Camera Stabilization

A system to mechanically stabilize and reduce unintended movement of a mounting platform through computational and electrical controls, intended for use with a camera.

Marit Brocker ● Mitchell Cook ● Caleb Disbrow ● Alex Wilson

Collaborators: Nathan Ostrout, Joshua Milas, Rick Tolleson, Robert Kraynik

12 May 2015

Table of Contents

[Table of Figures 3](#_Toc419027469)

[Overview 4](#_Toc419027470)

[Needs Statement 4](#_Toc419027471)

[Objective Statement 4](#_Toc419027472)

[Description 4](#_Toc419027473)

[Marketing Mockups 5](#_Toc419027474)

[Requirements Specifications 6](#_Toc419027475)

[Needs 6](#_Toc419027476)

[Engineering Specifications and Analysis 7](#_Toc419027477)

[Concept Selection 8](#_Toc419027478)

[Existing Systems and Concepts 8](#_Toc419027479)

[Benchmarking 8](#_Toc419027480)

[Rationale 9](#_Toc419027481)

[Design 10](#_Toc419027482)

[Overview 10](#_Toc419027483)

[Sensing 12](#_Toc419027484)

[Processing 13](#_Toc419027485)

[Actuation 13](#_Toc419027486)

[Power Management 15](#_Toc419027487)

[User Interface 16](#_Toc419027488)

[Engineering Standards 17](#_Toc419027489)

[Multidisciplinary Aspects 17](#_Toc419027490)

[Background 17](#_Toc419027491)

[Outside Contributors 17](#_Toc419027492)

[Constraints and Considerations 17](#_Toc419027493)

[Extensibility 17](#_Toc419027494)

[Manufacturability 17](#_Toc419027495)

[Reliability 18](#_Toc419027496)

[Societal and Economic Context 18](#_Toc419027497)

[Health and Safety Considerations 18](#_Toc419027498)

[Intellectual Properties 18](#_Toc419027499)

[Cost 19](#_Toc419027500)

[Testing Strategy 19](#_Toc419027501)

[Risks 20](#_Toc419027502)

[Schedule 22](#_Toc419027503)

[Gantt Chart 24](#_Toc419027504)

[Perspective 25](#_Toc419027505)

[Appendix 26](#_Toc419027506)

[A – Testing 26](#_Toc419027507)

[Unit Tests 26](#_Toc419027508)

[Integration Tests 37](#_Toc419027509)

[Acceptance Tests 41](#_Toc419027510)

# Table of Figures

[Figure 1 – Marketing mockup close-up 5](#_Toc419027511)

[Figure 2 – Marketing mockup in-use 6](#_Toc419027512)

[Figure 3 – Functional Decomposition 11](#_Toc419027513)

[Figure 4 – MPU-6050 block diagram 12](#_Toc419027514)

[Figure 5- Use of 120° offset in PWM control signals 13](#_Toc419027515)

[Figure 6 – Brushless gimbal motor drive schematic 14](#_Toc419027516)

[Figure 7 – Brushless gimbal motor drive PCB layout 14](#_Toc419027517)

[Figure 8 – Linear Fixed Voltage Regulator with Output Buffer and Error Correction 15](#_Toc419027518)

[Figure 9: User Interface Flow Diagram 16](file:///C:\Users\Marit%20Brocker\Downloads\CameraStabilizationII_revision_7.docx#_Toc419027519)

# Overview

## Needs Statement

Current systems for recording stable video can be costly, heavy, and difficult to get into small locations. While there are systems currently available, they are often expensive and are not readily available to amateur photographers. These work primarily through mechanical, rather than electrical and computational, corrections. Current systems are use-specific and are not easily able to switch between various shooting methods, such as handheld and vehicle mounted filming. A system is needed that is inexpensive, is modular, and makes use of electrical corrections to provide a more dynamically controlled camera stabilization method.

## Objective Statement

The objective of this project is to design and prototype a device that will reduce undesirable motion in video recording. The device will be mounted as part of a mechanical system that allows the user to specify the orientation of the camera and save that setting. The user will be able to select from several different modes of operation that will maintain a steady orientation toward a desired location or axis using computational corrections to the mechanical system. Once the user has selected an operating mode the user will be free to move the mount without disturbing the stabilization of the shoot.

## Description

The system will be a mechanical mounting platform which will be controlled by several stabilization components. The stabilization components will include actuation components which can adjust the camera based on the movement the system is experiencing. The movement of the system will be monitored by several input sensors to measure acceleration and direction of movement. Data will be analyzed by a processing unit and control algorithms, which will then feed back to the control mechanism and adjust the camera based on the system’s movement to stabilize it. The user will have controls to specify how the camera is stabilized by locking different axes. The frame will have a modular mounting bracket which will allow for using both a handheld attachment and vehicle mount.

## Marketing Mockups

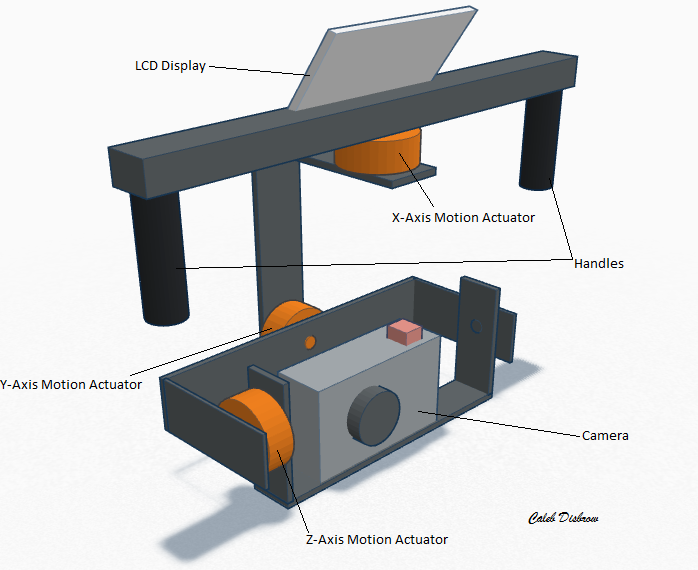


Figure – Marketing mockup close-up

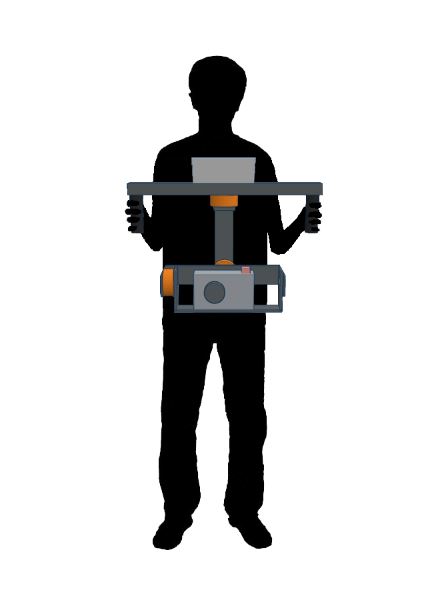


Figure – Marketing mockup in-use

# Requirements Specifications

The system is primarily intended to be carried by an individual for extended periods of time without excessive exertion, and thus needs to be lightweight and portable. The system should also reduce undesired movement along the axes of motion specified by the user.

## Needs

1. Lightweight
2. Portable
3. Dynamic mounting
4. Multiple axis control
5. Noticeable camera stabilization
6. Multiple user-settable modes
7. Easy to use

## Engineering Specifications and Analysis

|  |  |  |  |
| --- | --- | --- | --- |
| Customer Needs | Engineering Requirements | | Justification |
| 1,2,3,7 | A. | The system shall have a total mass less than 7 kg (15 lbs.) | A system with a mass of 7 kg should be light enough for a user to carry with minimal effort and be mountable to a vehicle with minimal strain to the mounting bracket. Based on system components this is a reasonable estimate. |
| 2, 7 | B. | The system shall be able to be powered for a minimum of 2 hours. | The user should be able to use the system for a reasonable amount of time, ideally approaching the battery life of the camera. This time is similar to existing products. |
| 2, 7 | C. | The system shall be within the dimensions of 0.5 m x 0.5 m x 0.5 m (1.6 ft.) | This size is approximately the same as existing systems and allows for a system that is not unwieldy for the user to hold and move. |
| 3, 7 | D. | The system shall have a universal connecting point for mounting. | The user not only should be able to mount the system to a variety of platforms from a mounting bracket but also should be able to use it in a handheld scenario. |
| 4 | E. | The system shall implement a 3-axis control system with a viewing range of ±160° for the pan (X axis) and tilt (Y axis), and ±50° for the pitch (Z axis). | These requirements provide a full viewing range in one direction while providing a small amount of clearance for the frame, and preventing any actuation components from over extending. |
| 4 | F. | The 3-axis control system shall be able to support a load of 1.5 kg. | This weight will allow for supporting a point and shoot camera and a small amount of mounting for the camera. The load is defined as a camera or other recording device added to the system, and does not include the minimal mounting platform. |
| 4 | G. | The system shall implement a 3-axis sensor control system for monitoring orientation. | This system will allow for monitoring 3-directional acceleration and angle in order to adjust the camera. |
| 5 | H. | The system shall stray no further than ±2° in any direction. | Based on existing systems this provides a reasonable tolerance. |
| 6,2,7 | I. | The system shall contain an integrated input device that allows the user to lock (stabilize) or unlock (freely move) each axis. | This allows for quick and easy changing of modes to stabilize different axes. The system will be integrated and will not need to be connected to a computer to adjust its mode. |

# Concept Selection

The following systems were surveyed as examples of existing stabilization systems with varying prices and functionality. Surveyed characteristics include price, weight, stabilization method, and number of stabilized axes.

## Existing Systems and Concepts

|  |  |  |
| --- | --- | --- |
|  | |  |
| Steadicam Merlin 2 | | $399 |
| Weight: | 1.4 lbs. | |
| Supported Load: | 5 lbs. | |
| Stabilization Method: | Metal gimbal/counterweighting | |
| Axes: | 2-axis (tilt, pan) | |
| Other: | Axis locking, manual adjustment | |
| Varizoom BlackHawk-VL | | $6200 |
| Weight: | 23 lbs. total system and harness | |
| Supported Load: | 10-25 lbs. camera | |
| Stabilization Method: | 3-axis mechanical gimbal mount, spring/counterweight based, attaches to vest | |
| Axes: | 3-axis | |
| Other: |  | |
| Run Movie RTR DIY Kit | | $975 |
| Weight: | 3 kg | |
| Supported Load: | 1.5 kg camera | |
| Stabilization Method: | 3-axis stabilization, electronic locking | |
| Axes: | 3-axis | |
| Other: | 5208 200T motors | |

## Benchmarking

Three existing systems were analyzed, and it was found that no one system satisfied both the desired price and desired functionality.

The Varizoom Blackhawk is an example of a high grade system currently used for professional recording applications. It operates mechanically and can support a much larger load than what is in the scope of this project. The system is vastly more expensive than the specified project budget, which makes it generally inaccessible to amateur videographers.

The Steadicam Merlin is within the project budget; however, it operates mechanically, is missing a third axis of control, and is difficult to adjust the sensitivity of the axes.

The Run Movie RTR DIY kit is comparable to this project in functionality and design, but the price is too high. It supports a similar size camera and uses electrical means of locking and stabilizing the camera, but at a higher price range.

Based on this research and the available expertise of the group, creating a system similar to the Run Movie RTR DIY kit, but for a lower price, will satisfy the customer needs. Additionally, the project team will be adding the ability to dynamically mount the system to make it a more applicable system for amateur use, which will reduce costs through modularity.

## Rationale

Stabilization method

Electric motors provide a compact and efficient way to independently control each axis of motion. Counterweights systems, like the Steadicam Merlin, are relatively heavy, are orientation dependent, and are difficult to stabilize without a long moment arm. Spring articulated systems, like the Varizoom Blackhawk, require large, heavy spring arms and are also orientation dependent. Conversely, electric motors can stabilize all three axes from a compact package and allow for mounting the system in any number of orientations. Using electric motors also aligns with the skill sets of the project team.

Number of axes

In order to fully implement a stabilization system for both taking still camera shots and panning video, it is beneficial to utilize all 3 axes in which the system might experience movement. Additionally, this provides a range of motion similar to that of a human. Two-axis stabilization systems, such as the Steadicam Merlin, still allow the user to move the camera sideways, which is undesirable in still photography.

Power

Lithium-polymer (Li-Po) rechargeable batteries are common in systems of this type, as seen in the Run Movie RTR DIY kit. Li-Po batteries provide a higher power-to-weight ratio than other rechargeable batteries, making them ideal for handheld use. For example, using NiMH rechargeable batteries to create a 10,000mAh battery with at least a 7.4 volt output would require seven 1.2 volt cells to be connected in series, producing 8.4 volt output. The Tenergy D size 10,000mAh cell provides 1.2 volts output per cell, weighs 5.76 ounces per cell, and has dimensions 60 mm x 33 mm. Seven cells would weigh 40.32 ounces and occupy approximately 457.4 cm3. A comparable Li-Po weighs 17.60 ounces and occupies 241.5 cm3. This means the Li-Po is roughly half the size and weight of the NiMH batteries, but provides comparable power.

Processing

The Arduino Due is a flexible, easy to program, and relatively inexpensive microcontroller. It supports the I2C protocol, has enough I/O to support motors, feedback, and a user interface, and it can power devices running at both 3.3 and 5V. Smaller devices, like the Teensy 3.1, lack the multiple output voltages and number of I/O pins available. The Due’s internal memory is more stable than external SD cards, like those used by the Raspberry Pi, which are susceptible to physical damage. Additionally, the small form factor and Arduino pin compatibility allows it to be used more compactly than devices such as the BeagleBone Black or TI Launchpads.

Sensors

Inertial measurement units combine accelerometers and gyroscopes onto a single board. This package makes them simple to mount and allows unified communication between the sensors and processing unit. Implementing the sensors individually would require either bulky breakout boards or custom designed ICs, greatly increasing the size and complexity of the sensor system.

Actuators

The product requires a high torque actuation component to properly stabilize the camera, and low audible noise to avoid disrupting video recording. Servo motors are precise, but have low holding torque and can be loud during actuation. Stepper motors have high holding torque and are quiet, but can move only in set intervals, potentially adding jagged movement to the system. DC brushless gimbal motors are quiet, and allow for high precision and torque in a compact form factor. Hall effect feedback sensors reinforce DC motors to provide increased precision and sensitivity.

Price

Ideally the final system will cost between $300 and $400 which will keep the project within its operating budget. Staying at the lower end of this range will allow for flexibility should the product require higher quality hardware. For a 3-axis electrical system, this is less expensive than most professional grade systems.

# Design

## Overview

The system has five main subsystems: sensing, processing, actuation, power management, and user interface (Figure 3.) The sensing subsystem captures data detailing the orientation and movement of the system which it then communicates to the processing subsystem. The processing subsystem performs the stabilization control algorithm and provides the necessary output signals to correct for undesired motion. The actuation subsystem receives these data and independently moves the three axes of the frame. The power management subsystem provides the current and voltage needed for each subsystem as well as diagnostic data to avoid damaging the system and inform the user of the remaining charge. The user interface allows the user to change how the system functions and displays data about the current state of the system.

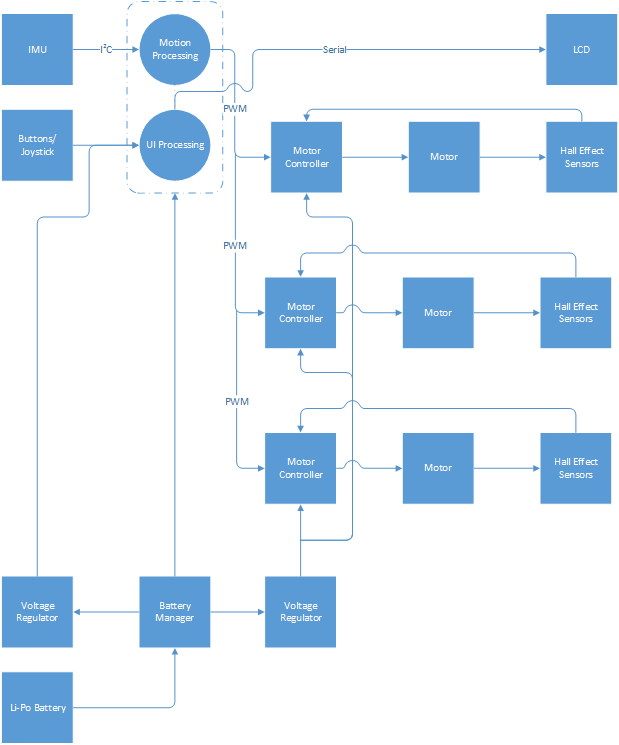


Figure 3 – Functional Decomposition

## Sensing

The sensing subsystem consists of two MPU-6050 inertial measurement units (IMU) which communicate using the I2C protocol with the processing subsystem (Figure 4.) An IMU contains three single-axis accelerometers and three single-axis gyroscopes to measure acceleration and angular momentum in all three axes of motion. These values are converted to digital signals using onboard analog-to-digital converters, processed and packaged as I2C signals, and sent to the microcontroller (MCU) for processing. The IMU and MCU communicate using two lines: I2C data (SDA) and clock (SCLK). The IMU also receives power directly from the MCU. The placement of the two IMUs allows for error correction; one IMU sits in the upper housing to detect motion that the system is experiencing, the second IMU sits on the camera platform to detect the amount that the system needs to adjust to return to its neutral position. Several multidisciplinary skills may be utilized to process the data that the IMU collects. Mechanical engineering and physics skills will be applied to accurately determine the state of the system based on the motion it may be experiencing in 3 axes.

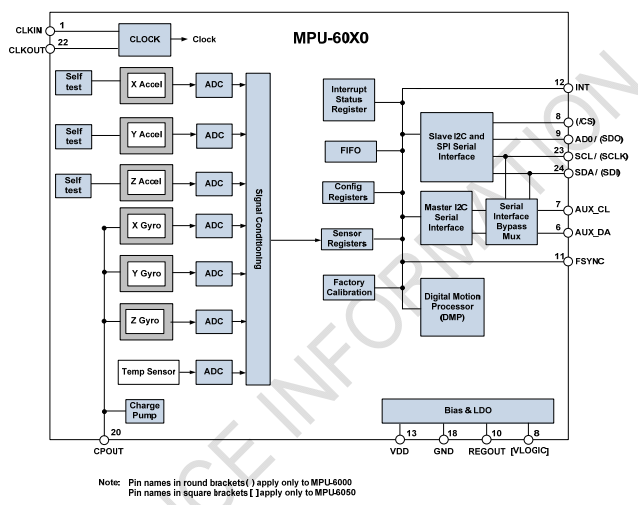


Figure 4 – MPU-6050 block diagram

(MPU-6050 datasheet)

## Processing

The processing subsystem consists of an Arduino Due. This microcontroller uses a 32-bit ARM Cortex-M3 CPU. The stabilization control algorithms are calculated on the microcontroller to provide the information necessary to provide motion correction in the system. The signals received by the controller are an I2C serial link from the IMU in the sensing subsystem, analog input from the controls in the user interface subsystem, and power and diagnostic data from the power management subsystem. The microcontroller sends an eight-bit control signal to the display in the user interface subsystem, DC control signals to the motor controllers in the actuation subsystem, and power to the user interface and sensing subsystems.

## Actuation

The actuation subsystem consists of two components, the Turnigy HD 5208 brushless three-phase gimbal motor and the L6235 motor controller. Figure 6 shows the control circuit used with the controller, adapted from the L6235 application note from STMicroelectronics. Figure 7 shows the final layout designed in Eagle CAD for fabrication on a PCB. The motor controller accepts control signals from the processing subsystem in the form of DC signals, which translate into forward/reverse control and a motor lock/brake control. The chip requires three Hall effect sensors to provide positional feedback to the motor controller by measuring the magnetic flux generated by the motor. The Hall effect sensors are placed around the motor at 120° intervals. The three-phase brushless motors require 3 separate PWM signals with an offset of 120°, as seen in Figure 5, which are provided by the brushless motor controller. These control signals sequentially power the motor windings to produce an electromagnetic field which applies a continuous, shifting force on the motor armature. This subsystem is duplicated three times to dynamically control 3 axes.

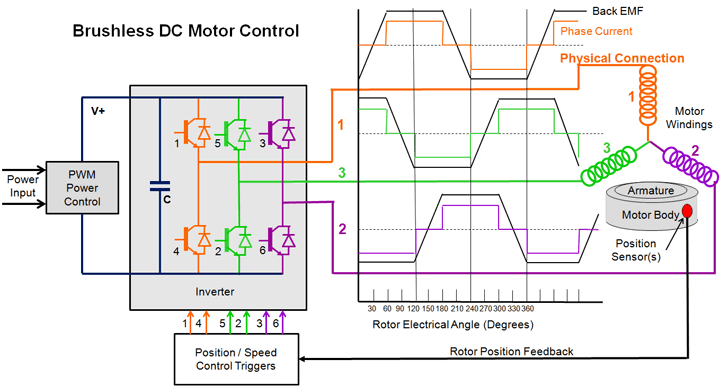


Figure - Use of 120° offset in PWM control signals

(http://www.mpoweruk.com/motorsbrushless.htm)

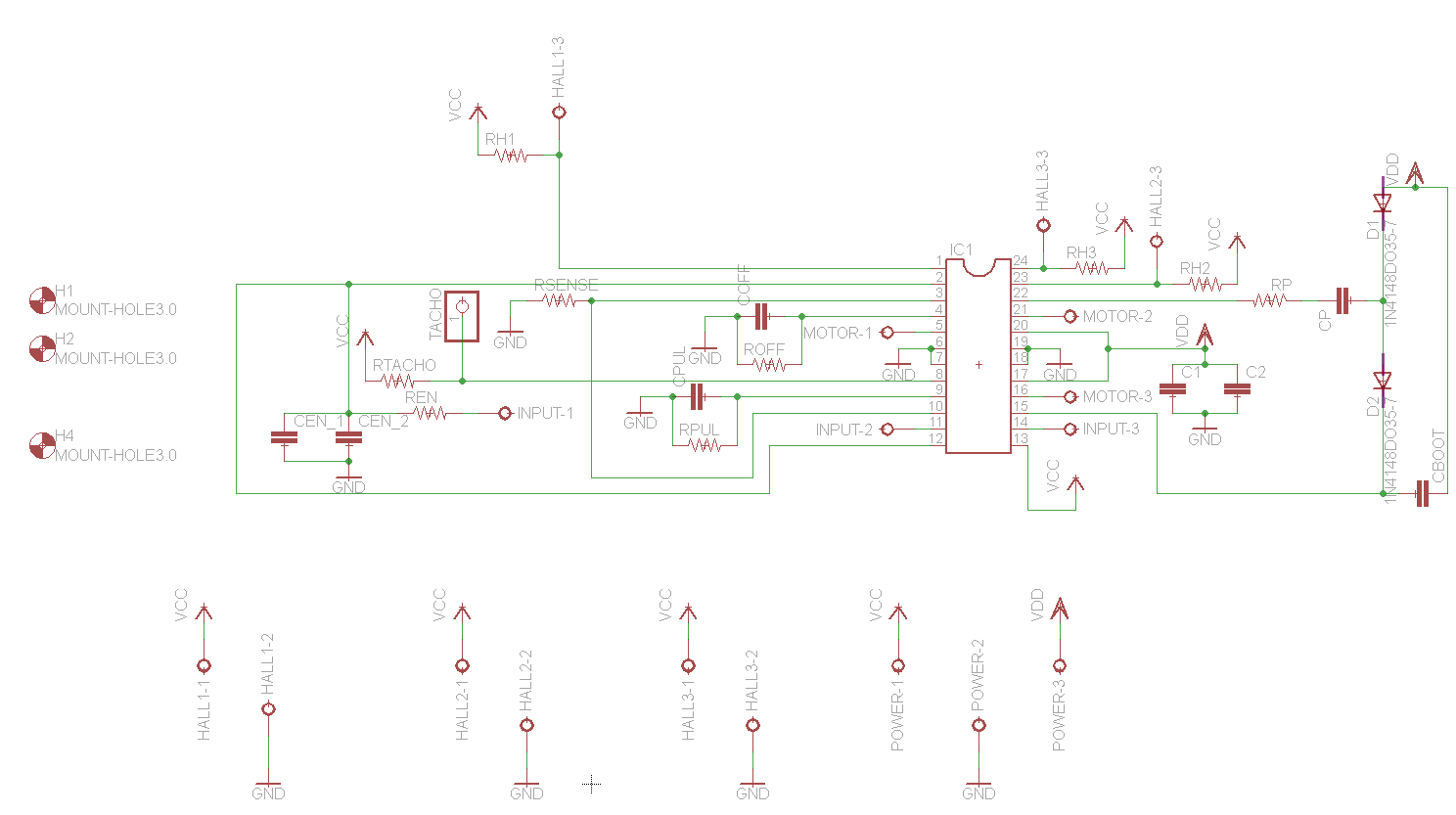


Figure 6 – Brushless gimbal motor drive schematic

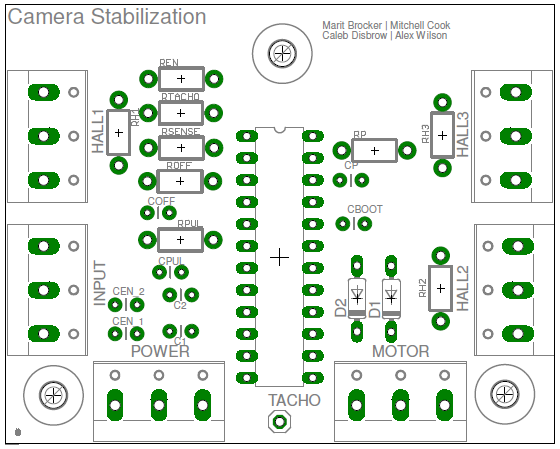


Figure – Brushless gimbal motor drive PCB layout

## Power Management

The power management subsystem consists of a 20,00mAh Li-Po battery, 9 volt and 12 volt step-up voltage regulation circuits, and a battery monitor system. The linear voltage regulators will transform the power provided by the Li-Po battery into the necessary voltage needed by the actuation and processing subsystems (Figure 8.) The user interface and sensing subsystems will be powered through the processing subsystem.

The step-up voltage regulators used where Pololu U3V50F9 and U3V50F12 for 9v and 12v transformation, respectively. The basic design of a linear voltage regulator can be seen in Figure 9.

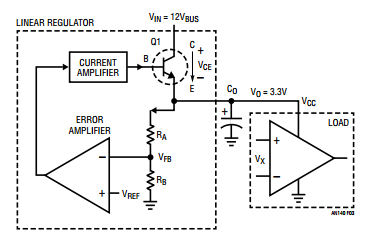


Figure 8 – Linear Fixed Voltage Regulator with Output Buffer and Error Correction

(http://cds.linear.com/docs/en/application-note/AN140fa.pdf)

The battery monitor system consists of a voltagesensing circuit which measures the current charge of each cell in the battery. The system has a buzzer and LED that will sound and flash when the voltage of any cell drops to unsafe levels.

## User Interface

The user interface consists of an LCD screen, a joystick, and two buttons. The joystick is used for navigating the textual menu represented on the LCD screen. The two buttons of the subsystem include an “enter” and a “back” button for entering and exiting menu options. The user interface contains the different modes in which the system can run, including locking and unlocking certain axes (Figure 9).

System Initializing

Please hold still

---- System Info ----

Bottom IMU Top IMU

X: 190 X: 87  
Y: 50 Y: 22  
Z: 0 Z: -5

Press FWD to enter UI

Initialization is complete

---- Axis Lock/Unlock ----

Note: Lock is stabilized

* Unlock X Axis (Pan)

Lock Y Axis (Tilt)

Unlock Z Axis (Pitch)

* Forward Button to toggle axis state.

Forward

Button

Back

Button

---- Home ----

Press FWD to select

* Axis Lock/Unlock

Settings

Press BCK to exit UI

Back

Button

Forward

Button

---- Settings ----

Save Changes with FWD

Use L/R to adjust

* Color [ Green ]

Cursor [ > ]

Forward

Button

Back

Button

* Forward Button to save changes
* Left and Right joystick to change settings

Figure : User Interface Flow Diagram

## Engineering Standards

This project will implement several standardized engineering for communications and control:

* The IMU will communicate with the microcontroller using the I2C protocol.
* The motors will be controlled using PWM signals of varying duty cycle.
* The microcontroller will be programmed using the standardized Arduino tool chain.

## Multidisciplinary Aspects

Multidisciplinary skills that will be utilized in this project include:

* Mechanical engineering and physics will be used to determine the physical tolerances of the system and help calculate the motion that the system is experiencing within the processing and control algorithms.
* Electrical engineering skills will be used to help develop the power management system and make sure that all components are properly and safely powered.

## Background

This project will utilize mostly Computer Engineering skills such as signal processing, interacting with both hardware and software subsystems, analyzing data gathered by sensors, and control algorithms. Coursework relevant to this project includes Digital Signal Processing, Physics, Electronics, Interface and Digital Electronics, and Software Engineering.

## Outside Contributors

* Nathan Ostrout – Contact at the Next 3D Print Lab. Printed all ABS plastic components.
* Joshua Milas – Contributed an Arduino library for adjusting PWM frequency.
* Rick Tolleson – Provided tools, equipment, hardware, and various services.
* Robert Kraynik – Contact at the machine shop. Constructed all aluminum frame components.

# Constraints and Considerations

## Extensibility

Since this project aims to create a low-cost but high-quality stabilization system, it could be extended into optimizing this concept to further reduce costs. By creating a dynamic mounting platform for the frame of the stabilization system, further products could also be created to allow for mounting the camera and frame to various other anchor points.

## Manufacturability

To assess the manufacturability of this system, several steps must be considered: the manufacturing of the frame components, acquiring the microcontroller, motors, and sensors, loading the control algorithms to the microcontroller, and attaching each of these components to the frame. Since most of these parts are pre-built, once the full specifications of the system are designed and the control algorithms are written, the assembly is relatively simple.

## Reliability

The reliability of this system is an important trait to consider. Since the system is to be both low cost and also high quality, it should be robust, rugged, and able to perform its function for several years at minimum. The systems that are most likely to fail are the microcontroller, the motors, and the battery. The microcontroller is susceptible to shock if handled improperly, but should not fail prematurely under normal usage. Should it fail, this component is difficult to replace because it is highly coupled with the rest of the system through physical connections. The motors are another difficult to replace component. If their torque thresholds are exceeded they may be damaged, or they may damage other components of the system. The battery will naturally decay over time; however it is a very easy component to replace.

## Societal and Economic Context

Since one of the main concerns for this project is keeping costs down, if this product were to be developed for retail it could have the societal impact of making a high quality stabilization system widely available to amateur photographers. This could potentially open new career opportunities for photographers seeking to develop professional level work, or could simply help hobbyists capture better quality images and videos.

## Health and Safety Considerations

A potential health concern related to the use of this product is weight. This is problematic because the user could experience back and/or arm strain by using this product for an extended period of time. This issue is addressed in the Customer Needs section, which states that the product should be lightweight. As a result of designing a lightweight system, there should be no risk of injury or fatigue when using this product for an extended period of time.

## Intellectual Properties

Due to patent considerations such as Patent-7642741(Handheld Platform Stabilization System Employing Distributed Rotation Sensors) and Patent-6611662(Autonomous, Self Levelling, Self Correcting Stabilized Platform), bringing this product to a market would be a difficult and costly procedure. This is because the proposed product employs stabilization systems that are similar to these existing patents. Instead, all hardware designs and software components developed for this project will be released online under the MIT license in order to help future developers create a working system.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Component** | **Unit Cost** | **#** | **Cost** | **Team Cost** | **Description** | **Vendor** |
| Motors | $41.37 | 3 | $124.11 | $124.11 | Turnigy 5208 Brushless DC Gimbal Motor | Hobbyking |
| IMU | $6.00 | 2 | $12.00 | $12.00 | GY-521 MPU 6050 | Amazon |
| MCU | $49.95 | 1 | $49.95 | $0 | Arduino Due | Owned |
| ProtoShield | $14.95 | 1 | $14.95 | $14.95 | Arduino ProtoShield | Jameco |
| Motor Control | $6.13 | 3 | $18.39 | $18.39 | L6235 | Arrow |
| Motor Board | $10.00 | 3 | $30.00 | $30.00 | Custom designed PCB | OSH Park |
| Hall effect sensors | $1.56 | 9 | $14.04 | $14.04 | A1104LUA-T | Sparkfun |
| Frame | $110.56 | 1 | $110.56 | $110.56 | Aluminum and Misc. Components | SMC Metal and Various |
| Battery | $56.16 | 1 | $56.16 | $56.16 | 10,000mAh Li-Po | HobbyKing |
| Battery Monitor | $1.99 | 1 | $1.99 | $1.99 | Low voltage alarm | HobbyKing |
| Power Manager | $13.95 | 2 | $27.90 | $27.90 | 12V and 9V Pololu Step-Up Regulators | Pololu |
| LCD | $18.95 | 1 | $18.95 | $18.95 | LCD for displaying UI (ST765) | Adafruit |
| Thumb Joystick | $5.95 | 1 | $5.95 | $5.95 | Joystick to navigate menu | Adafruit |
| Momentary Push Button | $0.95 | 2 | $1.90 | $1.90 | Buttons to enter/exit options | Adafruit |
| 3D Printed Components | $10.00 | 1 | $10.00 | $0 | ABS Plastic motor mounts and housing | Next 3D Print Lab |
| Misc. Electrical Components | $13.80 | 1 | $13.80 | $0 | Resistors, capacitors, and diodes | CE Department |
| **Total:** | | | **$510.65** | **$436.90** |  |  |

# Cost

# Testing Strategy

A thorough testing strategy was created to cover the various components, systems, and sub-systems of the unit. Unit testing covers the sensing, user interface, processing, power management, and actuation sub-systems. Integration testing covers how the different sub-systems work together and communicate. Acceptance testing ensures that the unit meets the defined engineering requirements. The testing document can be found in Appendix A – Testing.

# Risks

|  |  |
| --- | --- |
| **Sensors** | |
| Considerations: | * Different types of sensors: IMUs, Accelerometers, Gyroscopes * Protocols: Analog, SPI, I2C, PWM * Power Consumption * Sensitivity * Speed |
| Potential Failures: | * Sensors are unable to make fine enough measurements, fast enough * Power consumption is too high (negligible) * Sensors may have trouble communicating with processor |
| Risk Mitigation: | * MPU 6050 IMU * I2C Protocol * Low Power Consumption, could be powered off MCU * $40 * 3-axis Accelerometer and Gyroscope * No extraneous features |

|  |  |
| --- | --- |
| **Motor Controllers** | |
| Considerations: | * Different Control Methods: H-bridge, 3-phase motor controller, and modified ESC * Circuit protection * Response speed (time from turning on to moving the motor) |
| Potential Failures: | * Easily control the motor without having to provide extensive PWM signals. * Over Voltage/ Current could damage motors |
| Risk Mitigation: | * L6235 * 3-phase brushless motor controller * 8-52V, 5A out. * Over current protection * Easy to use |

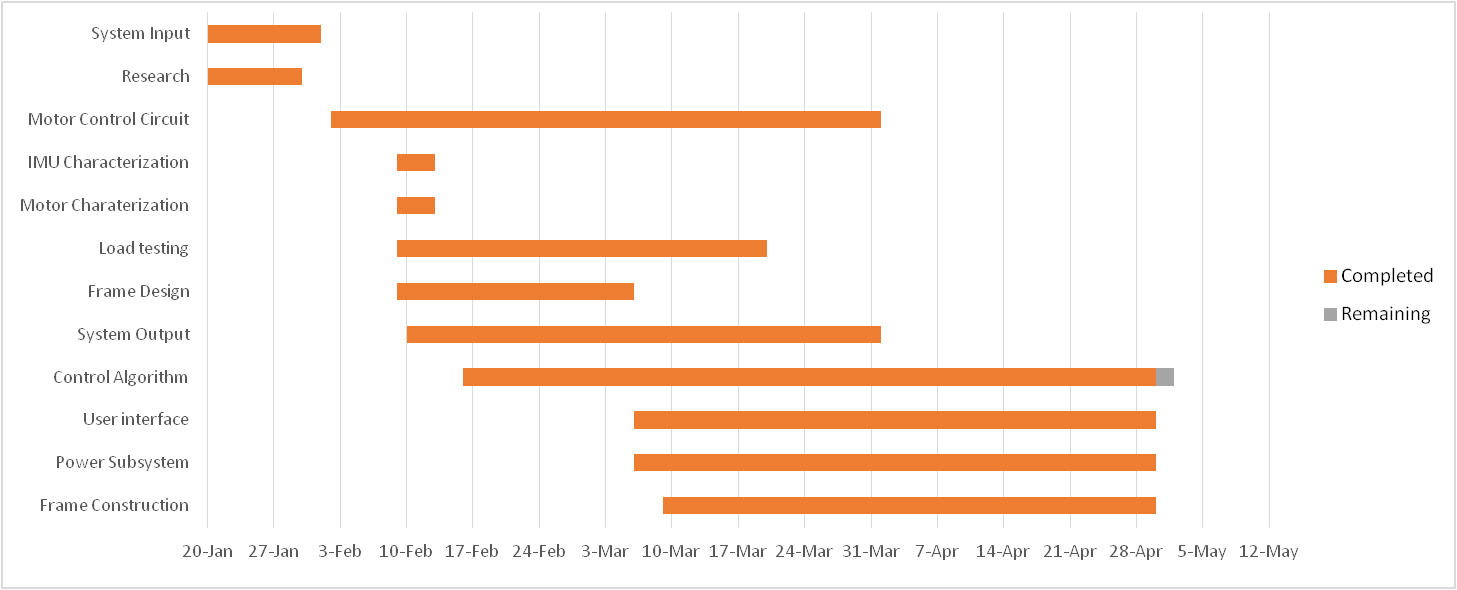
|  |  |
| --- | --- |
| **Motors** | |
| Considerations: | * Different Motor types: Continuous DC, Servo, Stepper, Brushless Motor * Power consumption, Torque, Weight, Noise, and Sensitivity |
| Potential Failures: | * Motors unable to support weight of load * Unable to stop precisely enough to produce noticeable stabilization * Motors cannot move slow enough or in small enough steps * Responsive enough to move when given a signal to move |
| Risk Mitigation: | * High turn, large Brushless Motor, low speed, high torque * Turnigy 5208 * $40-50 * Powered by offset PWM signals or controller |

|  |  |
| --- | --- |
| **Power Management** | |
| Considerations: | Knowing that a Li-Po battery would be used to power the system, the only remaining components to research were a battery monitor, a voltage regulator for the controller and the motor controllers, and the size of the battery to be used.  The following systems were researched:  **Voltage Regulation:**   * Linear Battery Eliminator Circuit * Switched Battery Eliminator Circuit * Fixed Voltage Regulator   **Battery Monitors:**   * Voltage Divider * Current Sensing   Methods for determining required battery size were also researched. |
| Potential Failures: | **Voltage Regulation:**   * Overheating as system is continuously used to regulate voltage * Inability to provide sufficient current * Inefficient conversion drastically reducing potential run time. * Unreliable output.   **Battery Monitor:**  Dropping below cutoff voltage of Li-Po battery can damage the battery cells.  Sudden power loss could cause damage to other parts of the system. |
| Risk Mitigation: | Use a current sensing system that reports to the controller the amount of charge left in the system so that the system can enter a safe state before losing power and the user can be notified of the need to power off the system.  Use a linear battery eliminator circuit to transform the voltage and current provided by the Li-Po battery to a safe range for the microcontroller and the motor controllers.  Once the max and nominal current consumption is measured for the different components, use the following equation to determine the size of battery required to power the system for an extended period of time under a max and nominal load. |

# Schedule

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Week One | | | | |
| Task | **Scheduled Completion** | **Responsibility** | **Modified Completion** | **Comments** |
| PWM research | January 30, 2015 | Mitch | January 30, 2015 | Complete. Discussed Arduino libraries for PWM. |
| Algorithm research | January 30, 2015 | Alex, Marit | January 30, 2015 | Complete. Algorithm defined as a PID. |
| Initial power research | January 30, 2015 | Marit | January 30, 2015 | Complete. Initial research completed using TI Power Bench. |
| I2C research | January 30, 2015 | Caleb, Mitch | February 6, 2015 | Complete. |
| Initial I2C connections made | February 06, 2015 | Mitch, Caleb | February 06, 2015 | Complete. |
| Initial motor circuit improved for consistency. | February 06, 2015 | Caleb, Alex | February 13, 2015 | Complete. Previously missing one screw bus for control inputs. |
| Initial frame design | February 13, 2015 | Everyone | February 13, 2015 | Complete. Measurements taken and initial components researched. |
| Motor load testing (weight) | February 13, 2015 | Marit, Mitch | February 13, 2015 | Complete. Some issues discovered. |
| Motor Characterization | February 13, 2015 | Caleb, Everyone | February 13, 2015 | Complete. |
| Characterize GY-521 IMU | February 13, 2015 | Caleb, Everyone | February 13, 2015 | Complete, sufficient for design |
| Run initial requirements verification tests | February 20, 2015 | Everyone | February 20, 2015 | Complete, several different tests run, results in status. |
| Control algorithm drafting | February 20, 2015 | Marit, Alex | February 20, 2015 | Complete. |
| Component Listing | February 27, 2015 | Everyone | February 27, 2015 | Complete |
| I2C communications with MCU | March 06, 2015 | Alex, Caleb | March 06, 2015 | Completed. Need to characterize the data provided by the IMU |
| Frame Construction | February 27, 2015 | Everyone | March 06, 2015 | Complete. All design elements complete, need to have the frame fabricated by the ME Department, to be completed by week 11. |
| Algorithm research 2 | March 13, 2015 | Marit, Alex | March 13, 2015 | Initial stub functions placed for PID control. |
| Initial UI | March 13, 2015 | Marit, Caleb | March 13, 2015 | Complete. Initial UI drafted and functional on LCD. |
| All parts ordered | February 27, 2015 | Everyone,  Rick Tolleson | March 14, 2015 | Complete. Frame parts ordered. |
| Motor circuit minimized | February 13, 2015 | Alex, Caleb | March 21, 2015 | Complete. Sent to OSH Fab. |
| Motor control with PWM | February 27, 2015 | Mitch, Caleb | March 21, 2015 | Complete. Need to manage software control now. |
| Motor control via MCU | March 06, 2015 | Mitch, Caleb, Alex | March 21, 2015 | Complete. Need to manage software control now. |
| Power consumption finalization | March 20, 2015 | Marit, Everyone | April 3, 2015 | Complete. |
| Battery system purchase | March 20, 2015 | Everyone,  Rick Tolleson | April 3, 2015 | Complete. |
| Control algorithms | April 03, 2015 | Alex, Marit, Mitch | April 3, 2015 | Complete. P Controller Designed. |
| Initial battery circuit constructed | April 10, 2015 | Caleb, Marit | April 10, 2015 | Complete. |
| UI finalization | April 17, 2015 | Caleb, Marit, Mitch | April 30, 2015 | All components are in and software is stubbed out. Final interrupts need to be written. |
| Control systems finalization | April 17, 2015 | Marit, Alex, Mitch | April 30, 2015 | Both IMU’s can be used on the data bus. The final algorithm now needs to use both. |
| Battery system finalization | April 17, 2015 | Everyone | April 30, 2015 | All components are in, just need to be connected. Charging works. |
| Design complete | April 24, 2015 | Everyone | April 30, 2015 |  |
| Frame Complete | April 17, 2015 | Everyone | April 30, 2015 | Need more motors and unfinished frame parts |
| Prepared for Imagine RIT | May 01, 2015 | Everyone |  |  |
|  |  |  |  |  |

## Gantt Chart



# Perspective

The finished project met or exceeded a number of the initial requirements, however it also misses several of our requirements. The system weighs only 6.8 lbs., which is 8.2 lbs. lighter than the requirement. The system can also remain powered for longer than the required 2 hour time limit, with the current maximum tested time being well past 4 hours of on-time. The system dimensions are roughly fitting with the required 0.5m cube required, although the outer handles do exceed this by 0.01m when they are in use on the system. The swappable handles are a proof of concept for a universal mounting system, although no vehicle mount was designed. The 3 brushless DC motors and the custom frame provide a platform that can turn in at least ±90° from the neutral position. The two 6-degrees of freedom IMUs provide linear and angular acceleration data in 3 orientations each. The LCD, joystick, and push buttons provide a full user interface providing the user with information about sensors and the ability to adjust system operation.

Despite the multiple successes in the design of the system, there are a few requirements that were not met that cause the system to not work as desired. Due to time constraints and the effectiveness of the control algorithm a few requirements have not been met. The full load capacity, 1.5kg, of the system could not be tested, although initial tests with a single motor suggest that the system could handle the required load. The incomplete control algorithm also made the ±2° stray impossible to properly test and verify. Additionally, the sensors chosen for the system did not provide stable feedback. This was found to be due to an inherent issue with gyroscopes dealing with drift due to the fact that they have no absolute point of reference. The IMUs chosen did include an advanced on-chip digital motion processing system that would have helped alleviate some of the inherent issues with the chip itself, however this feature needed faster I/O than was available and had conflicts with the UI subsystem. Some of these problems may have been fixed given additional time, however other issues came up regarding other components in the system.

Getting a large number of frame components custom designed and built proved to be a large investment in time and resources for the project. The 3D components were relatively quick and painless but the rest of the frame components proved to be expensive both time and money-wise, and the aluminum components were difficult to get manufactured by the mechanical engineering department. There was a large amount of time spent early on with issues such as the custom motor boards and motor torque that used a good deal of time that was designated for control systems work in the initial milestones. Given additional time for the project or better managed time, the various remaining problems associated with the system could be rectified.

In hindsight, work with third parties should have been started earlier and basic proof of concepts, like motor torque, should have been ironed out much sooner. Additionally, motor and sensor calibration and conditioning in the early stages would have left additional time to iron out the control algorithm. Sensor drift, which was found late in the project, could have been alleviated through the use of additional sensors. A magnetometer would provide an absolute point of reference for the gyroscope, which would allow the algorithm to account for drift. Some milestones could have also been re-organized to fix some of the issues that were encountered, such as putting more resources into the control algorithms early on to avoid problems with the core functionality later in the project.

# Appendix

## A – Testing

### Unit Tests

#### Sensing

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **IMU – Single Axis Sensing** | | | | | | | | | | | |
| **Description:** | | | Confirm that the IMU can measure changes in acceleration and angular momentum in a single axis. | | | | | | | **Type:** | White Box |
| **Tester Information** | | | | | | | | | | | |
| **Name of Tester:** | | | | Alex | | | | | | **Date:** | 1/30/15 |
|  | | | | |  | | | | |  |  |
| **Test Procedure** | | | | | | | | | | | |
| **Setup:** | | Establish I2C communications between the IMU and a receiving device. | | | | | | | | | |
| **Step** | **Action** | | | | **Expected Result** | **Pass** | **Fail** | **N/A** | **Comments** | | |
| 1 | Align the X axis of the IMU in parallel with gravity. | | | | The device should read out a 9.8 m/s2 change in acceleration in one axis via the IMU’s accelerometer. | X |  |  |  | | |
| 2 | Adjust the angular momentum of the IMU in the X axis only. | | | | The device should read out a change in angular momentum in one axis via the IMU’s gyroscope. | X |  |  |  | | |
| 3 | Repeat steps 1 and 2 for the Y axis. | | | |  | X |  |  |  | | |
| 4 | Repeat steps 1 and 2 for the Z axis. | | | |  | X |  |  |  | | |
| **Overall Test Result:** | | | | | |  |  |  |  | | |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **IMU –** I2C **Communications** | | | | | | | | | | | |
| **Description:** | | | Confirm that the IMU can establish I2C communications with a microcontroller or test PC. | | | | | | | **Type:** | Black box |
| **Tester Information** | | | | | | | | | | | |
| **Name of Tester:** | | | | Mitch | | | | | | **Date:** | 1/30/15 |
|  | | | |  | | | | | |  |  |
| **Test Procedure** | | | | | | | | | | | |
| **Setup:** | | Establish an I2C data line between the IMU and a receiving device using a physical connection and any required libraries. | | | | | | | | | |
| **Step** | **Action** | | | | **Expected Result** | **Pass** | **Fail** | **N/A** | **Comments** | | |
| 1 | Attempt to send data over the I2C data connection. | | | | Data sent at the source should be seen at the receiving device. | X |  |  |  | | |
| **Overall Test Result:** | | | | | |  |  |  |  | | |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **IMU – Multiple Axis Sensing** | | | | | | | | | | | |
| **Description:** | | | The IMU should be able to measure changes in multiple axes at once. | | | | | | | **Type:** | White Box |
| **Tester Information** | | | | | | | | | | | |
| **Name of Tester:** | | | | Alex | | | | | | **Date:** | 1/30/15 |
|  | | | |  | | | | | |  |  |
| **Test Procedure** | | | | | | | | | | | |
| **Setup:** | | Establish I2C communications between the IMU and a receiving device. | | | | | | | | | |
| **Step** | **Action** | | | | **Expected Result** | **Pass** | **Fail** | **N/A** | **Comments** | | |
| 1 | Adjust the IMU to make a change in acceleration in any two or more axes. | | | | The device should read out a change in acceleration in multiple axes via the IMU’s accelerometers. | X |  |  |  | | |
| 2 | Adjust the IMU to make a change in angular momentum in any two or more axes. | | | | The device should read out a change in angular momentum in multiple axes via the IMU’s gyroscopes. | X |  |  |  | | |
| **Overall Test Result:** | | | | | |  |  |  |  | | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **IMU - Speed** | | | | | | | | | | | | | | | | | | | | | |
| **Description:** | | | | Confirm that the IMU can communicate the data it is measuring to a microcontroller or test PC in a fast enough time margin that it does not impair stabilization algorithms. | | | | | | | | | | | | | | **Type:** | | Black Box | |
| **Tester Information** | | | | | | | | | | | | | | | | | | | | | |
| **Name of Tester:** | | | | | Mitch | | | | | | | | | | | | | **Date:** | | 1/30/15 | |
|  | | | | |  | | | | | | | | | | | | |  | |  | |
| **Test Procedure** | | | | | | | | | | | | | | | | | | | | | |
| **Setup:** | | Establish I2C communications between the IMU and a receiving device. | | | | | | | | | | | | | | | | | | | |
| **Step** | **Action** | | | | | | | **Expected Result** | **Pass** | | **Fail** | | **N/A** | | **Comments** | | | | | | |
| 1 | Make an adjustment to the IMU and record a time stamp both before and after this data is sent and received. | | | | | | | This data should be communicated in a negligible enough time that it does not impair real-time stabilization calculations. | X | |  | |  | |  | | | | | | |
| **Overall Test Result:** | | | | | | | | |  | |  | |  | |  | | | | | | |
| **IMU – Power Consumption** | | | | | | | | | | | | | | | | | | | | |
| **Description:** | | | | Confirm that the IMU has a negligible power draw and can be powered by the microcontroller. | | | | | | | | | | | | | **Type:** | | White Box | |
| **Tester Information** | | | | | | | | | | | | | | | | | | | | |
| **Name of Tester:** | | | | | | Marit | | | | | | | | | | | **Date:** | | 2/13/15 | |
|  | | | | | |  | | | | | | | | | | |  | |  | |
| **Test Procedure** | | | | | | | | | | | | | | | | | | | | |
| **Setup:** | | | Establish I2C communications between the IMU and a receiving device. | | | | | | | | | | | | | | | | | |
| **Step** | **Action** | | | | | | **Expected Result** | | | **Pass** | | **Fail** | | **N/A** | | **Comments** | | | | |
| 1 | Move the IMU such that there is a change in multiple axes for both the accelerometers and gyroscopes. | | | | | | The IMU should be able to perform all of its routine operation with a power draw under what the microcontroller can supply. No external power should be needed. | | | X | |  | |  | |  | | | | |
| **Overall Test Result:** | | | | | | | | | |  | |  | |  | |  | | | | |

#### User Interface

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **UI – Menu Navigation** | | | | | | | | | | | |
| **Description:** | | | Test that the UI can navigate to all possible menus. | | | | | | | **Type:** | White box |
| **Tester Information** | | | | | | | | | | | |
| **Name of Tester:** | | | | Marit | | | | | | **Date:** | 5/1/15 |
|  | | | |  | | | | | |  |  |
| **Test Procedure** | | | | | | | | | | | |
| **Setup:** | | Load the UI software to the microcontroller. | | | | | | | | | |
| **Step** | **Action** | | | | **Expected Result** | **Pass** | **Fail** | **N/A** | **Comments** | | |
| 1 | Attempt to navigate the UI using the joystick and control buttons. | | | | The user should be able to use the external controls to navigate through the UI options. | X |  |  |  | | |
| 2 | Attempt to successfully navigate to every single available UI screen shown in **Error! Reference source not found.** | | | | The UI software should be written such that all planned screens and features are available to the user. | X |  |  |  | | |
| **Overall Test Result:** | | | | | |  |  |  |  | | |

#### Processing

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **MCU -Programming** | | | | | | | | | | |
| **Description:** | | | Test that the MCU tool-chain can program the device. | | | | | | **Type:** | White box |
| **Tester Information** | | | | | | | | | | |
| **Name of Tester:** | | | | Alex | | | | | **Date:** | 1/30/15 |
|  | | | |  | | | | |  |  |
| **Test Procedure** | | | | | | | | | | |
| **Setup:** | | Attach the MCU to a PC with programming software and drivers installed. | | | | | | | | |
| **Step** | **Action** | | | **Expected Result** | **Pass** | **Fail** | **N/A** | **Comments** | | |
| 1 | Start the programming software. | | | The software should start and detect the connected device. | X |  |  |  | | |
| 2 | Open and compile a test program which toggles one GPIO pin. | | | The program should compile with no errors. | X |  |  |  | | |
| 3 | Upload the program to the MCU. | | | The upload should complete with no errors. | X |  |  |  | | |
| 4 | Remove the MCU from the PC, power it, and measure the GPIO output. | | | The GPIO pin should toggle at the set rate. | X |  |  |  | | |
| **Overall Test Result:** | | | | |  |  |  |  | | |

#### Power Management

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Battery – Accurate Charge Monitor** | | | | | | | | | | | |
| **Description:** | | | Confirm that the battery charge monitor accurately measures the voltage. | | | | | | | **Type:** | White Box |
| **Tester Information** | | | | | | | | | | | |
| **Name of Tester:** | | | | Marit | | | | | | **Date:** | 4/10/15 |
|  | | | |  | | | | | |  |  |
| **Test Procedure** | | | | | | | | | | | |
| **Setup:** | | Connect the battery monitor to a DC power supply. | | | | | | | | | |
| **Step** | **Action** | | | | **Expected Result** | **Pass** | **Fail** | **N/A** | **Comments** | | |
| 1 | Apply a voltage to the power supply greater than the critical cutoff voltage. | | | | The battery monitor should not raise an alarm. | X |  |  |  | | |
| 2 | Apply a voltage to the power supply that is equal to the critical cutoff voltage. | | | | The battery monitor should raise an alarm. | X |  |  |  | | |
| 3 | Apply a voltage to the power supply that is lower than the critical cutoff voltage. | | | | The battery should raise an alarm. | X |  |  |  | | |
| **Overall Test Result:** | | | | | |  |  |  |  | | |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Battery – Power Output with Microcontroller as Load** | | | | | | | | | | | |
| **Description:** | | | Determine the time to fully dissipate the battery to the cutoff voltage while powering the microcontroller. | | | | | | | **Type:** | White Box |
| **Tester Information** | | | | | | | | | | | |
| **Name of Tester:** | | | | Marit | | | | | | **Date:** | 4/28/15 |
|  | | | |  | | | | | | **Time:** |  |
| **Test Procedure** | | | | | | | | | | | |
| **Setup:** | | Ensure that the battery is fully charged. Connect the microcontroller to the battery through the boost converter and battery monitor. Load the microcontroller with a CPU intensive program. | | | | | | | | | |
| **Step** | **Action** | | | | **Expected Result** | **Pass** | **Fail** | **N/A** | **Comments** | | |
| 1 | Measure the power output every minute until the battery reaches its cutoff voltage or the time reaches two hours. | | | | The battery should maintain full output for at least 2 hours. | X |  |  |  | | |
| **Overall Test Result:** | | | | | | Battery lasted the full 2 hours with plenty of charge left. | | | | | |

#### Actuation

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Motor - Power Consumption** | | | | | | | | | | |
| **Description:** | | | Measure the maximum power consumption (Watts) of a motor with a load attached. | | | | | | **Type:** | White box |
| **Tester Information** | | | | | | | | | | |
| **Name of Tester:** | | | | Marit | | | | | **Date:** | 2/13/15 |
|  | | | |  | | | | |  |  |
| **Test Procedure** | | | | | | | | | | |
| **Setup:** | | Attach a 2.5 Kg load 15 cm away from the motor and attach an ammeter to the motor circuit. | | | | | | | | |
| **Step** | **Action** | | | **Expected Result** | **Pass** | **Fail** | **N/A** | **Comments** | | |
| 1 | Run for 10 rotations at maximum speed, oriented at 0° along the X-axis. | | | The current should stay below 1.2 A. | X |  |  |  | | |
| 2 | Rotate the motor 10° around the x-axis. | | | The current should stay below 1.2 A. | X |  |  |  | | |
| 3 | Run for 10 rotations at maximum speed. | | | The current should stay below 1.2 A. | X |  |  |  | | |
| 4 | Repeat steps 2 and 3 until 180° is reached. | | | The current should stay below 1.2 A. | X |  |  |  | | |
| **Overall Test Result:** | | | | |  |  |  |  | | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Motor – Hold Position Under Maximum Load** | | | | | | | | | | |
| **Description:** | | | Ensure the motors can maintain position holding the maximum supported load. | | | | | | **Type:** | White box |
| **Tester Information** | | | | | | | | | | |
| **Name of Tester:** | | | | Alex | | | | | **Date:** | 5/8/2015 |
|  | | | |  | | | | | **Time:** |  |
| **Test Procedure** | | | | | | | | | | |
| **Setup:** | | Attach a 2.5 Kg load 15 cm away from the motor, apply a stall signal to maintain position. | | | | | | | | |
| **Step** | **Action** | | | **Expected Result** | **Pass** | **Fail** | **N/A** | **Comments** | | |
| 1 | Turn the motor circuit on and position the motor at 0° along the X-axis and wait 30 seconds. | | | The motor should not have changed position and continues to maintain the original position. |  | X |  | Motor cannot support maximum load. Camera used for testing is usable due to lighter weight. | | |
| 2 | Rotate the motor 10° around the x-axis. | | |  |  | X |  |  | | |
| 3 | Wait 30 seconds. | | | The motor should not have changed position and continues to maintain the original position. |  | X |  |  | | |
| 4 | Repeat steps 2 and 3 until 180° is reached | | |  |  | X |  |  | | |
| **Overall Test Result:** | | | | |  |  |  |  | | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Motor – Consistent Movement Under Load** | | | | | | | | | | |
| **Description:** | | | Ensure the motor can move a maximum load consistently. | | | | | | **Type:** | White box |
| **Tester Information** | | | | | | | | | | |
| **Name of Tester:** | | | | Alex | | | | | **Date:** | 5/8/2015 |
|  | | | |  | | | | | **Time:** |  |
| **Test Procedure** | | | | | | | | | | |
| **Setup:** | | Attach a 2.5 Kg load 15 cm away from the motor, consisting of a camera and any additional required weight. | | | | | | | | |
| **Step** | **Action** | | | **Expected Result** | **Pass** | **Fail** | **N/A** | **Comments** | | |
| 1 | Run for 10 rotations at maximum speed, oriented at 0° along the X-axis | | | The motor should successfully move the load and the video should remain smooth. |  | X |  | Motor does not support recommended load | | |
| 2 | Rotate the motor 10° around the x-axis. | | |  |  | X |  |  | | |
| 3 | Run for 10 rotations at maximum speed | | | The motor should successfully move the load and the video should remain smooth. |  | X |  |  | | |
| 4 | Repeat steps 2 and 3 until 180° is reached | | |  |  | X |  |  | | |
| **Overall Test Result:** | | | | |  |  |  |  | | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Motor – Noise Production** | | | | | | | | | | | | | | |
| **Description:** | | | Measure the maximum noise production (dB) of a motor with a load attached. | | | | | | | | | | **Type:** | White box |
| **Tester Information** | | | | | | | | | | | | | | |
| **Name of Tester:** | | | | Caleb | | | | | | | | | **Date:** | 4/24/15 |
|  | | | |  | | | | | | | | |  |  |
| **Test Procedure** | | | | | | | | | | | | | | |
| **Setup:** | | Attach a 2.5 Kg load 15 cm away from the motor. Prepare a sound meter for use with test. | | | | | | | | | | | | |
| **Step** | **Action** | | | **Expected Result** | | **Pass** | | **Fail** | | **N/A** | | **Comments** | | |
| 1 | Orient the motor at 0° along the X-axis. | | |  | | X | |  | |  | |  | | |
| 2 | Move the sound meter 15 cm away from the motor. | | |  | | X | |  | |  | |  | | |
| 3 | Run for 10 rotations at maximum speed. | | | The sound production of the motors should remain under 10 dB for the duration of the movement. | | X | |  | |  | |  | | |
| 4 | Move the sound meter to 10 cm from the motor. | | |  | | X | |  | |  | |  | | |
| 5 | Run for 10 rotations at maximum speed. | | | The sound production of the motors should remain under 10 dB for the duration of the movement. | | X | |  | |  | |  | | |
| 6 | Move the sound meter to 5 cm from the motor. | | |  | | X | |  | |  | |  | | |
| 7 | Run for 10 rotations at maximum speed. | | | The sound production of the motors should remain under 10 dB for the duration of the movement. | | X | |  | |  | |  | | |
| 8 | Rotate the motor 10° around the x-axis. | | |  | | X | |  | |  | |  | | |
| 9 | Repeat steps 2 through 8 until 180° is reached. | | |  | | X | |  | |  | |  | | |
| **Overall Test Result:** | | | | |  | |  | |  | |  | | | |

### Integration Tests

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Integration – MCU to Motor delay** | | | | | | | | | |
| **Description:** | | | Measure the time from applying a change to the IMU signal line to motor movement. | | | | | **Type:** | Black box |
| **Tester Information** | | | | | | | | | |
| **Name of Tester:** | | | Mitch | | | | | **Date:** | 5/8/2015 |
|  | | |  | | | | | **Time:** |  |
| **Test Procedure** | | | | | | | | | |
| **Setup:** | | Set up the system (MCU, Motor circuit, and Motor) and zero out a timer. | | | | | | | |
| **Step** | **Action** | | **Expected Result** | **Pass** | **Fail** | **N/A** | **Comments** | | |
| 1 | Apply a change to the IMU I2C line. | | The motor should move within a short period of time from when the signal is applied. |  | X |  | Motor takes some small time due to control system calculations. | | |
| **Overall Test Result:** | | | |  |  |  |  | | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **MCU – Motor Control** | | | | | | | | | | |
| **Description:** | | | Use the MCU to control motor function. | | | | | | **Type:** | White box |
| **Tester Information** | | | | | | | | | | |
| **Name of Tester:** | | | | Caleb | | | | | **Date:** | 3/14/15 |
|  | | | |  | | | | |  |  |
| **Test Procedure** | | | | | | | | | | |
| **Setup:** | | Attach MCU outputs to the forward/reverse and brake pins of the motor control circuit. The motor enable should be off. | | | | | | | | |
| **Step** | **Action** | | | **Expected Result** | **Pass** | **Fail** | **N/A** | **Comments** | | |
| 1 | Compile and upload a program which uses a PWM signal to control the speed and direction of the motor. | | | The program should compile and upload with no errors. | X |  |  |  | | |
| 2 | Secure the motor and turn the enable on. | | | The motor should begin to move at the speed and direction set by the PWM signal. | X |  |  |  | | |
| **Overall Test Result:** | | | | |  |  |  |  | | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **MCU – Sensor Input** | | | | | | | | | | |
| **Description:** | | | Confirm that control algorithms properly associate IMU movement with motor adjustment. | | | | | | **Type:** | White box |
| **Tester Information** | | | | | | | | | | |
| **Name of Tester:** | | | | Caleb | | | | | **Date:** | 3/14/15 |
|  | | | |  | | | | |  |  |
| **Test Procedure** | | | | | | | | | | |
| **Setup:** | | Attach the IMU and motor control circuits to the MCU. | | | | | | | | |
| **Step** | **Action** | | | **Expected Result** | **Pass** | **Fail** | **N/A** | **Comments** | | |
| 1 | Compile and upload a program which takes IMU data and moves the corresponding motor in the opposite direction. | | | The program should compile and upload with no errors. | X |  |  |  | | |
| 2 | Move the IMU in the X-axis. | | | The X-axis motor should move in the opposite direction. | X |  |  |  | | |
| 3 | Repeat Step 2 for the Y and Z axes. | | |  | X |  |  |  | | |
| **Overall Test Result:** | | | | |  |  |  |  | | |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Battery – Power Output with Motors and Microcontroller as Load** | | | | | | | | | | | |
| **Description:** | | | Determine the time to fully dissipate the battery to the cutoff voltage while powering both the microcontroller and the motors. | | | | | | | **Type:** | White Box |
| **Tester Information** | | | | | | | | | | | |
| **Name of Tester:** | | | | Marit | | | | | **Date:** | | 4/28/15 |
|  | | | |  | | | | |  | |  |
| **Test Procedure** | | | | | | | | | | | |
| **Setup:** | | Ensure that the battery is fully charged. Connect the microcontroller and motors to the battery through the boost converter and battery monitor. Ensure that the motors are either constantly moving or moving at variable speeds. Load the microcontroller with a CPU intensive program. | | | | | | | | | |
| **Step** | **Action** | | | | **Expected Result** | **Pass** | **Fail** | **N/A** | | **Comments** | |
| 1 | Measure the power output every minute until the battery reaches its cutoff voltage or the time reaches two hours. | | | | The battery should maintain full output for at least 2 hours. | X |  |  | |  | |
| **Overall Test Result:** | | | | | |  |  |  | |  | |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **UI – System Info** | | | | | | | | | | | |
| **Description:** | | | Test that all of the features of the ‘System Info’ screen are working. | | | | | | | **Type:** | White box |
| **Tester Information** | | | | | | | | | | | |
| **Name of Tester:** | | | | Marit | | | | | | **Date:** | 4/28/15 |
|  | | | |  | | | | | |  |  |
| **Test Procedure** | | | | | | | | | | | |
| **Setup:** | | Load the menu software to the microcontroller, and initialize the full stabilization system. | | | | | | | | | |
| **Step** | **Action** | | | | **Expected Result** | **Pass** | **Fail** | **N/A** | **Comments** | | |
| 1 | Navigate to the system info screen on the UI. | | | |  | X |  |  |  | | |
| 2 | Check the available statistics on the system info screen. | | | | The system info screen should display battery life, X/Y/Z axis lock mode and orientation, and raw data from the sensors. | X |  |  | Battery level and axis status was removed from status screen. | | |
| 3 | Wait for the battery to drain over 20 minutes. | | | | The battery life statistics should update in real time. |  |  | X | Battery level is no longer monitored by the MCU. | | |
| 4 | Adjust the system so that the IMU experiences movement. | | | | The raw sensor data statistics should update in real time. | X |  |  |  | | |
| 5 | Navigate to the lock/unlock menus, and change the lock state and orientation of one axis. Navigate back to the system info screen. | | | | The axis lock mode and orientation status should update to reflect the changes to the adjusted axis. | X |  |  | UI no longer allows the user to orient individual axes. | | |
| **Overall Test Result:** | | | | | |  |  |  |  | | |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **UI – Axis Locking** | | | | | | | | | | | |
| **Description:** | | | Test that the user can adjust the axis lock state and orientation of each axis through use of the UI controls. | | | | | | | **Type:** | White box |
| **Tester Information** | | | | | | | | | | | |
| **Name of Tester:** | | | | Alex | | | | | | **Date:** | 4/28/15 |
|  | | | |  | | | | | |  |  |
| **Test Procedure** | | | | | | | | | | | |
| **Setup:** | | Load the menu software to the microcontroller, and initialize the full stabilization system. | | | | | | | | | |
| **Step** | **Action** | | | | **Expected Result** | **Pass** | **Fail** | **N/A** | **Comments** | | |
| 1 | Navigate to the axis modification screen, and select an axis to modify. | | | |  | X |  |  |  | | |
| 2 | Select the “adjust and lock” option, and then use the user controls to adjust the axis and set the mode to “axis lock”. | | | | The motor should rotate to the position dictated by the user. The axis should lock and hold this orientation when destabilized. | X |  |  |  | | |
| 3 | Select the “axis unlock” option for the same axis. | | | | The axis should unlock and no longer stabilize. | X |  |  |  | | |
| 4 | Select the “reset to neutral” option. | | | | The axis should reset to its neutral position automatically and no longer stabilize. | X |  |  |  | | |
| 5 | Repeat steps 1-4 for each axis. | | | |  | X |  |  |  | | |
| **Overall Test Result:** | | | | | |  |  |  |  | | |

### Acceptance Tests

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **IMU - Sensitivity** | | | | | | | | | | | |
| **Description:** | | | Confirm that the IMU can make sufficiently sensitive measurements to accommodate the smallest change that the system may experience. | | | | | | | **Type:** | Black Box |
| **Tester Information** | | | | | | | | | | | |
| **Name of Tester:** | | | | Mitch | | | | | | **Date:** | 5/8/2015 |
|  | | | |  | | | | | | **Time:** |  |
| **Test Procedure** | | | | | | | | | | | |
| **Setup:** | | Establish I2C communications between the IMU and a receiving device. | | | | | | | | | |
| **Step** | **Action** | | | | **Expected Result** | **Pass** | **Fail** | **N/A** | **Comments** | | |
| 1 | Make an adjustment of ±2° or more to the IMU. | | | | The IMU should be able to record this change so that the control algorithms may adjust and stabilize the system for this change. | X |  |  | IMU can detect small changes. | | |
| 2 | Make an adjustment of less than ±2° to the IMU. | | | | The IMU does not need to be able to record this change. | X |  |  |  | | |
| **Overall Test Result:** | | | | | |  |  |  |  | | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **MCU – Control Algorithm** | | | | | | | | | | |
| **Description:** | | | Test the system’s minimum stray. | | | | | | **Type:** | White box |
| **Tester Information** | | | | | | | | | | |
| **Name of Tester:** | | | | Alex | | | | | **Date:** | 4/29/15 |
|  | | | |  | | | | |  |  |
| **Test Procedure** | | | | | | | | | | |
| **Setup:** | | Attach a 2.5 Kg, 15 cm load to the motor, attach <orientation sensor> to the load. | | | | | | | | |
| **Step** | **Action** | | | **Expected Result** | **Pass** | **Fail** | **N/A** | **Comments** | | |
| 1 | Upload control software to the MCU. | | | Program uploads with no errors. | X |  |  |  | | |
| 2 | Move the system ±50˚ in the X-axis. | | | The load strays no further than ±2˚. |  | X |  |  | | |
| 3 | Repeat step 2 for the Y and Z axes. | | |  |  | X |  |  | | |
| **Overall Test Result:** | | | | | System moves with the user, adjusting back to ±2˚ after a delay. | | | | | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **System –Weight** | | | | | | | | | | |
| **Description:** | | | Confirm that the final, unloaded system has a weight that is less than or equal to 15 lbs., as specified in the engineering specifications. | | | | | | **Type:** | White Box |
| **Tester Information** | | | | | | | | | | |
| **Name of Tester:** Caleb | | | |  | | | | | **Date:** | 4/28/15 |
|  | | | |  | | | | |  |  |
| **Test Procedure** | | | | | | | | | | |
| **Setup:** | | Remove any load from the system. Place the system on a scale. | | | | | | | | |
| **Step** | **Action** | | | **Expected Result** | **Pass** | **Fail** | **N/A** | **Comments** | | |
| 1 | Measure and record the weight of the completed system. | | | The completed system should weight 15 lbs. or less. | X |  |  |  | | |
| **Overall Test Result:** | | | | |  |  |  |  | | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **System –Idle Operation Time** | | | | | | | | | | |
| **Description:** | | | Confirm that the final, loaded system remains operational for more than two hours while idle. | | | | | | **Type:** | White Box |
| **Tester Information** | | | | | | | | | | |
| **Name of Tester:** Marit | | | |  | | | | | **Date:** | 4/28/15 |
|  | | | |  | | | | |  |  |
| **Test Procedure** | | | | | | | | | | |
| **Setup:** | | Place the powered system on a flat surface where it will not move during testing. Place the maximum load on the system. | | | | | | | | |
| **Step** | **Action** | | | **Expected Result** | **Pass** | **Fail** | **N/A** | **Comments** | | |
| 1 | Measure the time it takes for the batteries to no longer maintain system operations, or until two hours is reached. | | | The system should maintain power for at least two hours. | X |  |  |  | | |
| **Overall Test Result:** | | | | |  |  |  |  | | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **System –Operation Time with Minor Adjustments** | | | | | | | | | | |
| **Description:** | | | Confirm that the final, loaded system remains operational for more than two hours while performing minor adjustments. | | | | | | **Type:** | White Box |
| **Tester Information** | | | | | | | | | | |
| **Name of Tester:** Marit | | | |  | | | | | **Date:** | 5/8/2015 |
|  | | | |  | | | | | **Time:** |  |
| **Test Procedure** | | | | | | | | | | |
| **Setup:** | | Suspend the system from a hanging mount that provides ample amounts of clearance. Place the maximum load on the system. Load a microcontroller with a simple test program that produces I2C output that mimics the IMU when minor oscillations of ±20◦ are observed. | | | | | | | | |
| **Step** | **Action** | | | **Expected Result** | **Pass** | **Fail** | **N/A** | **Comments** | | |
| 1 | Measure the time it takes for the batteries to no longer maintain system operations, or until two hours is reached. | | | The system should maintain power for at least two hours. | X |  |  | Battery is able to support desired on-time, but full load is not supported. | | |
| **Overall Test Result:** | | | | |  |  |  |  | | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **System –Operation Time with Major Adjustments** | | | | | | | | | | |
| **Description:** | | | Confirm that the final, loaded system remains operational for more than two hours while performing major adjustments. | | | | | | **Type:** | White Box |
| **Tester Information** | | | | | | | | | | |
| **Name of Tester:** Marit | | | |  | | | | | **Date:** | 5/8/2015 |
|  | | | |  | | | | | **Time:** |  |
| **Test Procedure** | | | | | | | | | | |
| **Setup:** | | Suspend the system from a hanging mount that provides ample amounts of clearance. Place the maximum load on the system. Load a microcontroller with a simple test program that produces I2C output that mimics the IMU when minor oscillations of ±50◦ are observed. | | | | | | | | |
| **Step** | **Action** | | | **Expected Result** | **Pass** | **Fail** | **N/A** | **Comments** | | |
| 1 | Measure the time it takes for the batteries to no longer maintain system operations, or until two hours is reached. | | | The system should maintain power for at least two hours. | X |  |  | Battery is able to support desired on-time, but full load is not supported. | | |
| **Overall Test Result:** | | | | |  |  |  |  | | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **System –Dimensions** | | | | | | | | | | |
| **Description:** | | | Confirm that the final system fits within a box of dimensions 0.5x0.5x0.5m. | | | | | | **Type:** | White Box |
| **Tester Information** | | | | | | | | | | |
| **Name of Tester:** | | | | Mitch | | | | | **Date:** | 4/29/15 |
|  | | | |  | | | | |  |  |
| **Test Procedure** | | | | | | | | | | |
| **Setup:** | | Place the system on a flat surface. | | | | | | | | |
| **Step** | **Action** | | | **Expected Result** | **Pass** | **Fail** | **N/A** | **Comments** | | |
| 1 | Measure and record the length, width, and height of the system. | | | The system should fit within a box of dimensions 0.5x0.5x0.5m. | X |  |  |  | | |
| **Overall Test Result:** | | | | |  |  |  |  | | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **System – Z-axis Range of Motion** | | | | | | | | | | |
| **Description:** | | | Confirm that the final, loaded system can move along the z-axis from -50◦ to +50◦. | | | | | | **Type:** | White Box |
| **Tester Information** | | | | | | | | | | |
| **Name of Tester:** | | | | Alex | | | | | **Date:** | 4/29/15 |
|  | | | |  | | | | |  |  |
| **Test Procedure** | | | | | | | | | | |
| **Setup:** | | Suspend the system from a hanging mount that provides ample amounts of clearance. Place the maximum load on the system. Ensure the system is in the default stabilization location. | | | | | | | | |
| **Step** | **Action** | | | **Expected Result** | **Pass** | **Fail** | **N/A** | **Comments** | | |
| 1 | Take control of the z-axis. | | | The x-axis and y-axis should no longer move. | X |  |  |  | | |
| 2 | Move the controller to the extremes of the axis, where the motors are not over torqued and the frame does not interfere with the system motion.  Record the angle of the extreme points | | | The system should be capable of moving ±50◦ from the default stabilization location. | X |  |  |  | | |
| **Overall Test Result:** | | | | |  |  |  |  | | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **System – Y-axis Range of Motion** | | | | | | | | | | |
| **Description:** | | | Confirm that the final, loaded system can move along the y-axis from -160◦ to +160◦. | | | | | | **Type:** | White Box |
| **Tester Information** | | | | | | | | | | |
| **Name of Tester:** | | | | Alex | | | | | **Date:** | 4/29/15 |
|  | | | |  | | | | |  |  |
| **Test Procedure** | | | | | | | | | | |
| **Setup:** | | Suspend the system from a hanging mount that provides ample amounts of clearance. Place the maximum load on the system. Ensure the system is in the default stabilization location. | | | | | | | | |
| **Step** | **Action** | | | **Expected Result** | **Pass** | **Fail** | **N/A** | **Comments** | | |
| 1 | Take control of the y-axis. | | | The x-axis and z-axis should no longer move. | X |  |  |  | | |
| 2 | Move the controller to the extremes of the axis, where the motors are not over torqued and the frame does not interfere with the system motion.  Record the angle of the extreme points | | | The system should be capable of moving ±160◦ from the default stabilization location. | X |  |  |  | | |
| **Overall Test Result:** | | | | |  |  |  |  | | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **System – X-axis Range of Motion** | | | | | | | | | | |
| **Description:** | | | Confirm that the final, loaded system can move along the x-axis from -160◦ to +160◦. | | | | | | **Type:** | White Box |
| **Tester Information** | | | | | | | | | | |
| **Name of Tester:** | | | | Alex | | | | | **Date:** | 4/29/15 |
|  | | | |  | | | | |  |  |
| **Test Procedure** | | | | | | | | | | |
| **Setup:** | | Suspend the system from a hanging mount that provides ample amounts of clearance. Place the maximum load on the system. Ensure the system is in the default stabilization location. | | | | | | | | |
| **Step** | **Action** | | | **Expected Result** | **Pass** | **Fail** | **N/A** | **Comments** | | |
| 1 | Take control of the x-axis. | | | The z-axis and y-axis should no longer move. | X |  |  |  | | |
| 2 | Move the controller to the extremes of the axis, where the motors are not over torqued and the frame does not interfere with the system motion.  Record the angle of the extreme points | | | The system should be capable of moving ±160◦ from the default stabilization location. | X |  |  |  | | |
| **Overall Test Result:** | | | | |  |  |  |  | | |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **System – Maximum Supported Load** | | | | | | | | | | |
| **Description:** | | | Confirm that the final system can handle a load up to 1.5 kg without failing to noticeably stabilize the load. | | | | | | **Type:** | White Box |
| **Tester Information** | | | | | | | | | | |
| **Name of Tester:** All | | | |  | | | | | **Date:** | 5/8/2015 |
|  | | | |  | | | | | **Time:** |  |
| **Test Procedure** | | | | | | | | | | |
| **Setup:** | | Suspend the system from a hanging mount that provides ample amounts of clearance. A camera with a mass of 0.184 kg will initially be placed on the system so that the stabilization effect can be observed. The basic stabilization for all axes will be used during this test. | | | | | | | | |
| **Step** | **Action** | | | **Expected Result** | **Pass** | **Fail** | **N/A** | **Comments** | | |
| 1 | Destabilize the system by moving the frame around, record video during this process. | | | No noticeable shaking should be seen in the recoded video, and the system should perform with no extreme stress. |  | X |  | System does not fully support maximum desired load, and has some stabilization issues due to control algorithm and sensor problems. | | |
| 2 | Add an additional 0.25 kg load to the system. | | |  |  |  |  |  | | |
| 3 | Repeat steps 1 and 2 until the system can no longer maintain stabilization or the motors can no longer support the load. | | | The system should maintain noticeable stabilization at least until the load is greater than 1.5kg. |  | X |  |  | | |
| **Overall Test Result:** | | | | |  |  |  |  | | |