Flying Penguins Group 3

Autonomous Underwater Vehicle
Final Design Report

Version 1.0

Revision History

Date	Version	Description	Author
4/13/2015	1.0	Initiate the document	DV
4/14/2015	2.0	Begin to populate the skeleton	CS
4/14/2015	3.0	Take control of the Document	DV
4/14/2015	4.0	Hazards and Cautions Manual	QD
4/14/2015	5.0	Budget	SD
4/14/2015	6.0	Introduction	SD
4/14/2015	7.0	System Construction and Assembly	PM
4/14/2015	8.0	Systems Engineering	CS
4/14/2015	9.0	User Manual	DV,SS
4/15/2015	10.0	Engineering Design and Calculations	PM
4/15/2015	11.0	Data Analysis	SD
4/15/2015	12.0	Development Section 5	CS PM QD DV
4/15/2015	13.0	Editing and Filling in Missing Items	CS

Abstract

In the 2014-2015 academic year, 6 ocean engineering undergraduates pursued a systems engineering endeavor. To design and produce an autonomous underwater vehicle capable of traveling in calm sea states at a user programed depth in up to 5 selectable directions to perform a box pattern.

As the topic of a system control and design project, the Flying Penguins are developing an Autonomous Underwater Vehicle (AUV). This project was chosen after careful debate of the Flying Penguins goals, skills, and ambitions. Preliminary designs were developed after a series of hierarchal and subsystem analysis. The criteria for analysis are the system level requirements, operational requirements, budgetary and manufacturing constraints. A detailed functional analysis revealed the various systems and sub systems that will need to be developed and how they have to be integrated. The team qualified three AUV control system designs as candidates for customer testing. Upon in-depth trade off analysis of design constraints and operating requirements, a unanimous decision was reached to pursue the concept of a vehicle with a cylindrical body, a single fixed thruster and an actuated rudder and stern plane surface for propulsion and navigation control. Subsystem integration analysis provided logical choices for AUV navigation and decision making techniques. Using an Arduino modified C interface an IMU and compass the Flying Penguins plan to measure and adjust the depth through thruster and stern plane surface control. The following document presents the preliminary design, decision methods for design development, and a plan of action for design fabrication and implementation. The six members of the Flying Penguins each received administrative and technical duties detailed in the management plan section. Through the course of several meetings with the advisory committee we have specified our deliverables and a timeline in which they will be assessed.

Table of Contents

Table of Tables

Table of Figures

Final Design Report

1. Introduction

The purpose of the senior design project is to build an affordable autonomous underwater vehicle (AUV). This is important because the wide spread applications both known and unknown for AUVs is limited by the high cost attached to purchasing AUVs. This project shall show that entry-level engineers can produce a functioning AUV with a budget of $2000. The AUV will be able to use user defined tasks and constraints to traverse an underwater environment. It will be created using both off the shelf and a few in house fabricated parts. The main objectives and goals of this project are to meet all requirements provided by the customer, meet all constraints defined by practicality, have a fully functioning electrical and mechanical system, working software, and to meet personal objectives. This project consists of six team members: Dietrich Vogel, Quintin Du Plessis, Steven Serbun, Stacey Darin, Chris Sullivan, and Pedro Muslera. Each member has individual tasks that support the goal of the Flying Penguins, which is: “to have a project that satisfies the customer requirements in a way we expect it to”.

1.1 Background and Needs Analysis

The background for AUVs and the needs analysis for this project are discussed in sections 1.1.1 and 1.1.2 below. The background includes a brief history of AUVs.

1.1.1 Background

An AUV is an autonomous underwater vehicle which is characterized by new technology, various types of controllers, and an array of sensors [1]. The advancement of technology has led to improvements in AUV’s. Figure 1 below shows a picture of an AUV called Remus-100 manufactured by Kongsberg [2].

Figure 1: Remus

Figure 2: Picture of an AUV, Remus-100 [2]

The first AUV started being developed in the 1960’s and was used to gather data in the sea [1]. From 1970- 1980, AUV’s started being developed by the University of Washington, University of New Hampshire’s Marine Systems Engineering Laboratory, and by the Russian Academy of Sciences. During the 1980’s-1990’s the development of software and computers helped to make the vision of AUV’s come true. From 1990-2000, all the technological advancements made it possible to fully develop an AUV. Instead of AUV’s being concepts, they lead to the first generation of operational systems. From 2000-2010, AUV’s became commercial products [3]. There are many needs for an AUV such as research and commercial use. For the purpose of research, AUVs can be used to gather data in the ocean or lakes. The AUV can collect data using different types of sensors. The type of sensors equipped on an AUV depends on the type of research that needs to be done. For example, AUVs aid in the exploration of the deepest parts of the ocean because the depth of the ocean makes it difficult to travel to. An AUV allows research to the deep ocean to be easily obtainable. This is because AUVs are autonomous and allows for users to decide where and at what depth the research will take place, and the AUV will perform the mission, collect data, and then resurface. AUVs are also beneficial to collecting data where sheets of ice lay since these are also difficult areas to reach [1]. Gathering this data allows for advancements in research and can also be used educationally.

Commercial AUV's are used for various applications. Some commercial AUVs are used to survey the ground at the bottom of the ocean. These AUV's scan the bottom of the sea floor and make a detailed map. These maps can then be used to determine the best location for building a subsea infrastructure [3]. Using an AUV is the most cost effective way to get this type of data. "For example, the oil company Shell has estimated that it can save up to US $100 million just by using AUVs during a period of five years [1]". These reasons are why the use of AUVs has increased over the past few years.

Building an AUV with a budget of $2000 will allow for further advancements in these application areas. This project will show that AUVs can be more marketable due to the low budget costs, and will allow for more users to obtain an AUV.

1.2 Problem Definition

The customer has requested the development of an AUV, which poses unique engineering challenges. The vehicle has to be designed, fabricated and built within a short time frame. Another significant challenge is the considerable low budget that has to be adhered to. In addition, a specific set of operational requirements has to be met. Design concepts have to be compared in a trade study and weighted by various criteria.

The objective of this project is to design and build an AUV with a budget of $2000 that meets all customer requirements. The design and build of this AUV needs to be completed by mid-April. The group working on this project has various backgrounds which help adhere to this project. For example, the group has taken previous engineering classes or participated in any other form of extracurricular activities. As previously discussed, an affordable AUV is not available on the market. This project will show that affordable AUVs can accessible to the general population. The design phase of this project took place from August 2014 through December 2014 and the build/testing phase took place from January 2015 through April 2015. This project was achieved through the help of FAU's faculty members and facilities. This being help from faculty in the areas of electronics, mechanics, and software. The facilities available are the machine shop, electronics shop, the Oceaneer (R/V), and SeaTech campus.

Three missions need to be executed and achieved for this project. For underwater motion, the AUV was designed and fabricated to be deployed by two people. Once in the water and user defined routines were established the data was logged. The development of a Functional Analysis and Decomposition discusses data logging and user defined routines in more detail. Along with user defined routines, the vehicle's position relative to the ocean has to be controlled by sensory input and feedback systems, for more detail on sensory input and feedback system consult the modern Arduino code. During the first mission, the AUV was required to dive to an operational depth of 10m (error < 20% of vehicle depth) in calm seas. The second mission, The AUV was required to adjust its position towards a user defined heading which was to be maintained for at least 5 minutes (error < 5°). During the third mission, the vehicle was required to go down to a depth of at least 10 meters, make a right turn of 90 degrees, and maintain this new heading for four minutes. At the 2nd waypoint the vehicle has to make another 90-degree right turn and proceed for another 4 minutes. Then the AUV makes a final 90 degree right turn and proceeds for four minutes. After a completed box pattern the vehicle is required to return to the surface and notify the operator of its location. Tito was then retrieved and the logged data transferred to a computer for analysis.

1.3 Project Objectives and Goal

The goal of this project is for a group of engineering students in their senior year to design and build an AUV. The objectives for this project are listed below:

- To design and build an AUV with a budget of $2000
- To design and build an AUV beginning in August and ending in April
- To ensure that all customer requirements are adhered the vehicle must be:
  - Two man deployable
  - Rated to a pressure of 50 meters
  - Have a maximum speed of 1.5 m/s
  - Able to keep depth for 5 minutes
  - Able to keep a heading for 5 minutes
  - Able to make a box pattern

1.4 Organization Chart of the Participants

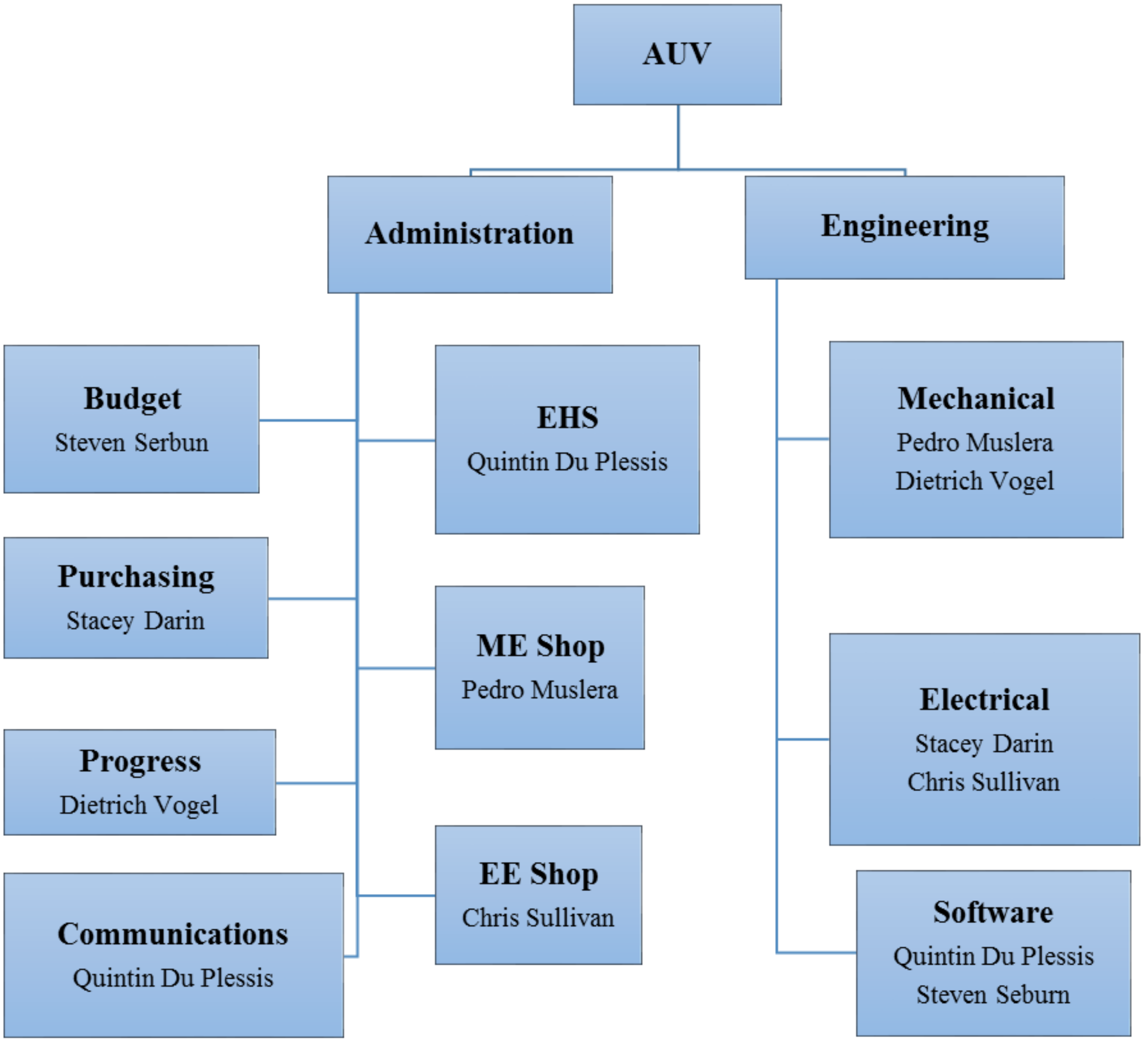


Figure 3: Group Tasks

1.5 Document Terminology and Acronyms

AUV - Autonomous Underwater Vehicle
IMU - Inertial Measurement Unit
GPS - Global Positioning System
FAU – Florida Atlantic University

ADCP – Acoustic Doppler Current Profiler

CTD – Conductivity Temperature Depth

FSM- Finite State Machine

DOF- Degrees of Freedom

ROV- Remotely Operated Vehicle

PCB- Printed Circuit Board

1.6 References

A. G. G. B. &. I. F. C. Ceron, "Development of Autonomous Underwater Vehicles; A brief overview," Mälardalen University School of Innovation, Design and Engineering, Västerås, Sweden.

K. Maritime, "Kongsberg Maritime," 22 September 2014. [Online]. Available: http://www.km.kongsberg.com/ks/web/nokbg0237.nsf/AllWeb/4C9D017AEEAA7AC3C12574B9002BB078.

R. D. Blidberg, "The Development of Autonomous Underwater Vehicles (AUV);," Autonomous Undersea Systems Institue, Lee, New Hampshire.

T. Prestero, "Verification of a Six-Degree of Freedom Simulation Model for the REMUS Autonomous Underwater Vehicle," MIT and WHOI, 2001.

"Appendix A Geometry for Aerodynamicists".

S. E. Bullard, "National Physical Laboratory," Kaye&Laby, [Online]. Available: http://www.kayelaby.npl.co.uk/general_physics/2_7/2_7_9.html. [Accessed 5 November 2014].

"NACA 0012 AIRFOILS (n0012-il)," Airfoil Tools, 2014. [Online]. Available: http://airfoiltools.com/airfoil/details?airfoil=n0012-il#polars. [Accessed 2 November 2014].

H. Y. &. M. Drela, "XFOIL 6.9 User Primer," 30 November 2001. [Online]. Available: http://web.mit.edu/aeroutil_v1.0/xfoil_doc.txt. [Accessed 1 November 2014].

"AA Portable Power Corp," [Online]. Available: http://www.batteryspace.com/smartcharger15afor222vli-ionpolymerrechargeablebatterypack.aspx. [Accessed 11 November 2014].

2. Systems Engineering

We used a Systems Engineering approach to develop Tito. “Systems engineering is an interdisciplinary approach and means to enable the realization of successful systems. It focuses on defining customer needs and required functionality early in the development cycle, documenting requirements, then proceeding with design synthesis and system validation while considering the complete problem.” [4] We first identified and defined the goal, which can be seen in Section 1.2. We then came up with alternative designs, and performed trade studies on system components. Then we developed Tito as he exists today with supporting documentation. These can be referenced in our critical design report

2.1 Concept of Operations

Tito is an Autonomous Underwater vehicle. The pressure vessel is rated up to 50 meters, and should not exceed this depth. The vehicle is capable of performing pre-programmed missions. These missions can be created in an Arduino C file. The missions can be sent to Tito via Xbee radio. Tito will then begin his mission after establishing a GPS fix. He will then use the depth sensor to control depth as defined by the mission. He will use the 3 axis compass to control heading as also defined by the mission. The batteries have the capacity for an hour of fully operational run time. Tito also has video recording capabilities via a Go Pro camera mount in the thruster shroud. He will return to the surface after the mission is completed, and will then communicate via Xbee his GPS location. The research vessel will maneuver to the broadcasted GPS signal and an operator will use a gaff, noose, or remove it by hand using the swim step.

2.2 Functional Analysis and Decomposition

The functional analysis is an in depth description of components and how they integrate to make subsystems and how those subsystems are connected to preform system tasks. What follows is a description of vehicle operation through a perspective of system integration.

2.2.1 High Level

The AUV functional analysis can be broken down into three top level functions: preparation, operation and recovery. Preparation will be the first function making sure the vehicle is ready before deployment. The AUV will then perform its mission during operation. We will then recover and de-rig the AUV, after it finishes operation, in recovery.

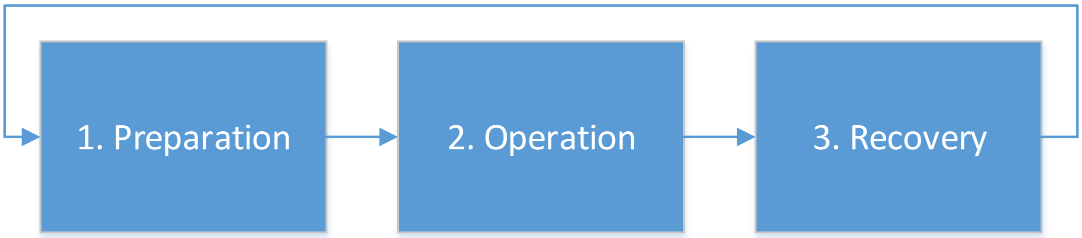


Figure 4:High Level FA

2.2.2 Preparation

The Preparation stage will be integral to the success of the goals set for the AUV. We will make sure that everything is in working order. It is essential that the pressure vessel be sound before deploying the vehicle. We will also set the parameters for the mission during this stage, this will include planned depth, headings, and time at each heading. Proper preparation is key to success for anything, as is the case here.

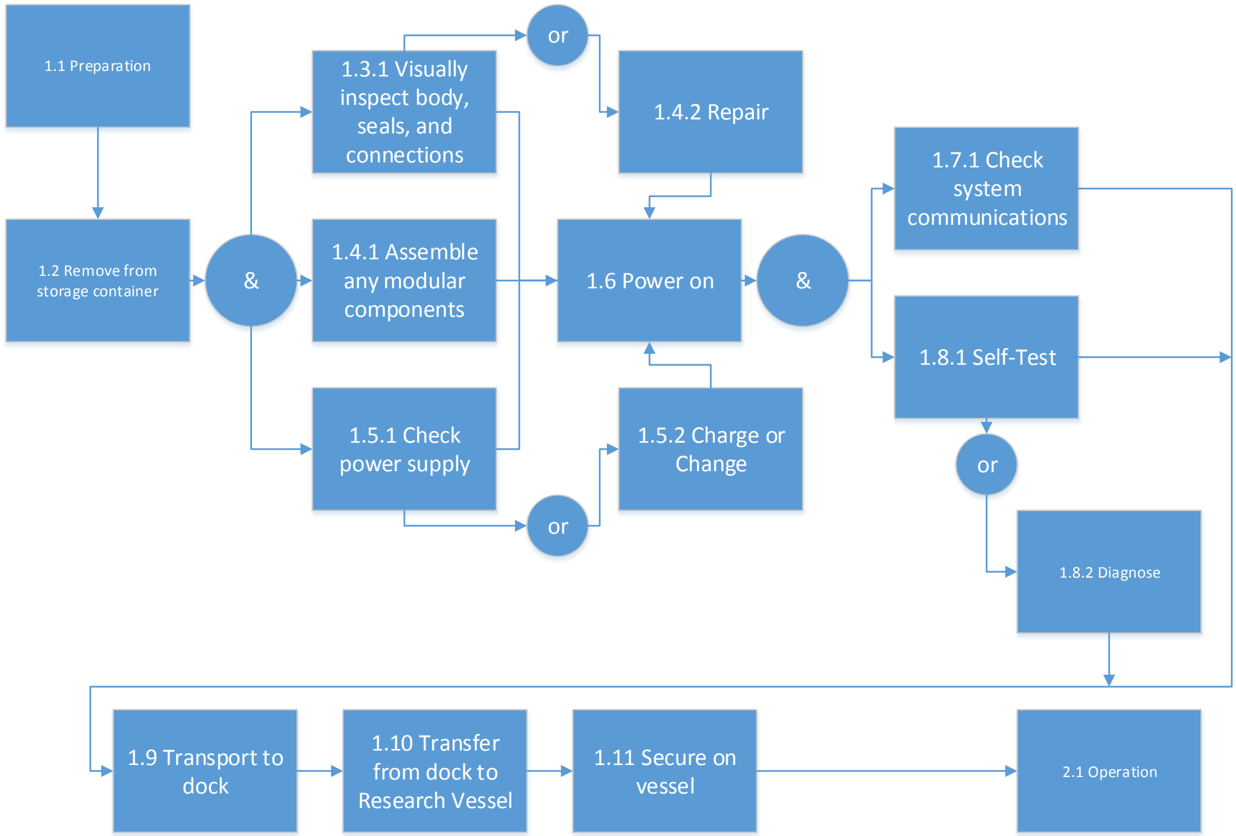


Figure 5:Preperation

2.2.3 Operation

Operation is from when the vehicle is launched until it reaches the end of its mission. This is a critical time for the AUV. We will have no control over it, once it is in the water. We will launch the AUV from the R/V, a conductivity sensor will then turn on the system. The AUV will initialize once in the water. It will be given a time delay of several minutes. This gives the AUV time to float away from the R/V, and also allows the sensors time to compensate any bias. The AUV will then begin its programmed mission. It will first descend to 10m, then it will take a user programmed heading. The AUV will traverse this heading for four minutes, it will then begin a right turn. It will repeat the turning process three times. The AUV will sound an acoustic ping and illuminate an LED, in case of a malfunction. This will aid in a diver recovery of the AUV. The AUV will begin its ascent after the box pattern is complete.

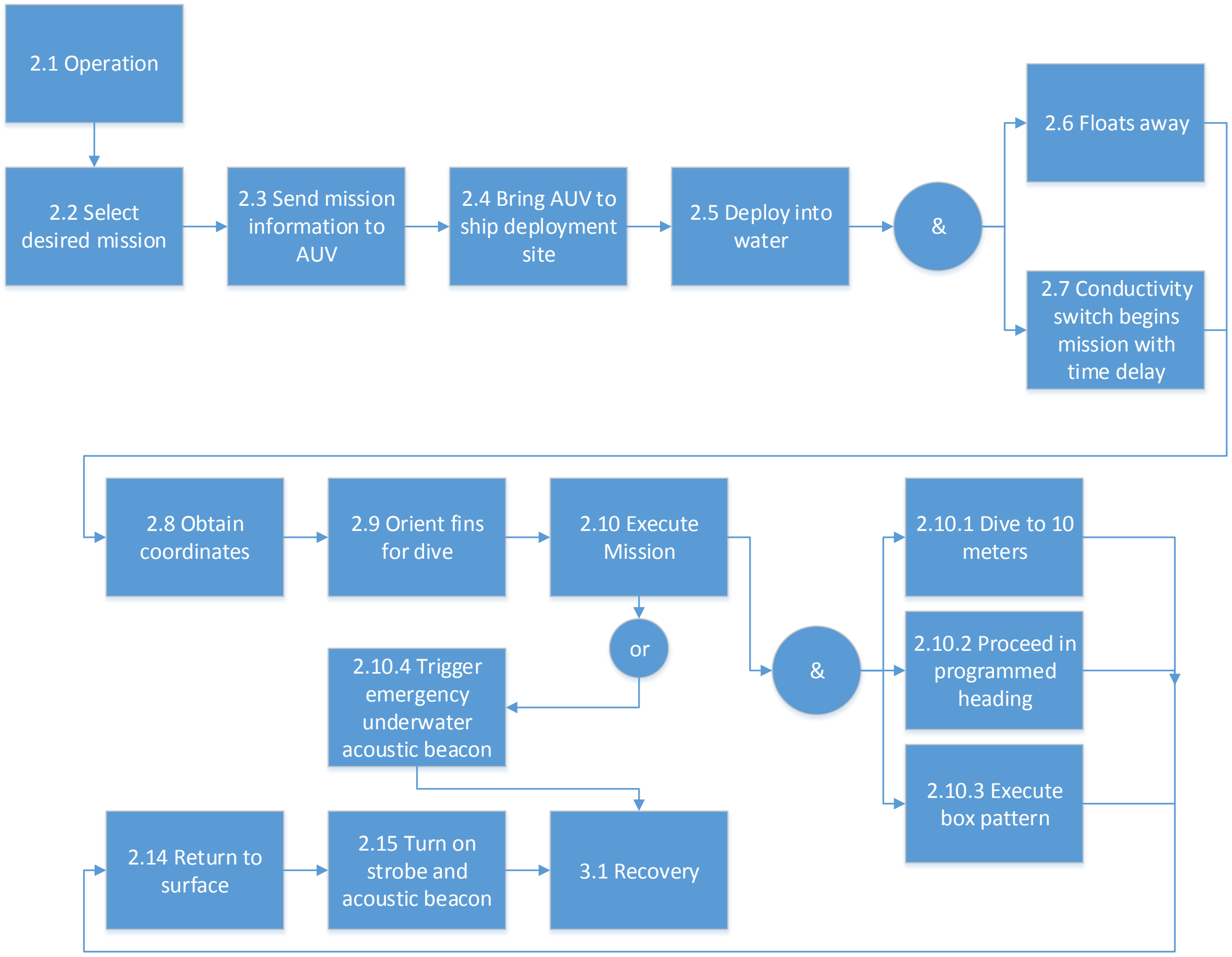


Figure 6: Operation

2.2.4 Recovery

Recoverability of the AUV is necessary after it has completed its mission. We can trace the AUV from an audio and/or visual beacon in addition to passive exterior coloring. The R/V will then proceed to the location of the AUV. The AUV will then be hooked using a pole and the D ring on the hull of the AUV and brought aboard the R/V. A scientific diver party may attempt to locate the AUV if the system malfunctions and does not return to the surface. The emergency acoustic ping will hone in on the search area. The data will be extracted from the AUV once on board. The AUV can then be cleaned and stored in its proper container. The post-processing of data can begin once on shore. The AUV can also be adjusted and repaired if needed.

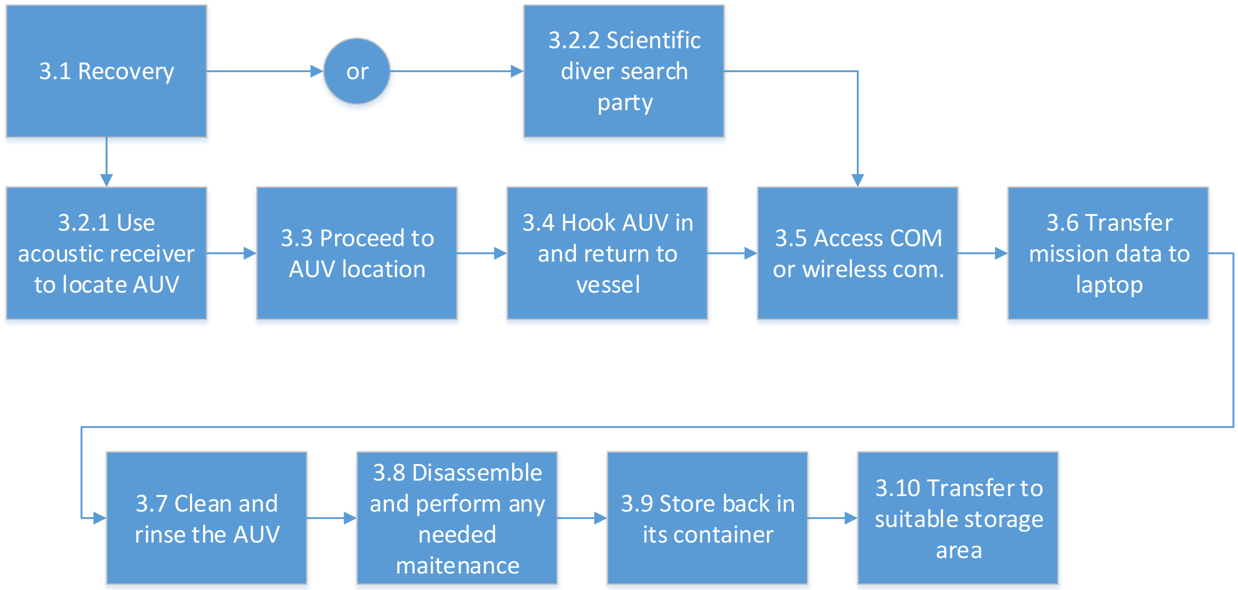


Figure 7:Recovery

2.3 Detailed Final Design Description

This section details the design of our vehicle in all aspects. The many systems of the vehicle are broken down here into sub systems and components. A map of the system and sub systems broken down is found in the physical architecture diagram in appendix. The schematics of the Motherboard and wiring diagrams can also be found in appendix

2.3.1 Overall System

The design of the AUV is a cylindrical body composed of two sections. The AUV has two sets of dynamic control surfaces in the rear. These control the navigation of the AUV. The wetted thruster is mounted on the back section of the AUV. The AUV has a half spherical nose cone for hydrodynamic efficiency. The larger front section houses the electrical components of the AUV. This includes the batteries, PCB, sensors, and communication mast. The communication mast houses antennas for the XBee and GPS. The smaller rear section houses the actuators for the AUV. This includes the Servos and their linkages, with the thruster mounted on the rear end cap of this section. The AUV is comprised of four main sub-systems. The sub-systems include: power, propulsion, navigation, control, and body. All four of them are inter-dependent on one another.

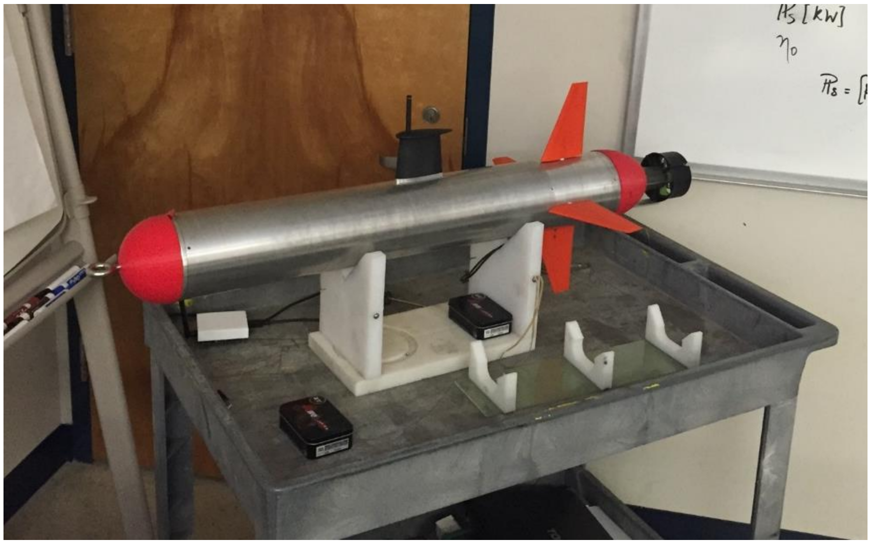


Figure 8: Tito

2.3.2 Power System

The power system of the AUV is integral to its operation. It provides and distributes to all sensors, actuators, and all other electrical components. The batteries or a shore power connection provide the power. These are then connected to the PCB, which regulates and distributes the power to all components.

2.3.2.1 Batteries

The AUV uses three battery packs. They contain six Nickel Metal Hydride (Ni-MH) battery cells, for a total of 18 cells. Each pack has a peak voltage of 8.5V and a working voltage of 7.2V. The three packs are connected in series and combine for a total working voltage of 21.6 V. Each battery pack is connected to the PCB using a Molex Mini-Fit Jr 4 pin connector. The batteries are mounted with PVC brackets on the underside of the PCB sled.

Figure 9: NiMH Battery Pack

2.3.2.2 Reed Relay

The vehicle will be powered on and off by Reed relay. The Reed relay is mounted on the interior of the front end cap. It uses a 2-Pin Molex Micro-Fit connector to connect to the PCB. It is a normally closed Reed switch. The vehicle is off when the magnet is in contact with the switch, and it is on when the magnet is out.

2.3.2.3 Voltage Regulators

The PCB has three voltage regulators on it. It has two Traco Power TSR-3-24150 DC/DC convertors. These take the battery voltage and shift it down 7.4 V for the two servo-motors for the control surfaces. These can handle an output current up to 3 A. There is also a Traco Power TSRN-1-2450SM DC/DC switching regulator. It shifts the battery voltage down to 7.4 V for the Arduino Due. It has a maximum output current of 1 A. All three of these chips are surface mounted on the PCB.

2.3.2.4 Motor Driver

The Seabotix BTD150 thruster is powered by a SyRen 10 regenerative motor driver by Dimension Engineering. The SyRen 10 is mounted on the aft part of the electronics sled. The motor driver can supply up to 8 A with a 24 V supply.

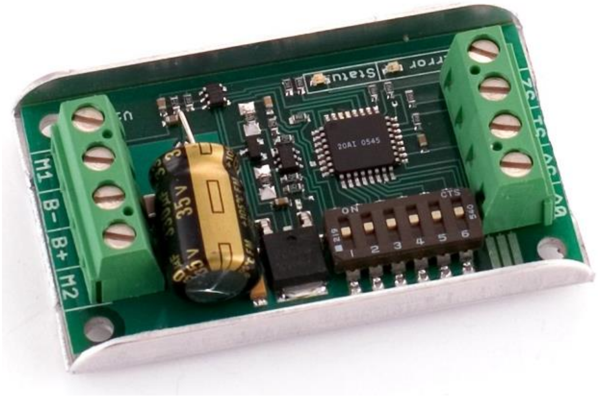


Figure 10: SyRen 10 Motor Driver

2.3.2.5 SEACON Connector

The SEACON connector allows to charge the batteries, and run the system off shore power while the vessel is sealed and closed. The connector used is a SEA CON All-Wet Series FAW-S-BC-R/A. The connector has the FAWM-8S-MP contact configuration. It is a female flat bulkhead connector. The connector is mounted on the front endcap of the AUV. The FAW-P-MD dummy connector is put on when it is not being charged or using shore power. We then use a SEA CON All-Wet Series FAW-P-MP to charge and provide external power for the AUV.

2.3.3 Actuation

The Actuation System are the mechanical devices that move and control the vehicle in the water. The thruster controls the forward movement of the vehicle. The two sets of control surfaces are useless without forward motion. The stern planes control the vehicles depth, and the rudders control the vehicles heading. The three systems work together to physically navigate the vehicle through its mission.

2.3.3.1 Control Surfaces

The two sets of control surfaces are mounted in the rear Servo-section of the vehicle. Two Servo-Motors are mounted to the pressure vessel. These are then linked to two shafts. The shafts exit the pressure vessel through four dynamics seals. Four fins are then mounted to each side of the shafts to make up the control surfaces.

Figure 11: Servo Assembly

2.3.3.1.1 Servo-Motors

There are two servo-motors that move the two pairs of control surfaces. They servos are PowerHD High Troque, High Voltage Digital Servos model 1218TH. The servos have an operating voltage between 6 and 7.4 V. The servos were chosen because of their high output voltage, and also the titanium shielded gears.

2.3.3.1.2 Servo Mount

The servo mount is two pieces an L-bracket attached to the servo mount block that mounts onto the tail end cap. The L-Bracket houses the two servo motors. One is oriented facing up to control the rudders. The other is oriented facing the side at 90 degrees this one controls the stern planes. The L-Bracket screws into the servo mount block with three screws, and then the servo mount block Screws into the end cap with three screws as well.

Figure 12: Servo Mount Bracket

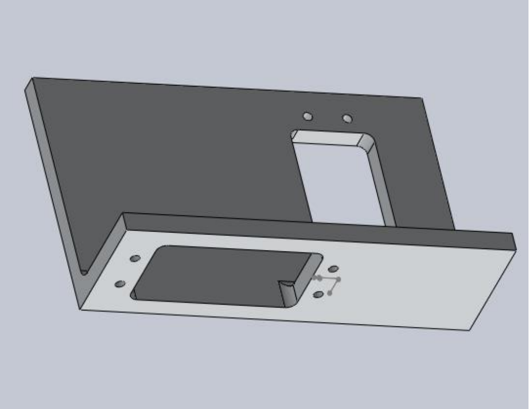


Figure 13: Servo Mount

2.3.3.1.3 Servo Linkages

The linkages are what connects the servo horns to the shafts arms of the control surface shafts. These are also made out aluminum. The linkage is attached to the servo horn with a screw. It also attaches to the shaft arm with a screw. The shaft linkage is held onto the shaft by a collar that is fastened to the shafts.

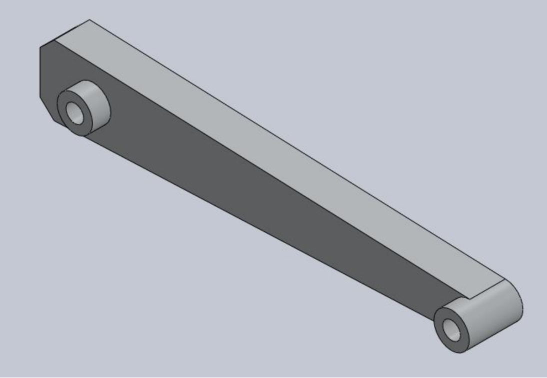


Figure 14: Servo Linkage

2.3.3.1.4 Shaft Arm

The shaft arm connects the servo linkage to the shaft. The holes on the linkage and shaft arm are aligned and then screwed together. The shaft arm slips over the shaft, and then tightens with a set screw. They are what rotates the shafts. They are also machined out of aluminum.

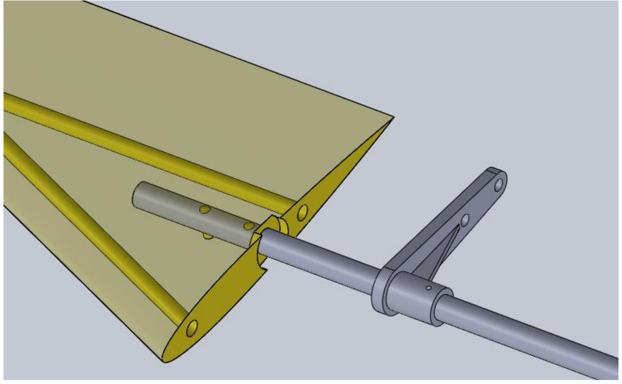


Figure 15: Servo Collar

2.3.3.1.5 Shaft Collars

The shaft collars are mounted on the shafts inside the pressure vessel next to the wall. They slip over the shaft when inserting it through the pressure vessel. These are mounted as close to the edge to prevent the shafts from moving at all. They are tightened with set screws.

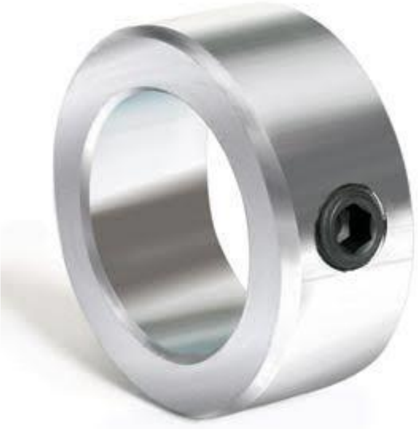


Figure 16: Shaft Collar

2.3.3.1.6 Shafts

The control surface shafts are made out of stainless steel. They are eight inches and length. They connect to servo linkages via the shaft arms. They then exit the pressure vessel, through four holes in the pressure vessel. The shaft collars mount on the shafts just before they exit the pressure vessel, and prevent it from moving. The holes are sealed by four Veri-Seals, and bushings mounted on top. The four fins are then mounted onto each side of the two shafts. The shafts are eight inches in length, and a quarter inch in diameter. The shafts have two screw holes on outside parts of the shafts, these are to screw the control surfaces down. This prevents them from moving around.

2.3.3.1.7 Dynamic Seals

The seals used for the control surface shafts were spring loaded 1/4in PTFE ring seals. They have a maximum pressure rating of 250psi. This equates to about 170m well below the vehicles operating depth. The seals lay on top of a 0.38in diameter counter sink in the shaft holes. A bushing is then placed above the seal over the shaft. This prevents any unwanted movement of the seal.

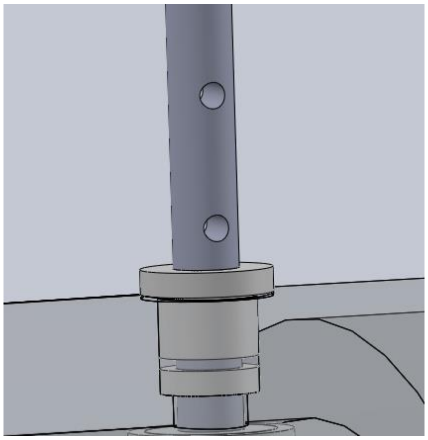


Figure 17: Dynamic Seal

2.3.3.1.8 Control Surfaces

The four control surfaces are NACA 0012 foils. They are 3-D printed out of NinjaFlex proprietary material.

2.3.3.2 Propulsion

The propulsion system moves the vehicle, and subsequently allows the control surfaces to work. The vehicle is powered by a SeaBotix BTD150 thruster. The thruster is controlled by the Dimension Engineering SyRen 10 motor driver which was described in Power System section 2.3.2.4.

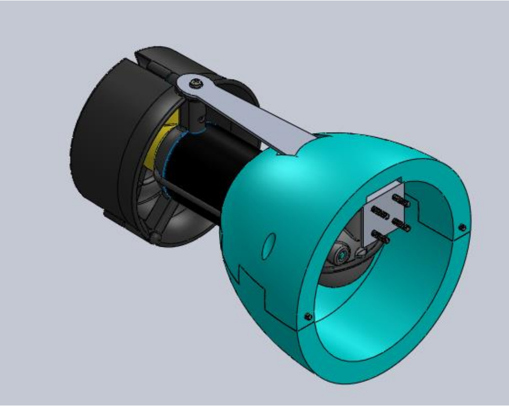


Figure 18: Thruster Assembly

2.3.3.2.1 SeaBotix BTD150 Thruster

The SeaBotix thruster is mounted on the aft end of the vehicle. It is mounted via an L-Bracket that mounts to the rear end cap of the vehicle. It is rated up to a depth of up to 150m. It operates at voltage of 19.1 V +/-10%. It can handle up to 4.25 A of continuous current. It will put out a maximum of 4.85 Ft/Lbs of continuous thrust.

2.3.3.2.2 Thruster Mounting Bracket

The thruster mounting bracket connects the thruster to the rear end cap. The thruster mounting screws are vertical. The bracket allows the thruster to line up directly in the center of the vehicle. This prevents a moment that would cause the vehicle to turn. The brackets is mounted onto the end cap with four horizontal screws. Then therer are two screws that go into the thruster mounts.

2.3.3.2.3 Thruster Shroud

The thruster shroud serves primarily to enhance the hydrodynamics of the vehicle. It also prevents any entanglement of the thruster or mounting bracket. The space inside the shroud can also be used to store foam to level the vehicle buoyancy. The shroud contains two pieces that connect to one another, and then are fastened using set screws. The thruster mounting bracket goes through a slit in the bottom shroud.

2.3.4 Control System

The control system is the brain of the vehicle. It senses and processes the internal and external environment of the vehicle. It then tells the actuation system of the vehicle what to do based upon the information it receives. The sensors, motherboard, and actuators are all controlled by the Arduino Due. This system relies on the power system to operate.

2.3.4.1 Arduino Due Microcontroller

The operations of the vehicle are processed by an Arduino Due microcontroller. The Due is based on an Atmel SAM3X8E ARM Cortex-M3 CPU. The Due contains 54 digital input/output pins, some of these have pulse with modulation (PWM) capabilities. It also houses 12 analog inputs, and 4 serial inputs. The Due has an operating voltage of 3.3V. The input voltage for the Due is between 6 and 16 Volts. The Traco Power TSRN-1-2450SM DC/DC switching regulator provides the Due with 7.4 V, as stated in section 2.3.2.3. The Due is mounted onto AUV Motherboard PCB with headers, and can be removed or replaced.

2.3.4.2 AUV Motherboard

The AUV motherboard was specifically designed for Tito Tuxedo. The board is mounted on standoffs on the upper side of the electronics sled. The board serves three main purposes. The first is to regulate and distribute the power from the batteries. The second is to channel data from all of the vehicles sensors to the Due. The third is to send the commands to the actuators of the vehicle. The Motherboard contains several break out boards mounted on headers. These include: an Adafruit 10-DOF IMU, Adafruit Ultimate GPS Breakout Board, Adafruit uSD, and Adafruit Xbee radio. The Arduino Due is centrally located on the board.

The board has several connectors on both ends. The aft end of board has three Molex Mini-Fit JR Four Pin Right Angle receptacles. These take in the three battery voltages, and then combine them. This end of the board also has another Molex Mini-Fit JR Four Pin Right Angle receptacle for the SyRen 10 motor driver. There is also a Molex Mini-Fit JR Six Pin Right Angle receptacle, this connector is for the SeaCon Bulkhead connector listed in section 2.3.2.5. The last connector on this side a Molex Micro-Fit Six Pin Right Angle receptacle for the two servo motors power and signal.

There are five connectors on the bow end of the AUV Motherboard as well. The first is a Molex Micro-Fit Two Pin Right Angle Receptacle, this is for the Reed Relay in section 2.3.2.2. There are then two Molex Micro-Fit Four Pin Right Angle Receptacles. One is for the Honeywell Depth Sensor and the other is for the TCM-2 Compass. Then there are two more Molex Micro-Fit Two Pin Right Angle Receptacles for the two leak sensors.

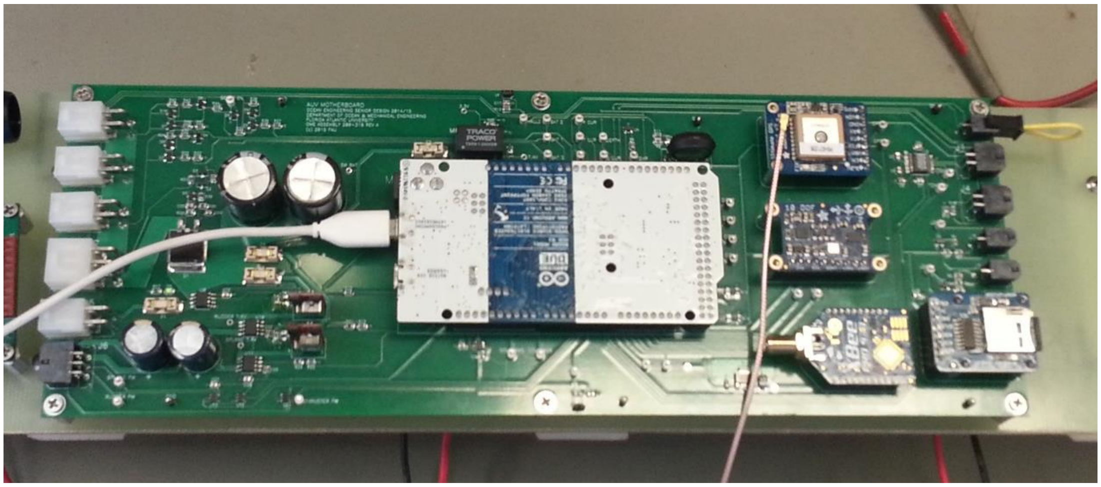


Figure 19: Assembled AUV Motherboard

2.3.4.3 Sensors

The vehicles sensors give vital health information, as well information about its surroundings and location. The vehicle uses them to navigate, and provide feedback. The vehicles sensors include: Honeywell PX2 pressure sensor, TCM-2 Compass, Adafruit 10-DOF IMU, Adafruit Ultimate GPS, two leak sensors, internal temperature sensor, and internal pressure sensor.

2.3.4.3.1 Honeywell PX2

The Honeywell PX2 Heavy Duty Pressure Transducers are built for industrial applications. The model number for the vehicle is PX2AS2XX100PAAAX. The S2 in the model number means it is 7/16in diameter with 20 UNF threading. The 100P means that the sensor is rated up 100PSI or about 68 m of seawater. The A means its reading is in absolute pressure. The last AA means that it reads ratiometric up to 5V, 10%Vs to 90%Vs. This model runs off a 5V supply voltage. The PX2 comes with an O-Ring on the end of the threading. The PX2 is just screwed into its threaded hole on the front end cap. It can easily be replaced if need be.

Figure 20: Honeywell PX2

2.3.4.3.2 TCM-2 Compass

The TCM-2 is a 3-D compass made by PNI Sensor Corporation. The compass runs off a 5V supply voltage, and uses RS232 serial communication. The compass is mounted on the front of the electronics sled, to keep it away from the noisy actuators. The compass is then connected to the PCB using a Molex Mini-Fit Right Angle 4-Pin connector. The signal is then sent through an RS232 Level Translator, this drops the signal from 5V down to 3.3V for the Due to process.

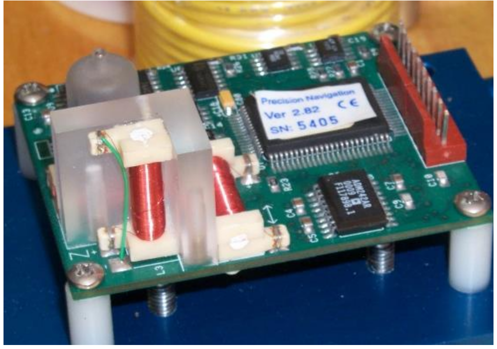


Figure 21: TCM-2 Compass

2.3.4.3.3 Adafruit 10-Dof IMU

The Adafruit 10-Dof IMU breakout board contains 11 sets of data. It records 3 axis accelerometer data, 3 axis gyroscopic, 3 axis magnetic compass, barometric pressure, and temperature. The board is mounted on the AUV Motherboard with standoffs, and is connected to one of the Due’s serial ports. It communicates via I2C. The breakout board runs off and communicates on 3V.

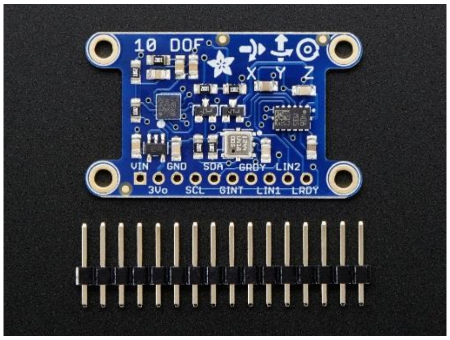


Figure 22: Adafruit IMU

2.3.4.3.4 Adafruit Ultimate GPS Breakout Board

The GPS breakout board also mounts on a socket on the AUV Motherboard. The breakout board is powered by the 3.3V supply of the Due. It also connects to one of the serial ports of the Due. The board has u.FL connector for an external antenna. A RP-SMA to u.FL Adapter cable is run into another RP-SMA cable. This cable can then mate with the connector for the GPS antenna mounted in the mast of the vehicle.

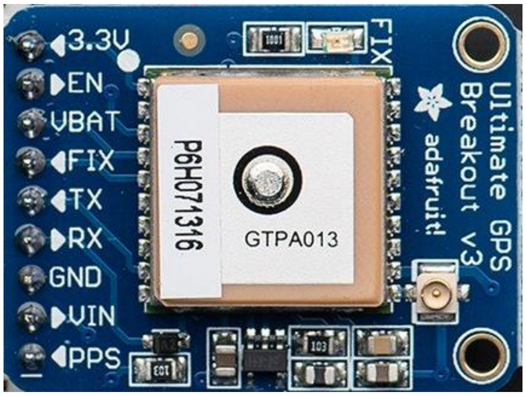


Figure 23: Adafruit GPS

2.3.4.3.5 uSD Breakout Board

The control system uses a uSD card to read and write memory. The board is mounted on a socket on the AUV Motherboard, and with four standoffs. It runs of the the Dues 3.3V supply. The break out board communicates with the Due via Serial Peripheral Interface (SPI). The uSD card is mounted in the small metal housing on the front of the breakout board.

2.3.4.3.6 XBee Radio

The XBee Pro S3B breakout board by DigiKey is used for wireless communication with the vehicle. The model number of the board is Module XBee Pro S3B 900HP RPSMA The breakout board is mounted on the AUV Motherboard with two sockets on each side. The radio runs off a 3.3V supply voltage. It operates on a 900MHz signal. It can communicate with a range of 28 miles with line of sight. It can communicate with data speeds up 200Kbps. An XBib with similar frequency is needed on the computer side to communicate with the vehicle. A coaxial cable connects directly to the breakout board. Then the other side mates with the connector of then RF antenna potted inside the communications mast. The vehicle can be communicated with in two ways. The first is through the programming port on the Due using a micro USB cord. This method requires the vehicle to be taken apart. The vehicle can also be communicated with its wireless Xbee radio.

Figure 24: XBee Radio

2.3.5 Body

The body of the vehicle is its physical structure. It houses all the other systems within its pressure vessel, and all other appendages are connected to it. It is comprised of four main sections: bow insert, electronics vessel, servo vessel, and rear end cap. These four components make one sound pressure vessel when assembled properly.

2.3.5.1 Bow Insert

The electronics sled is fastened to the front end cap of the vehicle. This comprises the bow insert of the vehicle. It also includes the nose cone, which screws into the front end cap. The electronics sled plug is also part of the insert it mounted on the opposite side of the front end cap.

2.3.5.1.1 Electronics Sled

The electronics sled houses the three battery packs on the underside of it. They are fastened to the sled by three battery mount brackets, and the battery plug. The AUV Motherboard is mounted to the top side of the sled by six standoffs. The TCM-2 compass is mounted on the sled directly in front of the AUV Motherboard. The SyRen 10 Motor Driver is mounted on the sled directly behind the AUV Motherboard. The electronics sled mounts to the front end cap with two screw. The battery plug slides into two notches on the rear end of the sled.

2.3.5.1.2 Front End Cap

The front end cap is machined out of aluminum. It can detach from the electronics sled. It has three threaded through holes in it. The lower starboard hole is for the SeaCon underwater bulkhead connector. The upper port hole is for the Honeywell PX2 depth sensor. The lower port hole is for creating a vacuum inside the pressure vessel. This is covered with a Teflon tape covered plug once the vacuum is created. It also has one partial hole for the nose cone’s eye hook to screw in. This is in the center of the cap. The groove in the upper center of the cap is for the magnet of the reed switch. There is a notch on the inside of the cap by the magnet grove, that house the reed switch. The end cap has two groves to seat two Parker 246 O-Ring seals, the first O-Ring is EPDM and the second is soft buna.

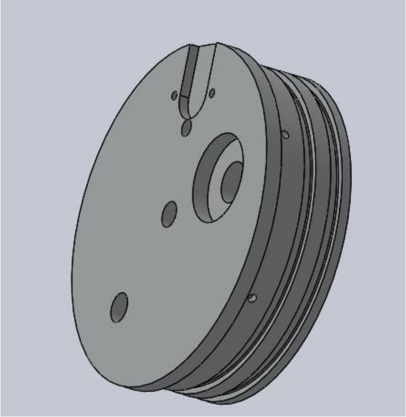


Figure 25: Front End Cap

2.3.5.1.3 Battery Plug

The battery plug mounts on the rear end of the electronics sled. It is machined out of acrylic. The plug has two grooves in it. These receive the two notches on the electronics sled to connect the two. The half that goes into the servo section has one Parker 242 EPDM O-Ring groves to separate the two compartments.

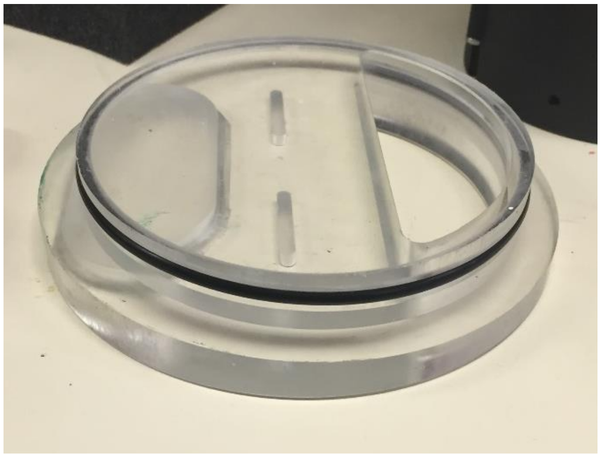


Figure 26: Battery Plug

2.3.5.2 Electronics Vessel

The electronics vessel is the tube that houses the electronics sled. It is made out of extruded aluminum with a 5in diameter. It has an inner diameter of 4.75in. The vessel has a total length 25.25 in. The front end cap seals the front end of the vessel. The servo section seals the rear end of the vessel, both are fastened by six sets screws each. The communications mast is mounted on the middle top section of the vessel.

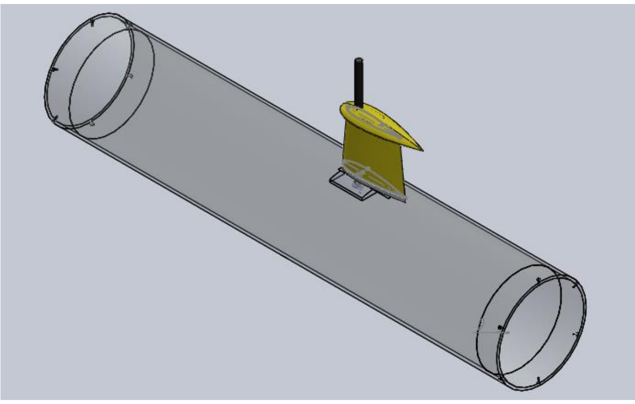


Figure 27: Electronics Vessel

2.3.5.2.1 Communications Mast

The communications mast is mounted on the top side of the electronics vessel. It contains both the GPS antenna and the Xbee antenna. The mast keeps the antennas from being submerged when the vehicle is on the surface. This ensures a better, stronger signal for them. The two coaxial cables go through a hollow threading on the base of the mount. They then enter the pressure vessel from this threading.

2.3.5.2.1.1 Potted Antenna Mast

The pressure vessel that houses the antennas is made out of 3m Scotchcast Flame Retardant Compund 2131. It was made from an injection molding. The mast is designed from three different NACA foils.

2.3.5.2.1.2 Mast Bottom Plate

The mast bottom plate is fabricated from aluminum. It has a threaded piece stainless steel tube that mounts in the countersink of the larger hole. The other larger hole is for the potting compound to leak out of the mold. The three small holes are for screws that enter into the mast. This gives it more stability. The notch in the middle of the two screw holes keeps the mast from spinning. The steel tube is then screwed into the tapped hole on top of the electronics vessel.

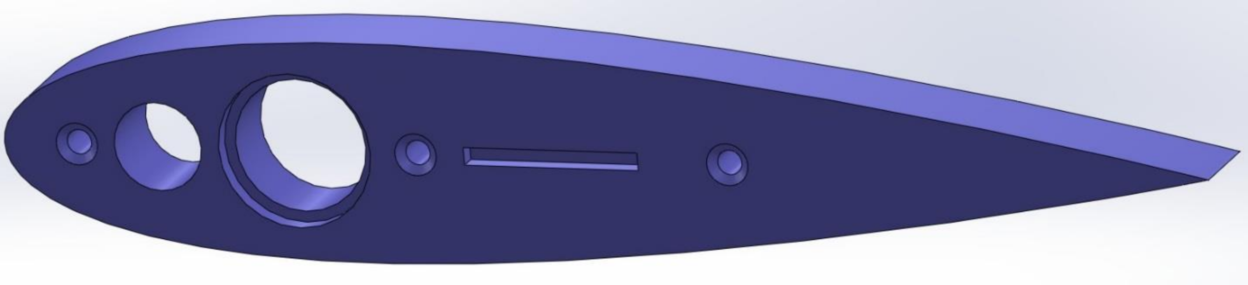


Figure 28: Mast Bottom Plate

2.3.5.2.1.3 Communication Mast Spacer

The spacer allows a flat surface for the nut to tighten onto the other end of the stainless steel tube.

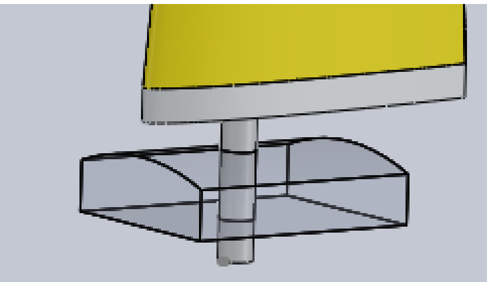


Figure 29: Mast Spacer

2.3.5.3 Servo Vessel

The servo vessel was machined out of a solid cylinder of aluminum. It has a total length of 6.75in, and varies in diameter throughout. The front end inserts into the rear of the electronics vessel. This end has two Parker 246 O-Ring seals, the first is EPDM and the second is soft buna. It fastens to the electronics vessel with six set screws. The rear end is sealed by the rear end cap. The vessel houses the entire servo assembly for the control surfaces described in section 2.3.3.1.

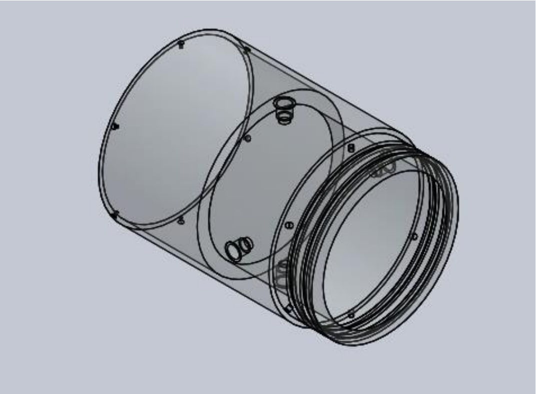


Figure 30: Servo Vessel

2.3.5.4 Rear End Cap

The rear end cap is also machined out of solid, extruded aluminum. The cap seals the rear end of the servo vessel. It has one through hole that the thruster cable is potted through. It also has four crew holes that are for the thruster L-Bracket as described in section 2.3.3.2. The end cap also has two Parker 246 O-Ring seals, the first EPDM and the second soft buna.

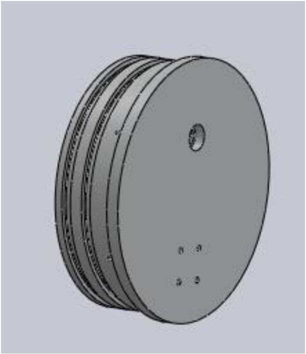


Figure 31: Rear End Cap

2.3.6 System Dependencies

Every system in the vehicle relies on one another in some type of way. This section describes what these dependencies are for systems, sub systems, and individual components.

2.3.6.1 Sub Systems

The diagram below explains the dependencies of the sub-systems explained above. Body is not listed here because it is all encompassing, all systems rely on it one way or another. The power system is the driver of control, propulsion, and navigation. These systems all need power to function. Propulsion and navigation also need the control system to operate. Control tells these systems what to do and when.

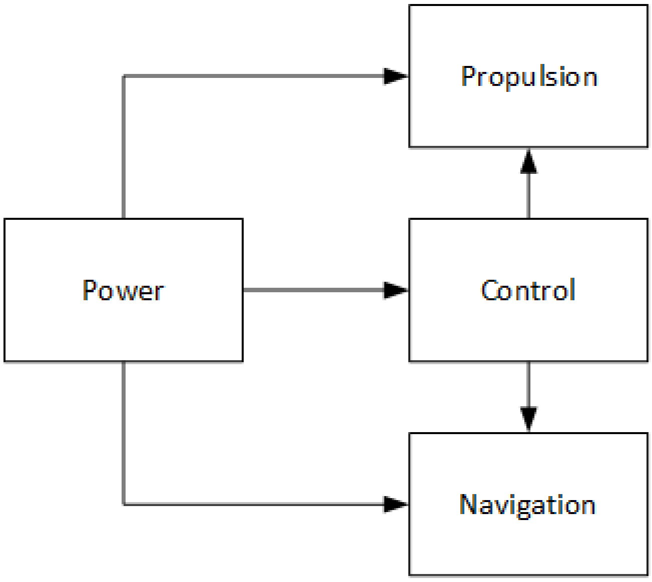


Figure 32: System Dependencies

2.3.6.2 Propulsion

The propulsion system is relatively simple. The batteries supply a voltage to the AUV Motherboard. The AUV Motherboard then supplies the SyRen 10 with power and a signal. The signal tells the SyRen 10 what voltage to supply the thruster. The SyRen 10 then sends the power to the SeaBotix thruster when commanded to.

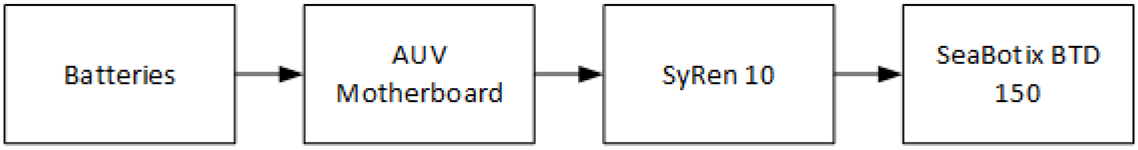


Figure 33: Propulsion System Dependecies

2.3.6.3 Navigation

The navigation system includes two sets of control surfaces the rudders and the stern planes. These systems rely on constant feedback from the sensors. They then can adjust their actuators according to this feedback and the desired set point of the mission.

2.3.6.3.1 Rudders

The rudders control the heading of the vehicle or yaw. The rudders rely on the servos and their linkages to move. The servo motors are supplied with power and a PWM signal from the AUV Motherboard. The AUV Motherboard is supplied with power from the batteries, and then distributes it properly. The signal comes from the Arduino Due on the AUV Motherboard. The Due takes in data from the TCM-2 3-axis compass, and then compares it to a set point for the mission. The Due then adjusts its signal to the servo accordingly to compensate for any difference in the two data points.

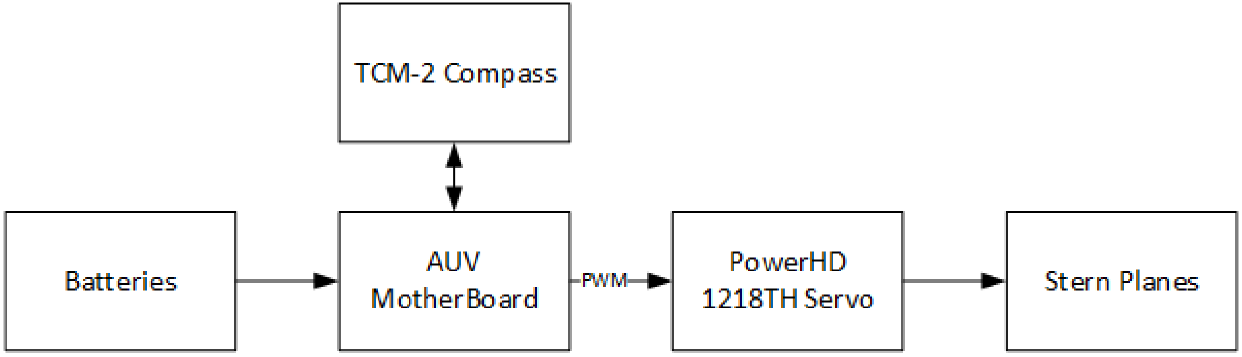


Figure 34: Rudder System Dependencies

2.3.6.3.2 Stern Planes

The stern planes operate in the same fashion as the rudder. The only exception being that it relies on feedback from the Honeywell PX2 Depth Sensor instead of the TCM-2.

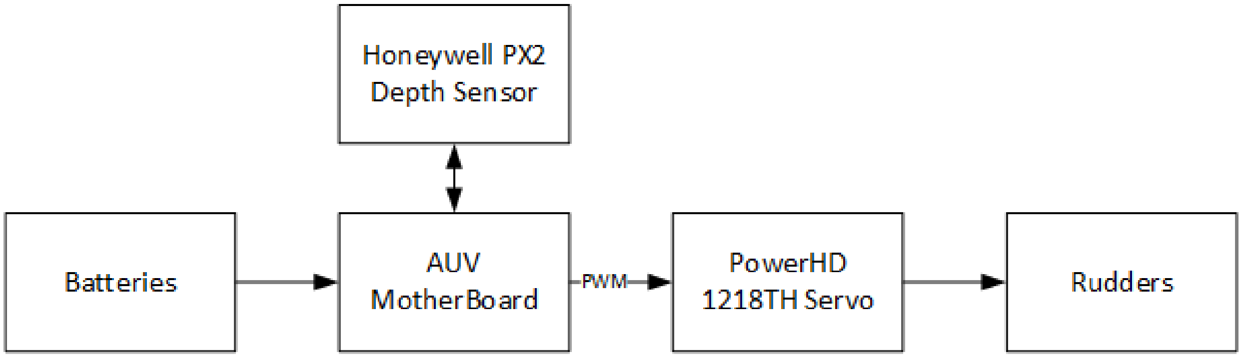


Figure 35: Stern Plane Dependecies

2.4 Requirements and Constraints

According to the customer contract the underwater vehicle must autonomously perform missions such as performing a commanded box pattern at 10 meters of depth in a calm sea state; success is defined by completing any of the following three user defined commands:

Float on the surface and then autonomously dive to a chosen depth (at least 10m ± 20% of vehicle depth) and travel at that depth for at least 5 minutes before returning to the surface for recovery
Float on the surface, autonomously descend to a chosen depth (at least 10m ± 20% of vehicle depth) and then travel in a chosen compass heading (± 5 degrees of intended heading) for at least 5 minutes before returning to the surface for recovery
Float on the surface, autonomously dive to a controlled depth (at least 10m ± 20% of vehicle depth) command a direction heading and follow that heading for 4 minutes. Then perform a starboard turn until the vehicle follows a heading 90 degrees from the initial heading. Hold that heading for 4 minutes. After 4 minutes take another right turn to head 180 degrees from the initial heading. Continue straight then after 4 minutes execute another right turn to travel 270 degrees from the initial heading for 4 minutes. After the 4th count of 4 minutes the vehicle will surface for recovery

2.4.1 Mission Requirements:

Types of requirements depend on where the requirement stems from. A customer requirement is an expectation or need in terms of mission objective, environment, constraint or measure of effectiveness. A functional requirement defines what the system should do. A performance requirement is based off of how well the system must complete the mission and a physical requirement is composed of physical limitations.
Descend to a depth of at least 10 m and travel for at least 5 minutes (error 20% of the vehicle depth)
Progress at a user defined depth of at least 10 meters in a user defined compass heading (error 20% of the vehicle depth and 5% of compass heading)
Descend to a user defined depth at least 10 meters, and command a box pattern of 4 user defined waypoints each point at least 4 minutes apart (error 20% of the vehicle depth and 5% of compass heading)
Instantaneous position estimation

Recoverability for more testing

2.4.2 Mission Constraints

Constraints directly depend on the environment, equipment, and capital available for AUV development and testing.

2.4.2.1 Operational Constraints:

- Operates in a sea state 1
- Tests in the open Atlantic ocean limited by the range of the Oceaneer
- Deployable from the Oceaneer
- No environmental considerations
- No mooring or a guide line (descend and ascend on its own)
- 2 man deployable
- Diverless operation
- Crew will not operate vehicle in water depths exceeding 50m

2.4.2.2 Project Constraints:

- Finish design by Nov 13
- Finish production and testing by April 20
- Use less than $2000 to develop a prototype
- Pressure vessel must fit in 6’X18”X6” pressure vessel testing chamber
- Pressure rated to at least 50m of hydrostatic pressure
- Must consist of commercial off the shelf parts or components simple enough to manufacture using the available FAU machine equipment
- Conflicting academic schedules

2.4.2.3 Mission Assumptions:

- Vehicle will operate in a seaway free of pollution and obstacles
- Temperate atmospheric environmental climates
- Operate in sea water between 18-22 degrees Celsius
- Operate in the water for no more than 1 hour at a time
- Operate at day time hours with the sun out
- On board crew proficient in AUV launch and recovery

2.4.3 AUV System Components

The requirements and constraints of the vehicles individual components.

2.4.3.1 External power and battery charging system

Requirements:

- Bulkhead external connector
- Batteries to charge
- Outside power
- Charging health monitor
- Intelligence to determine charging and discharging

Constraints:

- Not to exceed certain voltages or currents
- Size constraints
- Pin connector constraints

2.4.3.2 Battery Packs

Requirements:
- Must provide enough power for 1 hour
- Must have enough voltage

Constraints:
- Must be able to fit inside pressure vessel
- Must be below midline of cylinder for Cb and Cg seperation

2.4.3.3 Voltage Regulators

Requirements:
- Must fit on PCB

Constraints:
- Must regulate voltage to 7.4V
- Must not dissipate much heat
- Must have efficiency above 80%

2.4.3.4 Thruster

Requirements:
- Move vehicle at a maximum speed of 1.5 m/s
- Attach to the vehicle
- Water proof
- Pressure resistant to 50 m
- Constant thrust at constant voltage

Constraints:
- Smaller than six inches in diameter
- Draw less than 50% of the power budget
- Cost less than 10% of budget
- Rigidly mountable interface

2.4.3.5 Thruster mounting brackets

Requirements:
- Hold thruster rigidly to vehicle
- Symmetrical drag profile
- Utilize thruster mounting points

Constraints:
- Fit within the profile of the vehicle

2.4.3.6 Motor driver

Requirements:
- Demands power from the battery to provide regulated voltage and current to the thruster

Constraints:
- Driver cannot demand more voltage than the battery has available

- Driver cannot demand more current than the battery has available

2.4.3.7 Rudder

Requirements:

- Turn the vehicle
- Be able to attach to the body
- Be able to take small hits without breaking
- Same design as stern planes

Constraints:

- Have a side profile that is less than 4 inches squared
- Be able to accept a shaft of no less than 1.5 inches and 1/8 inch diameter
- Turn a maximum of 15° bi-directionally

2.4.3.8 Rudder Servo

Requirements:

- Produce enough torque to overcome the torque imposed by the control surface, dynamic seal and the lever arm couple
- Be mountable inside of the pressure housing
- Should not interrupt other functions
- Contain metal gearing

Constraints:

- Less than 10% of maximum power
- Fit within the pressure vessel

2.4.3.9 Stern Planes

Requirements:

- Direct the pitch angle of the vehicle
- Be able to accept a shaft of no less than 1.5 inches and 1/8 inch diameter
- Turn a maximum of 15° bi-directionally

Constraints:

- Have a side profile that is less than 4 inches squared
- Be able to take a shaft of no less than 1.5 inches

2.4.3.10 Stern Plane Servo

Requirements:

- Produce enough torque to overcome the torque imposed by the control surface, dynamic seal, lever arm couple and the righting movement of the vehicle
- Be mountable inside of the pressure vessel
- Contain metal gearing

Constraints:

- Less than 10% of maximum power
- Fit within the pressure vessel

2.4.3.11 Pressure vessel

Requirements:

- Hold all electronic components

- Hold mechanical components
- Keep sea water out
- Be rated for 50 meters of depth
- Allow for interfaces with exterior components
- Allow access to interior components
- Fit the profile of the vehicle design
- Provide a rigid structure for the vehicle

Constraints:

- Interior diameter must be greater than four inches
- Exterior diameter must be less than six inches

2.4.3.12 Dynamic seal in pressure vessel

Requirements:

- Hold sea water out of pressure vessel
- Be rated for 50 meters depth
- Operate with a speed of (CALCULATE SPEED OF THE SERVO according to Parker)

Constraints:

- Be less than ½ inch thick

2.4.3.13 O-ring seals

Requirements:

- Hold water out of pressure vessel
- Be rated for 50 meters depth

Constraints:

- Be smaller than the end caps and mounting sites

2.4.3.14 Battery Mount

Requirements:

- Hold battery in place
- Fit within the pressure vessel
- Add support to middle pressure vessel joint
- Guide cables along the length of the vessel
- Provide structural support to the electronics board

Constraints:

- Each battery mount must be less than 1.5 inches thick

2.4.3.15 Top Flat Surface Mount

Requirements:

- Allow for O-ring seal of GPS/Antenna mast

Constraints:

- As small as possible to minimize structural integrity of the vehicle

2.4.3.16 Communication Mast

Requirements:

- House GPS Antenna
- House Radio Antenna

- Located above the vehicle above the water
- Low drag shape
- Center of lift located at the center of mass of the vehicle
- Keep interior components dry

Constraints:

- Sit on top of vehicle so that it does not interfere with control surface function

2.4.3.17 Nose Cone

Requirements:

- Hydrodynamic, streamline shape
- Hold on to pressure vessel end cap
- Able to manufacture in house
- Allow for additional weight to level center of mass
- Allow for a solid mounting point or points for vehicle extraction methods

Constraints:

- Measure between 2 and 4 inches
- Made of a replaceable material
- Have the same diameter as the pressure vessel at contact point

2.4.3.18 Front End Cap

Requirements:

- Seal pressure vessel
- House vacuum, pressure plug
- Solid mounting point for the nose cone
- Hold the external pressure sensor
- Interface to mount the sled

Constraints:

- Have the same diameter as the pressure vessel at contact point
- Thick enough to fit two static O-ring grooves

2.4.3.19 PCB Shelf

Requirements:

- Organize most of the electronics
- Must remain stationary
- Must resist vibrations

Constraints:

- Fit inside of the vehicle
- Mount on mounting points on end cap and battery mount

2.4.3.20 PCB

Requirements:

- Hold electrical components to accomplish mission
- Regulate voltage and distribute to components
- Provide a hub for sensor communication

Constraints:

- Fit on the PCB sled

2.4.3.21 Nose Cone

Requirements:

- Must be able to hold D-ring
- Hydro-dynamically efficient

Constraints:

- Limited to the diameter of the vehicle

2.4.3.22 Microcontroller

Requirements:

- Must read all sensors and communicate to actuators
- Must be able to log data onto data logger

Constraints:

- Limited serial ports

2.4.3.23 Compass

Requirements:

- Must read heading, pitch, and roll

Constraints:

- Limited to inside pressure housing

2.4.3.24 Radio

Requirements:

- Has to be able to communicate from 1 mile away on the ocean

Constraints:

- Has to be located above still water line

2.4.3.25 External Pressure Sensor

Requirements:

- Much be able to give accurate depth

Constraints:

- Sensor section has to be wet
- Has to be able to function up to 50 meters in water

2.4.3.26 GPS

Requirements

- Must be able to get a fixed GPS

Constraints:

- Cannot be submerged

2.5 Management

This section shows how we managed the construction of our vehicle. It also details who worked on specific parts. The vehicle was assembled properly and timely due to good management plans leading up and throughout the build of the vehicle.

2.5.1 Component Breakdown

The component breakdown shows who worked on what part. It also shows specifically each member or faculties role in the part. All of the essential parts in the vehicle are listed

Table 1: Member Work

Sub System	**Component**	**Description**
1.Power	Batteries	Chris spent time creating the power budget, then finding a corresponding battery pack(s) that suited our needs. They also worked with Pedro, to ensure a good Cb and Cg separation, as this proved to be a serious risk. The only fabrication on the batteries was to extend the wires and install the Molex connectors, which was done by Stacey and Chris.
	Reed Relay	The reed relay was placed in the Wiring diagram designed by Stacey and Chris with the help of Ed. The relay was soldered onto a breakout board by Stacey and Quintin, and then the Molex connector was soldered to the board.
	Voltage Regulators	The three voltage regulators were placed in the Power Schematic by Stacey and Chris with the help of Ed. The regulators are surface soldered onto the board, and the board was assembled by Stacey and Chris.
	Motor Driver	The SyRen 10 was chosen by Stacey and Chris after we acquired the SeaBotix thruster. They chose the driver with the thruster's parameters. The driver was put on standoffs and drilled into the electronics sled by Stacey and Chris.
	SeaCon Connector	The SeaCon was donated to us by the department. Stacey, Steven, and Chris all worked together on assembling the internal soldering wire connections and the connector. The charging cord and shore power cord with the male mating SeaCon was made by Quintin. The threaded hole in the front end cap was tapped by Dietrich and Pedro.
2.Navigation	Servo-Motors	Stacey researched servo motors, with help from Pedro's calculations. The Molex connectors were installed by Stacey.
	Servo-Mount	Pedro and Dietrich came up for the Servo-Mount design. Pedro produced the SolidWorks drawings. They both milled out the L-Channel for the mount. The servo-mount block was made on the CNC mill.
	Servo Linkages	The servo linkages were designed and drawn in SolidWorks by Pedro and Dietrich. They were first 3-D printed by Pedro, to test for functionality. They were then made on the CNC mill by Tony, and then touched up on the mill by Pedro.
	Shaft Arm	The shaft arms were designed and drawn in SolidWorks by Pedro and Dietrich. They were first 3-D printed by Pedro to test for functionality. They were then made on the CNC mill by Tony, and then touched up on the mill by Dietrich.
	Dynamic Seals	Dietrich designed the dynamic seals. He found the spring loaded PTFE seals and bushings. He also bored the holes in the servo vessel for the seals.
	Control	Pedro researched what type of foil to use, and came up with the

	Surfaces	design in SolidWorks. He also 3-D printed the four control surfaces out of NinjaFlex.
3.Propulsion	SeaBotix BTD150	Stacey and Chris researched many thrusters, but couldn’t find a viable option. Dr. An decided to lend us a SeaBotix BTD150. The thruster cable was potted through the rear end cap by Pedro and Dietrich.
	Thruster Mount	The thruster mount was designed by Dietrich and Pedro. It was produced in SolidWorks by Pedro. The mount was machined by both Pedro and Dietrich.
	Thruster Shroud	Pedro produced the design and drawing in SolidWorks. He also 3-D printed out both pieces.
4.Control	Arduino Due	The group had an open discussion on which microcontroller to use, we decided on the Due with help from Ed. The headers to install the Due were soldered by Stacey and Chris. The programming for the Due and all its sensors was done by Steven and Quintin.
	AUV Motherboard	The schematics for the motherboard were done by Stacey and Chris with assistance from Ed. The PCB layout for the board was also done by Stacey and Chris with help from Ed. The production of the board was done by Stacey and Chris. The connectors were done by Stacey and Chris with help from Steven as well. The board was mounted onto the electronics sled with stand offs by Quintin and Dietrich.
	Honeywell PX2 Depth Sensor	The depth sensor was researched and chosen by Chris. The wiring and connector was done by Stacey. The hole in the nose cap was tapped by Dietrich and Pedro. The programming for the depth sensor was done by Quintin and Dietrich.
	TCM-2 Compass	Chris and Stacey researched IMU’s, couldn’t find anything reliable enough. The compass was donated by Dr. An. This was mounted on the electronics sled by Dietrich and Quintin. The connector and wiring was done by Stacey and Chris. The programming for the TCM-2 was done by Quintin and Steven.
	Adafruit GPS Breakout Board	The GPS Breakout Board was obtained from a previous senior design project. The receptacles and standoffs for the board were installed by Stacey and Pedro. The programming for the GPS was done by Quintin and Dietrich.
	Adafruit 10 DOF IMU	The IMU was obtained from a previous senior design project. The receptacles and standoffs for the board were installed by Stacey and Chris. The programming for the GPS was done by Quintin and Dietrich.
	Adafruit uSD Breakout Board	The uSD was obtained from a previous senior design project. The receptacles and standoffs for the board were installed by Stacey and Chris. The programming for the uSD was done by Quntin and Dietrich.

	XBee Radio	Stacey researched and found the proper XBee radio for the vehicle. Stacey and Chris installed the receptacles for the radio. The programming was done by Steven and Quintin.
5. Body	Electronics Sled	The entire body was designed and modeled by Pedro and Dietrich. The electronics sled was made by Pedro and Dietrich. The battery mount brackets were cut by them, and then CNC milled by Tony. The fiber glass top was cut by them, and the rest was assembled by them.
	Front End Cap	The end cap was lathed and milled by Dietrich.
	Battery Plug	Tony CNC lathed the plug, and then Pedro but the O-Ring groove on with the mill.
	Electronics Vessel	The vessel was lathed, drilled, and polished by Pedro and Dietrich with some help from Tony.
	Servo Vessel	The vessel was lathed, drilled, and polished by Pedro and Dietrich.
	Potted Antenna Mast	Pedro designed the NACA foils for the mast. The injection mold was made with the CNC by Tony. Pedro and Dietrich setup and made the molding.
	Mast Bottom Plate	The mast bottom plate was made on the CNC by Tony.
	Servo Vessel	The vessel was lathed, drilled, and polished by Pedro and Dietrich.
	Rear End Cap	The rear end cap was lathed, drilled, and tapped by Pedro. Dietrich also designed and helped with the potting of the thruster cable.

2.5.2 Gantt Chart

The Gantt chart kept us on track throughout project. We could see whether we were on track or not, and could adjust or workload according. We could also keep one another in check with the chart.

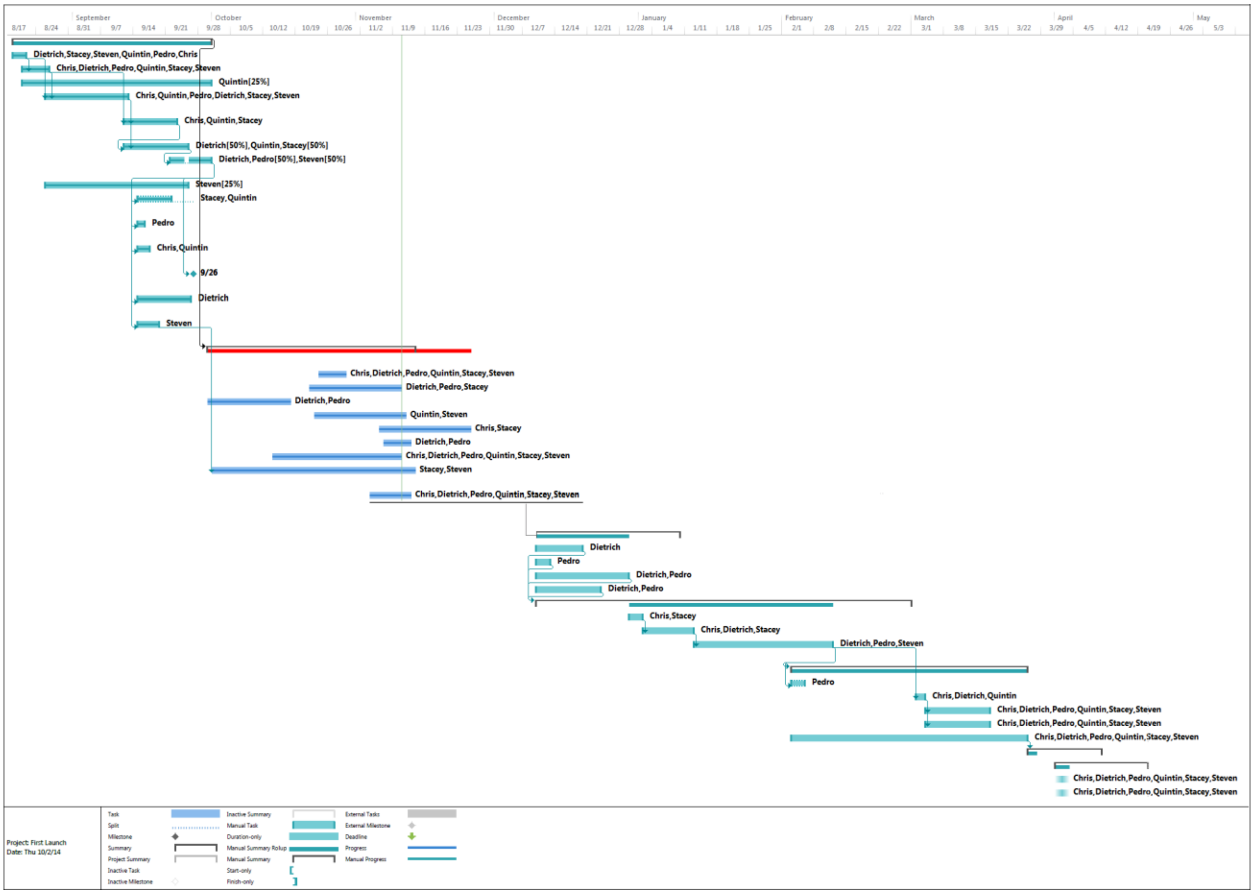


Figure 36: Gantt Chart

3. Design & Engineering

There were crucial mathematical calculations that went into the design of Tito. The vehicle may have never worked without these theoretical developments. The development for our software code is also shown here. As well as the design of our electrical components and AUV Motherboard.

3.1 Engineering Calculations

This section goes through calculations that were essential in designing the vehicle. These calculations verify that our design theoretically works.

3.1.1 Mechanical Calculations

Mechanical calculations deal with the equations of motion, hydrodynamics, and integrity of the vehicle. Some of the equations used are as follows:

$FD=-cd\varphi V2Af2$ (1)

$Af=\pi r2$ (2)

$cd=css\pi APA[1+60dl3+0.0025dl]$ (3)

$Lfin=12\rho cLSfin\delta eve2$ (4)

Required Thruster Force

The estimated mechanical force required to move the AUV is dictated by the drag that acts on the body of the AUV. The drag was estimated using only the axial drag expressed by:

$$F_D = -\left(\frac{c_d \varphi V^2 A_f}{2}\right) \qquad (1)$$

where ρ is the density of the fluid, in this case salt water with a density of 1027 $\frac{kg}{m^3}$, V is the velocity of the fluid traveling around the body, in this estimate it will be the max velocity of the AUV of 1.5 $\frac{m}{s}$, A_f is the frontal area of the body,

$$A_f = \pi r^2 \qquad (2)$$

where r is the radius of the body, 0.0635 m and c_d is the coefficient of drag estimated by:

$$c_d = \frac{c_{ss}\pi A_P}{A_f}[1 + 60\left(\frac{d}{l}\right)^3 + 0.0025\left(\frac{d}{l}\right)] \qquad (3)$$

Where c_{ss} is the Schoenherr's value for flat plate skin friction, $A_P = ld$, l is the length of the vehicle and d is the diameter. From *Principles of Naval Architecture* [3], the c_{ss} is estimated to be 3.397 x 10^{-3}. The coefficient of drag is calculated to be 0.12 which lead to a drag force of 1.76 N. To error on the side of caution a value of 0.3 was used for the drag force calculations. The estimate drag force is 4.4 N. to confirm this number a flow analysis was made in SolidWorks. The drag force obtained from SolidWorks was 6.7 N. To continue being conservative the larger of the two drag forces was used and then some, choosing a drag force of 9 N as the estimate.

3.1.1.1 Thrust Calculation Validation

A test was conducted to validate the drag calculations in order to determine thruster parameters. A similarly sized cylinder with two flat faces was tethered with a force sensor and dragged through the intercostal and the drag forces were measured. The drag forces measured are a very conservative estimate of the drag forces calculated previously for the vehicle. The cylinder tested had flat faces perpendicular to flow, a true bluff body with drag coefficient greater than 1. In actuality, the produced AUV will be a more streamlined shape with a drag coefficient closer to 0.1.

The drag on cylinder was tested by first finding the weight in water, then recording the motion of pulling the vehicle through the water. The force needed to pull the vehicle was measured off of the force gauge. The force gauge measured 1.86 and 2.02 pounds to pull the cylinder in two recorded trials. Post processing the videos provided the velocity of the cylinder. It was concluded that it took 1.86 lbs. to pull the cylinder 0.62m/s and 2.02 lbs. to pull the cylinder 0.68m/s. That data provides that we need approximately 4.45 to 4.5 lbs. of force to move the vehicle at 1.5 m/s. That force corresponds to 20 N of drag force. That measured drag force corresponds to a drag coefficient of about 1.2. A drag coefficient of 1.2 is an order of magnitude higher than drag coefficient used in the development of the analytical minimum. This drag force estimation is high due to the bluff body of the test cylinder in comparison to the more streamlined AUV design shape. In addition, the measurements obtained from the force sensor were fluctuating and the measurement used was the peak force observed during the trial. This test validates the analytical results. The AUV drag force must be less than the validation test but greater than the analytical minimum.

From our testing we calculated that the vehicles speed was around 2 m/s. This correlates with the numbers produced by the SolidWorks flow analysis. Our theoretical speed was 1.5 m/s with a conservative drag force of 9 N. Taking the theoretical speed and dividing it by the actual speed and then multiplying it by the 9N gives a value of 6.75 N which is close to 6.7 N obtained by the

SolidWorks flow analysis. That is an error of 0.75% from the SolidWorks model.

3.1.1.2 Fin Selection

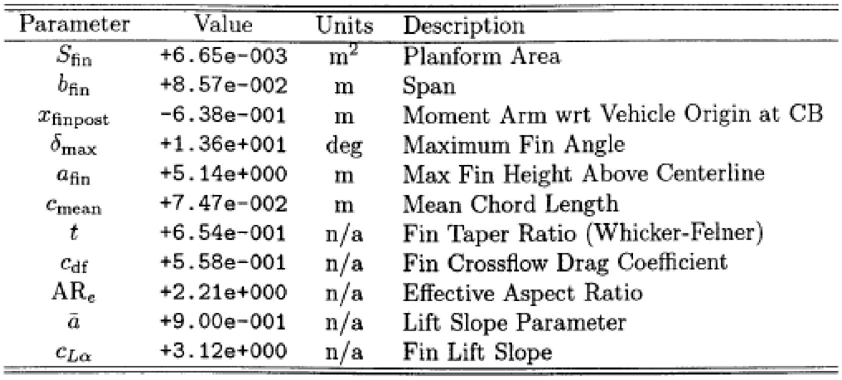
The fin design is dependent on the weight of the AUV and the rate at which the decent of the vehicle could be controlled. To find the weight of the AUV is purely dependent on the volume of the AUV; therefore a preliminary fin design must be created to decide on the mechanisms that will drive the fin and fit into the system. The fin design is dictated by a few items including the shaft required to hold and turn the control surface, the shafts relative location to both the vehicle and the control surface and the moments required to dive the vehicle, which are proportional to the weight of the vehicle due to the moment caused by the distance between the buoyancy force and the center of gravity, or the righting moment of the vehicle.

The shaft of the vehicle was chosen to be as 1/8" stainless steel rod. The material has a modulus of elasticity, E of 69 GPa and a room temperature yield strength of 55 MPa (Foundations of Materials Science and Engineering) If the rod only penetrates pressure vessel 1.5 inches the chances that a force applied to the end of the rod causing it to bend are negligible. The thickness and length of the rod in the fin provides enough structural support for the fin to attach and rotate with the rod.

The shaft should be positioned as far aft as possible to increase the lever arm between force applied by the fin on the sub and the center of mass. In this design that distance is restricted by the length of the pressure vessel. The total length of the vehicle is estimated to be 0.9 m or about 35 inches. The distance from the front of the vehicle to the control surfaces and the stern planes will vary between 0.7-0.8 m or 27.5-30 inches. This will provide a lever arm between 0.2-0.4 m or 7.8-15.9 inches, that should be sufficient to keep the size of the fin reasonable.

From the data collected from testing the vehicle it was possible to see that the vehicle dove at a rate of 0.1m/s, using this rate the vehicle will make it to a depth of 10 m in 1.67 mins. This is a reasonable time for diving being that the total mission is supposed to last around 15-20 min.
To obtain an estimated torque required by the motors to rotate the fins, the fins of a Remus 100, a commercial/industry standard AUV will be used as a rough estimate. The fin parameters assumed for this estimation are on Table 1.

Table 2: Remus fin Parameters, from [4]



Parameter	Value	Units	Description
S_{fin}	+6.65e-003	m^2	Planform Area
b_{fin}	+8.57e-002	m	Span
x_{finpost}	-6.38e-001	m	Moment Arm wrt Vehicle Origin at CB
δ_{max}	+1.36e+001	deg	Maximum Fin Angle
a_{fin}	+5.14e+000	m	Max Fin Height Above Centerline
c_{mean}	+7.47e-002	m	Mean Chord Length
t	+6.54e-001	n/a	Fin Taper Ratio (Whicker-Felner)
c_{df}	+5.58e-001	n/a	Fin Crossflow Drag Coefficient
AR_e	+2.21e+000	n/a	Effective Aspect Ratio
$\bar{a}$	+9.00e-001	n/a	Lift Slope Parameter
$c_{L\alpha}$	+3.12e+000	n/a	Fin Lift Slope

The lift force provided by the fins are found using the following equation:

$$L_{fin} = \frac{1}{2}\rho c_L S_{fin}\delta_e v_e^2 = \frac{1}{2}(1027)(3.12)(6.65e^{-3})(13.6)(1.5^2) = 326 \text{ N} \quad (4)$$

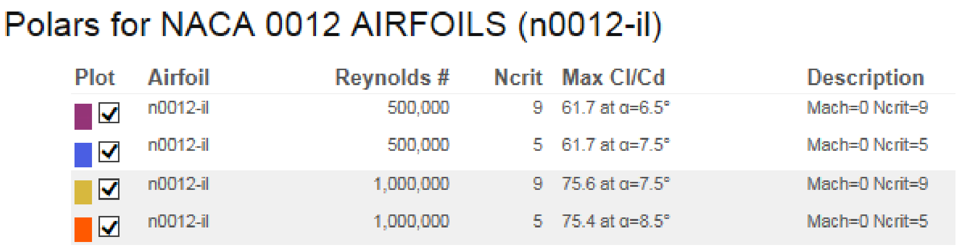
This is the major component of the moment caused by the fin about the point of rotation. The point where the lift force is applied on the fin, for a symmetrical NACA section is located around the thickest part of the chord or at a distance of 30% of the chord away from the leading edge [4]. To create stability in the fin the point of rotation will be chosen to be around 25% of the chord away from the leading edge, this will have the fin return to a neutral position if the motor stops applying a torque and it also decreases fluctuations when the fin is set to an angle of attack of 0°. (http://woodfreeman.com/pdf/rudders1.pdf) The control surfaces on the Remus have a 0012 NACA folio cross section, a span of 10.2 cm or 4 in and a maximum chord of 8.9 cm or 3.5 in. (REMUS 6 DOF nice) The fins conform to the shape of the vehicle. The vehicle has a tapered aft and therefore the fins trail back along the body. This will be disregarded in the design of the vehicle because the fins will be located before the tapering tail cone and sit along the pressure vessel.

To make calculations simpler both the chord and the span of the first iteration will be 4 in. At 4 in the lift force will act 1.2 in away from the leading edge and the point of rotation will be at 1 in from the leading edge. The distance between the two will be 0.2 in. Neglecting the drag force for a streamline body and using the result from the lift calculation above, 326 N or 1172.6 oz. of force and then multiplying that by the lever arm gives a torque of 235 oz.-in required to rotate the fin while the vehicle travels at a velocity of 1.5 m/s and the fin is turned to an angle of attack of 13.6°. This torque gives us a rough estimation of what the required torque from the control surface motors will look like.

To validate the lift coefficient of 3.12 from Table 1, a NACA 0012 cross section was reviewed. To do so it was necessary to find the Reynold's number for the region around the control surfaces at the required maximum velocity of 1.5 m/s. The Reynold's number acting at the location where the shaft has been placed is 1,000,000. This was obtained using a kinematic viscosity of sea water at 20°C of 1.05 x 10-6 m^2/s a length of 0.7 m and a velocity of 1.5 m/s [5]. It is possible to obtain the lift and drag polars of the NACA 0012 airfoils around that Reynold's number from Airfoil Tools, Figure 1 [6].

The web site also lets you choose the Ncrit value, which is used to model the turbulence of the fluid or roughness of the air foil. The developer defines it as, "… the log of the amplification factor of the most-amplified frequency which triggers transition. A suitable value of this parameter depends on the ambient disturbance level in which the airfoil operates, and mimics the effect of such disturbances on transition [7]." For this discussion all the parameters available in the region for a Reynold's number of 1,000,000 will be used. Figure 1 shows, going from left to right and top to bottom, the lift coefficient versus the drag coefficient, the lift coefficient versus the angle of attack, the pitching moment versus the angle of attach and the coefficient of drag versus the angle of attack.

In Figure 39, in the Cl/alpha plot it is possible to see a range for the lift coefficient of -1.5 to 1.5 for angles of attack of -20° to 20°. At around 13.6° the coefficient of lift can range from 1.2 to 1.4 depending on the Reynold's number and the Ncrit used. This is significantly lower than the 3.12 obtained from the Remus fin parameter table. To error on the side of caution the lift coefficient from the table will be used for the calculations. From the plot it is also interesting to notice that the stall angle for the lift coefficient ranges from around 14° to 16°. Looking at the Cd/alpha plot it is possible to see that the drag coefficient of the foil is normally around 0.01, going to angles greater than 5° it is possible to see the growth of the drag coefficient.



Polars for NACA 0012 AIRFOILS (n0012-il)

Plot	Airfoil	Reynolds #	Ncrit	Max Cl/Cd	Description
🟪 ☑	n0012-il	500,000	9	61.7 at α=6.5°	Mach=0 Ncrit=9
🟦 ☑	n0012-il	500,000	5	61.7 at α=7.5°	Mach=0 Ncrit=5
🟨 ☑	n0012-il	1,000,000	9	75.6 at α=7.5°	Mach=0 Ncrit=9
🟧 ☑	n0012-il	1,000,000	5	75.4 at α=8.5°	Mach=0 Ncrit=5

Figure 37: NACA Chart

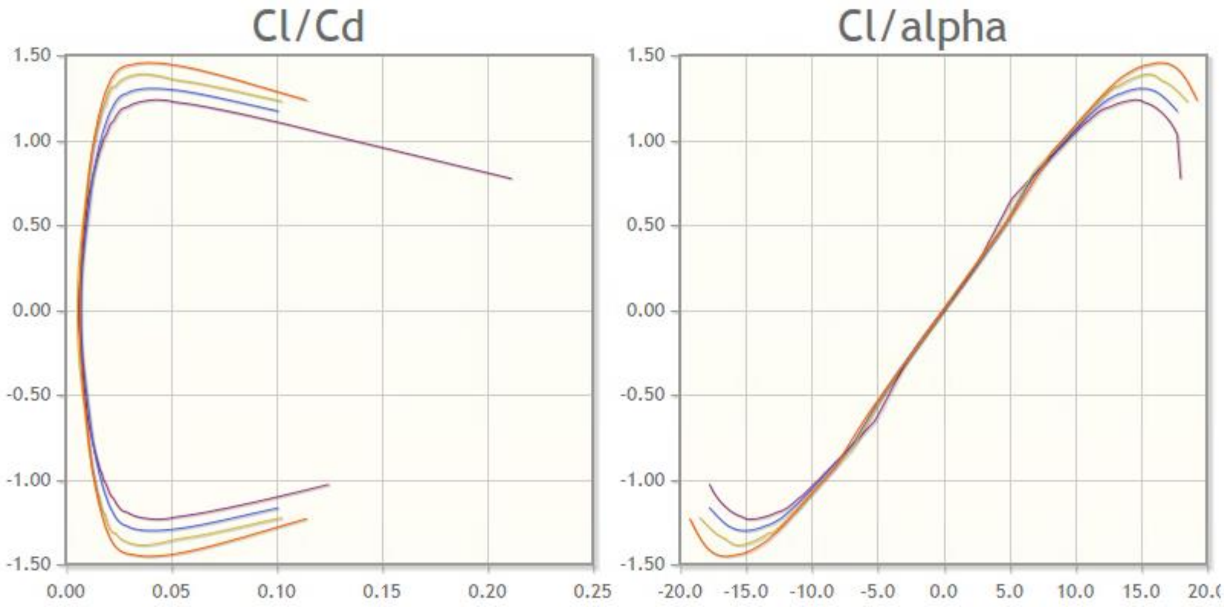


Figure 38: Cl/alpha and Cl/Cd Plots

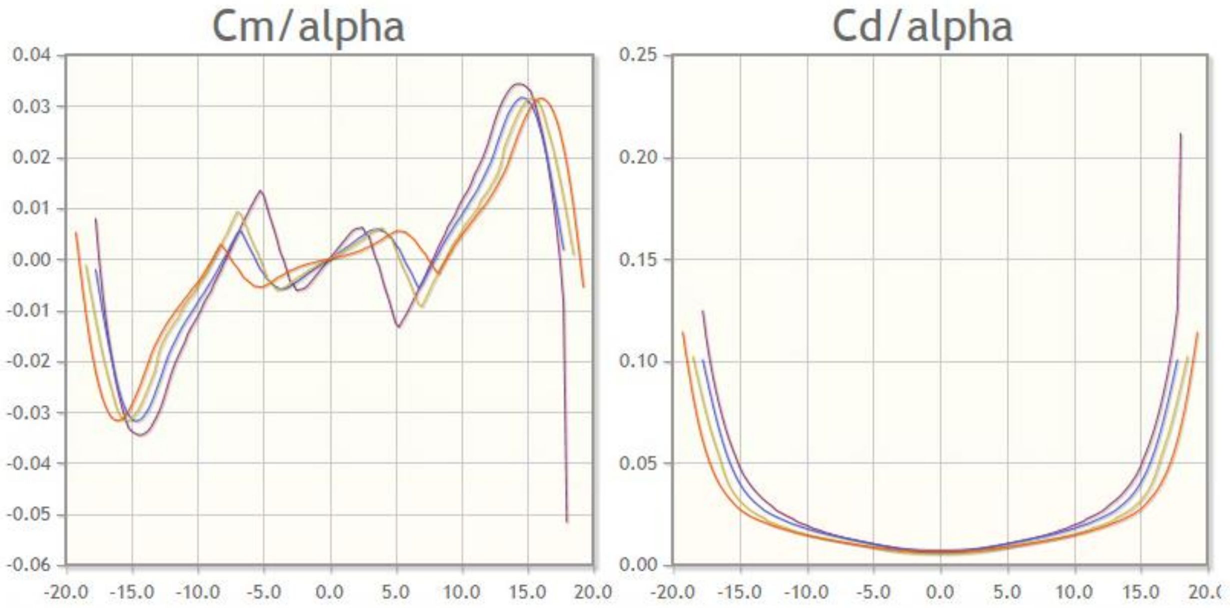


Figure 39: Cm/alpha and Cd/alpha Plots

Figure 39: Going from left to right and top to bottom the lift coefficient versus the drag

coefficient, the lift coefficient versus the angle of attack, the pitching moment versus the angle of attach and the coefficient of drag versus the angle of attack [7].

$$Ls = (F_B - F_g)\delta sin\theta$$

In order for the vehicle to dive the fins have to produce a torque that overcomes the righting moment. The larger the angle the vehicle needs to dive at the larger the righting moment. A decoupled equation of motion was used to compare the dive angle to the required torque that is needed to be overcome: where L is lift force produced by the stern planes, s is the distance from the stern planes to the center of buoyancy, F_B is the buoyant force, F_G is the weight δ is the maximum distance between the center of buoyancy and the center of mass and θ is the dive angle of the vehicle. This equation was put into MATLAB and run with values for the dive angle from 0 to 180°

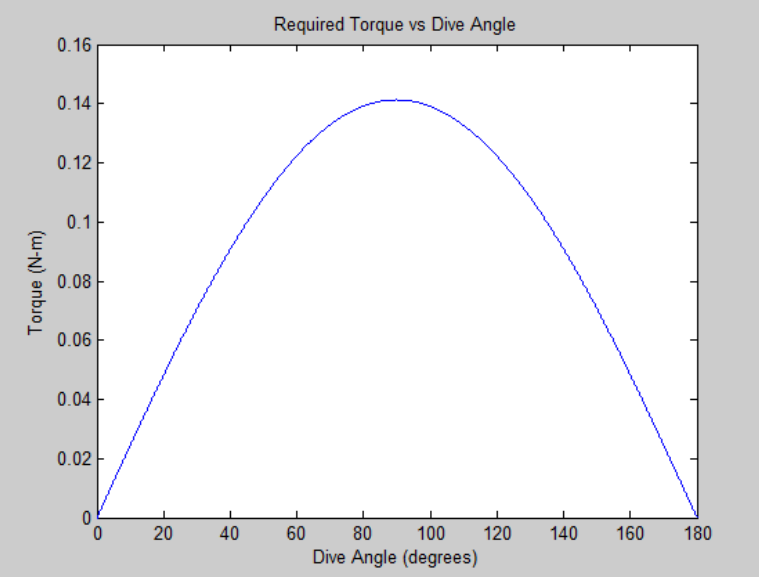


Figure 40: Required Torque vs Dive Angle

The maximum required torque is at 90° and correlates to the vehicle both diving straight up and diving straight down. The value of that torque is 0.1413 N-m or 20 oz-in. The required lift force needed to produce this torque is 0.7065 N.

To obtain a force of this magnitude the fins need to have a platform area of at least 3.2 in^2, this was calculated using Equation 4. A fin of that size is very small and would not meet the requirements to accept the 1/8 inch rod. Therefore to error on the side of caution, again, the fin will be designed to account for extraneous factors and will be able to fit the required shaft. From the vehicle testing it was possible to see that the servos used were strong enough to move the fins and control the vehicle. The 290 oz-in provided by the servos greatly surpassed the calculated 20 oz-in.

3.1.2 Electrical Calculations

The electrical calculations ensure the vehicle will have enough power, and that all devices will operate within their expected range. This will help prevent any damage to expensive electrical components of the vehicle.

3.1.2.1 Power Budget

Device	Type of Current	Current Range (A)	Voltage Draw (V)	Efficiency	Power (W)
Seabotix BTD150 Thruster	Max Continuous Amperage	4.25	19.1 +/- 10%	50%	81.175
Pololu HD-1218TH Servo (x2)	No Load	0.5	7.4	50%	37
	Stall	2.5	7.4		
PNI TCM-2 Compass	Standard	0.02	5		0.1
	Low Power	0.013	5		
Honeywell PX2 Pressure Transducer	Typical	0.005	5		0.025
Adafruit 10-DOF IMU	Typical	0.02	5		0.1
Adafruit GPS Breakout	Typical	0.02	3.3		0.066
Dimension Engineering Motor Driver SYSREN10	Driver Current	0.01	5		0.05
	Peak to Thruster (4.25)	0.6375	22.10	85%	14.1
D1D40 Relay	Input	0.0016	22.10		0.03552
MPX 4250 Pressure Sensor	Typical	0.007	5.1		0.051
	Max	0.01			
TC1047AVNBTR Temperature Sensor	Supply	0.000035	5.5		0.000175
A3212EUA Hall Effect Sensor (x2)	Supply	0.002	3.5		0.007
Arduino Due	Max	0.13	7.4		0.962
ACS711ELCTR Current Sensor	Max	0.0055	3.3		0.01815
ACS714ELCTR Current Sensor (x2)	Max	0.01	5		0.05
TSR3-24150 Switching Regulator (x2) to Servos	Max	3	22.2	85%	19.98
MAX3222ECAP RS232 Level Translator	Max	0.01	3.3		0.033
TSRN1-2450 Switching Regulator to Due	Max	1	22.2	85%	3.33
		Total Current			**Total Power**
		12.151635Amp			157.08
Battery		**Amp-Hours**			**Watt-Hours**
(3) NiMH 7.2V Battery Packs in Series		10			216

Table 3: Power Budget

The power budget in table 3 calculates the total current and voltage draw from all of the vehicles electrical components. We can then find how much battery capacity we need. This helps us to decide which battery pack(s) to choose, and gives us an estimated run time of our vehicle. The run time of the vehicle can be calculated using the following equations:

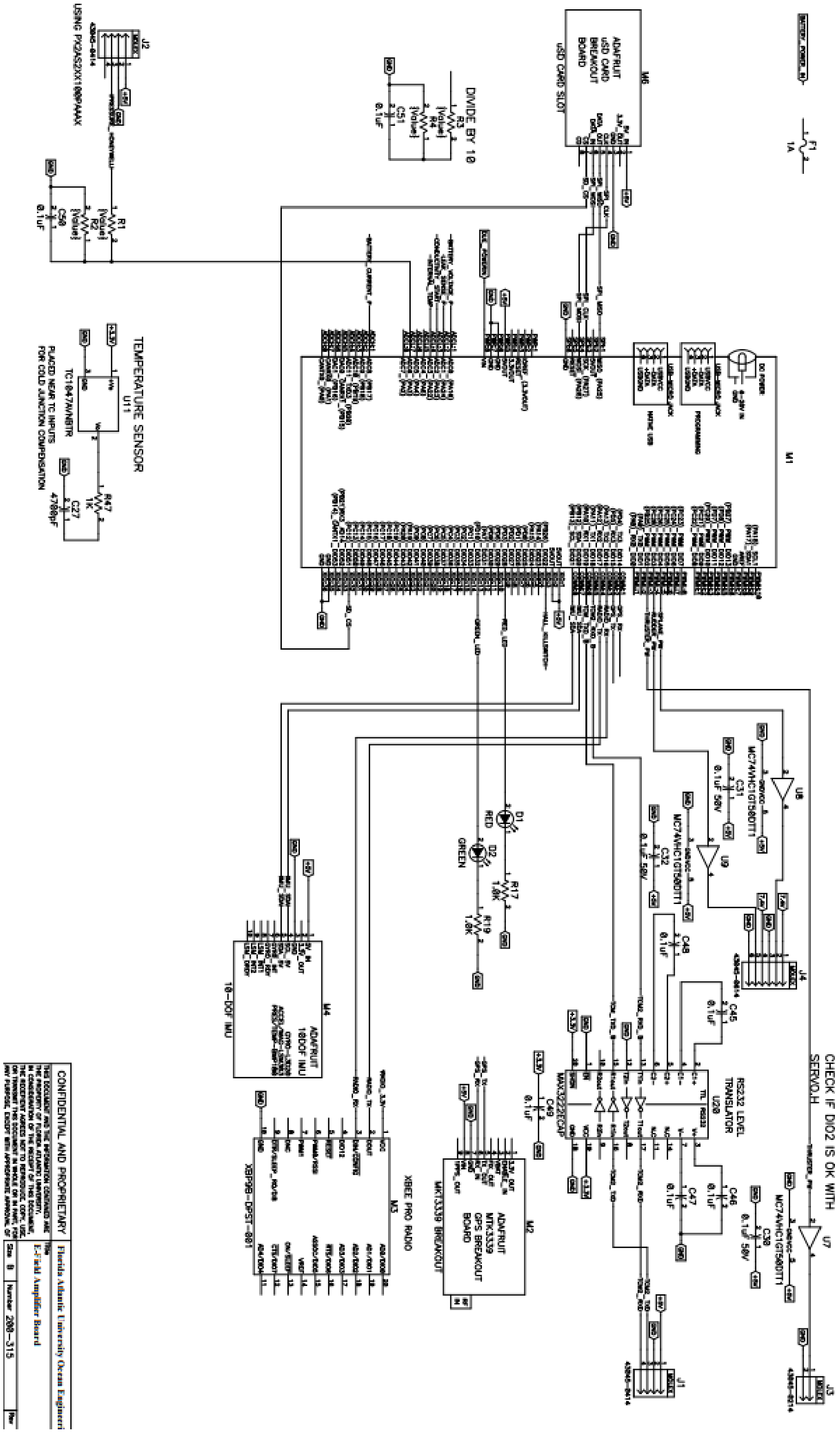
$$t_{run(hrs)} = \frac{capacity_{(AH)}}{i_{max}}$$

The run time is about 0.823 hours using the above equation. This is with the thruster and servos at maximum current, so the real run time may be much longer.

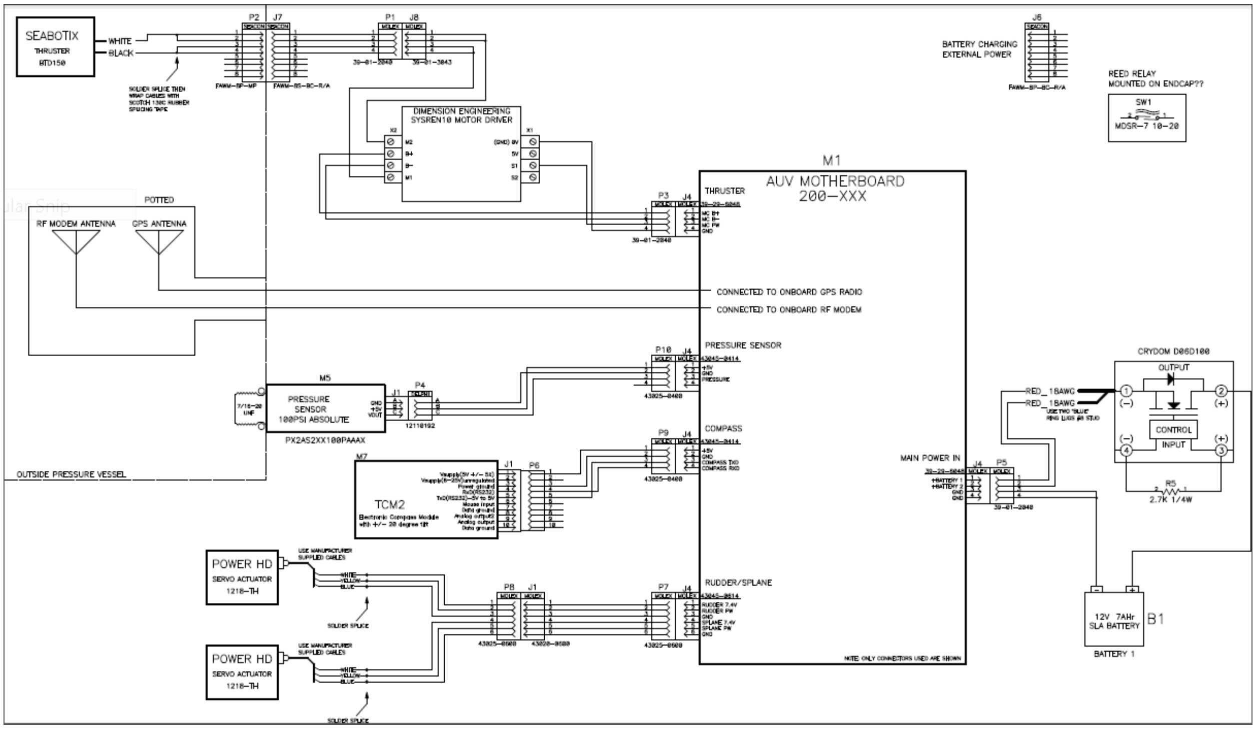
From tests the vehicle surpassed the 0.823 hours or 49 mins that were calculated above. Once again proving the accuracy of our design calculations.

3.2 Electrical Schematics

3.2.1 Arduino Schematic



3.2.2 Wiring Diagram



3.3 Software Logic Flow

The AUV is operated by a program that is installed on an Arduino Due micro-computer. The program consist of the standard elements required by Arduino in addition to a master program with sub-routines. The sub-routines will be individual libraries that are called in to the master program for execution in a pre-determined sequence. The following is a software flow diagram representative of the system.

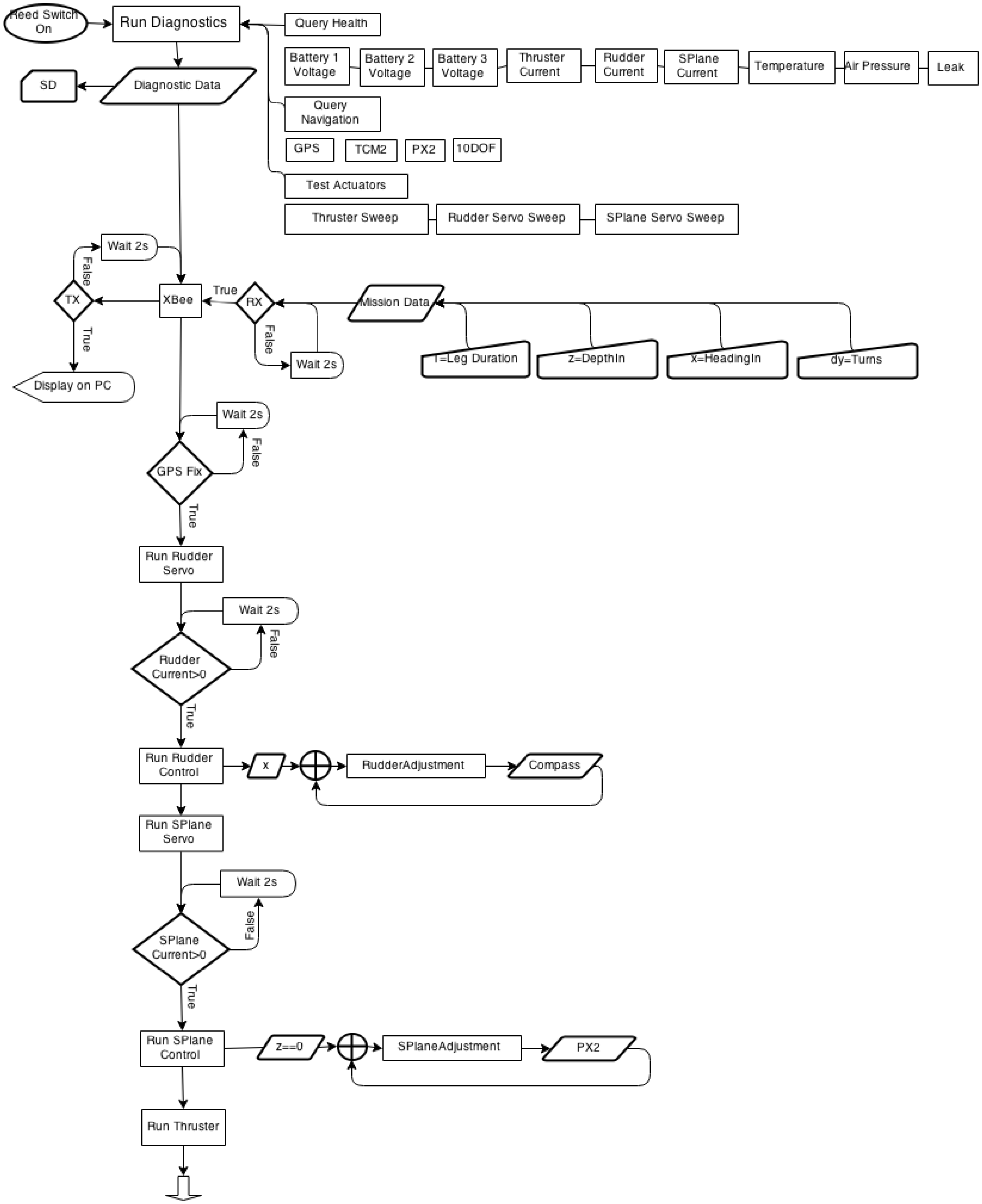


Figure 41: Software Logic

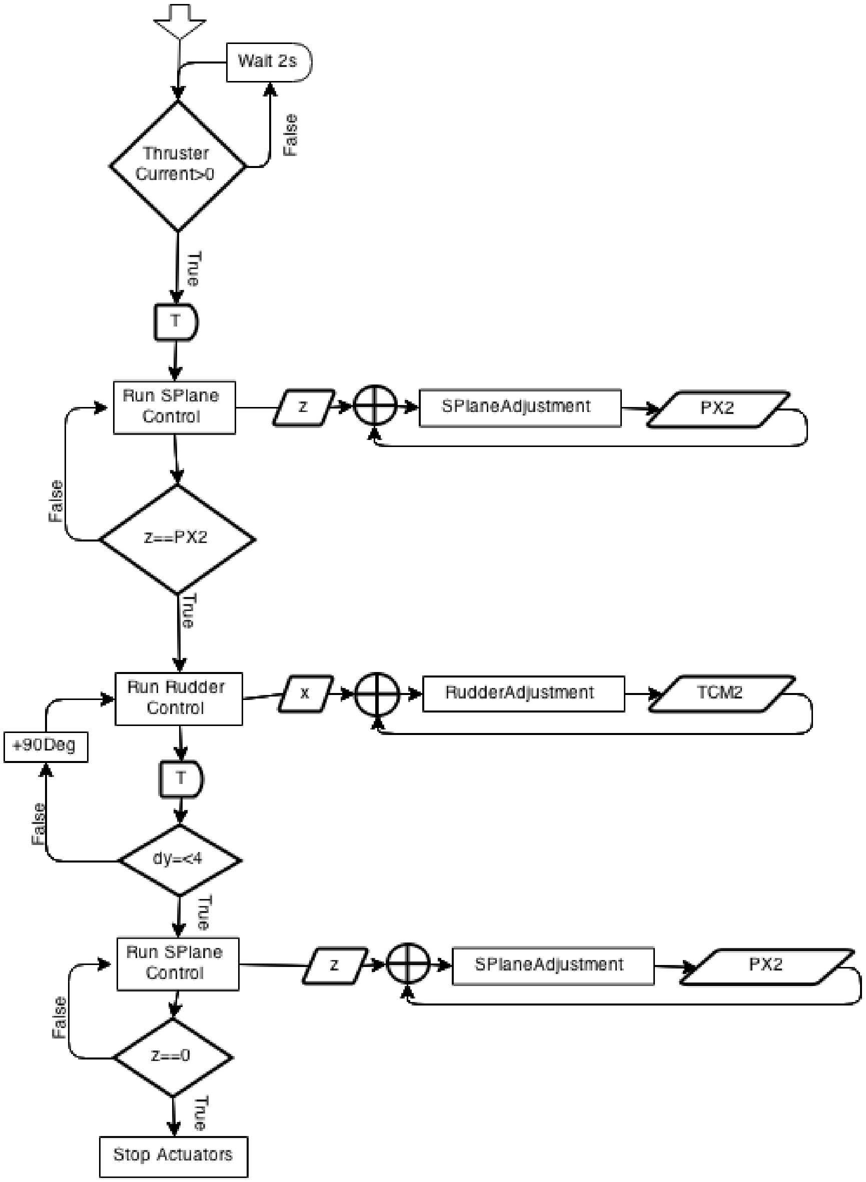


Figure 42: Software Logic Flow 2

Tito Tuxedo is an autonomous underwater vehicle capable of performing user defined missions. The user can enter a depth, heading and the number of turns that the vehicle has to make underwater. These instructions are sent to the Arduino processor via a digital radio communication system. When the master program TitoSoft executes a list of custom and standard libraries are called on. The program logic follows the sequence in Figures 41 and 42. The vehicle is powered either by an onboard batteries or via external power. A Reed switch turns the system on and a switching relay selects between the two power sources.

First a diagnostics program of the entire system is run; this is done in subroutines for health, navigation and actuators. The diagnostic information is stored on an SD card onboard the vehicle and transmitted via the XBee communication system to the PC for user analysis. The user defined values are transmitted to the vehicle via XBee. Tito now waits for a GPS fix before continuing. Once the GPS has a fix, the rudder control system is initialized followed by the stern plane control system. With the control surfaces active, the thruster is initialized.

Figure 43 illustrates the waypoints that have to be reached for the mission to be a success. From WP-1 While moving forward at the water surface, Tito finds the required heading and adjusts the rudder to turn towards it. After a time interval (T) at WP-2, Tito changes its stern plane angle to dive to the user defined depth. Tito now monitors the depth sensor and continues diving to WP-3. Now Tito maintains this depth for a period (T), then at WP-4 a 90-degree left turn command is executed and Tito finds the new heading. The preceding step is repeated for the number of turns selected to reach WP-7, then Tito resurfaces to WP-8.

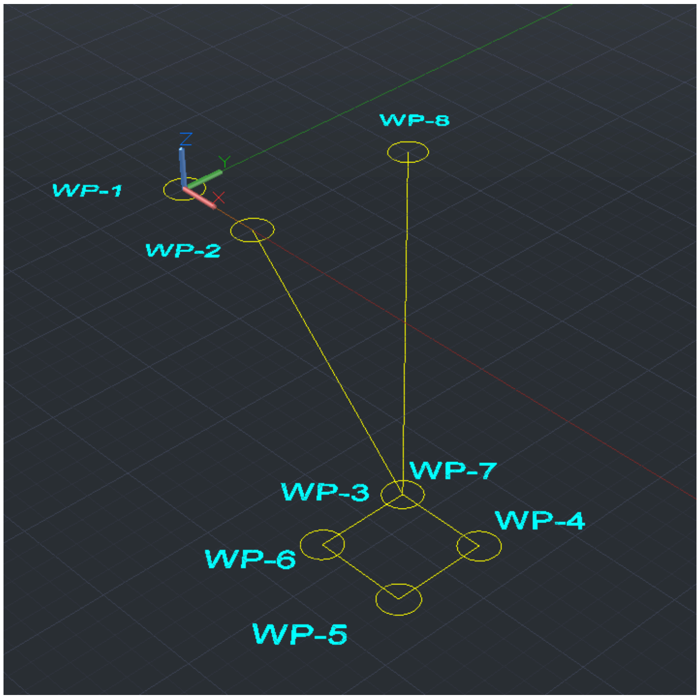


Figure 43: Mission Waypoints

4. System Construction and Assembly

4.1.1 AUV Body Construction

The whole AUV was drafted on SolidWorks and approved by the advisors and used for the construction of the vehicle. Each component was reviewed by the advisors and checked for machinability and potential issues. Once the draft was approved material was found to create the individual pieces. Some of the material was donated from scraps that were found around the machine shops and the rests was purchased from outside vendors.

4.1.1.1 Front end cap

The front end cap was machined on the lathe and other features were later added on the mill. The raw material was cut slightly larger than needed on the band saw from a piece of 5” diameter solid rod of 6061 Aluminum alloyed purchased from Alro Metals Plus. The raw material was chucked into the lathe and the outside was cleaned up by passing the cutting tool over the raw material, then turning the piece around in the chuck and facing the piece with the tool giving it a smooth perpendicular surface. Aluminum chucks were then cut to size and the piece was placed with the smooth perpendicular side chucked. While in this set up the other side was faced and the lip for the O-rings was cut to size with care. Then the O-ring grooves where cut into the piece making sure the tolerances for the O-rings were held, these tolerances were found using the Parker O-ring Handbook. With the outer diameters finished the piece was then dug out with a boring bar. The inside was dug out to leave enough metal for the tie-down to be attached on the outside part of the piece.

Once the end cap itself was finished the piece was taken to the mill and details were added. The end cap was chucked onto the turntable on the mill and the center of the piece was found and zeroed. The holes for the tie-down and the vacuum plug were made first, so that the piece could be pressure tested with the electronics pressure vessel and the rear end cap. The hole for the tie-down was made carefully so that the drill bit did not go through the entire piece and left enough metal to hold structural integrity. The vacuum plug was drilled with a very small bit first to make sure that the vacuum would hold as it was being plugged, but the drill bit broke inside of the piece and a larger bit had to be used for the hole. Then a larger hole was drilled and threaded for the plug itself. The larger hole for the vacuum hole was not a problem because a piece of polyurethane was cut to size and placed into the vacuum plug hole allowing for a smaller hole to be drilled into the polyurethane.

Once the piece was tested at the surface and in the pressure chamber the piece was chucked into the mill again using the turntable but turned around so that the part that it faced the inside of the pressure vessel faced up. The SEACON connector hole was then drilled by opening the through hole for the connector and then getting an end mill bit to make the inlay for the washer and nut. This inlay was needed because the end cap was too thick for the SEACON to penetrate and by grabbed by its nut. Then the pressure sensor hole was drilled and threaded and the holes for the electronics bench were added and threaded.

It was not until the two pressure vessels and the rear end cap were finished that that the set screw holes were created for the front end cap. The whole pressure vessel was put together and the holes were made in the correct orientation by chucking the entire system on to the mill using the turntable on its side and a tail stock with a piece of aluminum that was cut so that it sat at the center of one of the end caps when placed in the turntable chuck. Support shims were placed at the center of the pressure vessel to prevent bending as the holes were made. The set screws were set at a certain distance from the end cap lip to make sure that there was enough metal to prevent shearing from the pressure vessels. The distance drilled in was carefully monitored to practice for the set screws drilled into the servo pressure vessel. The threads were tapped as each hole was made to use the mill to start the taps and go in straight. The threads were finished by hand using a bottoming tap.

4.1.1.2 Rear end cap

The rear end cap was machined on the lathe and other features were later added on the mill. The raw material was cut slightly larger on the band saw from a piece of 5” diameter solid rod of 6061 Aluminum alloyed purchased from Alro Metals Plus. The raw material was chucked into the lathe and the outside was cleaned up by passing the cutting tool over the raw material, then turning the piece around in the chuck and facing the piece giving it a smooth perpendicular surface. Aluminum chucks were then cut to size and the piece was placed with the smooth perpendicular side chucked. While in this set up the other side was faced and the lip for the O-rings was cut to size with care. Then the O-ring grooves where cut into the piece making sure the tolerances for the O-rings were held, these tolerances were found using the Parker O-ring Handbook. With the outer diameters finished the piece was then dug out with a boring bar. The inside was dug out to reduce weight from the tail section of the vehicle. Before making any other modifications to the piece it was tested with the electronics pressure vessel and the front end cap on a surface test and a pressure chamber test.

Once the piece was tested it was placed into the mill using the turntable chuck with the inside upwards to make the thruster bulkhead. Fist the through hole for the cable was drilled slightly larger than the cable so that potting could sit around the cable. Then a larger end mill was used to create a concentric face on the inside of the end cap. Around that face four holes were made into the piece to hold a mold that was used to pour and hold the potting compound into place. Then the piece was turned around and another face was milled with the previous end mill that was not concentric with the cable through hole. This was done to prevent a rotation of the potting compound potentially causing a leak.

The mold for the potting was made of high-density polyethylene. A small block was first cut in half, four holes were made and threaded to keep the two halves together with screws. Then the two halves put together with screws were chucked into the mill. A tight through hole for the cable was drilled followed by a larger ball mill face and then a slight chamfer to the ball mill face. Then another four screw holes were made and threaded to grab the mold concentrically onto the four screw holes placed on the inside of the end cap. Then through the one of the sides a 1/8th nipple was threaded on so that the potting could fill the ball mill cavity.

The thruster cable was run through the end cap and then through hole drilled into the mold. The mold was then screwed shut and screwed onto the endcap. A small hose was attached to the nipple that was threaded to the mold and used to pot the thruster into place with the 3M Scotchcast, Flame Retardant Compound 2131. The compound was placed into a small syringe that attached to the hose. With the end cap sitting on the edge of the table with the thruster attached above it the syringe was squeezed slowly until the compound completely filled the ball end mill groove and the end cap face milled concentrically with the thruster cable through hole. Then it slowly poured past the thruster cable filling the non-concentric end mill face on the outside of the end cap.

The set screw holes were not made until the front end cap and the two pressure vessel sections were complete. The four pieces were oriented correctly and the holes were drilled and tapped one by one so that the mill could be used to begin the tap.

4.1.1.3 Electronics Pressure Vessel

The Electronics Pressure Vessel (EPV) was cut from raw material on the band saw then cut to size on the lathe, holes were opened in it and a flat surface was placed on it on the mill and then it was cleaned up on the lathe. The EPV was cut from a 5” diameter 6061 aluminum alloyed tube purchased from Alro Metal Plus. Internal supports had to be made to chuck the tube on the lathe to get a perpendicular edge for the tube and cut it down to an exact size. Two disks were created that fit perfectly into the tube. One disk had a lip to sit just outside one end of the tube and the other fit tightly inside the vessel. Both disks had a small round face on the inside facing face of the disk to place a support along it. With this internal support in place the tube was chucked into the lathe with the previously mentioned cut aluminum jaws and the tail stock holding the disk that was inside the tube. The tube was rotated and if any lateral motion was seen from the tube while rotating the chuck was re-opened and the tube was re-set. The lateral movement was still noticeable so an external chuck/support was used at the far end of the tube to minimize the lateral movement. Once the tube was rotating nicely the free end was faced to get a perpendicular face to the tube. Then the tube was removed and the supports were placed on the clean side and re-chucked. With the piece re-chucked the piece was measured using the lathe display and cut down to the correct size. With the clean perpendicular edges and the cut jaws the tube was now rotating true with the supports inside. The outside of the tube was lightly faced to remove some of the markings left by the external chuck/support. Before anything else was done to the tube, the two end caps were pressure tested, to check for any leaks.

Once the end caps were pressure tested, the tube was placed on the mill using the turntable on its side and the tail stock with the piece of aluminum to get it to the correct height. A flat spot was milled on the top of the tube with a small protruding rectangle. Then a through hole for the communications mast was made.

Once all the pieces for the pressure vessel were made the set screw through holes were made on the EPV.

4.1.1.4 Servo Pressure Vessel

Figure 44: Dietrich Lathe

The Servo Pressure Vessel (SPV) cut and lathed out of a 5” solid rod of 6061 aluminum alloyed and details were added on the mill. The 5” solid rod was chucked into the normal chuck of the lathe. It was faced and the exterior diameter was lightly faced to make sure the facing was perpendicular to the rod. The piece was turned around in the chuck and the other side was faced and cut down to size. The exterior that was not lightly faced was now lightly faced and the section of the external pressure vessel that held the O-rings was turned down. The O-ring grooves were then made to specification. The inside of the piece was then bored out to the required diameter for the O-ring section of the SPV required by the section that supported the shafts, PTFE ring seals and the bushings. The piece was then turned around and chucked onto the cut aluminum jaws and some of the material was removed to reduce weight on the tail section of the vehicle.

The shaft holes and the seal seats were placed using the mill. The SPV was then placed on the mill using the turntable chuck placed vertically. The disk with the lip was used to support the other side of the SPV with the tail stock. The hole for the shaft was drilled first. Then without moving the position of the SPV the inner face for the dynamic seal was created and finally a flat section for the bushing was created. The SPV was rotated 180 degrees and the opposite end was created. This was repeated for the other shaft that was perpendicular to the first and offset a distance away.

Once that was done the entire pressure vessel was put together, aligned and mounted on the mill with the turntable chuck on its side. One of the disks was placed on the tail stock section to add support through pressure. The set screws were then drilled carefully as to not go through all the way and then threaded. The internal edges of the through holes were then broken for all of the pressure vessel parts with a tapered bit, by hand. The entire vessel was then put into the lathe and the exterior diameter was lightly faced to get a nice shine on the aluminum. The edges of the O-ring grooves were then slightly broken with a file while turning the SPV on the lathe.

4.1.1.5 PCB Sled

The PCB Sled was fabricated from a piece of fiberglass found at the Boca Raton campus machine shop. The sled was created by cutting the fiberglass down to size using the circular saw and then placing it on the mill to clean up the edges and create the screw holes for the different components. When placed in the chuck on the mill the board was placed on parallels to get it close to the chuck top. Supports were created from a two by four. These supports held the protruding sections of the board to prevent bending while drilling holes at the required distances.

4.1.1.6 Battery Supports


Figure 45: Mill

The battery supports were created from high-density polyurethane by cutting blocks with the band saw and then placing them into the CNC. Scraps of HDPE were used to cut out blocks that were slightly larger than required for the three battery supports. A fourth block was used to create a die to attach the piece to the CNC securely while allowing the piece to be cut into the required shape. The drawing for the battery support was converted into G-code and sent to the CNC for milling. The code would first cut the outer diameter of the piece and then the piece was secured with screws along the perimeter to cut the inside. This was done three times one for each support. Then the supports were taken to the mill and the screw holes were open and then threaded by hand.


Figure 46: Battery Plug

4.1.1.7 Battery Plug

The battery plug was made from black Delrin by cutting a slightly larger block then necessary and placing it into the CNC, it was later fabricated from clear acrylic to make it easier to put in. The plug was created by cutting a slightly larger square piece of the material and placing the material into CNC. The G-code for the battery slot, sled mounting slots and the outer diameter were cut. The piece was then taken to the lathe and the O-ring groove was cut.

4.1.1.8 Servo Mount

To make the servo mount an aluminum 3”x3” L-channel was machined on the mill. The holes for the servos were opened. A test bench was created for the mount so that the fin actuation could be tested beforehand. The test bench was made of left over pieces of HDPE and fiberglass, put together with screws. Once the servo pressure vessel was created the servo mount was cut to size and placed into the pressure vessel.

Figure 47: Servo Mount Block

4.1.1.9 Servo Mount Block

To make the servo mount block a piece of aluminum was cut on the band saw, made parallel on the mill and then put into the CNC. A left over piece of aluminum was cut slightly larger than the block. The piece was then placed into the mill and the sides were made parallel. Then the piece was placed into the chuck of the CNC. The G-code for the bracket was created and sent to the CNC. The CNC then drilled the holes for the screws. The piece was then de-chucked and a die was made for the piece. The die consisted of the holes for the screws being drilled into a scrap piece of aluminum. The aluminum piece was then bolted down to the die and the rest of the G-code cut the outside of the piece and then dug out material from the piece to reduce weight at the tail section of the vehicle.

4.1.1.10 Thruster Mounting Bracket

The thruster mount bracket was made by cutting a piece of aluminum on the band saw and machining it on the mill. The piece was cut a bit larger on the band saw. Then it was placed into the mill and the cut horizontally. Then it was placed on the mill vertically to make the hole pattern to attach it to the rear end cap and finally it was cut down to the required length.

4.1.1.11 Tie-down

The tie down was made from a 3/8” stainless steel eye bolt. The bolt was taken down so that the nose cone was compressed by the eye. It was then placed into the lathe so that a little of the taper of the eye bolt was removed so that the nose cone could rotate freely allowing the weight added to the bolt to rotate and not interfere with the SEACON connector. The weight was added to the Tie-down by cutting a piece of lead to the required weight and then pushed into the cone to take its shape.

Figure 48: Nose Cone

4.1.1.12 Nose Cone

The nose cone was printed on a 3D printer using PLA. The drawing was saved as a .stl file and turned into g-code using Cura with the opening of the nose cone to sitting on the table. No supports were created for this part in the g-code. The g-code was then loaded into a Luzbot Taz 4 and printed.

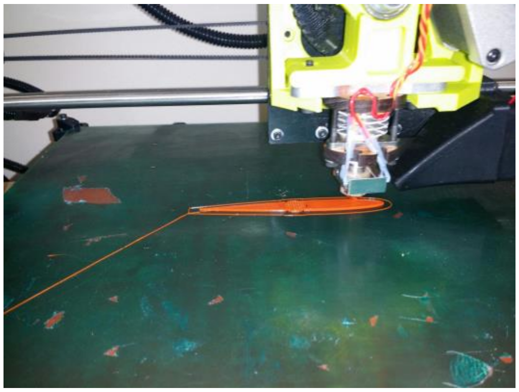


Figure 49: Control Surfaces

4.1.1.13 Control Surfaces

The control surfaces were printed on a 3D printer using a Thermoplastic Polyurethane called Ninja-Flex. The drawing was saved as a .stl file and turned into g-code using Cura with the larges chord of the control surface sitting on the table. The g-code was then loaded into a Luzbot Taz 4 and printed.

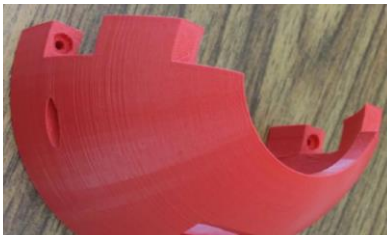


Figure 50: Thruster Shroud

4.1.1.14 Thruster Shroud

The thruster shroud was printed on a 3D printer in two parts using PLA. The drawings were saved as .stl files, turned into g-code using Cura and printed horizontally on the table one at a time. The pieces were then then filed because the thickness of the PLA bead was not taken into consideration when printing and the screw holes were threaded or opened to be through holes where needed. Foam was added into this section for added buoyancy.

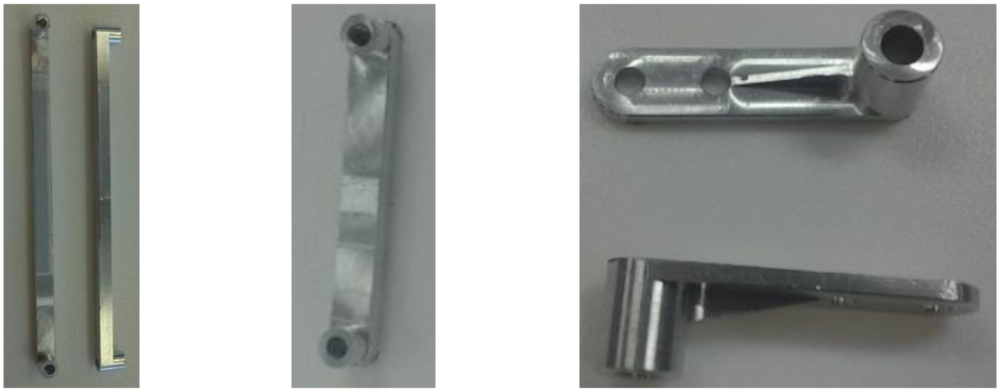


Figure 51: Servo Linkages

4.1.1.15 Servo Linkages

The servo linkages were created using aluminum scraps cut by the band saw and placed into the CNC. Before the linkages were made they were first printed out of PLA on a 3D printer to check functionality. The piece were saved as .stl files and then converted to g-code using Craftware. The pieces were printed on a CraftUnique CraftBot. Once the functionality was tested the drawings were turned into g-code for the CNC and then transferred to the CNC for printing. Then the pieces had to be cut apart using the mill and then shaved down to their final dimensions.

4.1.1.16 Control Surface Shafts

The control surface shafts were created from ¼" stainless steel rods that were cut using a hacksaw and then machining it in the mill. The shafts were first cut down to their size and then grinded down to a cleaner surface with the diamond wheel grinder. The shafts were then placed into the mill and the control surface screw holes were placed along with a flat surface on the rod for a set screw to hold the orientation of fins relative to the servo linkages.

4.1.1.17 Bushings

The bushings were purchased from McMasterCarr.com. They needed to be slightly smaller to fit with into the servo pressure vessel with the PTFE seals. They were grinded down using the diamond wheel grinder.

Figure 52: Mast Assembly

4.1.1.18 Communication Mast

The communication mast was created by making the bottom plate, the mast tube, and the mold used for potting. The bottom plate was drawn on SolidWorks, turned into g-code machined on the CNC. Three screw holes on the bottom plate were threaded and three screws were placed into the bottom plate to act as support for the potting. The mast tube was made from a stainless steel tube that was threaded by placing the tube on the lathe and turning a die while applying pressure to the die with the lathe's tail stock. The mold was created by making a negative of the required mast in SolidWorks. It was then converted into g-code and sent to the CNC for machining. The mast tube was placed into its hole in the bottom plate. The bottom plate was placed in its position on the mold. The cables for the GPS antenna and the XBee antenna were placed through the mast tube. A GPS antenna holder was printed using the CraftBot to hold the GPS antenna to the XBee antenna, so that potting completely encapsulated the GPS antenna and its connectors. The XBee antenna sat on its groove on the mold to keep its connectors completely potted. The cables were then tied down to the antenna using thin gauge steel wire. The mold was then closed and potting was injected into the mold through an NPT nipple with a hose using a caulking gun. A hole was left on the bottom section to make sure that the bottom of the mold was filled. Once the potting compound leaked out of that hole the hold was covered with a screw. Then when the potting was coming out the mast tube and leak hole on the mast bottom plate it was done. The mold was left to cure for 24 hours. The same 3M Scotchcast, Flame Retardant Compound 2131 that was used to pot the thruster cable.

4.1.2 Assembly

Assembling the AUV happens in three different sections: Bow Section, Servo Section and Thruster Section. The largest of the sections is the Bow section, it is comprised of most of the electronics, the front end cap, nose cone, tie-down communications mast and battery assembly. The Servo Section is the most difficult section to put together because the linkages need to be connected in a certain order so that they can be screwed together. This section is comprised of the servo assembly, linkages, shafts and control surfaces. The thruster section is comprised of the thruster bracket, thruster shroud, foam and the actual thruster.

4.1.2.1 Bow Section Assembly

The bow section assembly starts with the battery assembly. The battery assembly is comprised of the battery supports, batteries, electronics board, TCM2 Compass, motor controller, leak sensors and the PCB sled. The battery supports are first screwed onto the PCB board using ½” 4-40 screws. ¼” nylon standoffs with male threads and female threads are then attached to the PCB board using the male threads. The female threads are used in conjunction with 3/16” 4-40 screws to attach the electronics board, TCM2 compass and the motor controller.

After the battery assembly is together the front end cap is put into place. The front end cap holds a few things: the pressure sensor, two sled brackets, SEACON connector, reid switch and the reid magnet holder. The pressure sensor is screwed into the pressure sensor hole, each sled bracket is screwed into the front end cap with a ¼” 8-32 screw. The SEACON connector is put through its through hole and then attached with its washer and its nut to the front end cap. The reid switch is attached to a small breakout board and attached to the front end cap with two ¼” 4-40 screws. The reid magnet holder is held in place with two ¼” 4-40 screws on the front side of the front end cap. Two 246 O-rings are lightly greased with silicone and placed in the front end cap O-ring grooves. The front end cap is then attached to the PCB sled using the sled brackets and ¼” 4-40 screws. Once the board is together mechanically the individual electronic items are plugged into the electronics board.

To finish the bow section the mast assembly, electronics pressure vessel, battery assembly, front end cap, nose cone, and tie down are put together. The mast assembly cables are run through a lightly greased 010 O-ring and then the electronics pressure vessel through hole, the wires are then run through the mast assembly spacer, washer and nut so that the nut. Before the nut is tighten the mast bottom plate is set into place by lining up the protruding notch on the electronics pressure vessel with the indent on the bottom plate. Once the nut on the communications mast is tighten the cables are attached to the top of the pressure vessel with glue or tape. Before the battery assembly is slid into the electronics pressure vessel the two leak sensors, pressure sensor and reid switch need to be attached to the electronics board. Once connected the battery assembly is slid into the electronics pressure vessel and the front end cap and its O-rings are slid into the electronics pressure vessel while lining up the set screw holes. The six ¼” 6-32 set screws are then put in to close the front of the bow section assembly. On the back side of the bow section assembly connect the GPS antenna and the XBee antenna into the electronics board. Put the battery plug in, making sure the PCB sled notches and the battery fall into their slots. Pull out the thruster and servo cables and the servo leak sensor.

4.1.2.2 Thruster Section Assembly

The thruster section assembly was put together before the thruster cable was potted. The thruster bracket mount was attached to the rear end cap with four ¼” 4-40 screws. Then the thruster bottom cap was slid through the thruster bracket mount about half way, then screw in the two thruster screws closest to the rear end cap. Finish sliding the thruster bottom cap towards the rear end cap and attached it with two 1” 4-40 screws. Screw in the last thruster mount screw. Place any necessary foam in between the thruster and the bottom thruster cap. Close the thruster shroud by placing the top thruster cap onto the bottom thruster cap and screw them together with a ½” 4-40 screw.

4.1.2.3 Servo Section Assembly

The servo section assembly has a few assemblies: the servo assembly and the control surface assemblies. The servo assembly is comprised of the two servos, the servo bracket and the servo block. The control surface assemblies are made up of the control surface shafts, shaft arm, servo linkages, shaft collars, PTFE ring seals and bushings.

The servo assembly is set up outside of the servo pressure vessel. The servos are attached to the servo bracket with ¼” 4-40 screws. Then the servo bracket is attached to the servo mount block with three ½” 4-40 screws. The servo horns are then placed on the servos and screwed down with the servo screws. The servo assembly is then attached to the rear end cap with 3, ¾” 4-40 screws. Guide wires through the cable tie mounts and screw it in place.

Setting up the control surfaces correctly is crucial so that the system does not leak. First attach the servo linkages to the shaft arms using the ¼” truss headed 4-40 screws. Then slide the dive plane control surface shaft through the first penetration. Slide a nylon washer, a shaft collar, control shaft arm, shaft collar and nylon washer on to the shaft in that order. Slide the shaft through the second penetration. Place one fin onto one side of the shaft to guide the location of the shaft collars. Use a hex key to tighten the shaft collar as close to the servo pressure vessel wall as possible, then tighten the other shaft collar as close as possible to the other wall of the pressure vessel. This will prevent the shaft from moving axially and prevent the PTFE seals from moving out of place. Then place the hole of the shaft arm over the flat surface of the control surface shaft and tighten the 6-32 set screw with a hex key. Remove the fin. This process is then repeated for the rudder control surface assembly.

Once the control surface shafts are in place the external part of the assembly can be put in place.

Carefully slide a PTFE ring seal around one end of the shaft. The seal should be oriented with the metal spring towards the outside. Then with new un-altered bushing gently push the PTFE ring seal until it is seated inside the groove. Remove the un-altered bushing and place a shorten bushing until it is completely seated. This is then repeated for the other three shaft ends. Once the bushings have been placed the fins are placed on the shafts and the screws are put in until they lightly squeeze the flexible material of the fin.

Putting together these two assemblies can be tricky. First lightly grease two O-rings and place them on the rear end cap O-ring grooves. Then place the servo assembly inside the servo pressure vessel. Have another person hold the servo pressure vessel up so that you could place the servo linkages into the servo horn holes and screw them in place with the ¼” truss head 4-40 screws. This part can be difficult and might take a few tries but it is possible and has been done more than a handful of times. Then make sure the fin orientation is correct as you slide the pressure vessel onto the O-rings. Turn the assembly around with the thruster in the air so that pressure could be applied to seal the rear end cap against the servo pressure vessel. Pull the servo and thruster cables out through the third quadrant so that they do not interfere with the linkages. Put the 6-32 set screws into place to finish sealing the aft end of the vehicle.

With the three assemblies now two, closing up the system is the next step. Make sure the program is loaded and that cables are not in the way of the pressure vessel closing. Lightly grease two O-rings and place them in the servo pressure vessel O-ring grooves. Connect the thruster and servo cables and put the two assemblies together. Once the first O-ring goes past the set screw through holes a vacuum can be placed on the system using a vacuum pump. Using a 1/8” NPT nipple attach the vacuum pump to the vehicle. Turn on the pump and let the vacuum gauge read 23 psi. Remove the hose from the nipple and cap off the nipple as fast as possible. Unscrew the nipple while capped off with your finger. Have someone else put Teflon on the NPT plug and be ready to cap off the hole when the nipple comes off. Tighten the plug as tight as possible. Then line up the set screw holes on the servo pressure vessel and screw in the set screws. The final touch is adding the nose cone. To put the nose cone on make sure that the lead weight is seated correctly inside pass the tie-down through the nose cone and weight place a washer and spring in place and then tighten it down with a nut. Then screw the nose cone and tie-down into the front end cap and Tito is ready for a swim.

4.1.2.4 Printed Circuit Board

The PCB for the AUV was laid out using PCAD. Any circuit design program such as Eagle can be used for the schematics and layout of the PCB. The PCB layout then needs to be sent to a printing company. The board will then need to be assembled with all of the electrical components and standoffs. The connector pins all need to be soldered into their correct placements. The resistors and capacitors all need to be mounted in their respective places. Any sensors or other components such as the relay and voltage regulators that are board-mounted also will need to be soldered into place. The headers and receptacles all need to be soldered through the board as well.

4.1.2.5 Molex Connecotors

The Molex connectors will have to be handmade. The wires will be first cut to the proper size length, making sure to leave some extra for error. The ends of the wires then need to be stripped. Using the proper gauge size wire strippers about 1/8 inch of the insulation should be stripped off each end. Then the stripped wire is placed into the female end of the wire connector. A compression tool made for the specific connector is then used to seal the crimps on both the connector and the insulation. The wire connector is then snapped into place in the plastic Molex connector. This will be repeated for all of the wires on the connector, and all of the other Molex connectors.

5. Testing

Fabrication and assembly of Tito Tuxedo consisted of numerous discrete tasks to mitigate risk of project failure. Testing was crucial in gaining confidence that failure was extremely unlikely. Every component was tested and verified before integration. In addition, if a component needed integration such as the GPS and XBee in the mast the point of integration was tested thoroughly. There were many pressure tests at different stages of the fabrication to gain confidence one by one in the features of the exterior of the vehicle. Once confidence was gained in the shell, then the electronics were integrated for the full system. Once the electronics were integrated they were bench tested. Bench testing consisted of pressure sensor manipulation, servo manipulation and thruster characterization. After significant bench testing, penguins got wet in the ocean and the marina. After testing trim, Tito was capable of diving. After diving we tested heading control. After multiple iterations of testing in the water Tito was ready for the ocean. The Flying Penguins were constrained for time in this endeavor. Due to our constraint, we needed to test early and often. The personnel who were required for testing are listed in (4) and when the entire group of students were required they are collectively referred to as Penguins.

Table 4: Personnel

Name	Position	Roll
Pak-Cheung E An	Professor & Director	Senior Design Mentor
Douglas A. Briggs	Captain On-Call	R/V Oceaneer Captain
Robert K. Coulson	COORDINATOR,RES PROG/SVCS	Testing Equipment
John Charles Frankenfield	Technical Paraprofessional	AUV Specialist
Edward A. Henderson Jr	Technical Paraprofessional	Electrical Testing
John R. Kielbasa Jr.	SENIOR ENG TECHNICN/DSGNR	Electrical Testing
Anthony E. Lavigne	Machinist	Fabrication
Stacey Darin	Student	Purchasing, Electrical
Quintin Du Plessis	Student	EHS, Software
Pedro Muslera	Student	Machine Shop, Mechanical
Steven Serbun	Student	Budget, Software
Chris Sullivan	Student	Electronics Lab, Electrical
Dietrich Vogel	Student	Progress, Mechanical

Tito had to be fully operational; assembled, tested, quantified, and proven by April 15th 2015. Tito was wet testing on March 25, 2015; 26 days before the deadline but 5 days after we had ambitiously planned.

5.1 Individual System, Subsystem, Component Testing

The following details the tasks and tests with brief results and remarks. The overall system test plan consists of benchmarks for system completion, methods for testing the complete system, and necessary conditions for testing. At this point in the design process, the test plan and the actual test data can be compared. A comparison of the plan and the actual test will demonstrate how well we estimated testability. Each system test plan includes a description of the construction that was necessary to test, qualifications for test success, and deadlines for test completion. Below is a brief summary of higher level testing that was to be conducted at the corresponding dates. Latter is a discussion of the results.

Table 5: Testing Table

Day	Test	Location	Equipment	Outcome	Personnel
Aug					
18	Arduino Due and motor shield implementation	Laboratory	PC, Arduino Due, Adafruit Motor shield, USB cable	Upload code successfully	QD
21	Weigh AVP as AUV payload option	Laboratory		Establish feasibility of AVP payload by measuring weight and dimensions	Penguins
Sep					
10	Remus 100 Side-scan Sonar Data Collection	Sea State 1	R/V Oceaneer, Remus 100, PC	Record Side-scan Sonar data	Penguins, Mario, Boat crew
Oct					
10	Remus Depth Data Collection	Dania Marina	Remus 100, Tow Fish	Depth Data for Comparison	Penguins
21	Drag calculation check	Dania Marina	Cylinder with end caps, Tie-down, Steel Cable, Hand scale	Calculations were feasible	Penguins
24	TCM2: Initial setup and operation	Laboratory	TCM2, PC, RS232 connection, Multi-meter, Power supply	Display output word to an onscreen terminal	QD, J. Kielbasa

27	Thruster Test	Fountain	Seabotix Thruster, Power source, Lever arm, Rod, Hand scale	Thruster current and Force	DV, PM, SS, QD
31	TCM2: Field test	Outside, clear	TCM2, PC, RS232 connection, Multi-meter, Power supply	Display output word to an onscreen terminal, in a mobile setup	QD
Nov					
Dec					
	Xbee: Basic multi unit communication	Laboratory	2x PC, 2x Arduino Due, 2x USB cable, 2x OE lab board with Xbee	Send commands between 2 Xbee systems	SS, N.Burleson, E. Henderson
	GPS: Arduino Data collecting	Laboratory	PC, Arduino Due, USB cable, OE lab board with GPS	Display GPS location output word to an onscreen terminal	SS
	GPS: Field Test	Outside, clear	PC, Arduino Due, USB cable, GPS , GPS antenna	Display GPS location output word to an onscreen terminal	SS
Jan					
5	CNC milling machine test run	Machine Shop	ABS, Tools	Gain confidence in machining and the CNC operation	DV PM
8	Materials test	Boca Raton	G10 and cutting tools	Test the G10 material for strength and machinability	DV
10	TCM2: Data parsing	Laboratory	PC, Recorded TCM2 data	Matlab output: Heading, Pitch, Roll	QD
12	IMU: Collect data for error modeling	Laboratory	PC, Arduino Due, USB cable, OE lab board with 10DOF IMU	Plot error in measured heading vs. actual heading and show Soft-and-Hard Iron effects	QD, K. Heilije

12	Thruster Bulkhead Tests	AUV lab	Different AUVs, Different options for potting	check the feasibility of a pass through thruster potting	PM
13	IMU: Field data collection for error modeling	Outside, clear	PC, Arduino Due, USB cable, OE lab board with 10DOF IMU	Plot error in measured heading vs. actual heading and show Soft-and-Hard Iron effects	QD, SD
14	TCM2: Cal3 Calibration	Laboratory	TCM2, PC, RS232 connection, Multi-meter, Power supply	Use TCM2 commands for device calibration	QD
15	Servo mount bracket test	Machine Shop	Servo Mount Block, servo motors	Place the servos in the bracket to ensure tight proper fit	DV PM CS
16	SyRen10: Initial operation	Laboratory	Seabotix, power, motor controller, GOD	Implement code to control the thruster	SS
16	servo test stand	machine shop	servo bracket, servos, plastic, g10	created a stand to test the servo links and the construability of our linkage technique	DV PM
17	TCM2: Matlab integration	Laboratory	TCM2, PC, RS232 connection, Multi-meter, Power supply	Stream live data from TCM2 to Simulink model	QD
19	SyRen10: Arduino controlled thruster	Laboratory	Seabotix, power, motor controller	Streamline code to reflect responsive control	DV SS
27	Batteries: Charging and Discharging	Laboratory	Power Supply, Batteries, Multimeter, Infra Red Thermometer	Controlled and timed charge and discharge batteries	QD, SS, SD, CS, E. Henderson, J. Kielbasa
27	Initial pressure test	Shipping/rec	Nose cap, end cap, and	pressure tested the main tube on	PM

			electronics vessel	the surface by submerging the tube	
28	Weight check	corrosion lab	scale, caps, pressure vessel	Compare the weights of the finished parts with the designed parts	DV PM
Feb					
3	pressure vessel test	shipping receiving	Pressure Shell, Pressure Chamber	Pressure test to 50m then leave overnight and check for leaks	DV PM
4	Surface Pressure Test	Machine Shop	Trash can, Water, Electronic Pressure Vessel with End caps	Slight leak from plug, (Replaced)	DV, PM
5	Pressure Chamber Test	Pressure Cham.	Pressure chamber, Lead weights, Buoy, EPV with end caps Over Night	EPV did not leak	PM
10	Servo Linkage Range Test	Home	3D Print and test Servo linkage movement and range	40 deg of range	PM
10	Log Navigation Data	Laboratory	PC, OE Lab board	Successfully logged navigation data collected on OE Lab board	QD
11	Pressure Vessel: Air pressure	Laboratory	Air Compressor, Assembled AUV Pressure Vessel	Shows no leaks at 50m depth simulation	PM, D. Vogel, Tony
12	Pressure Vessel: Water pressure	Laboratory	Seatech Pressure tank, Assembled AUV Pressure Vessel	Shows no leaks at 50m depth simulation	PM, D. Vogel, CS, Tony
13	PCB	Laboratory	External Power, Assembled PCB, Sensors	Powered on the PCB and monitored all test points. Test sensor signals.	QD, SD, CS, Henderson, J. Kielbasa

14	Navigation hardware	Laboratory	PC, External Power, Assembled PCB, Sensors	Collected navigation data from PCB	QD, SS
19	Rudder Control	Laboratory	PC, External Power, Assembled PCB, Sensors, Actuators	Control the rudder with error signal from TCM 2.	QD, SS
20	Rudder Control	Laboratory	PC, External Power, Assembled PCB, Sensors, Actuators	Control the rudder with error signal from TCM 2.	QD
23	Mechanical Fit	Machine shop	Pressure vessel, Servo vessel, both end caps	fit together and place setscrews in place. No bends or breaks and the seam is invisible	DV
27	Battery System	Laboratory	PC, External Power, Assembled PCB, Batteries	Power PCB via batteries	QD
Mar					
3	Mast Wire fitting	AUV lab	Mold, and a tab to secure GPs and xbee	The mast wires fit perfectly in the mold	PM
3	Radio	Laboratory	PC, Xbee, Xbib	Serial communication between PC and PCB	QD, SS
4	Test bench	E-Lab	Compile all electronics and mechanics relating to electronics	tested that the electronics section is fully integrated meaning self powering and all components are on board	DV CS
4	Batteries	Laboratory	AUV, Batteries, PC	Run full system on battery power for an hour	QD
5	Bench testing	E lab	Electronics bench thruster and servos	servos move and data can be communicated	DV QD

				quickly	
6	GPS	Laboratory	PC, AUV	Communicated GPS data over radio signal	QD
7	Data Transmission	Laboratory	PC, AUV	Transmit health data from AUV to PC	QD
12	Mast pressure test	Ship/Rec	Exterior shell	successfully pressure tested the mast passing on the vehicle to 75psi or 50m	DV PM
11	External Power Cord	Laboratory	PC, Power supply, Multi-meter, External Power Cord, AUV	Successfully ran AUV systems and charged batteries via. External power cord plugged into Seacon connector with lab power supply	QD
14	Mechanical fit test	machine shop	end cap w/ thruster, servo section, servo block, servo bracket	successfully mount servos on the bracket then the block then to the end cap, and fit into the servo section	DV PM
16	Assembly test	AUV lab	Thruster section, communication section, electronics section	successful test of the integration between all parts of the vehicle. Full assembly	penguins
16	pressure test	Ship/rec	Assembled AUV with out Electronics	failed pressure test(@80PSI) with ring seals due to them being shaven upon installation	PM DV
18	pressure test	ship/rec	Assembled AUV with out Electronics	Passed pressure test after replacing and properly installing the seals	PM DV

18	Marina 2, surface and 1m depth	Dania Marina	Kayak, PC, Tito Tuxedo, battery charger, Travel Case	tested and changed the trim of the vehicle with a few weights in the nose cap	PM DV
19	more pressure testing	ship/rec	Assembled AUV with out Electronics	after Vacuuming the pressure vessel, tested over weekend at 75 psi and passed.	PM DV
23	Vacuum electronics test	E lab	Assembled AUV with electronics	successful vacuum and test of the internal health monitoring system	SS PM DV CS
24	wave tank movement testing	wave lab	fresh water, Tito	travel in the water and monitor any tilt or roll generated by the thruster	Penguins
25	ocean testing	ocean	Tito x bib pc	successful communication at range, successful health monitoring, successful surface launch and recovery	QD & SD
25	Ocean speed characterization	Beach	Tito, xbib, pc	message sent from pc to Tito, Tito executed the command (run thruster for specified intervals)	Penguins
26	Ocean Propulsion	Beach	AUV, PC	Successfully ran the AUV under thruster power via the PC and receiving data to PC	Penguins

27	Ocean Remote Control	Beach & marina	AUV, PC	Successfully ran the AUV remote controlled via the PC and receiving data to PC making turns, dives, forward reverse	Penguins
28	Turning testing	Marina	Tito, PC	focused on diving and characterization of the dive speed estimated 1m in 5 seconds	Penguins
31	Pressure test	Ship Rec	Tito without electronics	failed a pressure test due to being dropped on the ball valve	DV PM CS
Apr					
1	Pressure test	ship rec	Tito without electronics	Replace ball valve and pass overnight pressure test with boss plug	PM DV
2	pressure sensor mapping	E-Lab	Electronics section	install new pressure sensor and test its function to ensure proper mapping	SD
2	Fully Assembled AUV: Bench 2	Bench	PC, Tito Tuxedo, battery charger	Complete simulated mission 1,2,3 on the surface. Move it around the lab and check heading and depth control	Penguins
2	Marina speed characterization	Marina high tide	Tito PC	characterized speeds, 1700=1.5m/s 1600=1.1m/s and dive angles. Good straight line underwater travel	Penguins

4	Marina Surface test 1m depth	Dania Marina	PC, Tito Tuxedo, 2xKayak,	Surface test, Reverse, select diving control	Penguins
7	Beach, surface and 1m depth	Dania Marina	PC, Tito Tuxedo,	Linearize servos to even the ability to turn for both sides	Penguins
9	Beach, surface and 1m depth	Dania Marina	Kayak, PC, Tito Tuxedo	2 m dive, with 2 headings at slow speed	Penguins & Advisors
9	Marina, surface and 1m depth low tide	Dania Marina	Kayak, PC, Tito Tuxedo	Characterize speed and turning	Penguins & Advisors
10	Fully Assembled AUV: Sea Test, surface and 1m depth	Coastal Ocean	R/V Oceaneer, PC, Tito Tuxedo, battery charger, cuum pump, wrenches	Complete mission 2,2,2,3 recover, charge,3	Penguins & Rob C

Each component was tested then implemented into its sub system. Then sub systems were integrated into systems and finally the systems were integrated together for a final product. The most crucial test points were the mechanical shell pressure test once all of the features were refined (ie the mast, the boss plug, the dynamic seals) on 4/1. That was the point where all of the mechanical sub systems came together for a flawless test. Each component could not be tested on its own, but the sub system of the shell when the nose cap, communications section, thruster section components all meet the sub system takes a testable form.

The electronics sub system bench test on 3/11 where every electronic component was powered and integrated into the sub system. The electronics sub system section was tested first as standalone components, then one by one they were added. First, the PCB stood alone. Then we added the microprocessor, then we added battery power, then we added sensors, then we added output devices (ie motor controller). The construction and integration of parts was slow and iterative.

5.2 Systems Integration Testing

5.2.1 Pressure Vessel Testing

To start testing portions of the AUV body subsystem, unfinished end caps and the electronics pressure vessel were put together to make sure that the O-ring seals would seat correctly. Originally it was planned to have positive pressure inside the pressure vessel, this was to reduce risk by being able to see leaks in the system through bubbles. The system still did not have any set screws to hold the end caps in place yet and had minimal openings (no SEACON, thruster, or pressure sensor openings), and the system was being positively pressured it had to be kept together with a ratchet strap. This subsystem was placed into a shallow water tub and tested overnight. The subsystem was opened checked for water and there was none inside the vessel.

The system was prepped and closed for its next test in the pressure chamber, the test was successful. The pressure chamber was turned on by Mr. R. Coulson. The pressure chamber controller was not used this time around; it was set to city water pressure, about 50 psi and raised to 75 psi regularly, every 15 minutes until the end of the work day. In the morning the pressure chamber was raised to 75 psi and then turned off and opened. The subsystem was retrieved, opened and check for water. It was dry, the subsystem passed with flying colors confirming that the O-rings in the end caps and the vacuum plug were made to specification. Once the servo pressure vessel had O-ring grooves but before the shaft holes were opened the system was checked again for leaks. This was to integrate a new variable and piece into the subsystem without adding too much complexity. The dynamic seals placed around the shaft were thought to be a high risk leak point and if possible they were to be tested by themselves. The subsystem was prepped strapped up and pressurized and placed in the shallow water test bucket in the morning, right outside the machine shop. When it was checked midday a slow stream of bubble were noticed. The bubbles did not come from the new O-rings in the subsystem but from the vacuum plug. The plug that was originally used was not meant for the tapered NPT hole created, it was replaced and the subsystem stopped leaking air. The subsystem was removed from the shallow water test, dried, opened and inspected for leaks. The subsystem passed the shallow water test.

The subsystem was then prepped and placed into the pressure chamber, but this time the outcome was not so dry. After a three hour test in the pressure chamber being pressurized to 75 psi whenever possible, the system leaked. When the pressure chamber opened the subsystem had sunk to the bottom of the chamber and had to be pulled out by the buoy attached to the subsystem. While opening the system about a quart of water rushed out of the pressure vessel. The floor, cart and the subsystem were dried and the subsystem was prepped, pressurized and sealed. The system was left pressurized overnight to make sure the air did not leak out of the system. With a clear minds the next morning the system was checked. It had held pressure. After some thought, it was realized that before placing the system into the pressure chamber the plug was not tightened sufficiently. The previous plug had an O-ring and did not require for the plug to be cranked down, the new plug sealed because of a taper on the actual plug and hole that pushed against each other. The subsystem was re-pressurized, the plug was tightened correctly and placed into the pressure chamber again. When the chamber was opened the subsystem was still floating. When the pressure vessel was opened, it was dry.

Now that the subsystem was checked and the low risk O-rings had passed the test, it was time to open more holes in the pressure vessel. The holes for the SEACON connector and the pressure sensor were opened and the subsystem was checked again. The system passed without any issues. For good measure the SEACONs old O-ring was changed out after the test.

To keep moving forward in with the mechanical construction of the vehicle, the full pressure vessel was placed into the mill and the set screw holes and the communications mast though hole were opened. They were opened together to make sure that they lined up, but this meant that there would be no more pressure testing until the mast was created.

To confirm that potting the mast from a mold and that the components would fit into the desired mast half of the mast was 3D printed along with a shelled version of half the mast and a potential half of a mold for the mast. The shelled version was to see if it was possible to create a print that could be potted inside, this turned out to be too small. The components did not fit into the shell. They did fit into the mold, but the components would have been tight would possible touch the walls of the mold while being potted, leaving gaps in the potting that would allow this new subsystem to leak. This check lead to slightly scaling up the mast and creating a small piece to hold the GPS antenna to the XBee antenna, so that the only thing with external contact was the XBee antenna, which was water proofed. The new mold and pieces for the mast were created. This took some time and during this process the servo pressure vessel was finished and the through hole for the potting of the thruster cable were created. The communications mast seal, like the dynamic seals on the shafts, was an area of concern.

Once the communications mast, thruster cable potting and the shafts were ready to be added to the subsystem some of the team member were anxious to see the full subsystem perform in the pressure chamber. One of the mechanical team members set up the system hastily and put the dynamic seals in incorrectly. The subsystem was then prepped and instead of doing a surface test because it was late in the day, the system was placed straight into the pressure chamber. By this time the team had learned how to operate the controller. It was set for 75 psi and left running over night. In the morning the pressure chamber was opened and the subsystem was still floating. The system was dried and opened and there was two pints of water in the pressure vessel.

The issue was discussed, Mr. J. Frankenfield he recommended making a tool to push the dynamic seals in. He also recommended using a ball valve so that it was possible to know how much pressure was being introduced into the system. New PTFE seals were ordered to replace the ones put in hastily and damaged. The holes where the seals were pushed in through were chamfered slightly and the shafts where sanded and cleaned up to allow a smooth entrance for the seals and it was decided that the new seals would be lightly greased and put in with a new bushing and a tool that would push the bushing down evenly.
While the new seals were being shipped the ball valve was purchased and put to use. The team now remember to be patient with their testing and reverted to testing parts of the subsystem. The electronics pressure vessel with the communications mast and the end caps now completed were put together and sealed this time with a vacuum, to hold the rear end cap in place since the set screw holes were not meant to attach the rear end cap to the end of the electronics pressure vessel. This subsystem was left over night to check if it kept a vacuum. In the morning when it had passed the vacuum test it was placed into the pressure chamber and tested for two hours. The subsystem was not completely dry, there was about an ounce of water in the system. This was a much smaller leak than before. It was noticed that the ball valve was not completely tighten, so it was then tightened some more. The subsystem was then again placed into the pressure chamber this time over night. The next

morning the subsystem was pulled out and it was dry.

When the new PTFE seals arrived they were installed carefully and the full pressure vessel was re-assembled and pressurized. At this time, the fountain was up and running and the subsystem surface test was performed in the fountain. The system was placed into the fountain and carefully observed by the mechanical team. One of the members said he noticed a very small slow leak from the mast but it was not confirmed by the other member. The subsystem was placed into the pressure chamber over night at 75 psi. The next morning the subsystem was carefully retrieved and opened. There was a few drops of water in the system this again. This time about 3 ounce of water. The system was dried, re-pressurized and checked again in the fountain. A slight stream of bubbles was seen along one of the shafts. The shafts were then torqued and the stream increased. This was discussed with Mr. Frankenfield, and he agreed that the positive pressure could be pushing the seals out and that a vacuum should be pulled to set those seals in place and increase the pressure differential when external pressure acts on the pressure vessel.

Now with a vacuum on the pressure vessel it was not possible to see a stream of bubbles to check where the leak was coming from so it was carefully placed into the pressure chamber. The subsystem passed the test with no leaks. The next step was to put the electronics into the pressure. One positive test was not enough to take the risk of losing the electronics system to a leak, so the system was pressure tested again. When putting the system into the pressure chamber one of the shafts hit the wall of the pressure chamber. The test was still conducted. When the subsystem was pulled out and checked it had leaked. When the system was put back together with the new dynamic seals the shaft collars were not pressed against the wall of the pressure vessel. This allowed for a 2 mm axial movement of the shaft which was just enough to let water in through the dynamic seals. The shaft collars were tightened and set correctly. The subsystem was re-tested and there were no leaks. The system was tested two more times over the weekend to confirm that the pressure vessel did not leak before the pressure vessel was turned over to the rest of the team for software and electronics testing. When integrating the servo assembly into the servo pressure vessel it was noticed that the shaft collars were causing too much friction when the servo moved them. This caused higher amperages required from the servos and therefore higher power consumption. To remedy this issue nylon washers were purchased and placed between the servo pressure vessel wall and the shaft collars. This reduced friction significantly but still did not allow axial movement from the rods, keeping the dynamic seals in place.

5.2.2 Vehicle Testing

The integrated systems were tested for the first time on March 24th. The penguins took Tito into the SeaTech wave tank. Tito was tested for surge motion, and how the vehicle responded to thrust. He was then taken out to the beach to have more running area. We found the back to be sinking, and the nose floating too high. Lead weight was added to the nose cone, and foam blocks were added to the thruster section to counter this.

Tito was tested again March 25th with the added weight and foam. The vehicle was leveled out more now, with a slight nose up pitch. We put Tito in the wave tank again, and tested different thruster speeds. The thruster died after a few minutes of testing. We took him out and diagnosed the electronics for the thruster. We found a blown fuse caused the thruster to stop running. The blown fuse was caused by an amperage spike, due to holding the vehicle in place with the thruster running. We replaced the fuse, and took Tito to the beach for some more surface testing.

The nose cap was fixed with permanent lead weights, and we tested again in the wave tank on March 27th. Tito was now trimmed properly, and we found a suitable thruster speed at 1700. The code for the rudder control was ready, and we took Tito to the Marina. We used the kayak to chase Tito, and controlled Tito wirelessly through the XBee. The rudders worked well, and we could do full turns in either direction. The test was successful. The vehicle was unfortunately dropped on the front end cap on the way back to the lab for storage. There appeared to be no physical damage luckily.

Tito was again taken out in the Marina to test the rudder control on March 28th. There was some slight miscommunication between the operator and kayaker this day. A recreational vessel traveling through the slowly drove over Tito. Tito was immediately taken out of the water, and a small amount of water was found in the pressure vessel. The pressure vessel was tested on March 30th. It failed the test badly. We then positively pressurized the pressurized the vessel to find the leak with soapy water. We found it was leaking from the ball valve used to vacuum seal Tito. This was most likely caused by the fall onto the front end cap. We subsequently found the pressure sensor to be non-operational. This was also caused by the fall most likely.

We then proceeded to keep testing Tito in the Marina on April 2nd. This was our first time really testing Tito's diving capabilities. The first attempt was unsuccessful. An error in the code caused the thruster to run in reverse. Tito quickly descended to the bottom, and became stuck in the mud. We eventually recovered Tito after a few minutes. The code was then corrected, and we proceeded with testing. Tito ran several successful dives for 15 seconds, and then the thruster shut off. He would slowly float to the surface. Tito also kept a constant heading throughout the dives.

Tito's diving and rudder controls were again successfully tested on April 4th. He was found capable of keeping a user defined heading at user defined depth. The next few days were spent dialing in the code for the Ocean test on April 10th.

5.3 Factory Acceptance Testing

The following section details data obtained from marina and ocean testing. A proportional controller was developed in Arduino for the control of the rudder and stern planes. The controller attempted to maintain a proper heading and depth while running a mission. This proportional controller took the difference of the user defined heading and depth from the TCM2 compass and the PX2 Honeywell pressure sensor values. This difference was considered the error of the heading and/or depth of Tito. Both the heading and depth difference were then multiplied by a K_p (gain) coefficient which could be tuned to the correct parameters. The servos were then characterized through a fourth degree polynomial which linearized the servos. From this linearization the rudder and stern plane angles were constrained to ±10° from neutral (0°). Originally, the K_p values for both the rudder and stern planes were 1.5. Figure 50 and figure 51 show the results of marina testing for both heading and depth respectively. For this mission, Tito was given a depth of 0 meters, a heading of 275°, and a time of 20 seconds which corresponds to the length of the marina.

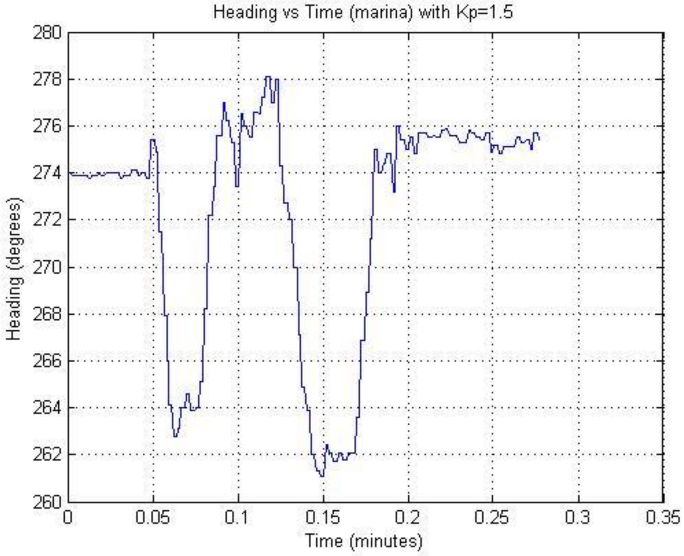


Figure 53: Marina Heading v Time

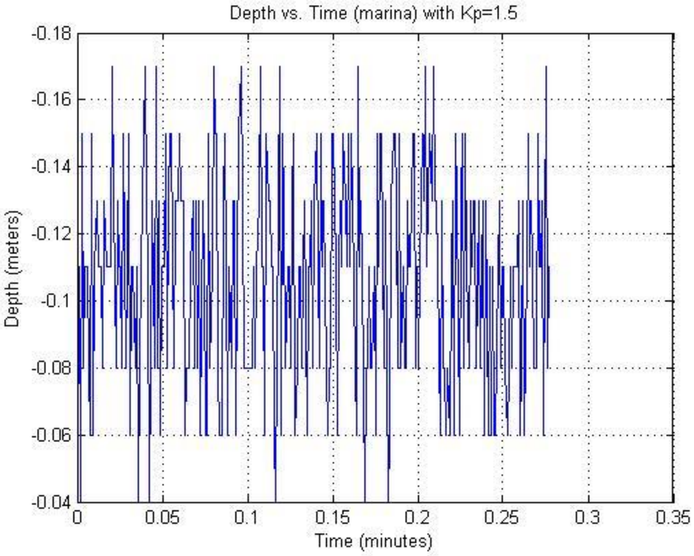


Figure 54: Marina Depth v Time

Figure 50 shows that the K_p value of 1.5 was too high and causing an overshoot for heading. The heading values ranged from 261° to 278° which were outside of the requirements for this project. After analyzing data, it was noted that the gain value for heading needed to be adjusted. The heading K_p term was then adjusted to 0.4 while the stern plane K_p value was left at 1.5. The next mission was run on the surface in the marina with a given heading of 105°, depth of zero meters, and a time of 8 seconds. Figure 52 and figure 53 show that the K_p

adjustment for the heading was within the project requirements. The heading in Figure 52 shows a fluctuation of ±5° from the 105° value given.

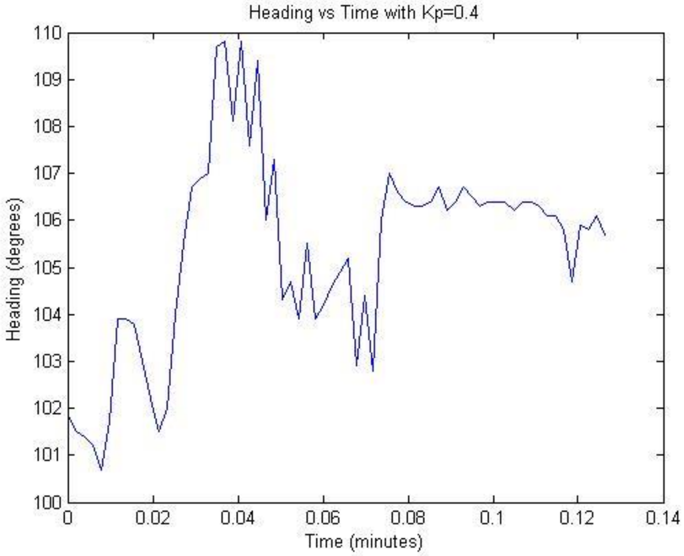


Figure 55: Filtered Marina Heading v Time

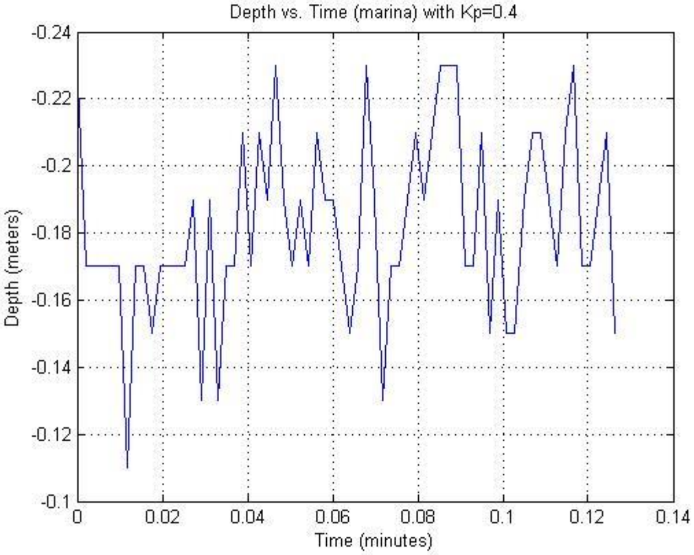


Figure 56: Filtered Marina Heading v Time

The next plan was to test the stern plane gain of 1.5 to see if this K_p value would cause an overshoot. Figure 54 and figure 55 show the heading of 82°, a depth of 1 meter, and a time of 2 minutes.

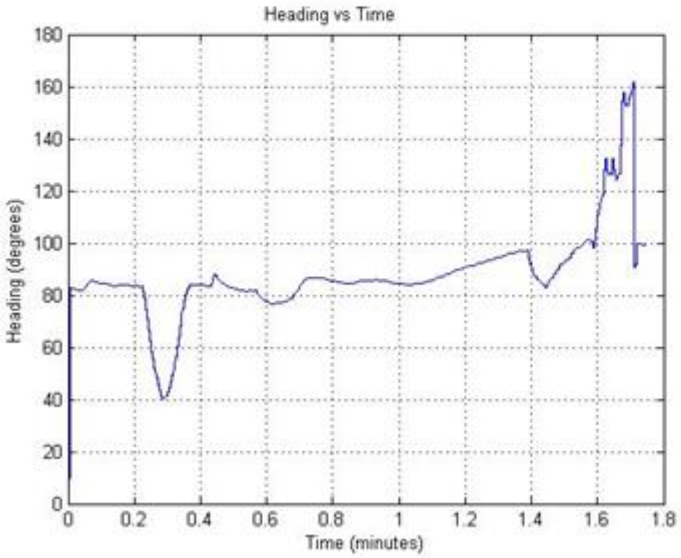


Figure 57: Marina 2 Heading v Time

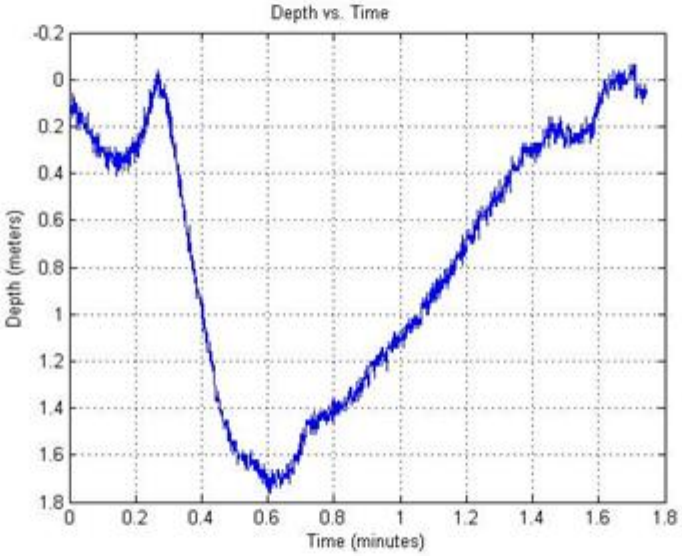


Figure 58: Marina 2 Depth v Time

From figure 55 it is clear that the depth had an overshoot of almost 100% which showed that the K_p value was too high. Comparing figure 54 and 55 shows that as soon as Tito dove he experienced roll which causes the TCM2 heading to have an inaccuracy. The same can be said during the ascent when Tito is about to breach the water surface. After analyzing this data, the K_p value was changed to 0.4 for the stern planes to match the rudders. Tito was then taken out on the Oceaneer for testing in the open ocean. While out at sea, Tito was tasked to perform his first full mission to meet project requirements. This included a depth of 10 meters, a given heading of 180°, and an elapsed time of 15 minutes. Figure 56 and figure 57 show the heading and depth of Tito respectively.

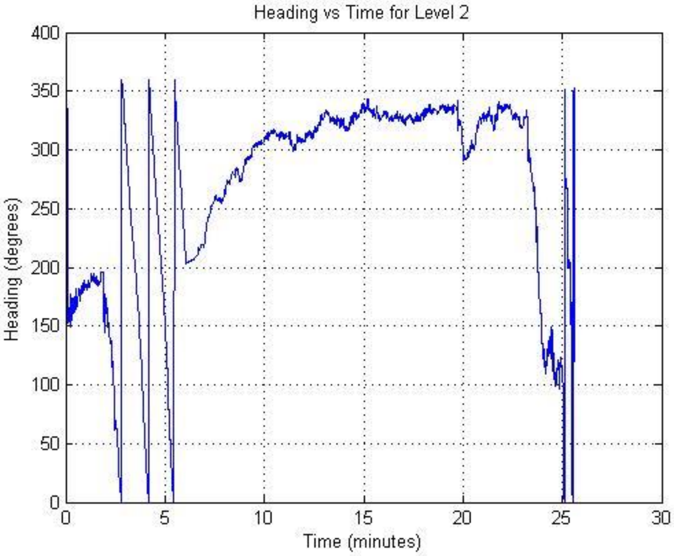


Figure 59: Level 2 Heading v Time

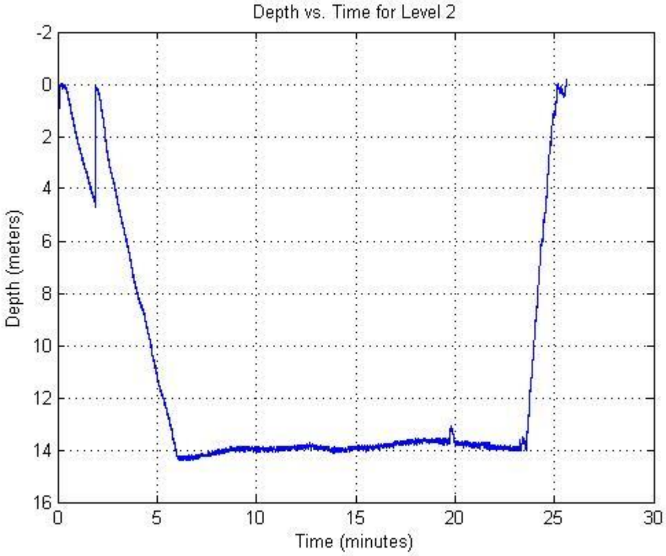


Figure 60: Level 2 Depth v Time

During his descent it is clear from figures 56 and 57 that Tito was performing a spiral dive which was caused by roll from the vehicle trying to adjust for heading while diving. The heading data also shows that after the spiral dive Tito went to a heading of 350° for approximately 15 minutes before starting to correct for heading. It is believed that Tito went straight to the ocean floor and was bouncing off the bottom, which caused the heading to not adjust properly. The reason for the depth error is because of the steady state error in the proportional controller implemented. Steady state error increased with depth and the pressure sensor was never properly characterized due to time limitations.

The second mission performed out in the ocean was the requirement for a box pattern. This mission was run at a depth of 10 meters, a time of 20 minutes (four minutes each leg), and four headings of 45°, 135°, 225°, and 315°. Figures 58 and 59 show the heading and depth for this mission respectively.

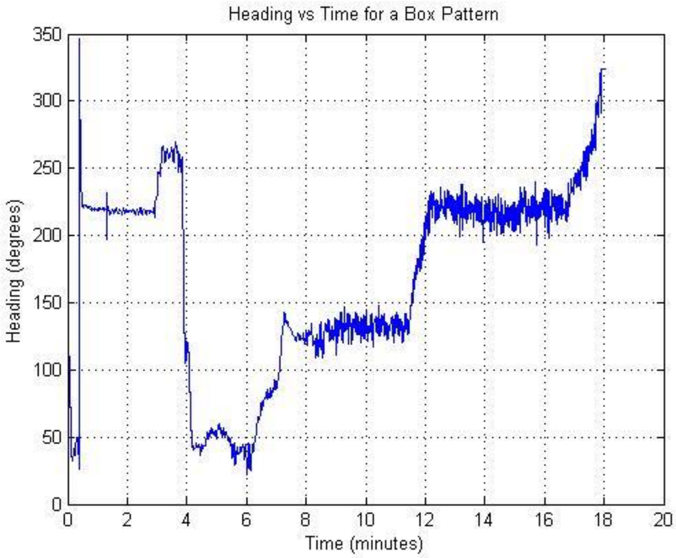


Figure 61: Box Heading v Time

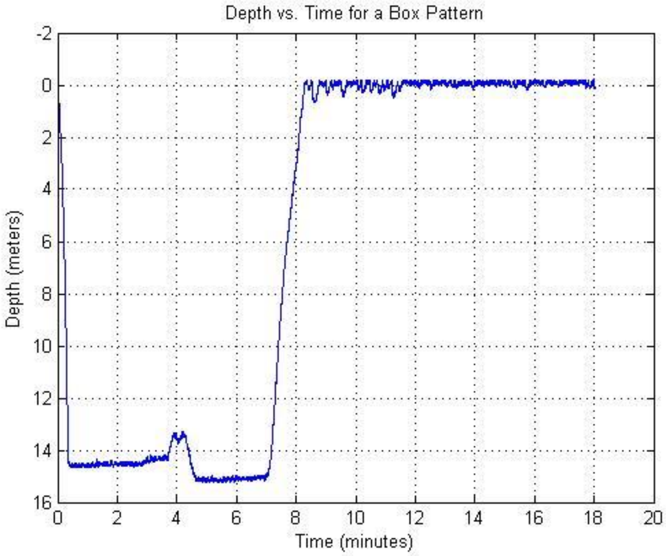


Figure 62: Box Depth v Time

Comparing figure 58 and 59 shows that Tito performed another spiral descent to the bottom of the ocean. After reaching the depth of 14 meters the heading was around 225° for approximately 1 minute before Tito tried to reach the heading input of 45°. After reaching the 45° mark, Tito then attempted to go to the desired heading of 135°. The same can be said for the 225° heading. While Tito was performing the second leg of his mission, the batteries on the vehicle drained to a voltage below what the thruster could sustain for underwater propulsion which caused the ascent of Tito around the seven minute mark. Although the battery voltages were low, they were not completely drained which allowed Tito to nearly complete the box pattern at the surface. At the 18 minute mark the batteries were completely drained and needed charging.

6. User’s Manual

The following guide exists to provide a clear and concise picture of the safety considerations, limitations, capabilities, and operations of Tito Tuxedo. The objective of the user manual is to explain how to assemble, operate, maintain, and troubleshoot the Spring 2015 iteration of Tito Tuxedo so that it can be used for future academic generations. Tito Tuxedo is an autonomous underwater vehicle that is mechanically robust and capable of traversing the underwater environment. The Flying Penguins hope to leave Tito as a legacy so that it can be improved on or used as an example for sleeker AUV’s. Possible uses will include education on IMU/ magnetometer comparisons for data analysis or underwater navigation exploration for control development. In addition to the inherent analytical academic value, Tito has the ability to detail important mechanical traits of an AUV as an educational example. Mechanically its strengths are buoyancy, weight distribution, righting moment, and durability. With the addition of a user manual, future uses of the vehicle will become more accessible.

6.1 Summary of System Operation

Tito Tuxedo is a modular system with three main parts. In order to understand the system and its complete operation, each module should be known. The three modules are the electronics sled, the communications section, and the thruster section. The electronics sled consists of the nose cap, batteries, and circuitry. The communications section consists of the mechanical interfaces connecting the two other modules as well as the communications mast. The thruster section contains the propeller as well as the control surfaces and the controlling mechanisms.

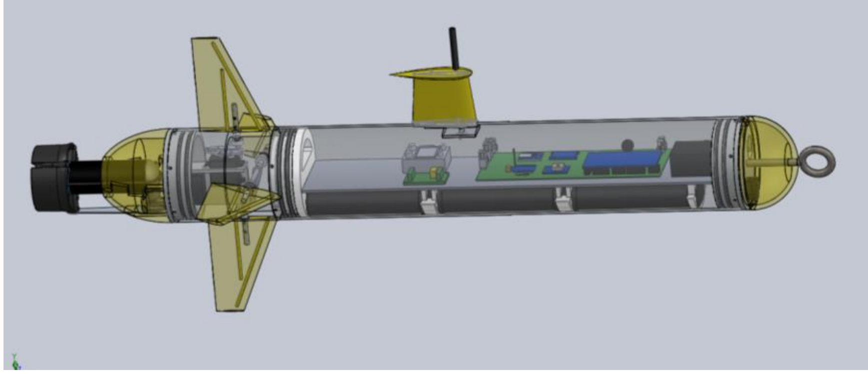


Figure 63: Transparent Tito

The electronics sled first slides into the communications section so that the o rings disappear. Then after communication cables are connected, a partition between sections to hold the battery stack and protect the electronics is added. After the partition is plugged in place, the thruster section can be slid on to the open end. When assembled, the three modules seamlessly combine to form Tito’s operating form. Complete assembly instructions can be found later in this manual.

6.1.1 Brief overview of system and how it works

Tito was designed, built and coded to perform a variety of tasks. Once mechanically closed, (procedure described later) Tito must be controlled using the Xbee radio. It is necessary to have a corresponding (XCTU unique ID) Xbib attached to a computer enabled with HyperTerminal. Once turned on, Tito will communicate through the Xbee to the Xbib on the computer. Once the communication is initialized, then Tito can be commanded with k statements which correspond to missions. While the mission parameters are being set Tito will be floating on the surface with the mast exposed. Once the mission parameters are communicated, Tito executes the commands. Tito communicates the commands to the propeller to go at the set speed, it will control the surfaces to control the attitude at which Tito propels itself in the water. After executing a mission Tito stays in recovery mode for the length of time set by the mission parameters. In this mode, Tito monitors health as well as facilitates GPS communication to a support vessel. The system is recovered using a gaff, noose, or swim step. Tito is one man recoverable using the 3/8" eye bolt on the nose like a narwhal. A gaff can be used to hook the eye bolt, the noose can be used to snare the eye bolt and a swim step can let a support researcher snatch Tito right out of the water.

After mission operation, the vehicle needs to be rinsed completely with fresh water then surface dried before removing the set screws and separating the modules. The system should be stored fully assembles to reduce chance of accident related failures such as falls, spills or thrills.

6.1.2 Basic operating conditions

Conditions for operating Tito are common throughout the year in south Florida. Depending on the mission objectives and mission occasion the operating conditions may change.
Missions departing from shore require two operators. One operator for the computer to execute mission statements, and another to cast and retrieve the vehicle. Shore missions should take place in sea states less than 6ft seas. In sea states more aggressive than 6 ft Tito can operate however its controllability and maneuverability may be compromised due to strong currents. In addition to a calm sea state, the weather should be clear so that the Xbee and GPS signal are not dampened by moisture or extreme weather.
Missions originating from a research vessel require only one operator with the exception of boat operating crew. The vehicle can be initialized, launched, instructed, monitored, and recovered by one operator. Tito should only be operated off a research vessel when the sea state is calmer than 3ft seas. Due to the risk of failing to obtain a GPS fix in sea states that swamp the mast, the vehicle muse be operated in the deep ocean on a calm sea day. However, Tito can be operated in aggressive sea states with the use of an attached tow float for easy surface tracking and identification. Operating in aggressive sea states is NOT recommended however it is possible with the use of recovery aids such as a tow float or tie line.

6.1.3 System Capabilities

As an integrated mechanical-electrical system, Tito is set up with many capabilities. The capabilities of the system were rigorously tested in the marina on different days and with different power levels. All the described capabilities are the results of averages on engineering estimates or the result of testing. For example, speed characterizations were done with a measured distance and a stopwatch. A total of three trials concluded the characterized speed at that setting. In ideal conditions (no current, no boundaries) Tito is mechanically capable of traverse a commanded route
Tito is mechanically tested to the following capabilities:

- Turning radius:~1m
- Rate of decent: 1m/10s
- Top speed: 2m/s Operational at 1.1m/s
- Pressure rated leak free to 50m operation but proved capable of 75m due to unintended

pressure testing
- Passive righting moment to ensure that the communications mast protrudes from the surface
- Precise leveling to ensure flat resting angle for zero pitch and roll compass measurements
- Precise trim to ensure positive buoyancy for recoverability and communication
- Drop tested at 1m heights to be resilient and functional after accidents
- Boat tested so that it can be unintentionally run over by ignorant vessels
- Exchangeable nose cone and tail cone to allow for different trimming or flow behaviors

Tito is electrically capable of the following:

- Sending and receiving health and mission parameters via XBee radio
  - Sending GPS, heading, battery level, internal pressure, and leak sensor information to computer
  - Capable of receiving mission parameters consisting of desired depth, thruster speed, 4 headings, interval time to travel in each heading, and the recovery time
- Monitoring depth to a tenth of a meter
- Turn on and off with a reed switch and a magnet
- Monitoring the heading within the tenth of a degree
- Adjust control surface orientation in real time due to difference between desired depth and heading and the measured depth and heading
- Monitoring the current values going to the thrusters and servo motors (control surfaces) to prevent overloading in cases of resisted movement (eg, thruster stuck in mud, or fins clamped and unable to move)

Combined, the mechanical and electrical capabilities allow Tito to float on the surface, receive mission parameters, proceed to travel in the directions and time lengths of the parameters, then return to the surface and broadcast GPS for recovery. Tito is capable of being deployed and recovered by one operator. Additional capabilities include modularity and expandability for future retrofits or iterations of the electrical or mechanical components. At the partition between the communications and thruster sections, a student could design a fourth module to be a payload or a more advanced sensor array.

6.2 Hazards and Safety

Use good engineering practices when operating Tito tuxedo. At all times awareness of surroundings are crucial for the safe operation of Tito Tuxedo. At all times Tito should be under the direct supervision of a student or employee of Florida Atlantic University department of Ocean Engineering. For understanding the basic benchmarks of vehicle operation contact Ed Henderson Ehender3@fau.edu. It is very possible to injure yourself, the environment or your pride if Tito is operated without proper knowledge of electromechanical devices. Thoroughly read this document for comprehension and ask questions if operations, commands or procedures do not make sense.

Tito is Property of Florida Atlantic University and should only be operated by upper class Engineers only. Understanding of electromechanical devices, microcontrollers, ocean conditions, and data analysis is REQUIRED for safe and effective operation of Tito.

6.2.1 Mechanical Hazards

User ignorance or error is the source of greatest risk throughout the operation procedure. Avoid haphazard alterations and note appearance. Do not operate if the vehicle appears physically compromised. It is important to note the following in consideration of physical logistics of the vehicle:

- The AUV weighs 26 pounds and has hard surfaces
- The AUV operates at up to 1.5 meters per second in the open water and could make unintended movements
- The AUV contains Lead and other hazardous materials under improbable environmental conditions
- Operate in areas clear of water traffic, bathers, or fragile environmental players to avoid improbable damage

6.2.2 Electrical Hazards

Knowledge of electromechanical devices are required before handling the vehicle electronics. Proper care should be given to each component to ensure safety. If wires appear wet frayed or disconnected, do not operate the vehicle. Operate the mission parameters with proper knowledge of health data transmissions. If the health data triggers electrical failure, power off the vehicle and disconnect the batteries before service. It is important to note the following:

- The AUV has a 24 DC volt circuit that can cause electrical shock and fire
- The high output batteries can cause electrical shock and fire
- The AUV can transmit electrical interference to the environment

6.2.3 Personal Safety

At all times use good engineering practices. Never work on Tito when you are tired, hungry or impaired. Always pay attention to the following when considering vehicle objectives:

- Wear protective gloves when handling lead inserts
- Wear covered shoes when moving the AUV
- Do not wear loose clothing when operating the AUV
- Secure loose hair and jewelry when operating the AUV
- Take plenty of fluids when performing missions outside
- Wear sunscreen and protective clothing when working outside

6.3 Cautions and Considerations

Every prototype and pre-production model has potential quirks. This section details potential cautions and considerations necessary for the safe and recoverable operation of Tito Tuxedo.

6.3.1 Operational Cautions

While operating the vehicle always consider the following:

- Keep out of reach of children
- Have at least two people present when operating the AUV
- Dropping the AUV may damage the integrity of the pressure vessel
    - If dropped pressure test to save electronics
- Do not operate in sea conditions higher than sea state one
- Use fully charged batteries for missions lasting longer than 20 minutes
- Use a tow float when operating with poor GPS reception in higher sea states

6.3.2 Mechanical Cautions

Even when not in operation the following information is important to consider:

- The AUV body has machined aluminum surfaces that could develop sharp edges which can cut skin or wires
- The AUV weighs 26 pounds and could cause injuries when picked up or dropped
- The propeller has sharp blades and turn up to 1700 revolutions per minute
- When performing a servo sweep of the control surfaces, the AUV can be rocked off balance, use a secure mounting
- Not following the assembly procedure in this manual could cause damage to mechanical parts

6.3.3 Electrical Cautions

While in operation and during storage, it is important to recollect the following:

- Do not touch Seacon charging spade connectors to each other, it may cause a short circuit
- Do not leave Seacon connection open, insert Loopback plug or power cable
- There is a 24 volt load at the Reed Switch, take caution not to cause a short circuit
- Make sure that the radio receivers are on the same frequency using the XCTU
- Ensure all surfaces are dry before disassembling the AUV
- Use caution when removing electronic components, the delicate pins can easily be bent
- Do not charge the batteries at a current higher than 0.9 Amps

6.4 Operational Notes

In the context of this user manual, the operational notes serve as a procedure for mission preparation, operation, and conclusion. The system level operation is a general mission procedure from start to finish. System level operation would detail a boat or shore excursion procedure. The component level details individual procedures for tasks such as rinsing, charging, and communicating.

System level operation notes describe the procedure for completing a mission and operating the complete powered on system.

6.4.1 System Level

The system level procedure begins after the preparation procedure ends. After the vehicle has been programed, closed, vacuumed, and sealed the power can be turned on and the system level procedure can begin.

After powered on, use the Xbib on a support laptop to talk to the xbee signal that Tito will be broadcasting.

Tito's software system uses six case statements to execute various tasks, this includes a health report, a control check, navigation check, the last known GPS coordinates, saving one string of data to an SD card, and the mission. It is recommended to follow an order to insure all systems are functioning properly. The first step into insuring a properly running vehicle is to have the Xbee connected to the Xbib, then have the Xbib connected to the laptop through the hyper terminal. The hyper terminal can be downloaded on various websites using a Google search engine. Once the hyper terminal has opened, check that the comport on the Xbib matches the hyper terminal comport. To do this go to the hyper terminal screen and select file→new connection→name→select icon→OK→Check COM port to match Xbib→OK. The hyper terminal will automatically call the Xbib and the two will be connected to each other. After this call is made, both Xbees are able to communicate with each other. Ensure that both Xbees are powered ON in order for this step to work.

Now that the communication has been setup, the first sequence for case statements is the health check. Pushing the ‘h’ key on the keyboard will send a list of health check from Tito. This includes elapsed time in minutes, battery 1-3 voltages, thruster current, rudder current, stern plane current, front and back leak sensor, temperature in Fahrenheit, and internal pressure in kilopascals. Reading the battery voltages allows the user to tell if the batteries need charging at a given moment. Battery voltages should be near 8 volts. Below 7.5v the batteries should be charged. The three current sensors are to ensure that the servos and thruster are within the range of amperage. This being the thruster within 4 amps continuously and the two servos within 2.5 amps. Over the safe amperage implies that there is a mechanical impediment and a situation to stop testing. The internal temperature sensor is to make sure that the insides of Tito are monitored and a mission can be stopped so that Tito does not get too hot. The temperature is ideally under 100 degrees Fahrenheit. The internal pressure sensor allows the user to check for slow leaks after a vacuum was put on Tito. Note that as internal temperature fluctuates so does internal pressure. The internal pressure is the first indication that there is a mechanical seal issue. The two leak sensors generally display a value around 3700. If at any time this value drops below 2000, this means that the circuit on the leak sensor has been completed and there is a high risk that there is water in the hull and Tito is leaking. If any of the health report looks to be outside of the parameters stated above, it is highly recommended to push ‘s’ on the keyboard to store the data on the SD card. Note that this will only store one string of data onto the card. It is also recommended not to continue onto the mission until all parameters are within the safety of margin.

Once the health has been initialized in a dry setting, Tito can now go through the launch procedure. After the launch procedure while Tito is steady on the surface of the water the other case statements come into play.

The third case statement to be performed is the control case statement. By pushing the ‘c’ key this will allow the servos to run a servo sweep and the thruster to ramp up to full speed then ramp back down to idle. This is to check that all connections to actuators are in tack and that there is no movement issues with the servos. This test exists to check for mechanical impediments in a new environment before mission execution.

The following case statement after control is ‘n’ which a navigation check is. It will simply send the TCM2 heading through the Xbee to the hyper terminal screen. This can be used to check if the TCM2 heading is correct or if the compass needs calibrating. To calibrate the compass go to the TCM2 datasheet and follow the instructions on how to calibrate the compass.

The succeeding case statement is ‘m’ which is the mission case statement. WARNING: Once this mission is executed, it will run a complete mission and the only way to terminate this course of action is by putting the reed switch into the appropriate position. After pushing m’ the hyper terminal will display “Enter depth in meters”. Type in the desired depth and push ENTER on the keyboard to go to the next step. The next step will display “Enter Heading 1”. Type in the desired heading in degrees and push ENTER on the keyboard. Follow this routine three more times for input heading 2, heading 3, and heading 4. The program will then ask for “Interval in seconds”, once again type in the appropriate value and push ENTER. “Enter Thruster value” will then appear on the hyper terminal. Any thruster value below 1500 will set the thruster in reverse, 1500 will put the thruster at idle, and anything above 1500 will send the thruster forward. The recommended forward value is any number between 1575-1600 which is a duty cycle of 15%-20%. The last question will ask “Time for recovery (in minutes)” which will send GPS coordinates for the amount of minutes established. Once all these values are inputted Tito will display “Ten seconds to Thruster Start” through the hyper terminal and Tito will run the mission accordingly. WARNING: If at any time the Xbee antenna is NOT able to communicate the user inputs may be invalid and a restart of Tito through the reed switch is necessary. For example, if Tito is in the water and a wave passes over the mast this could interfere with the communication of the Xbees. After Tito has completed a mission, pushing the ‘g’ key will show the last known fixed GPS coordinates. This is a safety key incase the recovery mode times out or if the hyper terminal data accidentally gets cleared. After mission completion Tito will send its own recovery signal until the prescribed recovery loop times out. Then Tito will conserve battery and only send GPS when interrogated by the XBee. Once the GPS signal is sent to the XBib, a support vessel can head to the coordinates for retrieval.

6.4.2 Component Level

The component level of operation notes details specific procedures for actions not described in the systems level. Detailed operations such as mission preparation, post mission storage, launching and recovering Tito are described as procedures with pictures to assist in understanding.

6.4.2.1 Software

To upload software, make sure that Tito is powered on. This includes the magnet on the reed switch being removed. After this, connect the USB micro to the programming port on the Arduino Due. This cable then gets connected to the laptop USB. Open the Arduino program and check that the correct comport is connected. To do this go to tools→port→select comport. Once connected upload the code to the Due.

6.4.2.2 Preparation

The preparation procedure picks up at the tail end of assembly. After everything has been fabricated and bench tested; Tito consists of three main parts. The thruster, communication, and electronics sections are the pieces needed for the preparation procedure. First take the communication section and guide the GPS and XBee wires to the rear of the cylinder. Then slide the electronics section into the communication section such that the mast is on the top and the batteries are on the bottom.

Slide the electronics section into the communications section until the o rings on the nose cap meet the communications section. Before sliding the orings into the communications section, connect 2 cables in the back. The GPS antenna, and the XBee antenna. Now that the wires are connected, ensure that the all cables are neatly placed above the pcb. After ensuring the wires are neat and comfortable, push the nose cap into the communications section past both o rings. It is important to take your time and ensure that the oring was not rolled or pinched between the nose cap and the communications section. Now that the nose cap is already on, the reed switch should align with the mast. If everything is aligned properly set screw the cap in place. After placing the set screws, locate the clear battery plug and servo partition. Align the servo partition with the electronics board and the battery so that the batter is held firmly in place. Once the plug is in place, the attached leak sensor must be attached to the corresponding connector on the extension wire protruding from the plug window.

After the battery plug is in place, locate the thruster section. The thruster section is a pre-assembled modular component which contains the secured control surface servo linkages and propeller motor mounts. First, hold the thruster section near the battery plug and connect the thruster (blue and red 2 prong cable) then connect the servo motors (multi colored ribbon cables) through the battery plug window to the respective wire extenders passing through the window. Once connected, streamline the wires to avoid pinching, tangling, or mechanical interference and then slide the thruster section onto the communications section. The thruster mount will be at the bottom or 6"oclock position. After the sections are connected, place the set screws in their holes to prevent shifting or sliding along the seal point.

Once the set screws are attached then locate a vacuum pump. With a barbed ¼"NPT fitting, suck a negative gauge pressure on the vehicle. Quickly remove the barbed fitting and exchange with a boss plug that has been previously prepared with Teflon tape. After the plug is torqued down, remove the reed switch and attach the nose cone via the 3/8" eye bolt. Monitor the internal pressure and when the internal pressure settles at a pressure less than ambient and holds the pressure for 10 minutes the preparation is complete and Tito is ready to run a mission.

6.4.2.3 Launching

Launching Tito is not an arduous or complex task. Launching Tito can occur after completion of the preparation procedures and after initializing of the system operation. After the health is monitored via XBee communication the launch can be initiated. Tito is positively buoyant and mechanically resistant to drops and dings. The method of launching Tito requires either a boat platform or a sea shore. From the seashore, Tito can be carried into water depths exceeding 1m then lowered. Tito will fall underwater then slowly arise to the surface a few feet from where he was lowered into the water depending on how fast it has hit the water. If Tito did not fully submerge, there may be bubbles in the nose cap which will temporarily inhibit diving until all of the air is displaced. The shore launching procedure concludes when Tito receives mission information and then travels away from the launching party.

For boat launching, the procedure begins after the preparation is concluded and health data is communicated to the Xbib. The proper procedure for a boat launch is a one man swing launch. The operator should grab Tito firmly with two hands and move close to the edge of the vessel. After communicating the will to launch with the boat captain and gaining his/her approval, gently push Tito out of your hands into the water and allow him to decent and travel away from the vessel. The purpose of the swing/push is to immediately separate the vessel and Tito in aggressive sea states. Launch procedure is completed when Tito is in the water floating on the surface ready to receive mission data.

6.4.2.4 Recovery

Recovery procedure is initiated by the completion of a mission. After Tito completes a mission it immediately enters a recovery state. In this state, Tito broadcasts GPS and health information through the Xbee. On the vessel, the operator will direct the captain to the expected retrieval GPS. Then the operator will monitor the Xbib serial monitor until Tito surfaces and extends GPS data. Once received, the GPS data will determine a vector for recovery. Once in sight, Tito can be recovered by one of three ways depending on the operator. First and most favorable, Tito is recovered by hand off of a swim step. Simply reaching in while the vessel motor is in neutral and pulling the vehicle out by the ring. Second, Tito is recovered using a noose on a pole. There is a rope that stays open and when the rope is around Tito, the operator quickly pulls the string and traps Tito between the noose and the pole that the noose is attached through. The pole can be pulled on board and a support crew member can grab the hanging Tito. Lastly, the most mechanically foolproof and least risky method of recovery is a hooked gaff. Once the vessel is close enough, the operator hooks Tito by the nose ring and pulls the vehicle from the water and onto the boat surface. Recovery ends when Tito is completely out of the water and powered off with the reed switch.

6.4.2.5 Post mission

Post mission procedures initialize after the vehicle has been recovered and the team is back at the mission base camp. The first and most important post mission procedure is rinsing Tito of all salt water. First, rinse all exterior surfaces very well. Including under the control surfaces where they meet the thruster section. Ensure to drench the thruster and the thruster housing with freshwater. Then remove the nose cone by twisting the eyebolt and rinse all the interfaces on the nose cap. After the vehicle is thoroughly rinsed surface dry the aluminum exterior so that it can be safely opened. Once the vehicle is dried, remove the set screws between the thruster and communications sections. Then unscrew the boss plug in the nose cap of the AUV. Removing the plug will allow the vacuum to dissipate and allow for the thruster section to be carefully pulled off. Pull the section off very slow because of the connecting cables. The thruster cable and servo cables should be immediately disconnected by the operator upon removal of the thruster section. Then carefully remove the leak sensor connector and the battery plug so that the GPS and XBee connectors can be disconnected. Once the thruster, servo, leak, GPS, and XBee connectors have been disconnected, then remove the set screws from the nose cap. Once the screws are removed, use the eye bolt to carefully pull the nose cap out of the communications section so that the electronics tray can be exposed. The SD card can then be safely removed and post processed for data.

For long term storage (over 1 week), remove the battery connections to the PCB board, then put the electronics section in the communications section. After the electronics are secured and not connected, secure the battery plug to prevent the chance of vibrations or shaking destroying components. Then place the thruster section in place. Store with all setscrews in place so that the entire system is stored in one specific place. After long storage, add the battery connections to the preparation procedure.

6.5 Trouble Shooting

If Tito is fully assembled and there is a malfunction in Tito it is necessary to narrow down the exact problem with Tito. Once the problem has been narrowed down go to the component level of troubleshooting.

6.5.1 Component Level

The component level of troubleshooting describes how to fix common problems observed throughout the project. This is broken down into power, propulsion, control, navigation, body and software levels.

6.5.1.1 Power

Ensure that all wires are connected properly. This includes the three battery connectors, the loopback connector, and the reed switch connector. Make sure that the batteries are charged and that the magnet is OFF the reed switch. If all these steps are followed and Tito still does not turn

ON, check the following:

- Check that the Seacon connector is in place and clean
- Check continuity throughout all wires
- Check test point voltages
- Check traces on the PCB

6.5.1.2 Propulsion

Ensure that the SeaBotix thruster and the motor controller are connected properly. If this is done and the problem still is not solved check the following:

- Continuity of the wires
- Dipswitches on motor controller
  - Look at the Syren10A data sheet for more information on the dipswitches located on the motor controller
- Check for obstructions in the propeller and that it is able to rotate freely

NOTE: The screw terminals located on the motor controller often get lose and disconnect wires. This is usually the main cause of thruster loss.

6.5.1.3 Control

Make sure that the servo motors are connected properly. NOTE: DO NOT remove the servo horns from the servo motors because they are positioned appropriately to never hit the wall. If the servo motors are connected and still not working, check the following:

- Fuses for the servo motors
- Individually check each servo using an external power suppy to make sure the servos are not broken

6.5.1.4 Navigation

Properly connect the TCM2 and the external pressure sensor. If this is done and navigation still isn’t working, check the following:

- Check that a magnet is not near the TCM2
- Check for obstruction in the pressure sensor
- Check the connectors on the TCM2 and pressure sensor
- If TCM2 heading is wrong, recalibrate the TCM2 according to the TCM2 User Manual
- Check for pressure fluctuations on the pressure sensor
  - An oscilloscope can check for changes in signal level

6.5.1.5 Body

- Body troubleshooting consists of picture comparisons.
  - If a system of component looks different than the pictures in this document then there is mechanical failure.
- With regular maintenance of seals leaks will not occur. Only under user error or extreme operating conditions will the vehicle break mechanically.
- If there is a grinding or moaning coming from the servo section
  - Preform a servo sweep(mentioned in the system operation notes)

6.5.1.6 Software

- The GPS has a fix, but data isn't being transmitted: Check electrical connections and battery level. Remove and reinsert the magnet on the Reed switch to reboot the system.
- The serial terminal is not receiving or transmitting data to the AUV: Check that the serial cable is connected to the correct communication port on the PC.

6.6 Maintenance Schedules

Tito tuxedo is designed to be operated for many years if the procedures are followed for operation and maintenance. Maintenance is traditional and inexpensive. Other than exterior washing to prevent oxidation or corrosion, Tito should be relatively maintenance free.

6.6.1 General Vehicle Maintenance

Every time the pressure vessel is sealed, the Boss plug must be Teflon taped.
Every 10 times sealing the pressure vessel, all #246 o rings should be changed on the nose cap and thruster section.
After 100 times sealing the pressure vessel, or approximately 10 days of salt water time the PTFE ring seals should be replaced and the mast o ring should be replaced.

6.6.2 Component Level Maintenance

Only in the case of individual failure from accidents, user error, or manufacturer error should components be maintained or replaced. Consult Ed Henderson as well as the associated material to track down the faulty component.

6.7 Repair Information

Repairs on Tito are likely electrical. Mechanical repairs are not expected if the vehicle is operated in only prescribed conditions. Specific information regarding the code iterations, electronic connections, and spare mechanical parts can be found in the appendix.

6.7.1.1 Propulsion

There are two main components attributed to the propulsion system which is the Syren10A motor controller and the SeaBotix thruster. The Seabotix thruster has a four pin connector (P4) that is attached to another connector (J4). J4 is connected to the motor controller which is then connected to the PCB connector (J5). To get further detail of the wiring of the system look in

6.7.1.2 Control

The control system consists of the two servo motors. These are connected to the PCB through connectors J6 and P3.

6.7.1.3 Navigation

The TCM2 and external pressure sensor are the two components consisted for navigation. The TCM2 compass is wired from the PCB connector J1 to the M1P1 mating connector. The TCM2 is facing forward if the connector on the compass is facing aft. The external pressure sensor is connected to the PCB (J4) and mated with connector M1P4. The pressure sensor then gets screwed into its appropriate hole on the front cap.

6.7.1.4 Body

Inspect all O-rings for defects and suitable lubrication. Make sure the vacuum plug has new tape tread each use. Make sure the feedback loop plug is in correctly. Spare parts can be fabricated by Anthony Lavigne in the Machine shop at Seatech. The drawings are in the appendix as well as in the Critical Design Report.

7. Detailed Budget

The detailed budget list is shown in the next table below. The table shows the vendor, description, quantity, price, and shipping. Many components were purchased in bulk which saved on shipping costs. The total cost of all the parts purchased was $1590.65 which left $409.35 from the $2000 budget imposed by the sponsor. The department lent The Flying Penguins a various amount of parts in order for the team to have a successful project and fall within budget. These items are listed in the smaller table below the previous table. As a team the Flying Penguins appreciate the efforts of the university to facilitate a successful senior design initiative.

Vendor	Description	Quantity	Price	Shipping	Cost w/ Shipping
AA Portable Power Corp	NiMH 7.4V	3	$180.00	$5.99	$185.99
Adafruit Industries	External Active Antenna	1	$12.95	$0.00	$12.95
Adafruit Industries	SMA to uFL	1	$3.95	$0.00	$3.95
Adafruit Industries	Micro-SD Card Breakout	1	$14.95	$0.00	$14.95
Advanced Circuits	P1.0KACT-ND	1	$33.00	$5.12	$38.12
Digi-Key Corp	Normally Closed	1			
Digi-Key Corp	Honeywell PX2	1	$89.48	$0.00	$0.00
Digi-key corp	ANT-916-CW-HW	1	$7.95	$0.00	$0.00
Digi-Key Corp	43045-0414	2	$3.52	$0.00	$0.00
Digi-Key Corp	43045-0214	2	$2.38	$0.00	$0.00
Digi-Key Corp	43045-0614	1	$2.24	$0.00	$0.00
Digi-Key Corp	39-29-6048	3	$5.94	$0.00	$0.00
Digi-Key Corp	43025-0200	2	$0.68	$0.00	$0.00
Digi-Key Corp	43025-0400	2	$0.90	$0.00	$0.00
Digi-Key Corp	43025-0600	2	$1.00	$0.00	$0.00
Digi-Key Corp	39-01-2040	4	$1.24	$0.00	$0.00
Digi-Key Corp	39-01-3043	1	$0.36	$0.00	$0.00
Digi-Key Corp	43020-0600	1	$0.50	$0.00	$0.00
Digi-Key Corp	PCB Parts		$232.16	$0.00	$232.16
Mouser	PCB Parts		$8.78		$8.78
Mouser	XBIB-U-ND	1	$60.00	$1.99	$61.99
Dimension Engineering	SyRen 10	1	$49.99	$2.50	$52.49
Mouser	12110192	1	$7.17		
Newark	MPX4250	1	$16.74		
Newark	TC1047AVNBTR	1	$0.62		
Mouser	Xbee 900HP Module	2	$39.00	$4.44	$82.44
Pololu	1218TH	2	$119.95	$6.95	$126.90
Power Sources Unlimited	TSR3-24150	2	$53.20	$17.11	$70.31
Power Sources Unlimited	TSRN1-2450SM	1	$9.75	$0.00	$0.00
John	Iso-Alcohol	1	$3.00	$0.00	$3.00
Digi-Key Corp	Honeywell PX2	2	$169.92	$0.00	$169.92
Adafruit	GPS	1	$39.99	$8.80	$48.79
McMaster Carr	6493k18	3ft	$38.53	$0.00	$0.00
McMaster Carr	4443k722	1	$3.40	$0.00	$0.00
McMaster Carr	13125k65	4	$56.80	$0.00	$0.00
McMaster Carr	90294a822	1	$5.26	$0.00	$0.00
McMaster Carr	2706t14	6	$25.62	$0.00	$0.00
McMaster Carr	9946K11	4	$8.36	$0.00	$0.00
McMaster Carr	95345A037	1	$10.73	$0.00	$0.00
McMaster Carr	95345A031	1	$7.96	$8.12	$164.78
Tony	Aluminum	1	$143.84	$0.00	$143.84
McMaster Carr	92311A190	1	$4.77	$12.23	$29.94
McMaster Carr	91772A076	1	$5.74	$0.00	$0.00
McMaster Carr	91780A016	20	$7.20	$0.00	$0.00
McMaster Carr	6366K47	1	$23.43	$2.00	$25.43
McMaster Carr	5018t592	1	$7.23	$8.00	$15.23
McMaster Carr	2706t14	7	$29.89	$12.00	$0.00
McMaster Carr	13125k65	4	$56.80	$0.00	$98.69

Table 6: Detailed Budget

Vendor	Description	Quantity	Price
SeaBotix	Thruster-BTD 150	1	$1,500.00
PNI Sensor Corp	Compass TCM2	1	$887.00
Parallax	RF XBee 900HP Module	2	$78.00
SparkFun Electronics	Arduino DUE	1	$49.95
Adafruit Industries	10-DOF IMU	1	$49.95
Adafruit Industries	Ultimate GPS breakout	1	$39.95
Seacon Connector	FAWM-8P-MP	1	$88.00
Seacon Connector	FAWM-85-BC-RIA	2	$88.00

In summation, the out of budget cost of fabricating and assembling Tito was $1590.65. While considering both donated parts and out of pocket purchases, the total production cost of Tito not counting labor, tools, or design fees is: $4371.50. With our out of budget cost the flying penguins were able to finish the project on time, under budget, and within the design specifications. In industry, our results would be a smashing success and our project manager would get a raise and the group would be recommended for a more ambitious design initiative. Without the support of the university donating crucial parts Tito would not have been a possibility.

8. Appendices

8.1 Photos

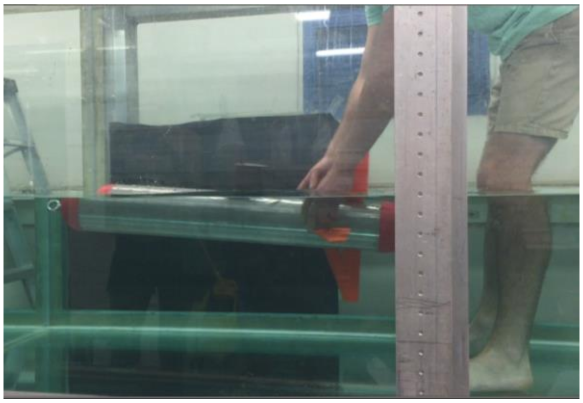


Figure 64: Tito Floating

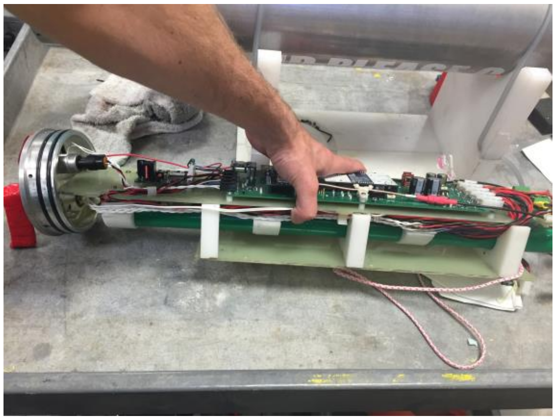


Figure 65: Electronics on Stand

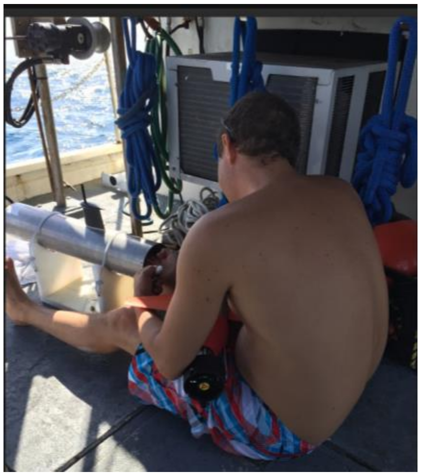


Figure 67: At Sea Charging

Figure 66: Thumbs Up

Figure 68: Beach Test

Figure 70: Thruster Testing

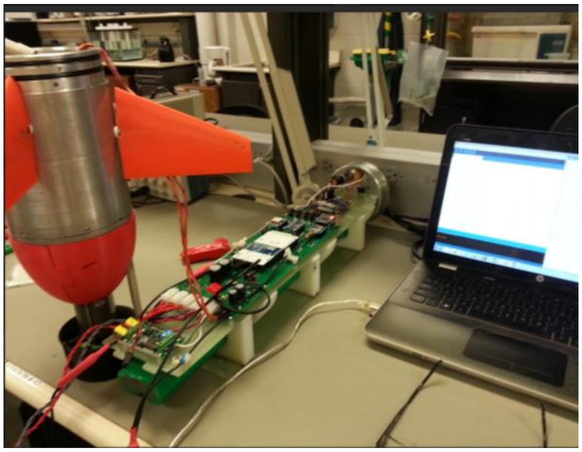


Figure 69: Control Surface Testing

Figure 71: Rudder ROM

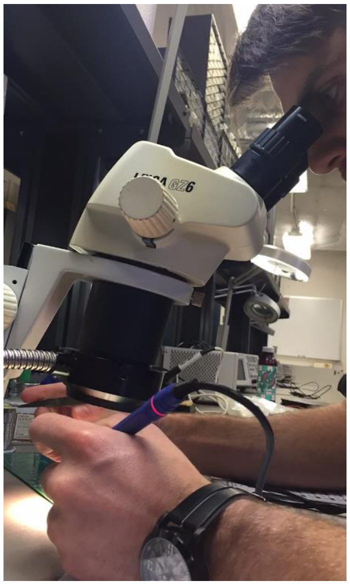


Figure 72: PCB Soldering

Figure 73: Machine Shop

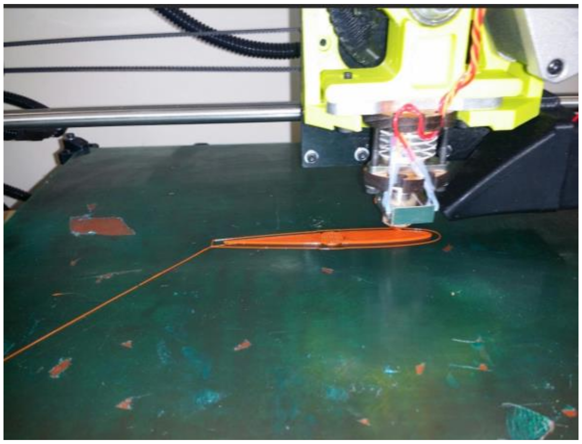


Figure 74: Printing Fin

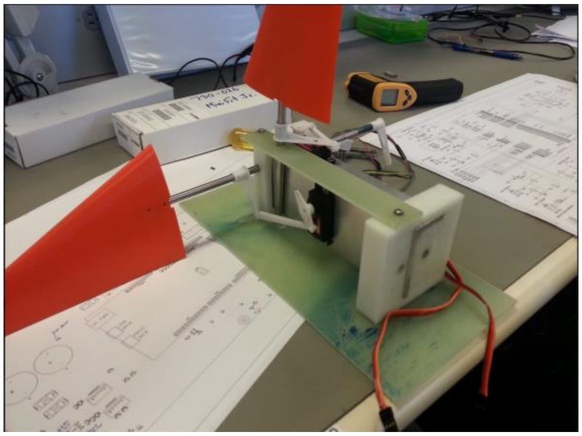


Figure 75: Servo Test Mount

Figure 76: Cheesin'

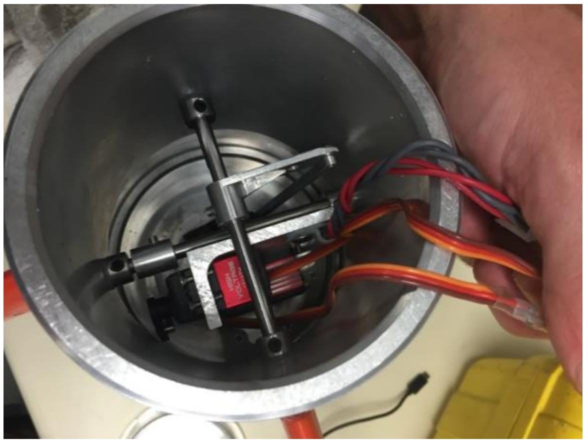


Figure 77: Servo Vessel Internals

8.2 Code

```
//////Int analog Pins
int battery1 = ADC0;
int battery2 = ADC2;
int battery3 = ADC1;
int pressureSensor = ADC3;
int thrusterCurrent = ADC4;
int rudderCurrent = ADC5;
int splaneCurrent = ADC6;
int depth = ADC7;
int leakSensor1 = ADC8;
int leakSensor2 = ADC9;
int temperatureSensor = ADC10;
///////Include Header files
#include <Servo.h>
#include <Adafruit_GPS.h>
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_LSM303_U.h>
#include <Adafruit_L3GD20_U.h>
#include <Adafruit_BMP085_U.h>
#include <Adafruit_10DOF.h>
#include <SPI.h>
#include <SD.h>
#include <Adafruit_Simple_AHRS.h>
#include <math.h>
#include <PID_v1.h>
///////Define Serial
#define GPS_Serial Serial1
#define tcm2Serial Serial2
#define XbeeSerial Serial3
///////10DOF
Adafruit_L3GD20_Unified gyroscope = Adafruit_L3GD20_Unified(20);
Adafruit_10DOF                dof   = Adafruit_10DOF();
Adafruit_LSM303_Accel_Unified accelerometer = Adafruit_LSM303_Accel_Unified(30301);
Adafruit_LSM303_Mag_Unified   magnetometer   = Adafruit_LSM303_Mag_Unified(30302);
Adafruit_BMP085_Unified       barometricPressure   = Adafruit_BMP085_Unified(18001);
Adafruit_Simple_AHRS          ahrs(&accelerometer, &magnetometer);
sensors_event_t accel_event;
sensors_event_t mag_event;
sensors_event_t event;
sensors_vec_t   orientation;
Adafruit_GPS GPS(&GPS_Serial);
float seaLevelPressure = SENSORS_PRESSURE_SEALEVELHPA;
///////HealthReport
float AdcConversion;
float ADC_Conversion;
float AdcConversion2;
float AdcConversion3;
float C_F;
float batt1;
float batt2;
float batt3;
float leak1;
float leak2;
float Temp;
float Temp2;
float pressure;
float pressure2;
float raw_pressure;
float raw_pressure2;
float TCurrent;
float RCurrent;
float SCurrent;
float depth2;
float depth3;
//SD Card
```

```
const int chipselect = 42;
File titoData;
String dataString = "";
//Servos/Thruster
Servo thruster;
Servo splane;
Servo rudder;
//Timer
int count;
float elapsedMillis;
float hertz;
long previousMillis = 0;
long intervalMillis = 1000;
//TCM2
int inChar = tcm2Serial.read();  // read the data and store it in the variable inChar as an integer
int p;
int r;
String inString = "";  // initializing the string inString
String TCM2output = "";
String headingstring = "";
float pitchTCM2;
float rollTCM2;
String pitchstring = "";
String rollstring = "";
//Tuning
int R_gain;
int S_gain;
String st = "";
//Controller
unsigned long starttime;
unsigned long endtime;
int loopcount;
unsigned long startleg;
unsigned long endleg;
int loopcountleg;
//Depth PID
double userDepth2, depth4, splaneAdjustment;
//Heading PID
double headingTcm2, rudderAdjustment, userHeading;
//User Inputs
char junk = ' ';
float userHeading1;
float userHeading2;
float userHeading3;
float userHeading4;
float userDepth1 = 0;
float userInterval;
float interval;
float userThrust;
float RetrievalTime;
//Case Statements
char keyhole;
//Vital Loop
String vitalString = "";
float depErr;
float hedErr;

void setup() {
  // put your setup code here, to run once:
  pinMode(50, OUTPUT); //Switch Battery
  digitalWrite(50, HIGH);
  analogReadResolution(12);
  analogWriteResolution(12);
  XbeeSerial.begin(9600);
  XbeeSerial.println("Tito is POWERED ON");
  XbeeSerial.println("");

  //Initialize TCM2, Use NMEA 0183 Format for Heading only, string: $HCHDM,<compass>,M*checksum<cr><lf>
  tcm2Serial.begin(9600);
```

```
  XbeeSerial.println("TCM2 Compass Heading Initialization to NMEA 0183");
  XbeeSerial.println("");
  tcm2Serial.write("go\r\n"); //send contineous output command
  tcm2Serial.write("sdo=n\r\n"); //set output word to NMEA 0183

  //Servos Attached
  splane.attach(3, 1000, 2000);
  rudder.attach(4, 1000, 2000);
  thruster.attach(2, 1000, 2000);
  //Intialize GPS
  GPS.begin(9600);
  GPS_Serial.begin(9600);
  GPS.sendCommand(PMTK_SET_NMEA_OUTPUT_RMCGGA);
  GPS.sendCommand(PMTK_SET_NMEA_UPDATE_1HZ);
  GPS.sendCommand(PGCMD_ANTENNA);
  Serial.begin(9600); // set up Serial library at 9600 bps
  //Initialise the 10 DOF IMU
  gyroscope.enableAutoRange(true);
if(!accelerometer.begin())
  {
    /* There was a problem detecting the LSM303 ... check your connections */
    XbeeSerial.println(F("Ooops(accel), no LSM303 detected ... Check your wiring!"));
    while(1);
  }
  if(!magnetometer.begin())
  {
    /* There was a problem detecting the LSM303 ... check your connections */
    XbeeSerial.println("Ooops(mag), no LSM303 detected ... Check your wiring!");
    while(1);
  }
  if(!barometricPressure.begin())
  {
     /* There was a problem detecting the BMP180 ... check your connections */
    XbeeSerial.println("Ooops(bar), no BMP180 detected ... Check your wiring!");
    while(1);
  }
  if(!gyroscope.begin())
  {
    /* There was a problem detecting the LSM303 ... check your connections */
    XbeeSerial.println("Ooops(gyro), no LSM303 detected ... Check your wiring!");
    while(1);
  }
  //Set PID in Automatic Mode
   //Initialize the SD Card Slot
   XbeeSerial.print("Initializing SD card...");
   Serial.print("Initializing SD card...");
   pinMode(10, OUTPUT); //For the SD Card to work
  if (!SD.begin(chipselect))
    {
      XbeeSerial.println("initialization failed!");
      return;
    }
      titoData = SD.open("titoData.csv", FILE_WRITE);
    if (titoData)
    {
      XbeeSerial.println("titoData file found...");
    }
          XbeeSerial.println("");
          XbeeSerial.println("Select a function in the recommended order:");
          XbeeSerial.println("h = Health, c = Control, g = GPS, s = SD card, ");
          XbeeSerial.println("t = TCM2 Calibration, n = Heading, m = Start Mission");
          XbeeSerial.println("");
}
uint32_t timer = millis();

void loop()
{
   if (XbeeSerial.available() > 0)
  {
```

```
    keyhole = (char)XbeeSerial.read();
    switch (keyhole)
    {
      case 'h':
      Health_Report();
      break;
      case'n':
      Heading_Check();
      break;
      case 'c':
      Control_Report();
      break;
      case 'g':
      GPS_Report();
      break;
      case 's':
      SD_Card();
      break;
      case'm':
      Control_Loopb();
      break;

    }
  }
}

void Control_Loop()
{
  User_Input_X();
  XbeeSerial.println("");
  XbeeSerial.println("10 seconds to Thruster Start");
  XbeeSerial.println("");
  delay(10000);
  thruster.write(userThrust);

      //Leg A
       starttime = millis();
       endtime = starttime;
       while ((endtime - starttime) <=interval) // do this loop for up to 1000mS
        {
          Time_Loop();
          Health_Conversion_Rates();
          IMU_Loop();
          TCM2_Loop();
          SD_Card();

          splaneAdjustment = (float)(userDepth2 - depth4); //Output(error) = Setpoint - Input
          float splaneAdjustmentgain = (splaneAdjustment*0.4);//gain
          float x = constrain(splaneAdjustmentgain, -10.0, 10.0);
          float f = (0.00025512*x*x*x*x);
          float g = (0.03702397*x*x*x);
          float h = (0.71319701*x*x);
          float i = (22.07402776*x);
          float j = 1661.95340080;
          int Splane_y = (int)(f - g + h - i + j);
          splane.write(Splane_y);

          rudderAdjustment = (float)(userHeading1 - headingTcm2); //Output(error) = Setpoint - Input
          float rudderAdjustmentgain = (rudderAdjustment*0.4);//gain
          float z =constrain(rudderAdjustmentgain, -10, 10);
          float a = (0.00187168*z*z*z*z);
          float b = (0.04999155*z*z*z);
          float c = (0.26418940*z*z);
          float d = (30.01342264*z);
          float e = 1607.25665254;
          int Rudder_y = (int)(a - b + c - d + e);
          rudder.write(Rudder_y);

          loopcount = loopcount+1;
```

```
        endtime = millis();
       }

      splane.write(1250);
      thruster.write(1900);
      delay(10000);

  XbeeSerial.println("Stop Thruster");
  XbeeSerial.println("");
  thruster.write(1500);
  XbeeSerial.println("End of Mission");
  XbeeSerial.println("");
         while ((endtime - starttime) <=(RetrievalTime)) // do this loop for 10 minutes
       {

        GPS_Report();

        loopcount = loopcount+1;
        endtime = millis();
       }
}
void Control_Loopb()
{
  User_Input();
  XbeeSerial.println("10 seconds to Thruster Start\n");
  delay(10000);
  thruster.write(userThrust);

      starttime = millis();
      endtime = starttime;
        //1st Leg, Diving at user heading

      while ((endtime - starttime) <=interval) // do this loop for up to 1000mS
       {


        Time_Loop();
        Health_Conversion_Rates();
        PX2_Loop();
        IMU_Loop();
        TCM2_Loop();
        SD_Card();

        splaneAdjustment = (float)(userDepth2 - depth4); //Output(error) = Setpoint - Input
        float splaneAdjustmentgain = (splaneAdjustment*0.4);//gain
        float x = constrain(splaneAdjustmentgain, -10.0, 10.0);
        float f = (0.00025512*x*x*x*x);
        float g = (0.03702397*x*x*x);
        float h = (0.71319701*x*x);
        float i = (22.07402776*x);
        float j = 1661.95340080;
        int Splane_y = (int)(f - g + h - i + j);
        splane.write(Splane_y);

        rudderAdjustment = (float)(userHeading1 - headingTcm2); //Output(error) = Setpoint - Input
        float rudderAdjustmentgain = (rudderAdjustment*0.4);//gain
        float z =constrain(rudderAdjustmentgain, -10, 10);
        float a = (0.00187168*z*z*z*z);
        float b = (0.04999155*z*z*z);
        float c = (0.26418940*z*z);
        float d = (30.01342264*z);
        float e = 1607.25665254;
        int Rudder_y = (int)(a - b + c - d + e);
        rudder.write(Rudder_y);

        loopcount = loopcount+1;
        endtime = millis();
       }
       //Make 1st turn
```

```
    while ((endtime - starttime) <=(interval*2)) // do this loop for up to 1000mS
     {


       Time_Loop();
       Health_Conversion_Rates();
       PX2_Loop();
       IMU_Loop();
       TCM2_Loop();
       SD_Card();

       splaneAdjustment = (float)(userDepth2 - depth4); //Output(error) = Setpoint - Input
       float splaneAdjustmentgain = (splaneAdjustment*0.4);//gain
       float x = constrain(splaneAdjustmentgain, -10.0, 10.0);
       float f = (0.00025512*x*x*x*x);
       float g = (0.03702397*x*x*x);
       float h = (0.71319701*x*x);
       float i = (22.07402776*x);
       float j = 1661.95340080;
       int Splane_y = (int)(f - g + h - i + j);
       splane.write(Splane_y);

       rudderAdjustment = (float)(userHeading2 - headingTcm2); //Output(error) = Setpoint - Input
       float rudderAdjustmentgain = (rudderAdjustment*0.4);//gain
       float z =constrain(rudderAdjustmentgain, -10, 10);
       float a = (0.00187168*z*z*z*z);
       float b = (0.04999155*z*z*z);
       float c = (0.26418940*z*z);
       float d = (30.01342264*z);
       float e = 1607.25665254;
       int Rudder_y = (int)(a - b + c - d + e);
       rudder.write(Rudder_y);

       loopcount = loopcount+1;
       endtime = millis();
     }
//Make 2nd turn

    while ((endtime - starttime) <=(interval*3)) // do this loop for up to 1000mS
     {


       Time_Loop();
       Health_Conversion_Rates();
       PX2_Loop();
       IMU_Loop();
       TCM2_Loop();
       SD_Card();

       splaneAdjustment = (float)(userDepth2 - depth4); //Output(error) = Setpoint - Input
       float splaneAdjustmentgain = (splaneAdjustment*0.4);//gain
       float x = constrain(splaneAdjustmentgain, -10.0, 10.0);
       float f = (0.00025512*x*x*x*x);
       float g = (0.03702397*x*x*x);
       float h = (0.71319701*x*x);
       float i = (22.07402776*x);
       float j = 1661.95340080;
       int Splane_y = (int)(f - g + h - i + j);
       splane.write(Splane_y);

       rudderAdjustment = (float)(userHeading3 - headingTcm2); //Output(error) = Setpoint - Input
       float rudderAdjustmentgain = (rudderAdjustment*0.4);//gain
       float z =constrain(rudderAdjustmentgain, -10, 10);
       float a = (0.00187168*z*z*z*z);
       float b = (0.04999155*z*z*z);
       float c = (0.26418940*z*z);
       float d = (30.01342264*z);
       float e = 1607.25665254;
```

```
    int Rudder_y = (int)(a - b + c - d + e);
    rudder.write(Rudder_y);

    loopcount = loopcount+1;
    endtime = millis();
   }
  //Make 3rd turn
  while ((endtime - starttime) <=(interval*4)) // do this loop for up to 1000mS
   {


    Time_Loop();
    Health_Conversion_Rates();
    PX2_Loop();
    IMU_Loop();
    TCM2_Loop();
    SD_Card();

    splaneAdjustment = (float)(userDepth2 - depth4); //Output(error) = Setpoint - Input
    float splaneAdjustmentgain = (splaneAdjustment*0.4);//gain
    float x = constrain(splaneAdjustmentgain, -10.0, 10.0);
    float f = (0.00025512*x*x*x*x);
    float g = (0.03702397*x*x*x);
    float h = (0.71319701*x*x);
    float i = (22.07402776*x);
    float j = 1661.95340080;
    int Splane_y = (int)(f - g + h - i + j);
    splane.write(Splane_y);

    rudderAdjustment = (float)(userHeading4 - headingTcm2); //Output(error) = Setpoint - Input
    float rudderAdjustmentgain = (rudderAdjustment*0.4);//gain
    float z =constrain(rudderAdjustmentgain, -10, 10);
    float a = (0.00187168*z*z*z*z);
    float b = (0.04999155*z*z*z);
    float c = (0.26418940*z*z);
    float d = (30.01342264*z);
    float e = 1607.25665254;
    int Rudder_y = (int)(a - b + c - d + e);
    rudder.write(Rudder_y);

    loopcount = loopcount+1;
    endtime = millis();
   }
   //Make 4th turn
  while ((endtime - starttime) <=(interval*5)) // do this loop for up to 1000mS
   {


    Time_Loop();
    Health_Conversion_Rates();
    PX2_Loop();
    IMU_Loop();
    TCM2_Loop();
    SD_Card();

    splaneAdjustment = (float)(userDepth2 - depth4); //Output(error) = Setpoint - Input
    float splaneAdjustmentgain = (splaneAdjustment*0.4);//gain
    float x = constrain(splaneAdjustmentgain, -10.0, 10.0);
    float f = (0.00025512*x*x*x*x);
    float g = (0.03702397*x*x*x);
    float h = (0.71319701*x*x);
    float i = (22.07402776*x);
    float j = 1661.95340080;
    int Splane_y = (int)(f - g + h - i + j);
    splane.write(Splane_y);

    rudderAdjustment = (float)(userHeading1 - headingTcm2); //Output(error) = Setpoint - Input
    float rudderAdjustmentgain = (rudderAdjustment*0.4);//gain
    float z =constrain(rudderAdjustmentgain, -10, 10);
```

```
     float a = (0.00187168*z*z*z*z);
     float b = (0.04999155*z*z*z);
     float c = (0.26418940*z*z);
     float d = (30.01342264*z);
     float e = 1607.25665254;
     int Rudder_y = (int)(a - b + c - d + e);
     rudder.write(Rudder_y);

     loopcount = loopcount+1;
     endtime = millis();
    }

   splane.write(1250);
   thruster.write(1900);
   delay(10000);
   XbeeSerial.println("Stop Thruster\n");
   thruster.write(1500);
   XbeeSerial.println("End of Mission\n");
   XbeeSerial.println("Sending GPS Fix, Come Get Me!!");

//Transmit GPS
       while ((endtime - starttime) <=(RetrievalTime)) // do this loop for 10 minutes
    {

     GPS_Report();

     loopcount = loopcount+1;
     endtime = millis();
    }
}

void Control_Report()
{
  XbeeSerial.println("Tito is preforming a Control Report");
  XbeeSerial.println("");
  XbeeSerial.println("Rudder Sweep");
 //Rudder is neutral at 1500
  rudder.write(1500);
  delay(2000);
  rudder.write(2000);
  delay(2000);
  rudder.write(1000);
  delay(2000);
  rudder.write(1500);
  delay(2000);
  XbeeSerial.println("Splane Sweep");
  //Splane is neutral at 1700
  splane.write(1700);
  delay(2000);
  splane.write(1000);
  delay(2000);
  splane.write(2000);
  delay(2000);
  splane.write(1700);
  delay(2000);
   XbeeSerial.println("Thruster Sweep");
  //Thruster stopped at 1500
  thruster.write(1500);
  delay(1000);
  thruster.write(1700);
  delay(2000);
  thruster.write(1800);
  delay(2000);
  thruster.write(1900);
  delay(2000);
  thruster.write(2000);
  delay(2000);
  thruster.write(1500);
  XbeeSerial.println("Control Report Complete");
```

```
}
void GPS_Loop()
{
  // Declare the averaged lat/lon as local variables (8 bytes, double precision!!!)
  char c = GPS.read();
  if (GPS.newNMEAreceived())
    {
      if (!GPS.parse(GPS.lastNMEA()))
    return;
    }

}
void GPS_Report()
{
    GPS_Loop();
if (timer > millis()) timer = millis();
  // Every second, print the current lat/long
if (millis() - timer > 1000)
  {
 timer = millis();
      if((GPS.latitude != 0.0) && (GPS.longitude !=0.0))
      {
         XbeeSerial.print("Latitude ");
         XbeeSerial.print(GPS.latitude,6);
         XbeeSerial.print(", ");
         XbeeSerial.print("Longitude ");
         XbeeSerial.print(GPS.longitude,6);
         XbeeSerial.print("\n");
         delay(1000);
      }
  }
}
void Heading_Check()
{
  TCM2_Loop();
  XbeeSerial.println(String(headingTcm2));
}
void Health_Conversion_Rates()
{
 AdcConversion = (((3.3)/4096)*10);  // The analog conversion for voltage
 ADC_Conversion = ((3.3/4095)*1000); //The analog conversion for mV
 AdcConversion2 = ((5/4095.0)*1000); // The analog conversion for 5.0 mV
 AdcConversion3 = (5/4095.0); // The analog conversion for 5.0 V

 batt1 = (analogRead(battery1)*AdcConversion);  // Converting the ADC of battery to volts and initializing variable
 batt2 = (analogRead(battery2)*AdcConversion);
 batt3 = (analogRead(battery3)*AdcConversion);

 leak1 = analogRead(leakSensor1);           // Reading the ADC of the leak sensor
 leak2 = analogRead(leakSensor2);

 Temp = (((analogRead(temperatureSensor)*ADC_Conversion)-500)/10);
 Temp2 = ((Temp*(1.8))+32);

 TCurrent = (((analogRead(thrusterCurrent)*(ADC_Conversion))-1650.0)/110.0);
 RCurrent = (((analogRead(rudderCurrent)*AdcConversion2)-2500)/185.0);
 SCurrent = (((analogRead(splaneCurrent)*AdcConversion2)-2500)/185.0);

 depth2=(analogRead(depth)*AdcConversion3);
 depth3=((25*depth2)-12.5);
 depth4=(double)((depth3-14.7)/1.47);

 raw_pressure = analogRead(pressureSensor);
 raw_pressure2 = raw_pressure*(5/4095.0);
 pressure2 = ((54.2*raw_pressure2)-10.84);
}
void Health_Report()
{
 Health_Conversion_Rates();
```

```
 Time_Loop();

XbeeSerial.println("Tito Health Report:");
XbeeSerial.print("Elapsed Time (minutes) = ");
XbeeSerial.println(previousMillis/60000.0);
XbeeSerial.print("Battery 1,2,3 [ V ] = ");
XbeeSerial.print(batt1-batt2,2); XbeeSerial.print(",  ");
XbeeSerial.print(batt2-batt3,2); XbeeSerial.print(",  ");
XbeeSerial.println(batt3,2); //Serial.print("");
XbeeSerial.print("Thruster Current [ A ] = ");
XbeeSerial.println(TCurrent,3); //Serial.print("");
XbeeSerial.print("Rudder Current [ A ] = ");
XbeeSerial.print(RCurrent,3); XbeeSerial.print(", ");
XbeeSerial.print("Splane Current [ A ] = ");
XbeeSerial.println(SCurrent,3);
XbeeSerial.print("Front Leak sensor = ");
XbeeSerial.print(leak2); XbeeSerial.print(", ");
XbeeSerial.print("Back Leak sensor = ");
XbeeSerial.println(leak1);// Serial.print("");
XbeeSerial.print("Temperature [ F ] = ");
XbeeSerial.print(Temp2,2); XbeeSerial.print(",  ");
XbeeSerial.print("Internal Pressure [ kPa ] = ");
XbeeSerial.println(pressure2,2); //Serial.print("");
XbeeSerial.println("");
}
void IMU_Loop()
{
  /* Calculate pitch and roll from the raw accelerometer data */
 accelerometer.getEvent(&accel_event);
  if (dof.accelGetOrientation(&accel_event, &orientation));
  /* Calculate the heading using the magnetometer */
  magnetometer.getEvent(&mag_event);
  if (dof.magGetOrientation(SENSOR_AXIS_Z, &mag_event, &orientation));
  /* Display the results (acceleration is measured in m/s^2) */
  accelerometer.getEvent(&event);
  /* Display the results (magnetic vector values are in micro-Tesla (uT)) */
  magnetometer.getEvent(&event);
  /* Display the results (gyrocope values in rad/s) */
  gyroscope.getEvent(&event);
}
void PX2_Loop()
{
depth2=(analogRead(depth)*AdcConversion3);
depth3=((25*depth2)-12.5);
depth4=((depth3-14.7)/1.47);
}
void SD_Card()
{

   String dataString = String(loopcount) + ", " +String(previousMillis/60000.0) + ", "+String(hertz) + ", " +String(batt3) + ", " +String((batt2-
batt3)) + ", "+String((batt1-batt2)) + ", " +String(leak1) + ", " +String(leak2) + ", " +String(Temp2) + ", " +String(pressure2) + ", "
+String(TCurrent,4) + ", " +String(RCurrent,4) + ", "+String(SCurrent,4) + ", " + String(depth4) + ", " + String(headingTcm2) + ", " +
String(GPS.latitude,6) + ", "+ String(GPS.longitude,6) + ", " + String(event.gyro.x) + ", " + String(event.gyro.y)+ ", " +String(event.gyro.z) + ", "
+String(orientation.roll) + ", "+ String(orientation.pitch) + ", "+ String(orientation.heading) + ", " + String(event.acceleration.x) + ", " +
String(event.acceleration.y)+", " + String(event.acceleration.z);

 File titoData = SD.open("titoData.csv", FILE_WRITE);
 titoData.println(dataString);
 titoData.close();
}
void Time_Loop()
{
      unsigned long currentMillis = millis();

  elapsedMillis = currentMillis - previousMillis;
    if(currentMillis - previousMillis > intervalMillis)
    {
      previousMillis = currentMillis;
    }
```

```
 hertz = ((1/(elapsedMillis))*1000);
}
void User_Input()
{
 /*****User input******/
   XbeeSerial.println("Tito's User Defined Mission Paremeters");
   XbeeSerial.println("");
/****Depth****/
   XbeeSerial.println("Enter value for Mission Depth in meters");
   delay(5000);
   while (XbeeSerial.available() == 0) ;  // Wait here until input buffer has a character
    {
      userDepth2 = (double)XbeeSerial.parseFloat();
      XbeeSerial.print("Depth = ");
      XbeeSerial.print(userDepth2);
      XbeeSerial.println(" meters ");
      XbeeSerial.println("");

      while (XbeeSerial.available() > 0)  // .parseFloat() can leave non-numeric characters
        {
          junk = XbeeSerial.read() ; // clear the keyboard buffer
        }
    }
/****Heading****/
      XbeeSerial.println("Enter value for Heading 1 in degrees");
    delay(5000);
    while (XbeeSerial.available() == 0) ;
    {
    userHeading1 = XbeeSerial.parseFloat();
    XbeeSerial.print("Heading 1 = ");
    XbeeSerial.print(userHeading1);
    XbeeSerial.println(" degrees\n");

      while (XbeeSerial.available() > 0)  // .parseFloat() can leave non-numeric characters
        {
        junk = XbeeSerial.read() ;
        }      // clear the keyboard buffer
    }
     XbeeSerial.println("Enter value for Heading 2 in degrees");
    delay(5000);
    while (XbeeSerial.available() == 0) ;
    {
    userHeading2 = XbeeSerial.parseFloat();
    XbeeSerial.print("Heading 2 = ");
    XbeeSerial.print(userHeading2);
    XbeeSerial.println(" degrees\n");

      while (XbeeSerial.available() > 0)  // .parseFloat() can leave non-numeric characters
        {
        junk = XbeeSerial.read() ;
        }      // clear the keyboard buffer
    }
  XbeeSerial.println("Enter value for Heading 3 in degrees");
    delay(5000);
    while (XbeeSerial.available() == 0) ;
    {
    userHeading3 = XbeeSerial.parseFloat();
    XbeeSerial.print("Heading 3 = ");
    XbeeSerial.print(userHeading3);
    XbeeSerial.println(" degrees\n");

      while (XbeeSerial.available() > 0)  // .parseFloat() can leave non-numeric characters
        {
        junk = XbeeSerial.read() ;
        }      // clear the keyboard buffer
    }
     XbeeSerial.println("Enter value for Heading 4 in degrees");
    delay(5000);
    while (XbeeSerial.available() == 0) ;
```

```
  {
  userHeading4 = XbeeSerial.parseFloat();
  XbeeSerial.print("Heading 4 = ");
  XbeeSerial.print(userHeading4);
  XbeeSerial.println(" degrees\n");

   while (XbeeSerial.available() > 0)  // .parseFloat() can leave non-numeric characters
    {
    junk = XbeeSerial.read() ;
    }     // clear the keyboard buffer
  }




/****Interval****/
  XbeeSerial.println("Enter value for Interval (seconds)");
  delay(5000);
  while (XbeeSerial.available() == 0) ;
   {
    userInterval = XbeeSerial.parseFloat();
    XbeeSerial.print("Interval = ");
    XbeeSerial.print(userInterval);
    XbeeSerial.println(" seconds ");
    XbeeSerial.println("");
    interval = userInterval*1000;
    while (XbeeSerial.available() > 0)  // .parseFloat() can leave non-numeric characters
     {
      junk = XbeeSerial.read() ;
     }     // clear the keyboard buffer
   }
/****Thrust****/
   XbeeSerial.println("Enter value for Thrust (Reverse 1000, Stop 1500, Full 2000)");
   delay(10000);
   while (XbeeSerial.available() == 0) ;
   {
  userThrust = XbeeSerial.parseFloat();
  userThrust = (userThrust);
  XbeeSerial.print("Thrust (Reverse 1000, Stop 1500, Full 2000) = ");
  XbeeSerial.print(userThrust);
  XbeeSerial.println("");

   while (XbeeSerial.available() > 0)  // .parseFloat() can leave non-numeric characters
    {
    junk = XbeeSerial.read() ;
    }     // clear the keyboard buffer
  }
  /****Retrival Time****/
   XbeeSerial.println("Enter retrieval time (minutes)");
   delay(10000);
   while (XbeeSerial.available() == 0) ;
   {
  RetrievalTime = XbeeSerial.parseFloat();
  RetrievalTime = (RetrievalTime*60000);
  XbeeSerial.print("GPS will transmit for  = minutes ");
  XbeeSerial.print(RetrievalTime);
  XbeeSerial.println("");

   while (XbeeSerial.available() > 0)  // .parseFloat() can leave non-numeric characters
    {
    junk = XbeeSerial.read() ;
    }     // clear the keyboard buffer
  }
}
void User_Input_X()
{
 /*****User input******/
  XbeeSerial.println("Tito's User Defined Mission Paremeters");
  XbeeSerial.println("");
```

```
/****Depth****/
  XbeeSerial.println("Enter value for Mission Depth in meters");
  delay(5000);
  while (XbeeSerial.available() == 0) ;  // Wait here until input buffer has a character
   {
    userDepth2 = (double)XbeeSerial.parseFloat();
    XbeeSerial.print("Depth = ");
    XbeeSerial.print(userDepth2);
    XbeeSerial.println(" meters ");
    XbeeSerial.println("");

    while (XbeeSerial.available() > 0)  // .parseFloat() can leave non-numeric characters
      {
       junk = XbeeSerial.read() ; // clear the keyboard buffer
      }
   }
/****Heading****/
    XbeeSerial.println("Enter value for Heading 1 in degrees");
   delay(5000);
   while (XbeeSerial.available() == 0) ;
   {
   userHeading1 = XbeeSerial.parseFloat();
   XbeeSerial.print("Heading 1 = ");
   XbeeSerial.print(userHeading1);
   XbeeSerial.println(" degrees\n");

    while (XbeeSerial.available() > 0)  // .parseFloat() can leave non-numeric characters
      {
     junk = XbeeSerial.read() ;
      }      // clear the keyboard buffer
   }

/****Interval****/
   XbeeSerial.println("Enter value for Interval (seconds)");
   delay(5000);
   while (XbeeSerial.available() == 0) ;
    {
     userInterval = XbeeSerial.parseFloat();
     XbeeSerial.print("Interval = ");
     XbeeSerial.print(userInterval);
     XbeeSerial.println(" seconds ");
     XbeeSerial.println("");
     interval = userInterval*1000;
     while (XbeeSerial.available() > 0)  // .parseFloat() can leave non-numeric characters
       {
        junk = XbeeSerial.read() ;
       }      // clear the keyboard buffer
    }
/****Thrust****/
    XbeeSerial.println("Enter value for Thrust (Reverse 1000, Stop 1500, Full 2000)");
    delay(10000);
    while (XbeeSerial.available() == 0) ;
     {
   userThrust = XbeeSerial.parseFloat();
   userThrust = (userThrust);
   XbeeSerial.print("Thrust (Reverse 1000, Stop 1500, Full 2000) = ");
   XbeeSerial.print(userThrust);
   XbeeSerial.println("");

    while (XbeeSerial.available() > 0)  // .parseFloat() can leave non-numeric characters
      {
     junk = XbeeSerial.read() ;
      }      // clear the keyboard buffer
   }
   /****Retrival Time****/
    XbeeSerial.println("Enter retrieval time (minutes)");
    delay(10000);
    while (XbeeSerial.available() == 0) ;
     {
```

```
    RetrievalTime = XbeeSerial.parseFloat();
    RetrievalTime = (RetrievalTime*60000);
    XbeeSerial.print("GPS will transmit for  = minutes ");
    XbeeSerial.print(RetrievalTime);
    XbeeSerial.println("");

      while (XbeeSerial.available() > 0)  // .parseFloat() can leave non-numeric characters
        {
        junk = XbeeSerial.read() ;
        }      // clear the keyboard buffer
    }
}
void Vital_Loop()
{
  String vitalString =String(hertz)+" Dpt "+String(depth4)+", "+String(splaneAdjustment)+" TCM2 "+String(headingTcm2)+",
"+String(rudderAdjustment)+" GPS "+String(GPS.latitude,6)+", "+String(GPS.longitude,6);
  XbeeSerial.println(vitalString);
}

  int readRadio(void)
{
  while (true) {
  if (XbeeSerial.available() > 0 )  // see if data is available by reading the xbee
  {
    char Red = XbeeSerial.read();                 // store data read from serial input to the integer Red
    if (isDigit(Red))                     // if Red is a digit then it will be read
    {
      st += (char)Red;                       // cast each of the integers of Red as a character and impliment each character into a string called st
    }
    if (Red == '\r')                    // If you are at the end of the string you do a character return
    {
      XbeeSerial.println(st);
      int ret = st.toFloat();
      st = "";       //clears the string to a new input
      return ret;
    }
  }
}
}


void rudder_gain(void)
{
 int verify = 0;
  R_gain = 0;
  XbeeSerial.println("What would you like the R_Gain to be?");
  R_gain = readRadio();
  while (verify != 1)
  {
    XbeeSerial.println("Is this the correct R_Gain?");
    XbeeSerial.println(R_gain);
    XbeeSerial.println("press 1 for yes, 2 or greater for no");
    verify = readRadio();
    if (verify == 1)
    {
      XbeeSerial.println("The R_Gain has been succesfully changed");
    }
    if (verify != 1)
    {
     rudder_gain();
     return;
    }
  }
}

void splane_gain(void)
{
 int verify = 0;
  S_gain = 0;
```

```
  XbeeSerial.println("What would you like the S_gain to be?");
  S_gain = readRadio();
  while (verify != 1)
  {
    XbeeSerial.println("Is this the correct S_gain?");
    XbeeSerial.println(S_gain);
    XbeeSerial.println("press 1 for yes, 2 or greater for no");
    verify = readRadio();
    if (verify == 1)
    {
      XbeeSerial.println("The S_gain has been succesfully changed");
    }
    if (verify != 1)
    {
     splane_gain();
     return;
    }
  }
}
```