2.945	0.603	29.45	6.03	23.42
3.195	0.810	31.95	8.10	23.85

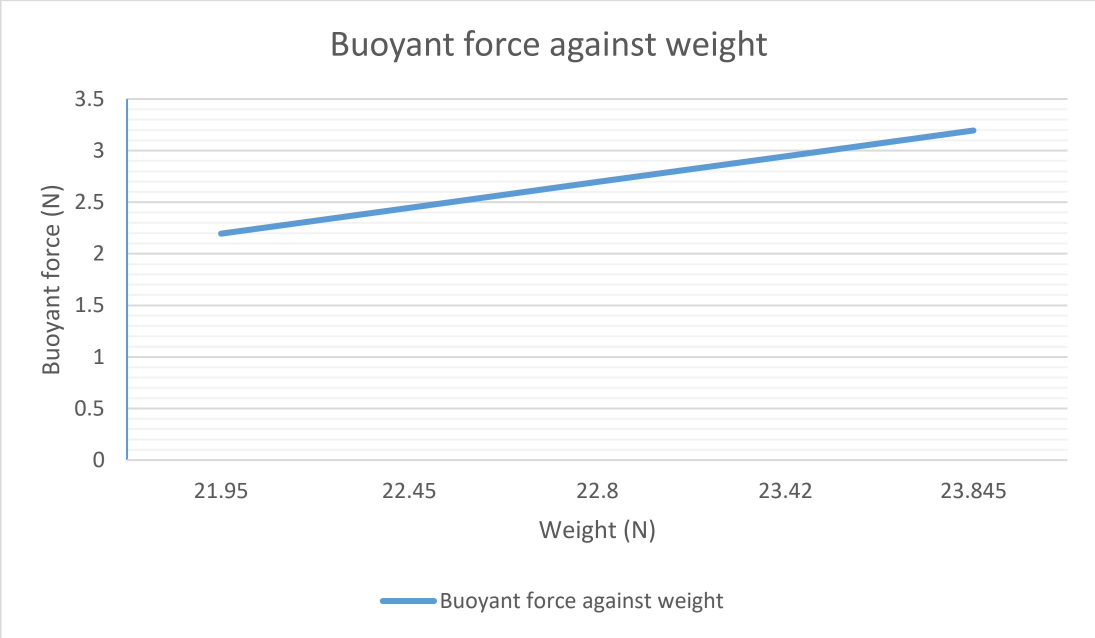


Figure 4.14: Graph of buoyant force against weight.

From Table 4.6, the mass of ROV with 2.195kg is the equilibrium mass as it neither float nor submerge. The experiment is only added until 1kg of weight because of the safety issue as the ROV itself is light. The added weight must be consistent to avoid inaccurate of the result. From the Figure 4.14, the buoyant force increases when the weight attached to ROV is increases. It has fulfilled the Archimedes principle stated on chapter 2.1.4 about buoyancy. The weight measured is repeated 3 times to get the average weight to reduce random error. Besides that, the electronic scale used is 3 decimal because the weight of ROV is light and the sensitivity is higher.

4.6 EXPERIMENT ON UNDERWATER MOTOR THRUST TEST.

Different motor speed will give different thrust in water. The motor is put in water as shown in Figure 4.15. The motor speed is not affect the thrust in air as result obtained in Table 4.7. The motor speed is differ a lot for the thrust in water as result obtained in Table 4.8. Form the graph shown in Figure 4.16, the motor speed is proportional to thrust. As the motor speed increased, the thrust of the motor also increased. Therefore at optimum motor speed will give the ROV a better stability because the angle of rotation will not be overshoot. The motor speed is set at 40% because the ROV itself is light and to avoid any overshoot occur for orientation control. The thrust obtained is repeated 3 times to get the average thrust in kilogram to prevent random error. The electronic scale used is with 3 decimal because it has higher sensitivity and to get a more accurate result.

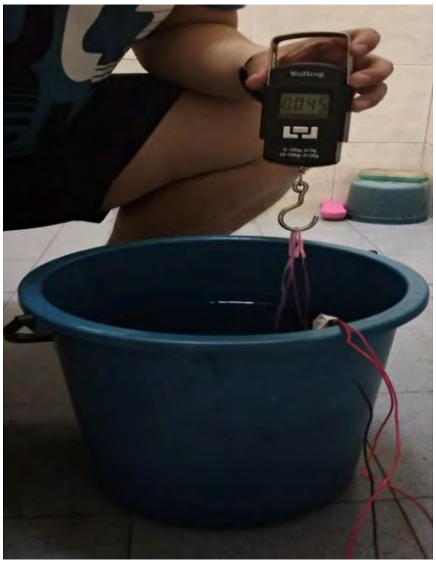


Figure 4.15: Measuring thrust of motor in water.

Table 4.7: Motor speed and average thrust in air.

Motor Speed (%)	Average Thrust (kg)
0	0.060
10	0.065
20	0.065
30	0.065
40	0.065
50	0.065
60	0.065
70	0.065
80	0.065
90	0.065
100	0.065

Table 4.8: Motor speed and average thrust in water.

Motor Speed (%)	Average Thrust (kg)
0	0.050
10	0.055
20	0.065
30	0.075
40	0.085
50	0.090
60	0.105
70	0.110
80	0.120
90	0.135
100	0.140

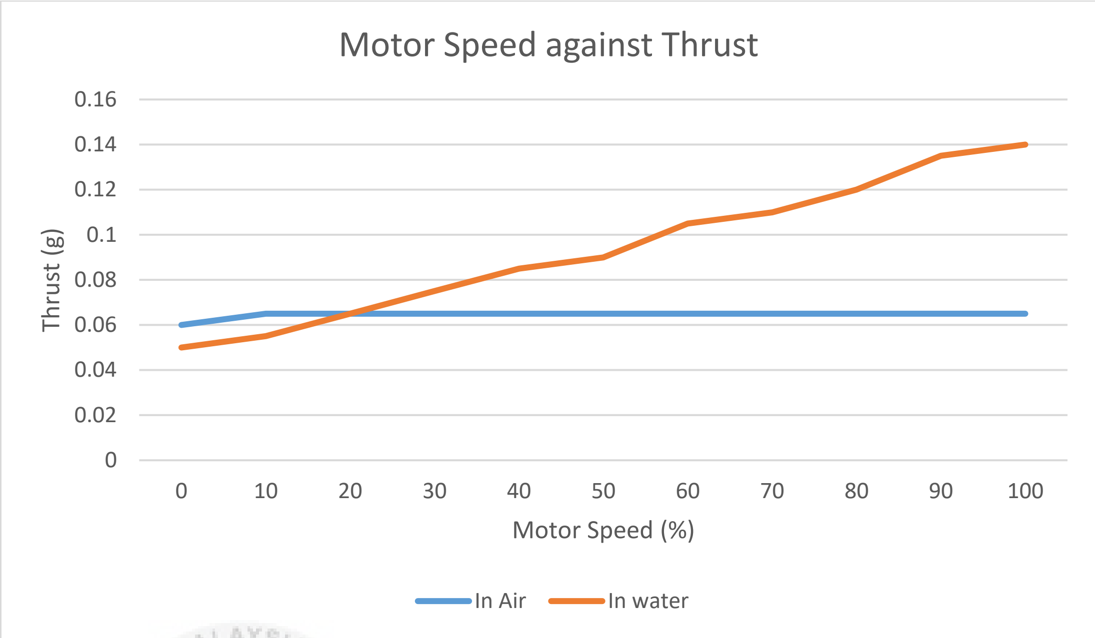


Figure 4.16: Graph of motor speed against thrust.

4.7 SUMMARY

Simulation tests such as center of gravity, sustainability test and stress test are successfully simulated by using SolidWorks Simulation Xpress. The results obtained were tabulated and discussed in this chapter. Besides that, for the hardware experiments such as MPU 6050, balancing test, buoyancy test and underwater motor thrust test were carried out in the lab pool. The design of ROV for this project was able to good stability in underwater based on the results obtained from experiments. The ROV is slightly positive in buoyancy force as the suggested from journals. Besides that, all the possible errors such as parallax error, random error and sensitivity error were eliminated. The MPU 6050 sensor was able to control the orientation of ROV in yaw, pitch and roll. However, the yaw angle was drifted and inaccurate during the test. This had caused the orientation control of ROV to less smooth. This problem can be solve by replacing the MPU 6050 sensor with MPU 9150.

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1 CONCLUSION

As a conclusion, design a remotely operated vehicle (ROV) was a challenging task. Before starting this project, obtain information and knowledge from other sources are very important. This is because there are plenty of design for ROV in articles. From the articles, the idea can be obtained to do modification and improvement for the performance of ROV. This project was to design a ROV with a control system for 3 degrees of freedom. The body frame was designed and constructed together with the electrical and software. Through the analysis using SolidWorks, the final design of ROV can be done with better performance. Several analysis were done such as the center of mass, sustainability and stress test, buoyant force, balancing test and motor speed test. All objectives are achieved by obtaining the result from the experiments. Objective 1 and 2 were achieved as the ROV is stable in underwater and had good orientation control. Besides that, the objective 3 was partially achieved as the angle of rotation on yaw is not able to control using MPU 6050 sensor.

5.2 FUTURE WORK

In future, student can replace the MPU 6050 sensor with MPU 9150 as the yaw angle obtained from sensor is not accurate. MPU 9150 is able to avoid the drift angle in all yaw, pitch and roll. Besides that, the control system such as PID controller and Fuzzy Logic controller is needed to enhance the steering system of the ROV for a better orientation control. This is because the controller can give feedback to motor if there is any deviation in angle during orientation control. Moreover, a better ballast system for ROV is needed for good stability in underwater orientation control.

REFERENCES

[1] "Hawaii Pacific University Oceanic Institute," [Online]. Available: https://www.oceanicinstitute.org/aboutoceans/aquafacts.html. [Accessed 16 November 2017].

[2] "Marine Technology Society," [Online]. Available: http://www.rov.org/rov_overview.cfm. [Accessed 17 November 2017].

[3] Capocci, R., Dooly, G., Omerdić, E., Coleman, J., Newe, T. and Toal, D., 2017. Inspection-class remotely operated vehicles—a review. Journal of Marine Science and Engineering, vol.5, no. 1, pp.13, 2017.

[4] D.-W. Kathryn Symes, "ROV Work-Class Operation Expenditure to Grow by 80%," 8 1 2014. [Online]. Available: https://www.spe.org/en/print-article/?art=232. [Accessed 17 November 2017].

[5] T. Week, "MH370: US team to be paid $15 miilion for finding missing plane," Jan. 11, 2017.

[6] Bagheri, A. and Moghaddam, J.J., 2009. Simulation and tracking control based on neural-network strategy and sliding-mode control for underwater remotely operated vehicle. Neurocomputing, 72(7-9), pp.1934-1950.

[7] S. W. Moore, H. Bohm and V. Jensen, Underwater Robotics: Science, Design & Fabrication, Marine Advanced Technology Edu; 1st edition (2010), 2010.

[8] K. T. R and D. A. Levinson, Dynamics, Theory and Applications, McGraw Hill, 1985.

[9] "ROV Applications - Design- Ballast, Buoyancy Control," [Online]. Available: http://www.rov.org/rov_design_ballast.cfm. [Accessed 18 November 2017].

[10] N. Gupta, S. E.Zeltamann, V. Chakravarthy, Shunmugasamy and D. Pinisetty, "Applications of Polymer Matrix Syntactic Foams," *The Journal of The Minerals, Metals & Materials Society (TMS),* vol. 66, no. 2, pp. 245-254, 2014.

[11] S. Robert D Christ and Robert L. Wernli, Components of an ROV system - Part 1: Mechanical and electromechanical systems, 2009.

[12] "PID Theory Explained," National Instruments, 29 March 2011. [Online]. Available: http://www.ni.com/white-paper/3782/en/. [Accessed 18 November 2017].

[13] N. Fenichel, "Ziegler-Nichols Tuning Rules for PID," Microstar Laboratories, 1994. [Online]. Available: www.mstarlabs.com/control/znrule.html.

[14] M. Diesel and Turbo, Basic Principles of Ship Propulsion, Loose Leaf - 2012.

[15] Chakrabarti and S. K., The Theory and Practice of Hydrodynamics and Vibration, Advanced Series on Ocean Engineering, 2002.

[16] Eng, Y. H., Lau, W. S., Low, E., & Seet, G. G. L. (2008, March). Identification of the hydrodynamics coefficients of an underwater vehicle using free decay pendulum motion. In International MultiConference of Engineers and Computer Scientists (Vol. 2, pp. 423-430).

[17] "Advantages of computational fluid dynamic," Pre Technologies, 2014. [Online]. Available: http://www.pretechnologies.com/services/computational-fluid-dynamics/advantage. [Accessed 19 November 2017].

[18] "What is a Joystick and its Type," Education and Information Technology, [Online]. Available: http://aumedu.blogspot.my/2016/08/what-is-joystick-and-its-type.html. [Accessed 19 November 2017].

[19] "Understanding the Progression of Joystick Technologies," CTI Electronics, [Online]. Available: http://www.ctielectronics.com/Potentiometer-Hall-Effect-Inductive-Joystick-Background.php. [Accessed 19 November 2017].

[20] García-Valdovinos, L.G., Salgado-Jiménez, T., Bandala-Sánchez, M., Nava-Balanzar, L., Hernández-Alvarado, R. and Cruz-Ledesma, J.A., 2014. Modelling, design and robust control of a remotely operated underwater vehicle. International Journal of Advanced Robotic Systems, 11(1), pp.1.

[21] A. Marzbanrad, M. Eghtesad, J. Sharafi and R. Kamali, "Design, Construction and Control of a Remotely Operated Vehicle (ROV)," *In ASME 2011 International Mechanical Engineering Congress and Exposition,* pp. 1295-1304, 2011.

[22] Nguyen, D. Anh, Q. H. Cao and P. H. Nguyen, "Research, Design and Control of a Remotely Operated Underwater Vehicle," *5th World Conference on Applied Sciences, Engineering & Technology,* pp. 391-396, 2016.

[23] R. Goodrich, "Accelerometer vs. Gyroscope: What's the Difference?," Live Science, 1 October 2013. [Online]. Available: https://www.livescience.com/40103-accelerometer-vs-gyroscope.html. [Accessed 20 November 2017].

[24] Abidin, A. Zainal, Mardiyanto, Ronny, Purwanto and Djoko, "Implementation of PID Controller for Hold Altitude Control in Underwater Remotely Operated Vehicle," *In Intelligent Technology and Its Applications (ISITIA), 2016 International Seminar on,* pp. 665-670, 2016.

[25] T. H. Koh, M. W. S. Lau, E. Low, G. Seet, S. Swei and P. L. Cheng, "Preliminary Studies of the Modelling and Control of Twin-Barrel Under actuated Underwater

Robotic Vehicle," *In Control, Automation, Robotics and Vision, 2002. ICARCV 2002. 7th International Conference,* vol. 2, pp. 1042-1047, 2002.

[26] Żak, A., 2011. Fuzzy controller for underwater remotely operated vehicle which is moving in conditions of environment disturbance occurrence. Journal of KONES, 18, pp.499-507.

[27] K. D. Le, H. D. Nguyen and D. Ranmuthugala, "A Self-tuning Nonlinear PID Controller for a Three-Thruster Remotely Operated Underwater Vehicle," *In The Second Vietnam Conference on Control and Automation,* pp. 1-8, 2013.

[28] Chin, C. S., Lau, M. W. S., E. Low, Seet and G. G. L., "Design of Thruster Configuration and Thrust Allocation Control for a Remotely Operated Vehicle," *Robotics, Automation and Mechatronics, 2006 IEEE Conference,* pp. 1-6, 2006.

[29] I. C. Rust and H. H. Asada, "The Eyeball ROV: Design and Control of a Spherical Underwater Vehicle Steered by an Internal Eccentric Mass," *2011 IEEE International Conference on Robotics and Automation,* pp. 5855-5862, 2011.

[30] Jayasundere, N. D., Gunawickrama and S. H. K. K., "Underwater ROV with Fuzzy Logic Motion Control," *Information and Automation for Sustainability (ICIAfS), 2016 IEEE International Conference,* pp. 1-6, 2016.

[31] "MPU-6050 Six-Axis (Gyro + Accelerometer) MEMS MotionTracking™ Devices," TDK InvenSense, [Online]. Available: https://www.invensense.com/products/motion-tracking/6-axis/mpu-6050/. [Accessed 20 November 2017].

[32] "Hobby King," Hobby King, [Online]. Available: https://hobbyking.com/en_us/turnigy-aerodrive-dst-700-brushless-outrunner-motor-700kv.html. [Accessed 14 December 2017].

[33] K. A. B. A. Rahim, Z. M. Zain, N. Harun and M. M. Noh, "Hull Design for ROV with Four Thrusters," *The National Conference for Postgraduate Research 2016, Universiti Malaysia Pahang,* 2016.

[34] I. S. V. Pennam Sanjay, "A Report On Ball and Beam Balancing," 2012.

[35] "OpenROV," OpenROV, [Online]. Available: https://store.openrov.com/products/neutrally-buoyant-tether. [Accessed 14 December 2017].

APPENDIX A

InvenSense	MPU-6000/MPU-6050 Product Specification	Document Number: PS-MPU-6000A-00 Revision: 1.0 Release Date: 11/24/2010

2 Purpose and Scope

This product specification provides preliminary information regarding the electrical specification and design related information for the MPU-6000™ and MPU-6050™, collectively called the MPU-60X0™ or MPU™.

Electrical characteristics are based upon design analysis and simulation results only. Specifications are subject to change without notice. Final specifications will be updated based upon characterization of production silicon. For references to register map and descriptions of individual registers, please refer to MPU-6000 Register Map and Descriptions document.

3 Product Overview

The MPU-60X0 Motion Processing Unit (MPU™) is the world's first motion processing solution with integrated 9-axis sensor fusion for handset and tablet applications, game controllers, motion pointer remote controls, and other consumer devices. The MPU-60X0 has an embedded 3-axis MEMS gyroscope, 3-axis MEMS accelerometer and Digital Motion Processor™ (DMP) hardware accelerator engine with an auxiliary I^2C port that interfaces to third party digital sensors, such as magnetometers. Interfacing with a 3-axis magnetometer delivers a complete 9-axis sensor fusion output to the MPU's primary I^2C or SPI port. (SPI is available on MPU-6000 only). This combines acceleration and rotational motion plus heading information into a single data stream for the application. This motion processing technology integration provides a smaller footprint and has inherent cost advantages compared to discrete gyroscope plus accelerometer solutions. The MPU-60X0 is also designed to interface with multiple non-inertial digital sensors, such as pressure sensors, on its auxiliary I^2C port. The MPU-60X0 is a second generation motion processor and is footprint compatible with the MPU-30X0 family.

The MPU-60X0 features three 16-bit analog-to-digital converters (ADCs) for digitizing the gyroscope outputs and three 16-bit ADCs for digitizing the accelerometer outputs. For precision tracking of both fast and slow motions, the parts feature a user-programmable gyroscope full-scale range of ±250, ±500, ±1000, and ±2000°/sec (dps) and a user-programmable accelerometer full-scale range of ±2*g*, ±4*g*, ±8*g*, and ±16*g*.

An on-chip 1024 Byte FIFO helps lower system power consumption by allowing the system processor to read the sensor data in bursts and then enter a low-power mode as the FIFO collects more data. With all the necessary on-chip processing and sensor components required to support many motion-based use cases, the MPU-60X0 uniquely supports a variety of advanced motion-based applications entirely on-chip and therefore is instrumental in enabling low-power motion processing in portable applications with reduced processing requirements for the system processor. By providing an integrated sensor fusion output, the DMP in the MPU-60X0 offloads the intensive motion processing computation requirements from the system processor, minimizing the need for frequent polling of the motion sensor output.

Communication with all registers of the device is performed using either I^2C at 400kHz or SPI at 1MHz (MPU-6000 only). For applications requiring faster communications, reading of the sensor and interrupt registers may be performed using SPI at 20MHz (MPU-6000 only). Additional features include an embedded temperature sensor and an on-chip oscillator with ±1% variation over the operating temperature range.

By leveraging its patented and volume-proven Nasiri-Fabrication platform, which integrates MEMS wafers with companion CMOS electronics through wafer-level bonding, InvenSense has driven the MPU-60X0 package size down to a revolutionary footprint of 4x4x0.9mm (QFN), while providing the highest performance, lowest noise, and the lowest cost semiconductor packaging required for handheld consumer electronic devices. The part features a robust 10,000*g* shock tolerance, and has programmable low-pass filters for the gyroscopes, accelerometers, and the on-chip temperature sensor.

For power supply flexibility, the MPU-60X0 operates from VDD power supply voltages of 2.5V±5%, 3.0V±5%, or 3.3V±5%. Additionally, the MPU-6050 provides a VLOGIC reference pin (in addition to its analog supply pin, VDD), which sets the logic levels of its I^2C interface. The VLOGIC voltage may be 1.8V±5% or VDD.

InvenSense	MPU-6000/MPU-6050 Product Specification	Document Number: PS-MPU-6000A-00 Revision: 1.0 Release Date: 11/24/2010

The MPU-6000 and MPU-6050 are identical, except that the MPU-6050 supports the I^{2}C serial interface only, and has a separate VLOGIC reference pin. The MPU-6000 supports both I^{2}C and SPI interfaces and has a single supply pin, VDD, which is both the device's logic reference supply and the analog supply for the part. The table below outlines these differences:

Primary Differences between MPU-6000 and MPU-6050

Part / Item	MPU-6000	MPU-6050
VDD	2.5V±5%, 3.0V±5%, or 3.3V±5%.	2.5V±5%, 3.0V±5%, or 3.3V±5%.
VLOGIC	n/a	1.71V to VDD
Serial Interfaces Supported	I^{2}C, SPI	I^{2}C
Pin 8	/CS	VLOGIC
Pin 9	AD0/SDO	AD0
Pin 23	SCL/SCLK	SCL
Pin 24	SDA/SDI	SDA

4 Applications

- *BlurFree*™ technology (for Video/Still Image Stabilization)
- *AirSign*™ technology (for Security/Authentication)
- *TouchAnywhere*™ technology (for "no touch" UI Application Control/Navigation)
- *MotionCommand*™ technology (for Gesture Short-cuts)
- Motion-enabled game and application framework
- InstantGesture™ iG™ gesture recognition
- Location based services, points of interest, and dead reckoning
- Handset and portable gaming
- Motion-based game controllers
- 3D remote controls for Internet connected DTVs and set top boxes, 3D mice
- Wearable sensors for health, fitness and sports
- Toys

5 Features

5.1 Gyroscope Features

The triple-axis MEMS gyroscope in the MPU-60X0 includes a wide range of features:

- Digital-output X-, Y-, and Z-Axis angular rate sensors (gyroscopes) with a user-programmable full-scale range of ±250, ±500, ±1000, and ±2000°/sec
- External sync signal connected to the FSYNC pin supports image, video and GPS synchronization
- Integrated 16-bit ADCs provide simultaneous sampling of gyros
- Enhanced bias and sensitivity temperature stability reduces the need for user calibration
- Improved low-frequency noise performance
- Digitally-programmable low-pass filter
- Gyroscope operating current: 5mA
- Standby current: 5µA
- Factory calibrated sensitivity scale factor
- User self-test

5.2 Accelerometer Features

The triple-axis MEMS accelerometer in MPU-60X0 includes a wide range of features:

- Digital-output tri-axis accelerometer with a programmable full scale range of ±2*g*, ±4*g*, ±8*g* and ±16*g*
- Integrated 16-bit ADCs provide simultaneous sampling of accelerometers while requiring no external multiplexer
- Accel normal operating current: 500µA
- Low power accelerometer mode current: 40µA at 10 Hz
- Orientation detection and signaling
- Tap detection
- User-programmable interrupts
- Free-fall interrupt
- High-G interrupt
- Zero-motion/Motion interrupt
- User self test

5.3 Additional Features

The MPU-60X0 includes the following additional features:

- 9-Axis sensor fusion via on-chip Digital Motion Processor (DMP)
- Auxiliary master I^2C bus for reading data from external sensors (e.g., magnetometer)
- 5.5mA operating current for all 6 axes
- VDD supply voltages of 2.5V±5%, 3.0V±5%, 3.3V±5%
- Flexible VLOGIC reference voltage allows for multiple I^2C interface voltages (MPU-6050 only)
- Smallest and thinnest package for portable devices: 4x4x0.9mm QFN
- Minimal cross-axis sensitivity between the accelerometer and gyroscope axes
- 1024 byte FIFO reduces power consumption by allowing host processor to read the data in bursts and then go into a low-power mode as the FIFO collects more data
- Digital-output temperature sensor
- User-programmable digital filters for gyroscope, accelerometer, and temp sensor
- 10,000 *g* shock tolerant
- 400kHz Fast Mode I^2C for communicating with all registers

InvenSense	MPU-6000/MPU-6050 Product Specification	Document Number: PS-MPU-6000A-00 Revision: 1.0 Release Date: 11/24/2010

- 1MHz SPI serial interface for communicating with all registers (MPU-6000 only)
- 20MHz SPI serial interface for reading of sensor and interrupt registers (MPU-6000 only)
- MEMS structure hermetically sealed and bonded at wafer level
- RoHS and Green compliant

5.4 Motion Processing

- Internal Digital Motion Processing™ (DMP™) engine supports 3D motion processing and gesture recognition algorithms
- MPU-60X0 collects the gyroscope and accelerometer data while synchronizing data sampling at a user defined rate. The total data set obtained by the MPU-60X0 includes 3-axis gyroscope data and 3-axis accelerometer data, and temperature data. The MPU calculated output can also include compass data from a digital 3-axis third party magnetometer.
- FIFO buffers the complete data set, reducing timing requirements on the system processor and saving power by letting the processor burst read the FIFO data, and then enter a low-power sleep mode while the MPU collects more data.
- Programmable interrupt supports features such as gesture recognition, panning, zooming, scrolling, zero-motion detection, tap detection, and shake detection
- Programmable low-pass filters.
- Low-power pedometer functionality allows the host processor to sleep while the DMP maintains the step count.

5.5 Clocking

- On-chip timing generator ±1% frequency variation over full temperature range
- Optional external clock inputs of 32.768kHz or 19.2MHz

APPENDIX B

User Manual of Brushless Speed Controller

Thanks for purchasing our Electronic Speed Controller (ESC). High power system for RC model is very dangerous, please read this manual carefully. In that we have no control over the correct use, installation, application, or maintenance of our products, no liability shall be assumed nor accepted for any damages, losses or costs resulting from the use of the product. Any claims arising from the operating, failure or malfunctioning etc. will be denied. We assume no liability for personal injury, property damage or consequential damages resulting from our product or our workmanship. As far as is legally permitted, the obligation to compensation is limited to the invoice amount of the affected product.

Specifications

Model	Cont. Current	Burst Current (≤10s)	BEC Mode	BEC Output	BEC Output Capability				Battery Cell		Weight	Size
					2S Lipo	3S Lipo	4S Lipo	6S Lipo	Lipo	NiMH		L*W*H
Skywalker-6A	6A	8A	Linear	5V/0.8A	3 servos				2S	5-6 cells	5.5g	32*12*4.5
Skywalker-12A	12A	15A	Linear	5V/1A	3 servos	2 servos			2-3S	5-9 cells	9g	38*18*6
Skywalker-12AE	12A	15A	Linear	5V/2A	5 servos	4 servos			2-3S	5-9 cells	10g	38*18*7
Skywalker-15A	15A	20A	Linear	5V/2A	5 servos	4 servos			2-3S	5-9 cells	16.5g	48*22.5*6
Skywalker-20A	20A	25A	Linear	5V/2A	5 servos	4 servos			2-3S	5-9 cells	19g	42*25*8
Skywalker-30A	30A	40A	Linear	5V/2A	5 servos	4 servos			2-3S	5-9 cells	37g	68*25*8
Skywalker-40A	40A	55A	Linear	5V/3A	5 servos	4 servos			2-3S	5-9 cells	39g	68*25*8
Skywalker-40A-UBEC	40A	55A	Switch	5V/3A	5 servos	5 servos	5 servos		2-4S	5-12 cells	43g	65*25*12
Skywalker-50A-UBEC	50A	65A	Switch	5V/5A	8 servos	8 servos	6 servos	6 servos	2-4S	5-12 cells	41g	65*29*10
Skywalker-60A-UBEC	60A	80A	Switch	5V/5A	8 servos	8 servos	6 servos	6 servos	2-6S	5-18 cells	63g	77*35*14
Skywalker-60A-OPTO	60A	80A	N/A	N/A					2-6S	5-18 cells	60g	86*38*12
Skywalker-80A-UBEC	80A	100A	Switch	5V/5A	8 servos	8 servos	6 servos	6 servos	2-6S	5-18 cells	82g	86*38*12
Skywalker-80A-OPTO	80A	100A	N/A	N/A					2-6S	5-18 cells	79g	86*38*12

Programmable Items **(The option written in bold font is the default setting)**

1. Brake Setting: **Enabled / Disabled**
2. Battery Type: **Lipo** / NiMH
3. Low Voltage Protection Mode(Cut-Off Mode): **Soft Cut-Off (Gradually reduce the output power)** /Cut-Off (Immediately stop the output power)
4. Low Voltage Protection Threshold(Cut-Off Threshold): Low / **Medium** / High
   1) For lithium battery, the battery cell number is calculated automatically. Low / medium / high cutoff voltage for each cell is: 2.85V/3.15V/3.3V. For example: For a 3S Lipo, when "Medium" cutoff threshold is set, the cut-off voltage will be: 3.15*3=9.45V
   2) For NiMH battery, low / medium / high cutoff voltages are 0%/50%/65% of the startup voltage (i.e. the initial voltage of battery pack), and 0% means the low voltage cut-off function is disabled. For example: For a 6 cells NiMH battery, fully charged voltage is 1.44*6=8.64V, when "Medium" cut-off threshold is set, the cut-off voltage will be: 8.64*50%=4.32V.
5. Startup Mode: **Normal** /Soft /Super-Soft (300ms / 1.5s / 3s)
   a) Normal mode is suitable for fixed-wing aircraft. Soft or Super-soft modes are suitable for helicopters. The initial acceleration of the Soft and Super-Soft modes are slower, it takes 1.5 second for Soft startup or 3 seconds for Super-Soft startup from initial throttle advance to full throttle. If the throttle is completely closed (throttle stick moved to bottom position) and opened again (throttle stick moved to top position) within 3 seconds after the first startup, the re-startup will be temporarily changed to normal mode to get rid of the chance of a crash caused by slow throttle response. This special design is suitable for aerobatic flight when quick throttle response is needed.
6. Timing: **Low** / Medium / High,(3.75° /15° /26.25°)
   Usually, low timing is suitable for most motors. To get higher speed, High timing value can be chosen.

Begin To Use Your New ESC

IMPORTANT! Because different transmitter has different throttle range, please calibrate throttle range before flying.

Throttle range setting (Throttle range should be reset whenever a new transmitter is being used)

Switch on the transmitter, move throttle stick to the top position	Connect battery pack to the ESC, and wait for about 2 seconds	The "Beep-Beep-" tone should be emitted, means the top point of throttle range has been confirmed	Move throttle stick to the bottom position, several "beep-" tones should be emitted to present the amount of battery cells	A long "Beep-" tone should be emitted, means the lowest point of throttle range has been correctly confirmed

Normal startup procedure

Move throttle stick to bottom position and then switch on transmitter.	Connect battery pack to ESC, special tone like " ♪ 123" means power supply is OK	Several "beep-" tones should be emitted to present the amount of lithium battery cells	When self-test is finished, a long "beep-----" tone should be emitted	Move throttle stick upwards to go flying

Protection Function

1. Start up failure protection: If the motor fails to start within 2 seconds of throttle application, the ESC will cut-off the output power. In this case, the throttle stick **MUST** be moved to the bottom again to restart the motor. (Such a situation happens in the following cases: The connection between ESC and motor is not reliable, the propeller or the motor is blocked, the gearbox is damaged, etc.)
2. Over-heat protection: When the temperature of the ESC is over about 110 Celsius degrees, the ESC will reduce the output power.

3. Throttle signal loss protection: The ESC will reduce the output power if throttle signal is lost for 1 second, further loss for 2 seconds will cause the output to be cut-off completely.

Trouble Shooting

Trouble	Possible Reason	Action
After power on, motor does not work, no sound is emitted	The connection between battery pack and ESC is not correct	Check the power connection. Replace the connector.
After power on, motor does not work, such an alert tone is emitted: "beep-beep-, beep-beep-,beep-beep-" (Every "beep-beep-" has a time interval of about 1 second)	Input voltage is abnormal, too high or too low.	Check the voltage of battery pack
After power on, motor does not work, such an alert tone is emitted: "beep-, beep-, beep- "(Every "beep-" has a time interval of about 2 seconds)	Throttle signal is irregular	Check the receiver and transmitter Check the cable of throttle channel
After power on, motor does not work, such an alert tone is emitted: "beep-, beep-, beep-" (Every "beep-" has a time interval of about 0.25 second)	The throttle stick is not in the bottom (lowest) position	Move the throttle stick to bottom position
After power on, motor does not work, a special tone "♪ 56712" is emitted after 2 beep tone (beep-beep-)	Direction of the throttle channel is reversed, so the ESC has entered the program mode	Set the direction of throttle channel correctly
The motor runs in the opposite direction	The connection between ESC and the motor need to be changed.	Swap any two wire connections between ESC and motor

Program the ESC with your transmitter (4 Steps)

Note: Please make sure the throttle curve is set to 0 when the throttle stick is at bottom position and 100% for the top position.

1. Enter program mode
2. Select programmable items
3. Set item's value (Programmable value)
4. Exit program mode

1. Enter program mode

1) Switch on transmitter, move throttle stick to top position, connect the battery pack to ESC
2) Wait for 2 seconds, the motor should emit special tone like "beep-beep-"
3) Wait for another 5 seconds, special tone like "♪ 56712" should be emitted, which means program mode is entered

2. Select programmable items

After entering program mode, you will hear 8 tones in a loop with the following sequence. If you move the throttle stick to bottom within 3 seconds after one kind of tones, this item will be selected.

1. "beep" brake (1 short tone)
2. "beep-beep-" battery type (2 short tone)
3. "beep-beep-beep-" cutoff mode (3 short tone)
4. "beep-beep-beep-beep-" cutoff threshold (4 short tone)
5. "beep-----" startup mode (1 long tone)
6. "beep-----beep-" timing (1 long 1 short)
7. "beep-----beep-beep-" set all to default (1 long 2 short)
8. "beep-----beep-----" exit (2 long tone)

Note: 1 long "beep-----" = 5 short "beep-"

3. Set item value (Programmable value)

You will hear several tones in loop. Set the value matching to a tone by moving throttle stick to top when you hear the tone, then a special tone "♪ 1515" emits, means the value is set and saved. (Keeping the throttle stick at top, you will go back to Step 2 and you can select other items; or moving the stick to bottom within 2 seconds will exit program mode directly)

Items \ Tones	"beep-" 1 short tone	"beep-beep-" 2 short tones	"beep-beep-beep" 3 short tones
Brake	Off	On	
Battery type	Lipo	NiMH	
Cutoff mode	Soft-Cut	Cut-Off	
Cutoff threshold	Low	Medium	High
Start mode	Normal	Soft	Super soft
Timing	Low	Medium	High

4. Exit program mode

There are 2 ways to exit program mode:

1. In step 3, after special tone "♪ 1515", please move throttle stick to the bottom position within 2 seconds.
2. In step 2, after tone "beep-----beep-----"(that is: The item #8), move throttle stick to bottom within 3 seconds.

APPENDIX C

Motor: A2212 KV:980					
Technical Datas		**Recommended Prop(inch)**			
KV	980	Standard	2s-1147/1155	Max thrust	2s-9050
Configu-ration	12N14P		3s-7035/8040		3s-8043
Stator Diameter	22mm				
STator Length	12mm				
Shaft Diameter	3mm				
Motor Dimension(Dia.*Len)	Φ27×30mm				
Weight(g)	50				
Idle current(10)@10v(A)	0.5				
No.of Cells(Lipo)	2-3S				
Max Continuous current(A)180S	14.5				
Max Continuous Power(W)180S	160				
Max. efficiency current	(4-9A)>80%				
internal resistance	120mΩ				

Tested with Angel 20A ESC					
Prop	**Volts (V)**	**Amps (A)**	**Watts (W)**	**Thrust (g)**	**Efficiency (g/W)**
9x4.7	7	4.4	30.8	360	11.69
	8.5	5.7	48.45	510	10.53
	10	7.3	73	610	8.36
	11	8.3	91.3	720	7.89
10x4.7	7	7.5	52.5	540	10.29
	8.5	9.6	81.6	660	8.09
	10	11.2	112	760	6.79
	11	12.4	136.4	820	6.01

APPENDIX D

```
/* Copyright (C) 2012 Kristian Lauszus, TKJ Electronics. All rights reserved.
 This software may be distributed and modified under the terms of the GNU
 General Public License version 2 (GPL2) as published by the Free Software
 Foundation and appearing in the file GPL2.TXT included in the packaging of
 this file. Please note that GPL2 Section 2[b] requires that all works based
 on this software must also be made publicly available under the terms of
 the GPL2 ("Copyleft").

 Kristian Lauszus, TKJ Electronics
 Web      :  http://www.tkjelectronics.com
 e-mail   :  kristianl@tkjelectronics.com
 */
#include <Wire.h>
#include "Kalman.h" // Source: https://github.com/TKJElectronics/KalmanFilter
#include<I2Cdev.h>
#include<MPU6050.h>
#include <Servo.h>
#define RESTRICT_PITCH // Comment out to restrict roll to ±90deg instead - please read: http://www.freescale.com/files/sensors/doc/app_note/AN3461.pdf
```



```
Kalman kalmanX; // Create the Kalman instances
Kalman kalmanY;

/* IMU Data */
double accX, accY, accZ;
double gyroX, gyroY, gyroZ;
int16_t tempRaw;

double gyroXangle, gyroYangle; // Angle calculate using the gyro only
double compAngleX, compAngleY; // Calculated angle using a complementary filter
double kalAngleX, kalAngleY; // Calculated angle using a Kalman filter
```

```
Servo servo1;
Servo servo2;
Servo servo3;
Servo servo4;
Servo servo5;
Servo servo6;
const int relayPin = 22;
const int relayPin1 = 23;
const int relayPin2 = 24;
const int relayPin3 = 25;
const int relayPin4 = 26;
const int relayPin5 = 27;
const int relayPin6 = 28;
const int relayPin7 = 29;
const int relayPin8= 30;
const int relayPin9 = 31;
const int relayPin10 = 32;
const int relayPin11 = 33;
const int relayPin12 = 34;
const int relayPin13 = 35;
const int relayPin14 = 36;
const int relayPin15 = 37;
const int relayPin16 = 38;
const int relayPin17 = 39;
const int relayPin18 = 40;
const int relayPin19 = 41;
const int relayPin20= 42;
const int relayPin21 = 43;
const int relayPin22 = 44;
const int relayPin23 = 45;
uint32_t timer;
uint8_t i2cData[14]; // Buffer for I2C data
```

```
// TODO: Make calibration routine
void setup() {
  Serial.begin(115200);
  Wire.begin();
  TWBR = ((F_CPU / 400000L) - 16) / 2; // Set I2C frequency to 400kHz
  i2cData[0] = 7; // Set the sample rate to 1000Hz - 8kHz/(7+1) = 1000Hz
  i2cData[1] = 0x00; // Disable FSYNC and set 260 Hz Acc filtering, 256 Hz Gyro filtering, 8 KHz sampling
  i2cData[2] = 0x00; // Set Gyro Full Scale Range to ±250deg/s
  i2cData[3] = 0x00; // Set Accelerometer Full Scale Range to ±2g
  while (i2cWrite(0x19, i2cData, 4, false)); // Write to all four registers at once
  while (i2cWrite(0x6B, 0x01, true)); // PLL with X axis gyroscope reference and disable sleep mode
  while (i2cRead(0x75, i2cData, 1));
  if (i2cData[0] != 0x68) { // Read "WHO_AM_I" register
    Serial.print(F("Error reading sensor"));
    while (1);
  }
  delay(100); // Wait for sensor to stabilize
  /* Set kalman and gyro starting angle */
  while (i2cRead(0x3B, i2cData, 6));
  accX = (i2cData[0] << 8) | i2cData[1];
  accY = (i2cData[2] << 8) | i2cData[3];
  accZ = (i2cData[4] << 8) | i2cData[5];
  // Source: http://www.freescale.com/files/sensors/doc/app_note/AN3461.pdf eq. 25 and eq. 26
  // atan2 outputs the value of -π to π (radians) - see http://en.wikipedia.org/wiki/Atan2
  // It is then converted from radians to degrees
#ifdef RESTRICT_PITCH // Eq. 25 and 26
  double roll  = atan2(accY, accZ) * RAD_TO_DEG;
  double pitch = atan(-accX / sqrt(accY * accY + accZ * accZ)) * RAD_TO_DEG;
```

```
#else // Eq. 28 and 29
  double roll  = atan(accY / sqrt(accX * accX + accZ * accZ)) * RAD_TO_DEG;
  double pitch = atan2(-accX, accZ) * RAD_TO_DEG;
#endif
  kalmanX.setAngle(roll); // Set starting angle
  kalmanY.setAngle(pitch);
  gyroXangle = roll;
  gyroYangle = pitch;
  compAngleX = roll;
  compAngleY = pitch;
  timer = micros();
  pinMode(relayPin, OUTPUT);
 pinMode(relayPin1, OUTPUT);
 pinMode(relayPin2, OUTPUT);
 pinMode(relayPin3, OUTPUT);// Set Pin connected to Relay as an OUTPUT
 pinMode(relayPin4, OUTPUT);
 pinMode(relayPin5, OUTPUT);
 pinMode(relayPin6, OUTPUT);
 pinMode(relayPin7, OUTPUT);
 pinMode(relayPin8, OUTPUT);
 pinMode(relayPin9, OUTPUT);
 pinMode(relayPin10, OUTPUT);
 pinMode(relayPin11, OUTPUT);
 pinMode(relayPin12, OUTPUT);
 pinMode(relayPin13, OUTPUT);
 pinMode(relayPin14, OUTPUT);
 pinMode(relayPin15, OUTPUT);// Set Pin connected to Relay as an OUTPUT
 pinMode(relayPin16, OUTPUT);
 pinMode(relayPin17, OUTPUT);
 pinMode(relayPin18, OUTPUT);
 pinMode(relayPin19, OUTPUT);
 pinMode(relayPin20, OUTPUT);
```

```
 pinMode(relayPin21, OUTPUT);
 pinMode(relayPin22, OUTPUT);
 pinMode(relayPin23, OUTPUT);
 servo1.attach(3);
servo2.attach(4);
servo3.attach(8);
servo4.attach(9);
servo5.attach(10);
servo6.attach(11);
}
void loop() {
  /* Update all the values */
  while (i2cRead(0x3B, i2cData, 14));
  accX = ((i2cData[0] << 8) | i2cData[1]);
  accY = ((i2cData[2] << 8) | i2cData[3]);
  accZ = ((i2cData[4] << 8) | i2cData[5]);
  tempRaw = (i2cData[6] << 8) | i2cData[7];
  gyroX = (i2cData[8] << 8) | i2cData[9];
  gyroY = (i2cData[10] << 8) | i2cData[11];
  gyroZ = (i2cData[12] << 8) | i2cData[13];
  double dt = (double)(micros() - timer) / 1000000; // Calculate delta time
  timer = micros();
  // Source: http://www.freescale.com/files/sensors/doc/app_note/AN3461.pdf eq. 25
and eq. 26
  // atan2 outputs the value of -π to π (radians) - see
http://en.wikipedia.org/wiki/Atan2
  // It is then converted from radians to degrees
#ifdef RESTRICT_PITCH // Eq. 25 and 26
  double roll  = atan2(accY, accZ) * RAD_TO_DEG;
  double pitch = atan(-accX / sqrt(accY * accY + accZ * accZ)) * RAD_TO_DEG;
#else // Eq. 28 and 29
  double roll  = atan(accY / sqrt(accX * accX + accZ * accZ)) * RAD_TO_DEG;
  double pitch = atan2(-accX, accZ) * RAD_TO_DEG;
```

```
#endif

  double gyroXrate = gyroX / 131.0; // Convert to deg/s
  double gyroYrate = gyroY / 131.0; // Convert to deg/s
#ifdef RESTRICT_PITCH
  // This fixes the transition problem when the accelerometer angle jumps between -180 and 180 degrees
  if ((roll < -90 && kalAngleX > 90) || (roll > 90 && kalAngleX < -90)) {
    kalmanX.setAngle(roll);
    compAngleX = roll;
    kalAngleX = roll;
    gyroXangle = roll;
  } else
    kalAngleX = kalmanX.getAngle(roll, gyroXrate, dt); // Calculate the angle using a Kalman filter
  if (abs(kalAngleX) > 90)
    gyroYrate = -gyroYrate; // Invert rate, so it fits the restriced accelerometer reading
  kalAngleY = kalmanY.getAngle(pitch, gyroYrate, dt);
#else
  // This fixes the transition problem when the accelerometer angle jumps between -180 and 180 degrees
  if ((pitch < -90 && kalAngleY > 90) || (pitch > 90 && kalAngleY < -90)) {
    kalmanY.setAngle(pitch);
    compAngleY = pitch;
    kalAngleY = pitch;
    gyroYangle = pitch;
  } else
    kalAngleY = kalmanY.getAngle(pitch, gyroYrate, dt); // Calculate the angle using a Kalman filter
  if (abs(kalAngleY) > 90)
    gyroXrate = -gyroXrate; // Invert rate, so it fits the restriced accelerometer reading
  kalAngleX = kalmanX.getAngle(roll, gyroXrate, dt); // Calculate the angle using a Kalman filter
#endif
```

```
  gyroXangle += gyroXrate * dt; // Calculate gyro angle without any filter
  gyroYangle += gyroYrate * dt;
  //gyroXangle += kalmanX.getRate() * dt; // Calculate gyro angle using the unbiased
rate
  //gyroYangle += kalmanY.getRate() * dt;
  compAngleX = 0.93 * (compAngleX + gyroXrate * dt) + 0.07 * roll; // Calculate
the angle using a Complimentary filter
  compAngleY = 0.93 * (compAngleY + gyroYrate * dt) + 0.07 * pitch;
  // Reset the gyro angle when it has drifted too much
  if (gyroXangle < -180 || gyroXangle > 180)
    gyroXangle = kalAngleX;
  if (gyroYangle < -180 || gyroYangle > 180)
    gyroYangle = kalAngleY;
  /* Print Data */
#if 0 // Set to 1 to activate
  Serial.print(accX); Serial.print("\t");
  Serial.print(accY); Serial.print("\t");
  Serial.print(accZ); Serial.print("\t");
  Serial.print(gyroX); Serial.print("\t");
  Serial.print(gyroY); Serial.print("\t");
  Serial.print(gyroZ); Serial.print("\t");
  Serial.print("\t");
#endif
  Serial.print(roll); Serial.print("\t");
  Serial.print(gyroXangle); Serial.print("\t");
  Serial.print(compAngleX); Serial.print("\t");
  Serial.print(kalAngleX); Serial.print("\t");
  Serial.print("\t");
  Serial.print(pitch); Serial.print("\t");
  Serial.print(gyroYangle); Serial.print("\t");
  Serial.print(compAngleY); Serial.print("\t");
  Serial.print(kalAngleY); Serial.print("\t");
#if 0 // Set to 1 to print the temperature
```

```
 Serial.print("\t");


 double temperature = (double)tempRaw / 340.0 + 36.53;
 Serial.print(temperature); Serial.print("\t");
#endif
motorstop();
 Serial.print("\r\n");
 delay(2);
  if(kalAngleX > -95 &&kalAngleX < 80 && kalAngleY < -10 && kalAngleY > -
86){
 rollleft();
 delay(500);}
 else if(kalAngleX > 80 &&kalAngleX < 98 && kalAngleY < 3.5 && kalAngleY >
-86){
 rollright();
 delay(500);}
 else if(kalAngleX > -180 &&kalAngleX < 80 && kalAngleY < -5 &&
kalAngleY > -85){
 pitchup();
 delay(500);}
  else if(kalAngleX > 6 &&kalAngleX <80 && kalAngleY < -5 && kalAngleY > -
84){
 pitchdown();
 delay(500);}
}
void motor1B(){
digitalWrite(relayPin,  HIGH);  // Set Pin to LOW to turn Relay OFF
digitalWrite(relayPin1, HIGH);
digitalWrite(relayPin2, LOW);
digitalWrite(relayPin3, LOW);
 delay(500);
}
void motor1F(){
```

```
digitalWrite(relayPin,  LOW);  // Set Pin to LOW to turn Relay OFF
digitalWrite(relayPin1, LOW);
digitalWrite(relayPin2, HIGH);
digitalWrite(relayPin3, HIGH);
 delay(500);
}
void motor2B(){
  digitalWrite(relayPin4,  HIGH);  // Set Pin to LOW to turn Relay OFF
digitalWrite(relayPin5, HIGH);
digitalWrite(relayPin6, LOW);
digitalWrite(relayPin7, LOW);
 delay(500);
}
void motor2F(){
digitalWrite(relayPin4,  LOW);  // Set Pin to LOW to turn Relay OFF
digitalWrite(relayPin5, LOW);
digitalWrite(relayPin6, HIGH);
digitalWrite(relayPin7, HIGH);
 delay(500);
}
void motor3B(){
 digitalWrite(relayPin8,  HIGH);  // Set Pin to LOW to turn Relay OFF
digitalWrite(relayPin9, HIGH);
digitalWrite(relayPin10, LOW);
digitalWrite(relayPin11, LOW);
 delay(500);
}
void motor3F(){
digitalWrite(relayPin8,  LOW);  // Set Pin to LOW to turn Relay OFF
digitalWrite(relayPin9, LOW);
digitalWrite(relayPin10, HIGH);
digitalWrite(relayPin11, HIGH);
```

```
 delay(500);
}
void motor4B(){
digitalWrite(relayPin12,  HIGH);  // Set Pin to LOW to turn Relay OFF
digitalWrite(relayPin13, HIGH);
digitalWrite(relayPin14, LOW);
digitalWrite(relayPin15, LOW);
 delay(500);
}
void motor4F(){
digitalWrite(relayPin12,  LOW);  // Set Pin to LOW to turn Relay OFF
digitalWrite(relayPin13, LOW);
digitalWrite(relayPin14, HIGH);
digitalWrite(relayPin15, HIGH);
 delay(500);
}
void motor5B(){
digitalWrite(relayPin16,  HIGH);  // Set Pin to LOW to turn Relay OFF
digitalWrite(relayPin17, HIGH);
digitalWrite(relayPin18, LOW);
digitalWrite(relayPin19, LOW);
 delay(500);
}
void motor5F(){
digitalWrite(relayPin16,  LOW);  // Set Pin to LOW to turn Relay OFF
digitalWrite(relayPin17, LOW);
digitalWrite(relayPin18, HIGH);
digitalWrite(relayPin19, HIGH);
 delay(500);
}
void motor6B(){
digitalWrite(relayPin20,  HIGH);  // Set Pin to LOW to turn Relay OFF
```

```
digitalWrite(relayPin21, HIGH);
digitalWrite(relayPin22, LOW);
digitalWrite(relayPin23, LOW);
 delay(500);
}
void motor6F(){
digitalWrite(relayPin20,  LOW);  // Set Pin to LOW to turn Relay OFF
digitalWrite(relayPin21, LOW);
digitalWrite(relayPin22, HIGH);
digitalWrite(relayPin23, HIGH);
 delay(500);
}
void motorstop(){
 digitalWrite(relayPin, HIGH);
digitalWrite(relayPin1, HIGH);
digitalWrite(relayPin2, HIGH);
digitalWrite(relayPin3, HIGH);
digitalWrite(relayPin4, HIGH);
digitalWrite(relayPin5, HIGH);
digitalWrite(relayPin6, HIGH);
digitalWrite(relayPin7, HIGH);
digitalWrite(relayPin8, HIGH);
digitalWrite(relayPin9, HIGH);
digitalWrite(relayPin10, HIGH);
digitalWrite(relayPin11, HIGH);
digitalWrite(relayPin12, HIGH);
digitalWrite(relayPin13, HIGH);
digitalWrite(relayPin14, HIGH);
digitalWrite(relayPin15, HIGH);
digitalWrite(relayPin16, HIGH);
digitalWrite(relayPin17, HIGH);
digitalWrite(relayPin18, HIGH);
```

```
digitalWrite(relayPin19, HIGH);
digitalWrite(relayPin20, HIGH);
digitalWrite(relayPin21, HIGH);
digitalWrite(relayPin22, HIGH);
digitalWrite(relayPin23, HIGH);
 servo1.writeMicroseconds(1000); // Send signal to ESC.
servo2.writeMicroseconds(1000);
servo3.writeMicroseconds(1000);
servo4.writeMicroseconds(1000); // Send signal to ESC.
servo5.writeMicroseconds(1000);
servo6.writeMicroseconds(1000);
}
void yawright(){
  motor1B();
  motor3F();
  motor4F();
  motor6B();
 servo1.writeMicroseconds(1400); // Send signal to ESC.
servo2.writeMicroseconds(1000);
servo3.writeMicroseconds(1400);
servo4.writeMicroseconds(1400); // Send signal to ESC.
servo5.writeMicroseconds(1000);
servo6.writeMicroseconds(1400);
 delay(3000);
}
void yawleft(){
  motor1F();
  motor3B();
  motor4B();
  motor6B();
  servo1.writeMicroseconds(1400); // Send signal to ESC.
servo2.writeMicroseconds(1000);
```

```
servo3.writeMicroseconds(1400);
servo4.writeMicroseconds(1400); // Send signal to ESC.
servo5.writeMicroseconds(1000);
servo6.writeMicroseconds(1400);
delay(3000);
}
void pitchup(){
  delay(500);
  motor2F();
  motor3B();
  motor5F();
  motor6B();
servo1.writeMicroseconds(1000); // Send signal to ESC.
servo2.writeMicroseconds(1400);
servo3.writeMicroseconds(1400);
servo4.writeMicroseconds(1000); // Send signal to ESC.
servo5.writeMicroseconds(1400);
servo6.writeMicroseconds(1400);
delay(3000);
}
void pitchdown(){
  delay(500);
  motor2B();
  motor3F();
  motor5B();
  motor6F();
servo1.writeMicroseconds(1000); // Send signal to ESC.
servo2.writeMicroseconds(1400);
servo3.writeMicroseconds(1400);
servo4.writeMicroseconds(1000); // Send signal to ESC.
servo5.writeMicroseconds(1400);
servo6.writeMicroseconds(1400);
```

```
delay(3000);
}
void rollright(){
  delay(500);
  motor2F();
  motor5B();
  servo1.writeMicroseconds(1000); // Send signal to ESC.
servo2.writeMicroseconds(1400);
servo3.writeMicroseconds(1000);
servo4.writeMicroseconds(1000); // Send signal to ESC.
servo5.writeMicroseconds(1400);
servo6.writeMicroseconds(1000);
delay(3000);
}
void rollleft(){
  delay(500);
  motor2B();
  motor5F();
 servo1.writeMicroseconds(1000); // Send signal to ESC.
servo2.writeMicroseconds(1400);
servo3.writeMicroseconds(1000);
servo4.writeMicroseconds(1000); // Send signal to ESC.
servo5.writeMicroseconds(1400);
servo6.writeMicroseconds(1000);
delay(3000);
}
```