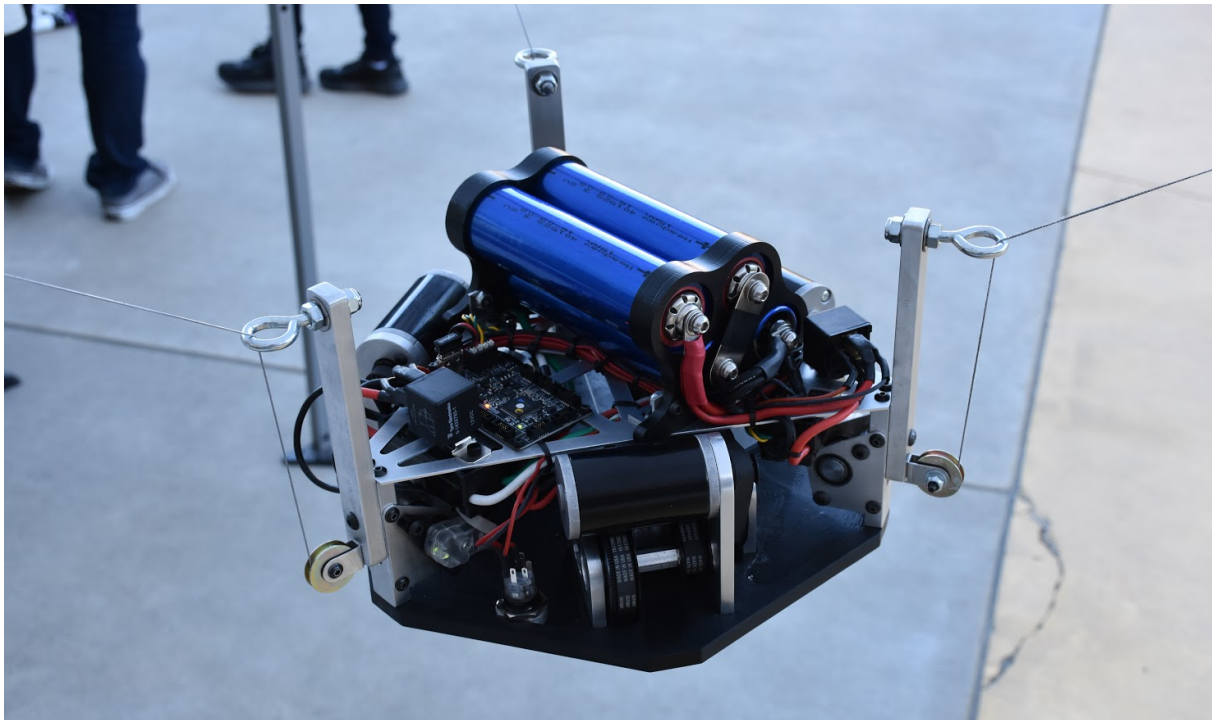
# Remote Cable Gantry



Senior Project

By

Allen Bailey

ELECTRICAL ENGINEERING DEPARTMENT

California Polytechnic State University

San Luis Obispo

2017

# Table of Contents

# Abstract

The Remote Cable Gantry is a robotic system that was initially intended to aid in the art of aerial videography. It was designed to enable novice and expert users alike to capture both video footage and audio from perspectives unachievable by current methods. This system uses a series of cables to control the position of a camera gimbal in a defined 3D space and, as a self-contained unit, is portable and easy to use. The Remote Cable Gantry offers a quiet, intuitive, and safe alternative to existing technology, which has been limiting the market and potential of aerial photography and videography. Although this 3D positioning system can be applied to any system that requires being controlled in 3D space, the scope of this project is limited to operating with current camera stabilization gimbal systems for aerial videography.

# Background and Introduction

Around the start of the 1800s, both photography and aerial photography took flight. Some of the earliest instances of aerial photography included pigeons, kites, blimps, and “roofs of very tall buildings”[1]. As technology progressed, the methods transitioned to airplanes, rockets, satellites and many other methods.

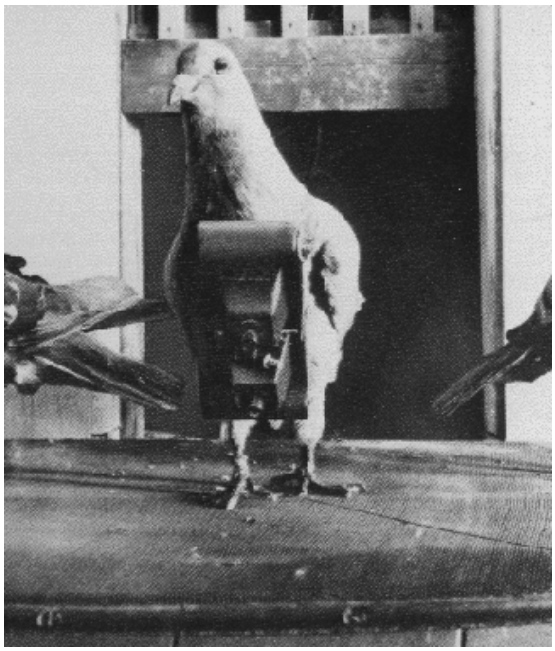


Figure 1: Pigeon carrying a camera [1]

For local aerial photography, current methods include specialized equipment, quadcopters, and remote controlled planes. Aerial photography and videography presents a perspective that is different to everyday experiences, however it is not easily achievable. There are whole companies, namely SkyCam[2], where aerial videography is their business. SkyCam has contracts with the National Football League and the International Olympic Committee to operate cameras at their events to capture angles of view not accessible by ground-based cameras.


Figure 2: SkyCam at a football stadium [2]

Some solutions for indoor situations include studio tracks for cameras to ride around on. These are typical in newsrooms where many camera angles are needed [3].

Lastly, in recent years, quadcopters have taken off in function and accessibility. Many quadcopters come with a camera built in to give a first person view of the flight. This is done on a separate communication system than the control of the quadcopter and typically relays the image from the camera to either a stand-alone monitor or video goggles.

# Project Description

The Remote Cable Gantry is a system of cables and winches that define a position in 3d space by controlling the cable length. Since there are three cables and anchor points, defining the line lengths limit the position to two unique spots that share an axis normal to the plane of the anchor points. With the addition of gravity, a single unique equilibrium position can be controlled. This project is in essence a miniature version of a product produced by SkyCam. This implementation is optimized for portability and ease of use. The Remote Cable Gantry is an alternative to current systems to capture aerial footage while being affordable to the general public use.

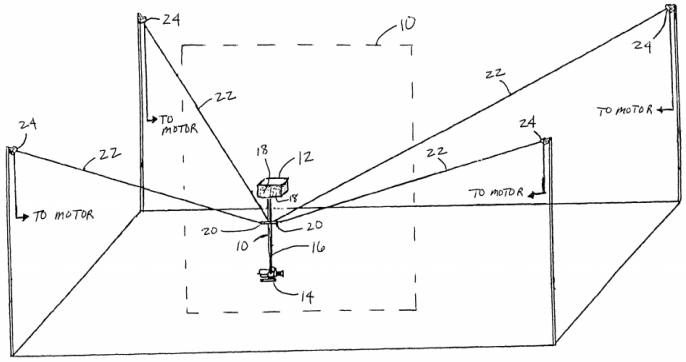
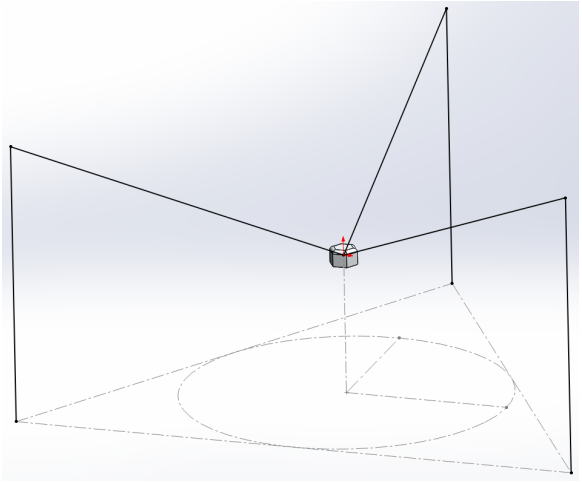



Figure 3: SkyCam cable system [2]



Where SkyCam relies on having many cables or other control points, the Remote Cable Gantry gives up some control for the benefit of ease of setup. To fully constrain all six degrees of freedom of movement in 3D, (three translational, three rotational), the gantry would need seven or more cables. Since a gimbal can be used to counter the rotational errors, I optimized it to be easy to set up as opposed to defining the orientation with more cables.

Figure 4: Remote Cable Gantry setup initial sketch

# Market Research and Analysis

There are a few options on the market for aerial photography which address slightly different applications. The most relatable is SkyCam as mentioned previously. However this is built for large events and financially large customers. Quadcopters is another similar solution that is more in the same customer market. Although the mobility of quadcopters exceeds the performance expectation of the Remote Cable Gantry, they are limited by battery life, noise and safety concerns. Other more niche solutions are studio tracks and camera boom arms, which are similarly designed for indoor use, but lack mobility.

## Customer Archetype

The main group of customers will be freelance photographers and videographers that are mostly middle class to upper class in financial standing. It is a product that goes along with the expensive hobby of photography. These may be used by youtubers that currently use quadcopters but want the audio from their videos, or photographers hired for small events like weddings or concerts. The quiet and safe operation becomes a more important role to photographers who operate around people or animals. College age students are also a natural customer for this product as it is on the edge of being tech savvy with some understanding of interfaces of new technologies. Although this is on the edge of the price range of a college student, some students will already have this hobby developed and relatively speaking the addition of this product is inexpensive. Another possible need is to pursue a degree in the arts where this will be a useful asset to have. In comparison to alternatives, this offers a price range that is just in reach of technology indoctrinated youth. The Remote Cable Gantry brings the shot angles of outdoors aerial photography, inside. This allows for new content that previously is inaccessible to the customers mentioned above.

Another group that this product should be attractive to is management of theater and performing arts. Colleges and performing arts centers could enhance the documentation of shows or performances by incorporating the Remote Cable Gantry into their stage system. As this customer is already familiar with running stage equipment, the operation entry level for this product is negligible.

## Pain Relievers, Gains, and Improvements

Video and photography technology is becoming more accessible. This is allowing for much more content creation and dilution of unique footage. This product adapts an underutilized perspective to a new market of hobbyist and freelance work. Current alternatives of similar functionality are as expensive or more so than the proposed cost of the Remote Cable Gantry. Quiet operation is the next big advantage. There is currently a huge market for quadcopter and RC planes. The benefit of this product is it does not require noisy and potentially dangerous props for it to function. The RC community already uses photography equipment which will aid in acceptance of this new product.

## Market Description

The Remote Cable Gantry is system to control the position of a platform in 3D space. By careful design, the construction of this product will fulfill demands of the market by meeting current individual amateurs and professionals needs. Current limitations to available solutions are centered around mobility, ease of use, and acoustic noise level. A comparison of these technologies can be seen in the graph below.

Table 1: Limitation comparison of current technologies

| Product | Pros | Cons |
|---|---|---|
| Remote Cable Gantry | Moderate coverage in 3d space<br>Inexpensive<br>Easy to setup and use | Requires bounded area to set up anchor points |
| SkyCam | Large coverage in 3d space<br>Silent | Contracted company limits availability for personal use. Very expensive |
| QuadCopters (Phantom 4)[4] | Portable and relatively inexpensive | Very noisy<br>Short operating time |
| Camera Boom Arm | Simple to use<br>Silent<br>inexpensive | Limited mobility as the camera boom arm can only reach so far from a grounded center point. It is also not very portable |
| Studio Track system | Simple to use and good mobility indoors | Not portable<br>Expensive |

The main leverage point for the Remote Cable Gantry to get into the market is for indoor operation. As opposed to landscape aerial photography, the indoor setting is bounded, allowing for easy setup of the Remote Cable Gantry. If there are no anchor points, the Remote Cable Gantry will have to take extra steps to bring in anchor points to operate. This makes outdoor operation more challenging.

## Market Size and Actionable Market

A decent assumption of market size can be seen in the growth of quadcopter sales. As seen in the chart below, popularity of drones has greatly increased in the last few years. This has been enabled by development of inexpensive drone technology and driven by large emerging market making development cost investment worthwhile.

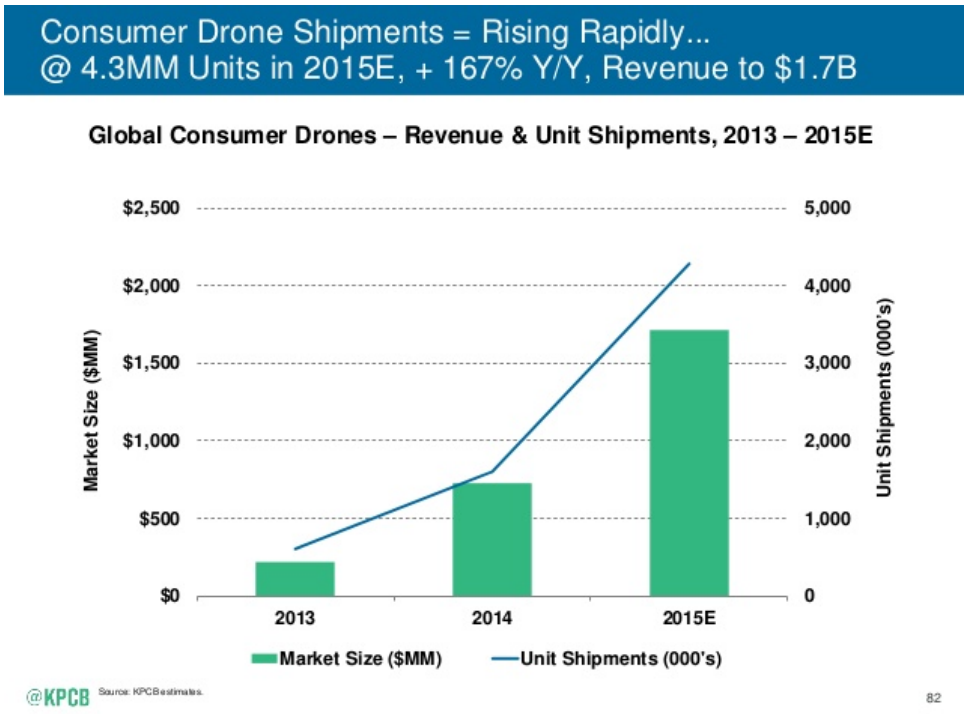



Figure 5: Growth of Drone Shipments from 2013 to 2015 [5]

This primarily targets outdoor use where the Remote Cable Gantry could only get a small percentage of the market where audio is important to the consumer. Indoor markets at the consumer level are practically untouched. Although you could fly a drone inside, there aren't currently many other products for indoor 3d navigation at the general consumer level. This is where the Remote Cable Gantry can get a footing in the market.

## Entry to market

Table 2 below shows an estimate cost to enter the market. By exploring crowd funding and sponsorships, some of the cost could be alleviated. Key potential customers can follow other markets of quadcopters and photo equipment. Primary distributors may include companies like Amazon or Bestbuy.

Table 2: Estimated Cost to Enter Market

| Item | Cost | Explanation |
|---|---|---|
| Prototype | $1000 | Some iteration will need to take place as part of the design process. This iteration will artificially inflate the final unit cost. |
| Marketing | $1000 | Though websites, expositions, and promotions, marketing will take a large portion of the total cost |
| Initial product supply investment | $5000 | To be prepared to sell many products, an initial investment must take place to stock up on final units |
| Total | $10,000 | Rounding up to account for miscellaneous costs unseen in the initial estimate and assuming labor is done in house |

Although the Remote Cable Gantry is solving a niche problem, there are still existing competitors that impact the economic success of marketing this project. Table 3 below outlines some of the competitors and evaluates the specific solutions on mobility, acoustic level, overall cost, and ease of use to the customer.

Table 3: Marketing Comparison of Remote Cable Gantry to Existing Competitors

| Product | Mobility | Acoustic Level | Cost | Ease of Use |
|---|---|---|---|---|
| Remote Cable Gantry | Moderate coverage in 3d space | Quiet | <$500 | Easy, Practice is required for quality of outcome |
| SkyCam | Large coverage in 3d space | Silent | Contract Based (thousands of dollars) | SkyCam provides trained technicians to operate their camera system |
| QuadCopters (Phantom 4)[4] | Most mobile with battery and radio limitations | Very noisy | >$500 | Moderate, practice is required for safety and quality of outcome |
| Camera Boom Arm | Typically bound radially to a point on the ground | Silent | >$500 | Very simple, almost no training required |
| Studio Track system | Limited to track setup | Low | Contract Based (thousands of dollars) | Very simple, may require trained technician |

# Engineering Requirements and Specifications

## Functional Decomposition

In table 4 below, the marketing requirements and the engineering requirements are correlated. The marketing requirements were deduced from the market analysis, whereas the engineering requirement shows the specific engineering goals to accomplish the marketing requirement.

Table 4: Customer Needs and Levels of Importance

| | **Marketing Requirement** | **Engineering Requirement** |
|---|---|---|
| 1 | The system must be remotely operated | Solid wireless communication between a remote and gimbal unit/s with error checking |
| 2 | The system must be easy to setup and use | Simple connection points and initialization software |
| 3 | The system will interface with common camera gimbals | Properly sized mounting holes and electrical connections |
| 4 | The system must have a decent battery life | Power management and correct battery sizing >30min, <4hrs |
| 5 | The system must operate as quiet as possible | <30dB |
| 6 | The system must be lightweight | ~ < 30lbs |
| 7 | The system must be able to operate outside in mild weather | IP54 |

As detailed in the table above, For both operation and safety, the communication between the remote and the unit/s must be well implemented. The system also needs to be somewhat automated to have a working out of the box experience. This ease of use and common mount camera gimbals will make the gantry more easily accepted in the market. Since this system generally is competing against products with short battery life, the minimum is pretty low but it does compete with non powered systems that it has to out compete with other aspects. The system needs to also allow for audio recording by being as quiet as possible, which some of the competitors do really well. For portability, it can not be too heavy to move and setup. However, in use some weight of the end effector will add damping to it and slow it down. Lastly the system needs to operate where there could be some water and dust and loosely follow the regulation that the camera has so an IP54 rating is appropriate.

## Level 0 Block Diagram

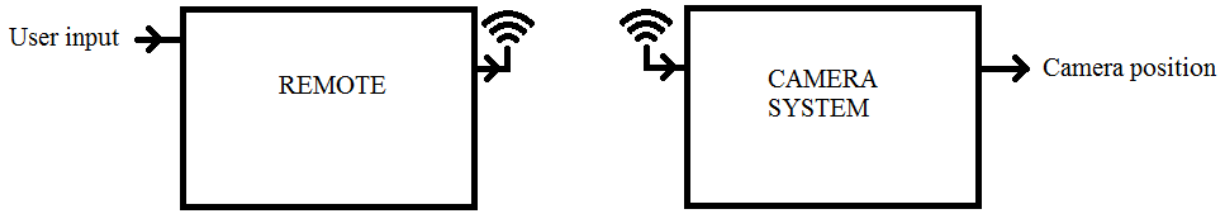



Figure 6: level 0 block diagram

The very top functionality diagram captures the main aspects of this project. The remote takes user input and wirelessly transmits it to some camera system module which controls the camera's position.

## Level 1 Block Diagram

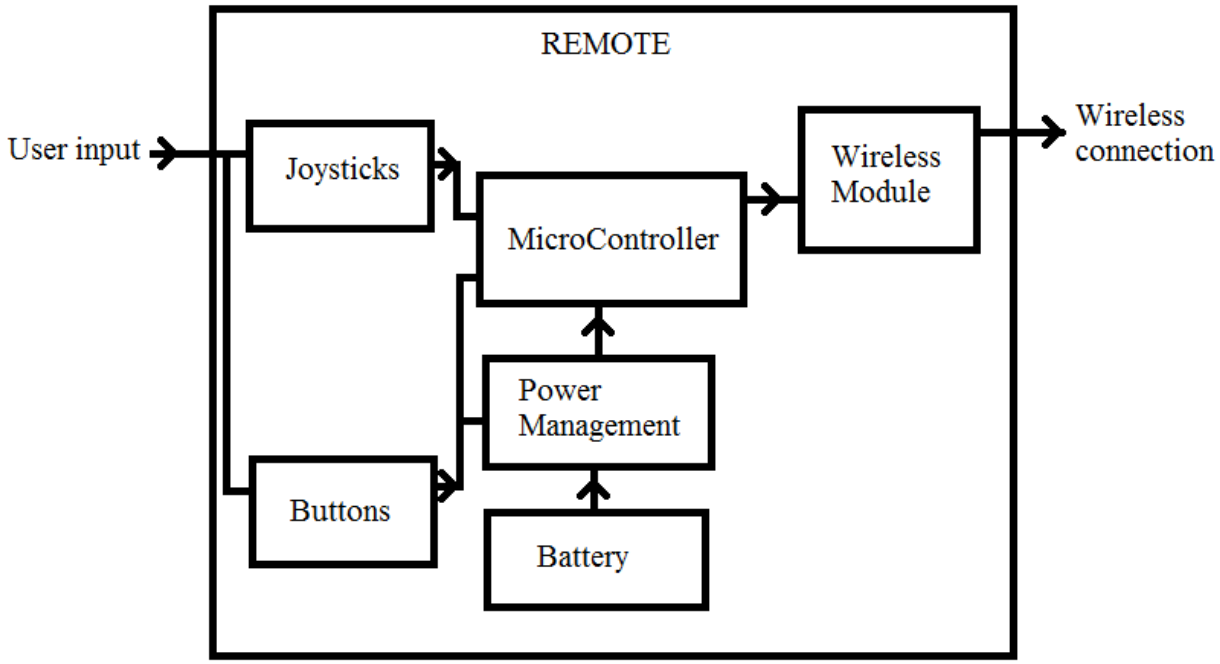



Figure 7: Level 1 Remote Diagram

The Remote has many subunits inside it. Joysticks and Buttons give the microcontroller the ability to sense the user inputs. It needs power management and distribution within the remote. It also needs a wireless subunit to relay information from one microcontroller to the other in the Camera System module.

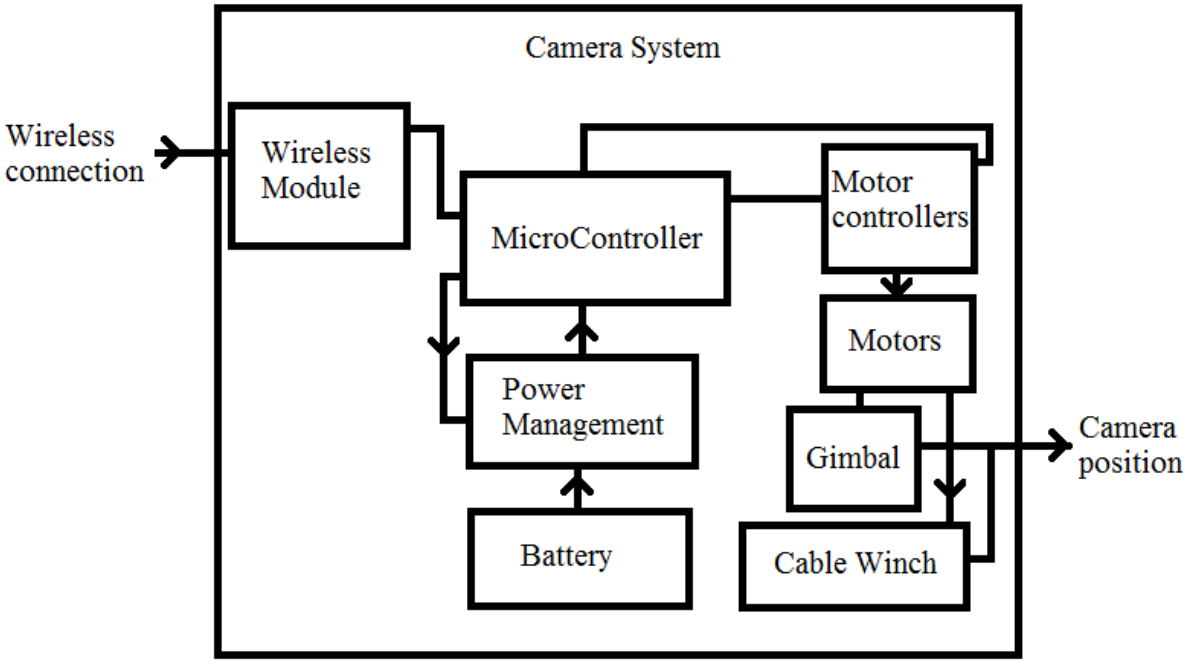



Figure 8: Level 1 Camera System Diagram

Similar to the Remote, the Camera System needs components like the wireless module, microcontroller, and power management. Since the camera has to convert electrical signals to a physical movement, motor controller systems also have to be incorporated. Both the Remote and Camera System are battery powered as shown by the battery subunit.

# Testing/Verification Plan

Table 5: Verification test chart

| Engineering Requirement/Specification | Plan of verification | Result |
|---|---|---|
| Wireless communication robustness | Distance can be tested outdoors of sending a simple signal many times and increasing the distance between the wireless modules. | Wireless gamepad used achieved approximately 45ft before becoming unreliable |
| Weight | During design, the weight can be modeled into the CAD program. The full unit can also be weighed at the full assembly and operation stage. | The final weight came in at 12.3lbs |
| Noise level | During construction of the cable mechanics, the noise level can be monitored to verify the end product will operate silently | No objective result was recorded. Subjectively, at low speeds the gantry was quiet enough for on board audio recording |
| Battery life | The battery life can be tested both by power draw of subunits, and also duration during the functionality testing. | The Remote Cable Gantry ran for over an hour in a high load scenario |
| Functionality | A full systems test that can pan a camera around from point A to B fills this requirement of functionality | The system could pan from point A to B. |
| Panning speed | The panning speed of the module will be calculated during the design phase, however it can be verified at the end when the unit is fully operational | Theoretical calculations concluded the gantry could perform with 1 m/s line speeds |
| IP54 | During testing, subjecting the units to conditions that are specified in the IP54 definition will fulfill this test. | With an enclosure, the gantry would meet this specification |

## Preliminary Design Analysis

There are two competing main concepts for the implementation of the Remote Cable Gantry. The first is having all of the winch cables powered from a central end effector. This centralizes the controls and electronics. This design has the benefit have a smaller number of components, less cost, easier setup, and an overall cleaner unit. It however is potentially much louder, heavier and smaller battery life due to moving around more mass.

Another possible design solution is breaking up the winch modules into separate winches on the anchored side of the cable, similar to what SkyCam currently does as seen in the picture below. Some of the advantages of this are reduced constraints on size speed and noise level, as they are not the things moving around with the camera. There is an additional complexity that is associated with more modules; Each one has to have its own power and communication and it has more parts to set up to get going. If the winches are separate, more management of the cable is possible as size constraints are reduced. Since the top acceleration is limited by the pull of gravity because the lines are assumed in tension, the benefits of a lighter and faster end effector diminishes after a certain point. The separate modules are also more difficult to set up initially and software has to be more diligent on edge cases between modules.

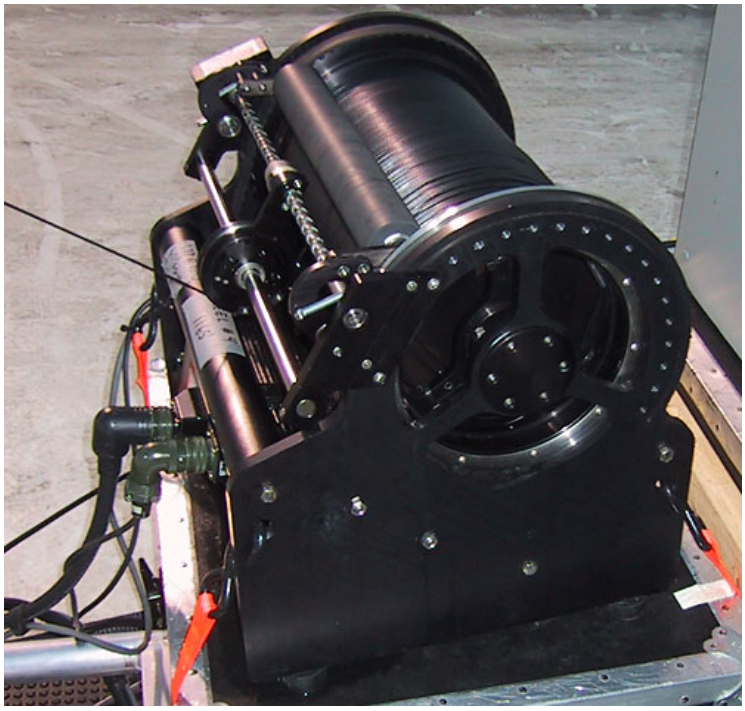


Figure 9: Skycam Winch[6]

## Schedule

The main critical path relies on a detailed design in CAD completed by the time specified below. If this time gets pushed back, the entire project could be delayed. Since the minimum viable product should be easily achieved, there is some buffer to a completed project as anything past the MVP is extra features to the scope. The main milestones are highlighted in table 6.

Table 6: Major Milestones

| Milestone | Date |
|---|---|
| EE460 Report | 11/28/16 |
| Design Review | 2/16/17 |
| Design Lock-in | 2/18/17 |
| Project Expo | 6/2/17 |

# Implementation

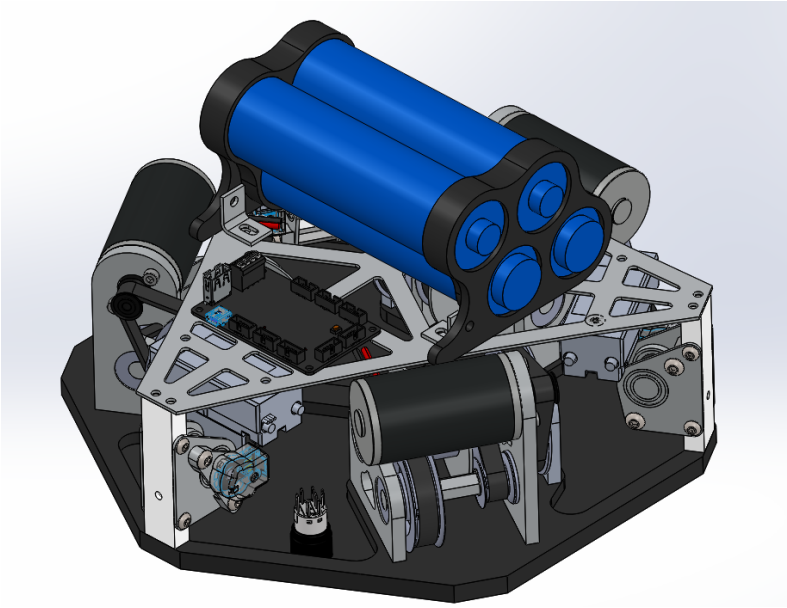


Figure 10: CAD model of Remote Cable Gantry

The implementation and design evolved greatly over time. Most of this iteration was performed with the use of Solidworks, a 3D CAD software. The fabrication was performed on CNC machines at FIRST Robotics team 973 in Atascadero and Trust Automation in San Luis Obispo. Both Solidworks and the manual and CNC machines at these locations made this project a reality. To meet the requirements listed in the initial analysis, the overall size and weight ended being 12.3lbs and 12" x 11.5" x 8". This made it portable and relatively light. Many safety concerns were addressed with sizing the strength of the line and motors. The line is 1mm spectra cable rated to 300lbs of tension. With the selected motors and gearing, there is about a x2 times safety factor as the motors fully stalled can only pull around 150lbs. Many other safety factors come from implementation in software to either limit the amount of tension in the lines, operating zones, and top speeds. Transitioning from CAD to the real world, there were very minimal changes, demonstrating the effectiveness of designing in Solidworks first before investing energy in a physical device.

## Electronics

The electronics of this project changed dramatically from its initial conception. Overall, this transition moved from custom built solutions to finding pre-made off the shelf items that served my function. The electronics can be broken up into two main categories, signals and controls and power distribution.

The initial design called for a substantial custom solution, building up the system from the microcontroller level. This would have required a lot more hardware development and software to support it. The main challenges were the communications and interface to the user and the processing onboard to send signals to motor speed controllers. Initial thoughts were along the lines of using an Arduino based board or similar pre-built micro controller that had a lot of support online. I explored the option of the Arduino Fio, which has a built in socket for interfacing with an xbee wireless module. This

option still seems very attractive as the xbee wireless communication is a well supported high performing wireless solution.

There would have been quite a bit of development required to set up a robust communication. With robotic systems one major safety concern is having control of the system at all times. This requires some method to determining if the current commands are accurate. A “handshake” and “heartbeat” detection would be pretty important to implement to avoid potential problems if the remote lost communications. The handshake makes sure that the Remote Cable Gantry connects to a unique, known controller. However unlikely, if multiple modules operated in the same vicinity, separating the signals would be very important. The heartbeat is another important concept as it ensures that the control signal that is being processed is new and relevant information. If the control system is acting on stale data, the system can become unstable from a user point of view.

Aside from the main gantry system, the user interface is another system that requires some development. As stated before, initially this was going to be a custom solution, building up a user interface controller from buttons, joysticks and other discrete components. As seen in the initial engineering requirements black box diagram, this requires its own signal processing and power management system. It would communicate with an xbee module and send packets, including a heartbeat signal, to the main gantry system. All of the computations would be performed with the knowledge of the current state of the user interface controller. This method is not very challenging but it did require time in development that I could not afford.

Figure 11: CTRE Hero Board

The solution to all of the electronics and development challenges presented above showed itself in the form the Cross the Road Electronics (CTRE) Hero system. The Hero development board is designed specifically for robotic control applications. It is based on a STM32F4 chip and supports all of the common peripherals that you find with other embedded development boards. It runs the .NET Micro Framework and supports with a many libraries and example code, development in C# using Microsoft’s Visual Studio. CTRE also developed extremely cost effective motor speed controllers that the Hero board was designed to support. Although the Hero board and the rest of the CTRE system was more than I planned on spending on the control system, It saved me time, money, and energy in the development stage as most of the development of this system was directly applicable to what I set out to do.

A bonus feature of the Hero board was there was already support for a wireless logitech gamepad, F710. This meant that I had a fully functioning user interface already analyzed and optimized by logitech with many buttons and joysticks. The only thing that I had to ensure was the signal distance, as this wireless gamepad was developed for use at a computer or gaming station at a distance of a few

meters at most. I tested the signal distance experimentally and got about fifty feet before the signal started failing. This unfortunately can not be a permanent solution as fifty feet is a bit limiting for the scope of this project's operation area. It did, however, quickly solve my communication problem and allowed me to develop other aspects of the project.

The Talon SRX motor speed controllers were the key to the whole system for my application. The motor speed controller in its simplest form transforms a low power signal (typically PWM) and modulates a high power line to control the output to a motor. The Talon SRX however is much more advanced. Internally it has a PIC microcontroller that makes more intelligent decisions of how to modulate the output. The key to it being the best solution for my project was its ability to conduct a closed loop PID controller on board. This relieved a lot of development of the actual controller and made the whole control system easier as it abstracted the PID controller from the rest of the system. The Talon also monitors current, temperature, output and input voltage, bus voltage, directly interfaces with both analog and digital sensors, runs control loops at 1kHz rate, and much more. I used the Talon to interface with a magnetic quadrature encoder that I used in the line encoder assemblies. The magnetic encoder I used was also developed by CTRE and had built in support with the Talon. The talon was an obvious solution to my project. With the inclusion of the Talon, all I had to do is communicate position and Velocity setpoints over a CAN bus to the three winch systems. This broke down the development into fewer, more manageable pieces.

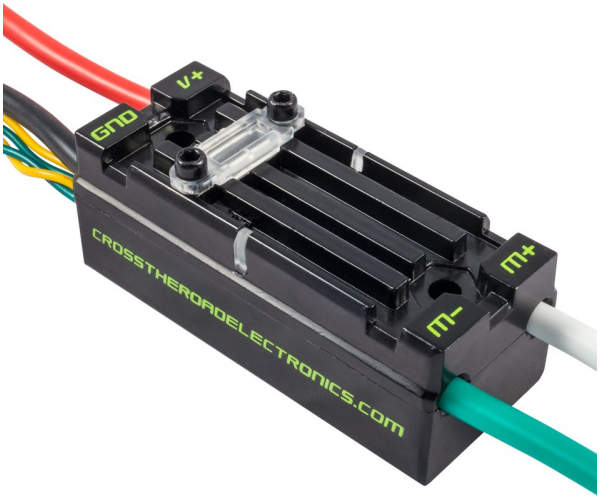


Figure 12: Talon SRX

The power distribution started with battery selection. Building off of the amature and hobbiest aspect, I first designed the system to use common 3cell lipo batteries seen in quadcopters. With further investigation and consulting, I transitioned to a series of LiFePO4 batteries. I stayed with the Lithium Ion battery family for their gravimetric energy and power densities over other chemistries. While performing the calculations for this project, I found a direct relationship to weight and runtime, i.e. half the weight, double the runtime. More about this can be found in the calculations section. I ended with four "LFP-40152SE"15AH 3.2V LiFePO4 batteries, for both their electrical and mechanical specifications. With threaded ends, it was very easy to interface with and the discharge characteristics worked with my application. With the Motor Selection Calculations (see section), these batteries gave sufficient runtime at around an hour of continuous movement at the top of the operating zone where it takes the most power..

After the battery, I wanted to be able to switch the whole system on and off with a simple switch. Most regular button switches are not rated for more than a few amps let alone the current that

the three motors would draw. The solution to this was a relay triggered by a switch. This would allow the high current to be turned on and off with a much lower signal. However, with three motors and accounting for worst case in terms of current, the relay would still be pretty big and expensive. Alternatively, I used three smaller relays that were rated for much less. Although it is more components, it was much cheaper and simpler to implement. I placed the three relays between the battery and each motor speed controller controller and drove them and the Hero board off of the current through the switch.

The switch was selected to conform to the IP54 rating that I wanted and aesthetics. I ended with a PV4F23011-3R4 switch from E-Switch which is rated for 2 amps and is IP65 certified. It also has three built in leds that I later plan to use for signaling at the gantry side of the system.

Each relay for the winches is a 40A non latching automotive relay. Although the motors are quite capable of pulling more than 40A, normal operation should not be far above 15A. These specific relays were mainly chosen by cost.

## Mechanics

Bridging the gap between the electronics and the mechanics are the motors. Motor selection was an important process as it dictated both mechanical and power requirements of the system. Through my research I found that ideally I would use a low RPM brushless motor because of its acoustic and electrical noise performance. I opted to work with a brushed motor, however, due to cost, time, and prior knowledge of its characteristics and performance. Through the FIRST robotics program I have had exposure to both the control system mentioned above and a set of DC brushed motors. The decision between brushed and brushless was also determined by the ability to use the Talon SRX with the motor I selected (the Talon only works with a brushed motor system). Using performance curves from dynamometer tests performed by the distributer, I selected a 150W BAG motor distributed by VexPro and West Coast Products as the motor to use in my project. One of the major deciding factors was the motor's thermal performance in how well it managed stalling and dissipating heat. Its small form-factor and closed body design made it easy to work with and relatively quiet between the motors I was comparing.

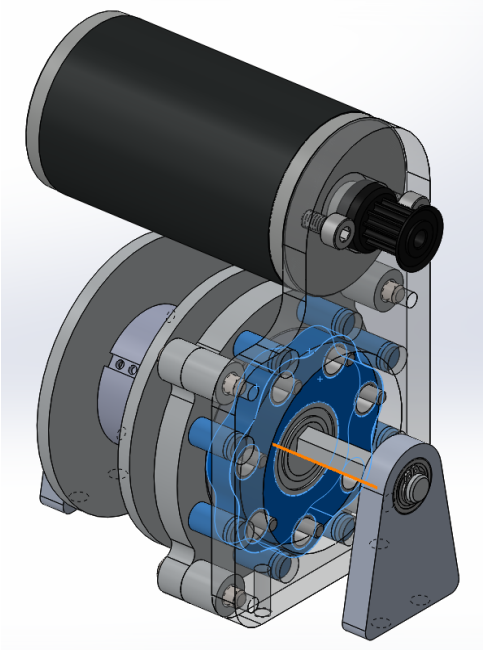


Figure 13: Cycloidal gearbox

The largest mechanical design challenge was the winch. The winch gearboxes went through four different designs, two of which were manufactured. The initial design called for a cycloidal gearing stage. A cycloidal gearbox is a

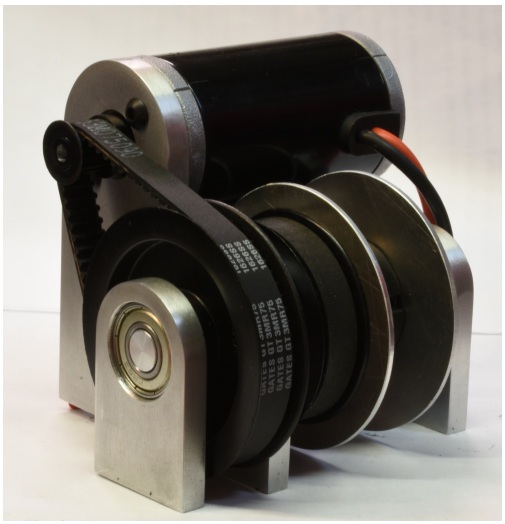


Figure 14: Belt stage gearbox

high efficiency, high gear ratio setup that seemed perfect for my project. During production of this design, the challenges of this design became very apparent. For the sake of time, I abandoned that design and made a three stage belt drive gearbox. The total gearing was 35:1 to reduce speed and increase torque. The design, fabrication, and assembly turned out to be much simpler than the cycloid and was a better choice.

Equally important to the motor and gearbox is the line encoder. To get feedback on the current position of the gantry it is important to accurately measure the line length. The spool poses a challenge with its changing diameter depending on how much line is taken in or out. Putting the sensor on the spool would have been easier to implement but instead I solved the spool problem by making the encoder its own unit that directly measured the line as it passed around a drum. By doing this it removed the nonlinearity that the spool introduced. However it added its own complications. The assembly was simple to implement with the line wrapped around a drum that was tangential to the lines path. To ensure it read correctly, the line always has to be under tension. This is a safe assumption under normal operation, however, the constant tension means the winch can go unstable if you relieve some of the weight with either the ground or another object. This could be solved similar to the end of spool detection discussed in the future improvements section by monitoring accumulation of large error. Overall, this was a clever solution for linearizing the system for ease in determining the position. The encoder is a 1024 count quadrature encoder that gave accuracy down to the thousands of an inch. This was to have more resolution than how accurately I wanted to control. The main structural component also provided frame to hold up the electronics board.

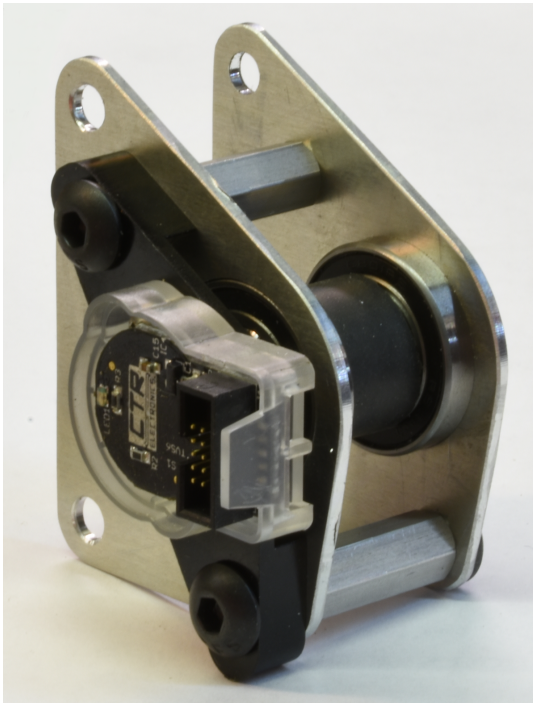


Figure 15: Line encoder module

The frame structure of the gantry was designed to be compact and lightweight. The main base of the gantry was made out of ⅜” ABS plastic. I relieved quite a bit of the weight out of it with pocketing and I wanted the thickness for stiffness. The line encoders held the electronics plate which was made of 0.063” thick aluminum sheet.The spacing and shape helped with the overall rigidity of the frame. One of the design constraints was to make it portable so I made the base plate and whole assembly fit inside a

foot by foot square. Making this dimension larger would have made many other things easier because the size constraint was very challenging to meet.

One aspect of the mechanical that did not come to fruition was a line spooling mechanism. For proper line management fishing reels use a carriage screw to move the line back and forth to evenly load the spool. Because of time constraints I did not develop that for the winch spools. So far it has not posed an issue with the wrapping on the spool.

## Bill of Materials

Below in Table 7 is a rough bill of materials. Some prices were overestimated to leave margin.

Table 7: Remote Cable Gantry Preliminary Bill of Materials

| Camera Cable Gantry | | | | | Total |
|---|---|---|---|---|---|
| | | | | | $985.00 |
| Item | | Source | QTY | Cost/per | Price |
| Remote | | | 1 | $50 | |
| | Logitech F710 | Amazon | 1 | $50 | $50 |
| Cable Module | | | 1 | $935 | |
| | Frame | custom | 1 | $20 | $20 |
| | Battery | Battery space | 1 | $15 | $15 |
| | microcontroller | CTRE | 1 | $60 | $60 |
| | Motor | Vex pro | 3 | $25 | $75 |
| | motor controller | CTRE | 3 | $80 | $240 |
| | cable | mcmaster | 100 | $0 | $15 |
| | gearbox/pulley | custom | 3 | $100 | $300 |
| | switch | digikey | 1 | $20 | $20 |
| | encoder | USdigital | 3 | $40 | $120 |
| | gimbal | hobbyking | 1 | $70 | $70 |

# Calculations

All calculations were performed in a google spreadsheet found at this link: https://docs.google.com/spreadsheets/d/19SguAiz-cQS4HVzyxhnL_cIEUwSOjGKcnN6uHBLw3QE/edit?usp=sharing

## Three Cable Tension

To get an understanding of what size motors I had to use, I had to calculate the tensions and forces involved with a three cable system. I first solved the two cable system at rest by drawing the free body diagram and breaking up the vectors into their components using Newton's second law. With this you get two equations,

$$\sum F_x = L1_x + L2_z = 0 \text{ and } \sum F_z = L1_z + L2_z + g = 0$$

Where:
$L1_n$ is the x or z component of the tension vector from line 1
$L2_n$ is the x or z component of the tension vector from line 2
g is the force of gravity on the gantry in the negative z direction

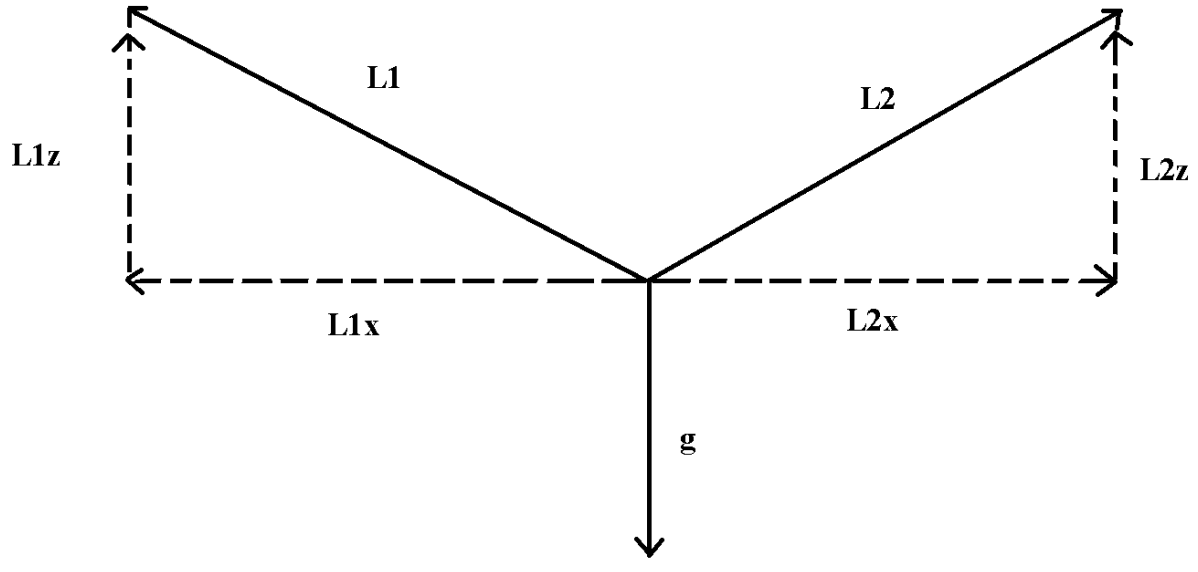



Figure 16: Free body diagram of two cable system

These equations can be solved in a system of equations using the trigonometric relationship between the x and z components of a vector namely, the pythagorean theorem . By setting the sum of

the forces to zero you assume the gantry is at rest as the zero acceleration drives that side of the equation to zero. This can be broadened to three lines and three dimensions.

$$\sum F_x = L1_x + L2_x + L3_x = 0,\ \sum F_y = L1_y + L2_y + L3_y = 0 \text{ and } \sum F_z = L1_z + L2_z + L3_z + g = 0$$

Again these equations assume the gantry is at rest and can be solved with a system of equations using trigonometry. It is computationally intensive to perform trigonometric functions on an embedded platform. For this reason I decided to solve the system of equations using information given about the x,y,z coordinates of the three anchor points and the gantry. By forming a unit vector along the direction of the line from the gantry, I could project the line's tension into the x,y, and z components using vector projections. From there it became a system of three equations (force equations detailed above) and three unknowns (tension in lines 1,2, and 3) which could be solved with more computationally friendly linear algebra.

Given the relative coordinates of the anchor points and gantry I could now calculate tensions in all three lines at any point due to gravity pulling down on the gantry. To solve the dynamic aspect of the gantry moving, I no longer assumed the acceleration was zero and accounted for the mass of the object too.

$$\sum F_n = L1_n + L2_n + L3_n = m * a_n$$

Where:
m is the mass of the gantry
$a_n$ is the desired acceleration in the n component either x,y, or z.

These calculations would be necessary to control the acceleration and calculate the load on the motors at all points. They also gave me better insight of how the tension operated in the defined space. The chart below is a three dimensional graph that shows the tension for the two line system, where x and y are the x and z coordinates in a 2D plane and the z axis is the tension in one line.

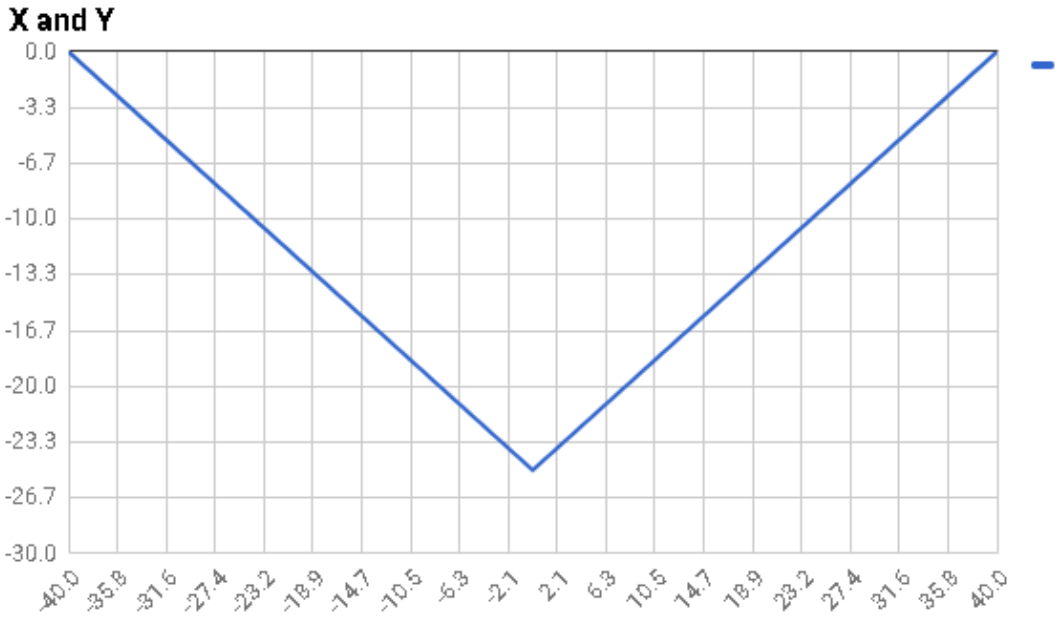



Figure 17: Position of the lines in XZ on plane XY

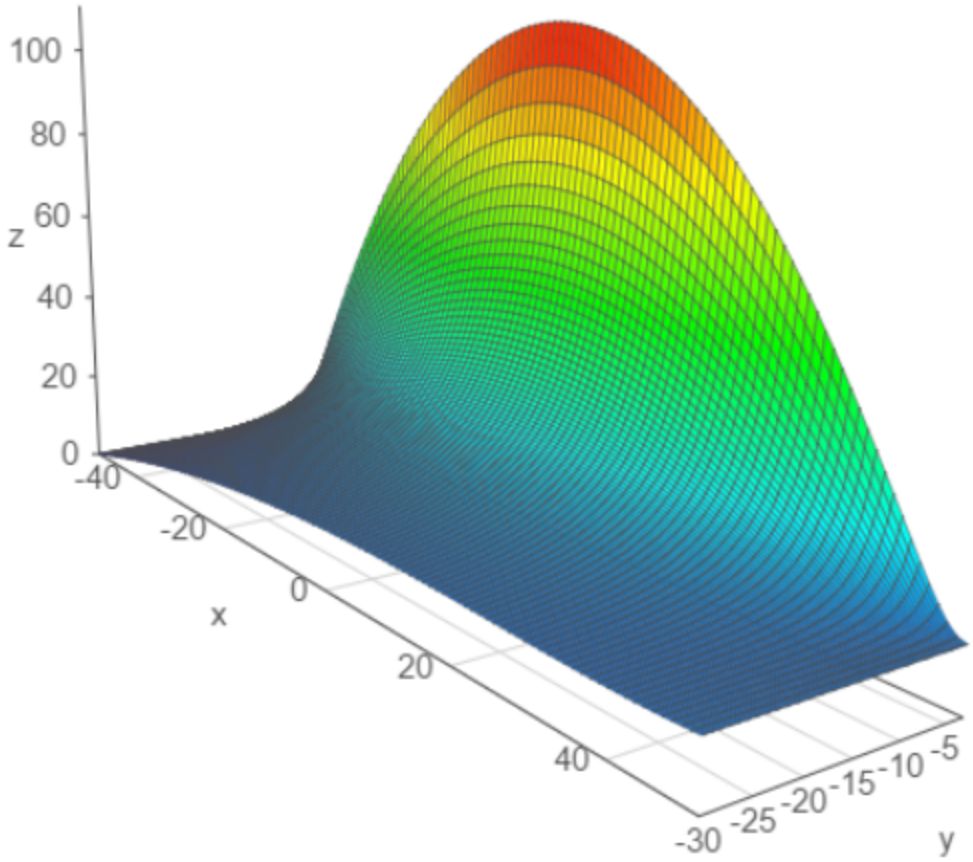



Figure 18: Tension graph in one line of a two cable system

There are three observations to be made from Figure 18. The tension is constant and proportional to gravity directly underneath the line being analyzed. In this case the line being analyzed is the right line in a two line system on the right side of the graph. The tension in the line, again represented by the z axis, plateaus to a constant, non-zero value. This indicates that when the gantry is directly below one line, that line supports all of the weight of the gantry, but no additional force is need. This is supported by the other side of the graph where, when the gantry is directly under the other cable, the cable being analyzed has zero tension. The second realization was as you ascend in the Z axis in the 2D system (represented by the y axis in this graph), the tension in the line goes up exponentially. This is intuitive as if you are trying to raise a weight in the middle of a rope, it will take more and more tension to make it raise higher and higher. This comes from the geometry and directions of forces. As the weight rises the cables are pulling more to the left and right than they are to lift the weight up. Finally, the least intuitive conclusion was that at any fixed altitude in Z the tension in the lines are greatest when the gantry is at the midpoint between the anchor points. This is demonstrated by the parabolic nature in x and z. This result also gave me the understanding that directly in the center of the three cable system triangle contained the most tension in all three lines. As the gantry moves toward any anchor point, the tensions in all of the lines go down. With this analysis I knew to select my motors based on the point where there was the most tension, in the direct center.

## Coordinate Transformation

In order to accurately move the gantry around in position, I had to solve the problem of calculating line lengths. This calculation is simply the distance formula in 3D.

$$L_A=\text{sqrt}((A_x-G_x)^2+(A_y-G_y)^2+(A_z-G_z)^2)$$

Where:
$A_n$ is the n coordinate of the anchor attached to the line in question.
$G_n$ is the n coordinate of the gantry.

These are the core calculations for this project. I can pass distance setpoints to the PID loops for each line depending on where I want the gantry to be relative to the anchor points.

Taking this equation one more step, the velocity transformation can be found by differentiating the distance formula. This allows the gantry to be given a velocity setpoint like move 1 m/sec in the x direction and it can calculate the instantaneous velocity for each line. I say instantaneous velocity because the velocity of the line is highly dependent on the current position of the gantry relative to the anchor points. The inputs to the velocity then become the desired x,y, and z velocities, as well as the current positions of the gantry and anchor points. This method could be used to calculate the instantaneous acceleration of each cable and then compare it against the calculations of tension based on the current acceleration. Since the control loops run much faster than the system can react, position

control seemed sufficient. When acceleration limiting comes into play, see below in the Future Improvements section, these other controls will become more important.

## Motor Selection Calculation

Motor selection is one of the most important decisions that had to be made in the design process. Many factors went into selecting the motors for the winches. For the sake of time, I limited myself to motors I have had experience with in the FIRST Robotics program. All of those options were brushed DC motors with extensive testing and data available. To understand how much power I would need in each motor, I first had to calculate what loads were involved with running the gantry. By giving a ballpark weight for the gantry of about 11lbs, later refined by a CAD model, and realizing the load was maximum directly in the center of the field of operation (see Three Cable Tension), I settled with Vex Pro's BAG motor by iteratively going through different gear ratios to determine the effect on output torque and freespeed. This motor was also most desirable from the selection of motors Vex Pro sells as the small form factor and lock rotor/thermal test proved to be very good relative to the other motors. Since the brushed DC motor can be modeled with a simple line between free speed and stall torque, it was easy to analyze the performance of the motor under different loads calculated in the tension calculations.

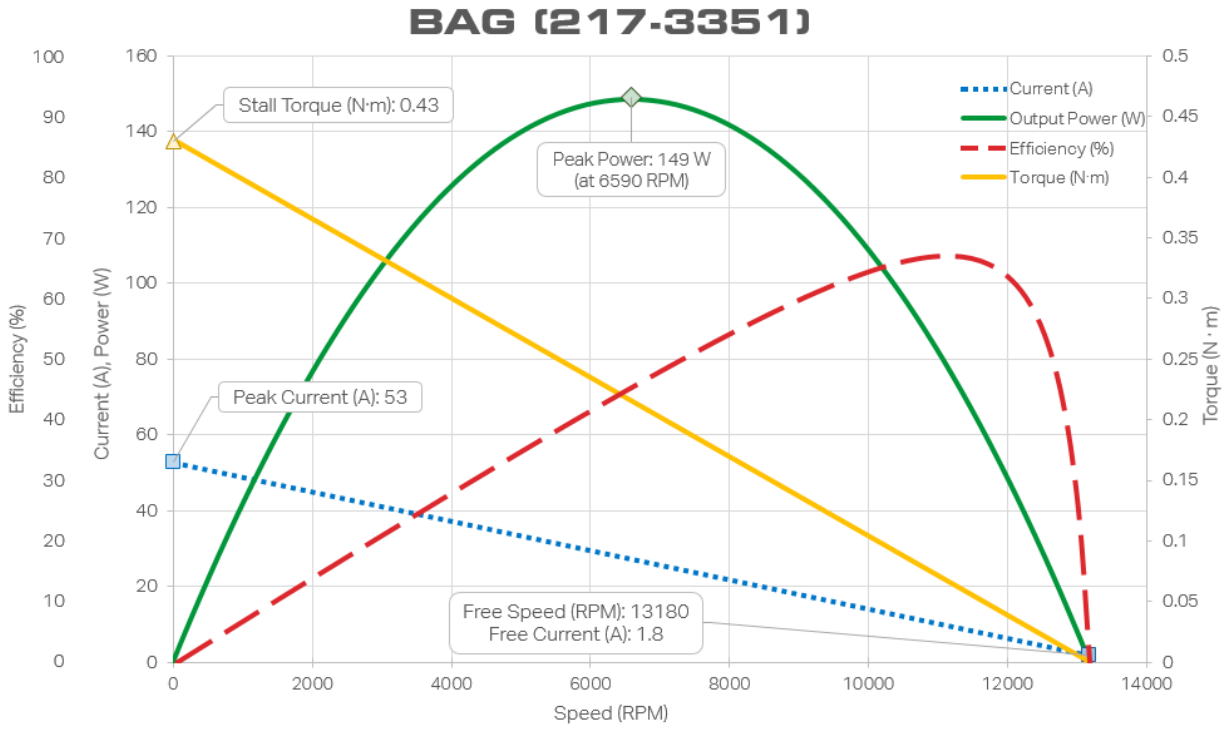



Figure 19: BAG motor curve [7]

Figure 19 above shows the motor curve for the BAG motor that was supplied by Vex Pro. The conversion from tension in the line to torque on the motor is just a matter of multiplying the tension by the spool radius.

$$\tau = F*d$$

Where:
$\tau$ is the torque
F is the tension in the line
d is the radius of the spool

I settled on having a gear ratio of 35.6:1 as it gave me a theoretical stall torque of 16.3N*m and 395.4RPM. At the average radius of the spool of 0.938 inches, This equated to roughly 154lbs of force on the lines and a cable linear speed of almost 1 meter/sec. With these ratios I could also calculate that the overall voltages and currents to the motor were low enough to operate the motor in a thermally safe way. Using the provided graph from Vex Pro, I determined that when the gantry is holding position, it shouldn't use more than around 10A (2.24V). This put the motor in a decent range for operating for a long period of time. With a maximum of 154lbs, this also gave a x2 safety factor between the rating of the line I am using at 300lbs and the maximum force the motors could impose on the line.

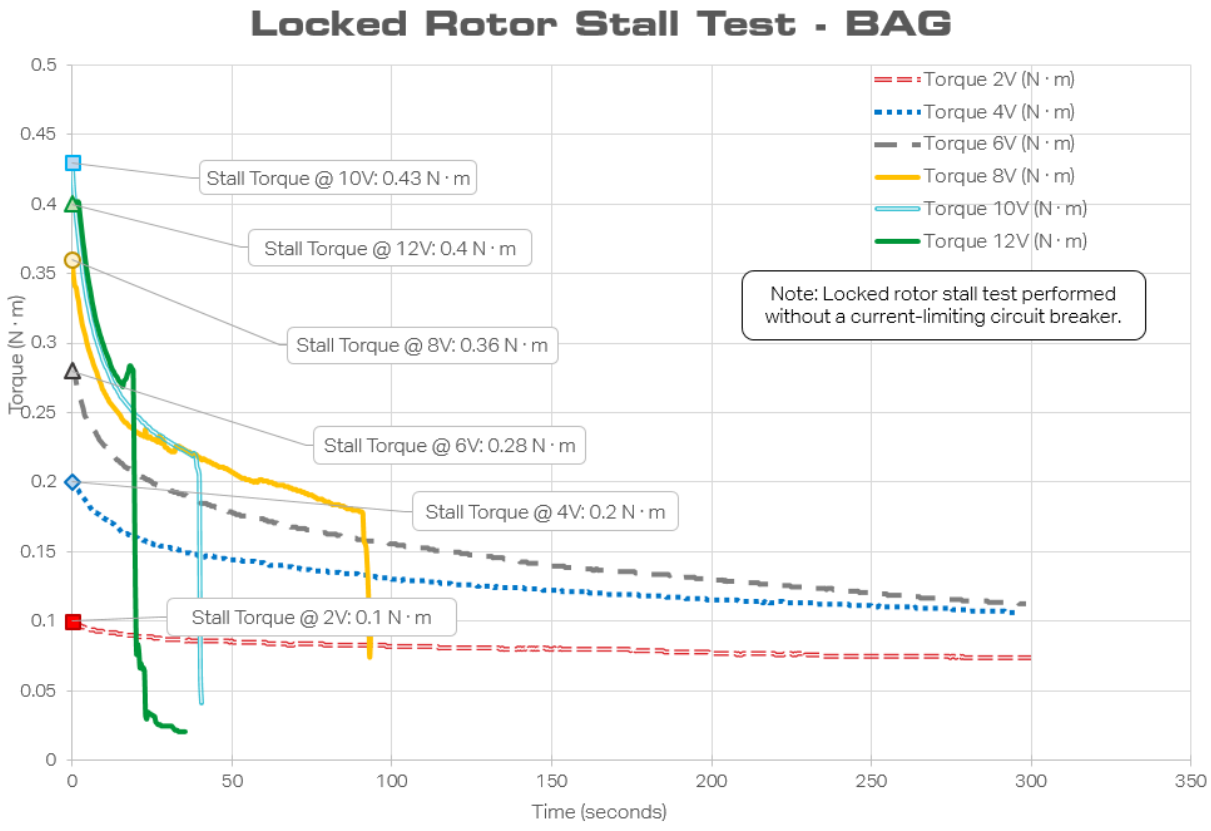



Figure 20: BAG locked rotor stall test [1]

The Locked Rotor Stall Test chart gave me test data that showed me that for extended periods of time, 4V is about the maximum voltage that I would want to stall at indefinitely. Putting 4V into my calculations to determine maximum free speed and stall torque, I found that the gantry could hold its position in the middle of the triangle formed by the anchor points at a height that is 5% of the total width of the triangle. For example, if the width of the triangle was 100ft, these settings would allow the gantry to hold position in the center of the triangle, 5ft below the lowest anchor point. This performance seemed sufficient so I continued through the design phase.

## Acceleration Limit

The acceleration limit exists because of the dependence on gravity. Since gravity is a constant force at all times, it sometimes limits the operation of the gantry's translational moves. This problem also stems from the nonlinear behavior of rope. The line can transmit tension only, never in compression. The Z direction is the easiest to relate to initially. Since gravity is the only force pushing the gantry down, it can not accelerate faster straight downwards than 9.8 m/sec^2. If the gantry position and anchor points are known, the limit to xy is a function of the angle formed by each line. The maximum acceleration towards any anchor point will be when there is no tension pulling the gantry back, so for this analysis each line can be analyzed independently. Since the Z component of maximum acceleration is fixed due to gravity, it forces one solution to the tension vector of the line. For there not to be any acceleration in Z as the gantry traverses in XY the acceleration must be limited. To find the maximum acceleration at any given point it is necessary to find the minimum of the maximum acceleration in each line.

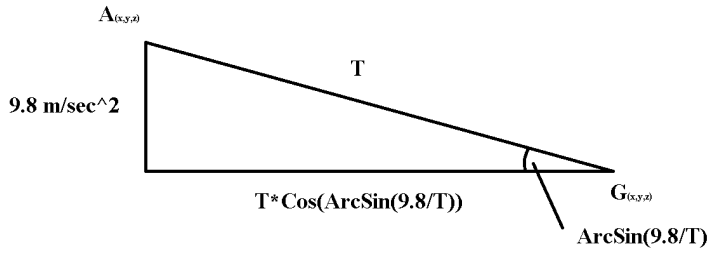



Figure 21: Trigonometric relationship between Z and XY acceleration

Following through with the trigonometry, the relationship can be reduced to a relatively simple equation.

$$\alpha_{max} = 9.81*(\sqrt{((A_x-G_x)^2+(A_y-G_y)^2)/(A_z-G_z)})$$

Where:
$A_n$ is the nth component of the position of anchor A (x,y,z)
$G_n$ is the nth component of the position of anchor G (x,y,z)

With this relation, the maximum acceleration can be calculated. The general trend follows the more acute the angle that is formed from horizontal by the positions of the gantry and anchor, i.e. the closer to the top the gantry goes, the higher translational acceleration can be realized. This is logical as the related triangles of the tensions and positions make for a larger XY component with a fixed Z component if the angle is very small.

# Future Improvements

## Automated anchor point calculation

The setup of the Remote Cable Gantry is still not as intuitive as the project first seeked to solve. Using a system that is very similar to how GPS works, the gantry would calculate the anchor points using multilateration[8]. In essence, by giving the system known positions of the gantry and measuring the change in line length between them the gantry can multilaterate and resolve the positions of the anchor points. This is intended to be a setup procedure after which it would go back to the basic line length calculation. The minimum amount of points needed to solve the position in 3D is four, however, better accuracy can be obtained if more data points are taken and using a least squares approximation to solve the overdefined system.

## Live error correction using IMU

The Remote Cable Gantry is currently pretty limited with the information it receives. By using an inertial measurement unit (IMU) it would be able to understand how it is moving beyond relying on the lengths of the line. This makes the string line lengths and calculations a little less critical and allows for live error correction. The easiest case to analyze is if the string wraps around a foreign object and is messing up the active location of the anchor point. With an IMU, the change in the dynamics of the system can be accounted for by analyzing how exactly the gantry is moving and recalculating the anchor points using multilateration as shown above. In this way it gives the navigation a more robust solution giving the user the easiest operation.

## Haptic feedback

With the current control system, the Logitech joystick used has rumblers in it. The feature of haptic feedback would be very useful to get feedback from the gantry. Alarms could range from low battery to attempting to navigate outside of defined boundary limits, see boundary limits below. Currently the haptic feedback is the only signal the operator can get locally about what is happening with the gantry. Lights could be added to the gantry itself but that reduces the stealthy and non intrusive aspect. Until the control system gets an upgrade to where you get a screen or other indicator at the joysticks, haptic feedback is one of the best ways to convey information back to the operator.

## Waypoint navigation

One feature that more and more quadcopters are coming out with is waypoint navigation. This is autonomously moving from point to point along a predetermined path. The implementation of this is one of the easier tasks especially with how the structure of the control system is set up currently. Since I only have to give each motor controller a setpoint to navigate to, I can make the setpoints driven from a collection of predetermined points that make up a path. This idea would be very useful for an operator to fly in very specific arcs or, as illustrated in the Other Uses section, draw specific shapes.

## End of spool detection

One issue in the current system is the edge case where the spool runs out of line and starts spooling in the opposite direction. With the current setup, this unfortunately immediately makes the system unstable as it is returning positive feedback. This is caused by the reverse of direction of the cable even though the motor continues to spin the same way. One solution to this is having sensors built into both the spool and line encoder. By comparing the signals from these, it could be concluded that the system has reached the end of the spool. This could be determined as well with the current hardware, but it requires setting a trigger for detecting a spike in the error signal. Lastly, the implementation of an IMU as stated above, would simply flip the position of the anchor point. Although this doesn't seem to be a very robust solution, it would technically allow the gantry to continue running. It does make the rest of the automation much harder to catch all of the edge cases. The best solution would be to add another sensor and monitor the error between them, or from a user standpoint, never size your triangle of operation above a precalculated area.

## Gimbal support

The Remote Cable Gantry was designed for general use with cameras on gimbals, hanging underneath the gantry module. Software support for the general camera gimbal would complete the all in one package for aerial videography. The most challenging feature that could be added with gimbal control is a gimbal-centric control of the camera module. This would augment the controls to always be from the perspective of the camera so as the camera rotates, forward would remain forward as observed by the camera. This makes flying the camera module via just the camera feed, much more intuitive.

## Boundary limits

There are many limits to the motion of the camera module during operation. The biggest constraint is the camera module can only move within the projection of the triangle formed by the anchor points. Points outside of the triangle require the opposing line have a negative tension, in other words, pushing rope. Limits can be imposed in software to keep the module inside this triangle. Calculations for

determining if the desired point lies within a triangle can be found in the calculations section. Other boundary limits include a ceiling and floor limit. Ceiling limits keeps the camera module attempting to go too high and draw massive amounts of tension in the lines. A simple limit to possible z coordinate values would keep the camera module within range. However, if tension is the main concern for a ceiling, surfaces of constant tension could be calculated in the initialization. These surfaces of constant tension are different for each anchor and are teardrop shaped with the lowest point being in the center of the triangle. A similar constraint on z height can be implemented to make a virtual floor. Other, more complex boundary limits could be user defined to keep the camera module from flying in specific zones.

## Acceleration limit

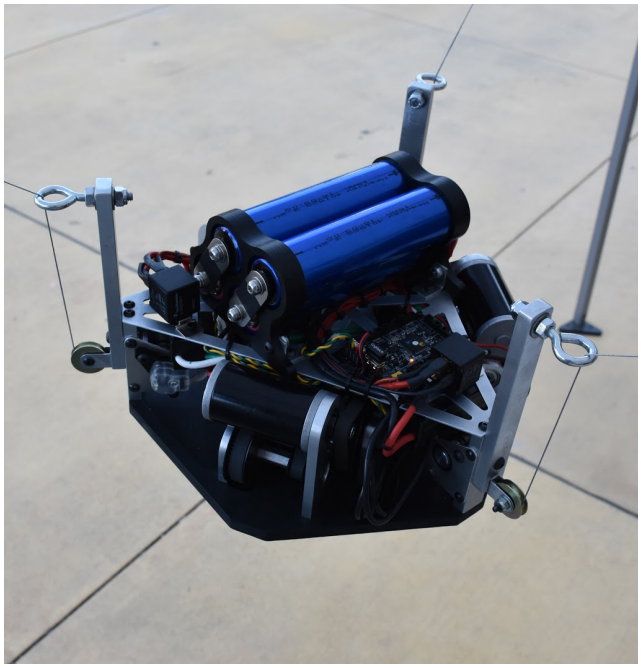
Due to the dependance of gravity to fully constrain the camera module in 3d space, there is an acceleration limit to its movement. The Z component of the tension vector, that is colinear to the line, must equal the desired Z acceleration. This limit is only a function of the angle of the line from the camera module to the anchor point. For example, the camera can not accelerate very fast in xy without any z, when it is close to underneath the anchor point as the vector in the xy plane is a very small component of the tension vector, correlating to force and acceleration. An additional line and anchor could be added out of plane of the original 3 lines to fully constrain the system, or a software acceleration limit can be implemented. The software implementation would require calculating, in real time, the acceleration limit at the current position. Acceleration and deceleration would also be different for almost every point as the angles to the 3 anchor points are usually different. This does make it more stable around the limits of operation because as it approaches a boundary, the distance and angle to opposing anchors becomes shallower and more forgiving.

# Alternate Uses

This project was first intended to be a solution to amature aerial videography, however through the process, many other possible uses arose. The closest alternate use still focuses on using a camera but since the gantry knows its position very well, it can be used in surveying. Possibilities range from a field of plants or animals that people would like to know the location of, surveying a crime scene where minimal disturbance is needed, or inspections in areas unfit for people, drones, or other methods. Another possibility is using the gantry as a large scale plotter. An example of this is drawing markers or artwork on football fields. There are robotic solutions that sketch out insignias on fields, however their operation is limited in speed as their precision is determined by reacting off of the ground and beacon system. Since the location and calculations of the Remote Cable Gantry are greatly intertwined, the gantry would be able to sketch out field markings much faster than current solutions as it reacts off of solid measurements of distance. I have already seen similar systems work as large scale 3D printers, however other design choices would have to be made. This system is optimal for tasks that require low load, high precision movement in either 2D or 3D.

# Closing Remarks

The Remote Cable Gantry started out as an idea for a personal project in the summer of my junior year. Although it isn't as focused on Electrical Engineering as most other senior projects, this project brought together many aspects of things I have learned over the course of my college career, both inside and outside of classes. This project touched many different subjects including controls, power electronics, embedded systems, system level integration, and even mechanical design. The time frame to complete all of the tasks to get a minimum viable product was very challenging. In hindsight, I should have included a couple more people to share the tasks. Although there are many more improvements to apply, as seen by the Future Implementations section, I was overall please with the outcome (seen on the cover page). As the concluding project to my Cal Poly career, the Remote Cable Gantry taught me many things from battery technologies to time management and reflects upon all of the skills and knowledge I have aquired over the past few years.



# References

[1] Paul R. Baumann. “History of Remote Sensing, Aerial Photography.” Internet: http://www.oneonta.edu/faculty/baumanpr/geosat2/RS%20History%20I/RS-History-Part-1.htm, [OCT. 22,2016].

[2] “Skymo.” Internet: http://skycam.tv.s28625.gridserver.com/skymo/, [OCT. 22,2016].

[3] “Telemetrics’ New TG4M TeleGlide Camera Track System Optimizes Space Requirements” Internet: http://www.telemetricsinc.com/press-releases/telemetrics-new-tg4m-teleglide-camera-track-system-optimizes-space-requirements, [OCT. 23,2016].

[4] ”About DJI” Internet.http://www.dji.com/company, [OCT 24,2016].

[5] ”The Consumer Drone Market: Trend Analysis.” Internet. http://emberify.com/blog/drone-market-analysis/, [OCT 24,2016].

[6] ”Monday Night Football: Behind the Scenes” Internet. http://www.digitalproducer.com/articles/viewarticle.jsp?id=29472-5

[7] ”VEX Robotics Motor Data - BAG Motor” Internet. http://motors.vex.com/vexpro-motors/bag-motor

[8] ”Multilateration” Internet. https://en.wikipedia.org/wiki/Multilateration

# Appendix A: Initial Senior Project Analysis

Project Title: Remote Cable Gantry
Student: Allen Bailey
Advisor: Clay McKell

1. Summary of Functional Requirements
    a. Overall capabilities:
        i. The Remote Cable Gantry shall easily control the position of a camera in a defined 3d space
        ii. The system shall be portable and lightweight
        iii. The system shall operate as quietly as possible
2. Primary Constraints
    a. Challenges associated with the project
        i. Accurate control systems for the cables
        ii. Power management
        iii. Mechanics of a silent operation
3. Economic
    a. What will impact the result?
        i. Human capital: If this product is produced for consumers, there is an investment in the employees that are hired to make these products as well as supply jobs.
        ii. Manufactured Capital: The outcome of the project is a manufactured good.
        iii. Financial Capital: If successful the Remote Cable Gantry could produce a net profit and add competition to the market.
        iv. Natural Capital: Since this product uses electronics, especially the battery system, there will be an effect on nature.
        v. Costs: On a small scale where I am doing all of the design and manufacturing, the main cost is in components, mainly the electromechanical components and batteries. On a large scale, more of the cost is shifted to overhead of running a business as the overall cost of a unit is rather small in that scheme.
4. If manufactured on a commercial basis:
    a. Expect <100 units sold in the first year
    b. Total cost of the prototype is around $500 as per table 9
    c. Actual manufacturing costs of each system will be possibly as low as $300
    d. Should the device sell for $500 a unit, the profit would be <$20000 for the first year
    e. To run the device constantly, maintenance and replacement of the battery could be as much as $50 a year

5. Environmental

    a. The environmental impact is common to most electronic consumer goods. Since this project is specifically using a lipo battery, that contributes the most environmental impact to to manufacturing of the product. There is a similar environmental impact when the product reaches end of life, but as it is made of many subunits that can be replaced, the overall waste of the product is minimized. The majority of the environmental impact follows that of battery production.

6. Manufacturability

    a. The manufacturing of the silent motored systems will be the most challenging manufacturing on this project. The precision needed for silent gearbox is much greater than just a functional one. The two gearboxes that are being explored for acoustic qualities are the cycloidal gearbox and a belt drive system.

7. Sustainability

    a. The first thing that, with proper use, will decay is the battery system. Depending on use, the battery could last as little as a few months to a couple years. After the battery the mechanical and motors are next likely to fail. The scale between the a mechanical failure and a battery failure is pretty large so the lifespan is mostly determined by the battery and maintenance of the unit.

    b. The battery is also the biggest environmental impact as the means of making the battery are not the cleanest methods naturally. As battery technology advances, the impact of this product on the environment will decrease and become more sustainable.

    c. Most upgrades of the project will be in software. Other that efficiency of operation leading to a longer life span this will have a small impact of sustainability.

    d. The only limitations on software upgrades is analytical power invested and potentially a hardware change.

8. Ethical Considerations

    a. If misused, the Remote Cable Gantry could invade people's privacy as it is designed to be a nonintrusive system achieving alternate perspectives of areas. Also as it is not a standalone product, there is a dependence of safe operation to not damage other equipment like cameras.

9. Health and Safety

    a. The main health and safety concern is upon failure of the Remote Cable Gantry. Since this product is designed to be over things and people, if the system's lines fail it could cause harm to either things below it or the cargo on board. Another consideration is the use of lipo batteries and their safety hazards.

10. Social

    a. The Remote Cable Gantry benefits society through the development of the creative outlet of videography and furthers the technology in that field by applying a different background of electrical engineering to the art of photography.

11. Development

a. Through the use of a couple CAD programs and CNC machining, the design and manufacturing will not utilize any new tools. However, during the analysis of all the systems together, performance will have to be measured by new and old techniques as each subsystem is nothing new, but the combination of them has not been configured this way before.

# Code

This is the C# code ran at the 2017 Spring Senior Project Expo.

```
/*
 *Camera Cable Gantry
 * 5-10-17
 * Allen Bailey
 * Cal Poly San Luis
 */
using System;
using System.Threading;
using Microsoft.SPOT;
using System.Text;
using Math = System.Math;

namespace CameraCableGantry
{
    public class Program
    {
        static CameraCableGantry _Gantry = new CameraCableGantry();
        public static void Main()
        {
            _Gantry.Init();
            while (true)
            {
                _Gantry.Run();
            }
        }
    }

    public class CameraCableGantry
    {
        /*coord. of important points*/
        float[] currentPos = { 0, 0, -10 };
        float[] targetPos = { 0, 0, -10 };
        float[] anchorA = { 0, 57.735f, 0 };
        float[] anchorB = { 50, -28.868f, 0 };
        float[] anchorC = { -50, -28.868f, 0 };
```

```
float[] lineLengths = { 60, 60, 60 };

/** USB gamepad plugged into the HERO */
CTRE.Gamepad _gamepad = new CTRE.Gamepad(CTRE.UsbHostDevice.GetInstance());

/** Make talons with deviceId 1,2,3 */
CTRE.TalonSrx _talon1 = new CTRE.TalonSrx(1);
CTRE.TalonSrx _talon2 = new CTRE.TalonSrx(2);
CTRE.TalonSrx _talon3 = new CTRE.TalonSrx(3);


/*gamepad references*/
const int _A = 2;
const int _B = 3;
const int _X = 1;
const int _Y = 4;
const int _LB = 5;
const int _LT = 7;
const int _RB = 6;
const int _RT = 8;
const int _LJX = 0;
const int _LJY = 1;
const int _LJB = 11;
const int _RJX = 2;
const int _RJY = 5;
const int _RJB = 12;
const int _BACK = 9;
const int _START = 10;

public enum State {Disabled, Enabled, Initialization, Multilateration};
State _State = State.Disabled;

/** hold the current button values from gamepad*/
bool[] _btns = new bool[13];

/** hold the last button values from gamepad, this makes detecting on-press events trivial */
bool[] _btnsLast = new bool[13];

/* hold the current axis values from gamepad*/
```

```
        float[] _axis = new float[6];

        /** some objects used for printing to the console */
        StringBuilder _sb = new StringBuilder();
        int _timeToPrint = 0;

        float _targetPosition = 0;

          uint[] _debLeftY = { 0, 0 }; // _debLeftY[0] is how many times leftY is zero, _debLeftY[1] is
how many times leftY is not zero.

        public void Init()
        {
            InitTalons();
            //Multilateration();
        }

        public void Run()
        {

            /* zero the sensor and throttle */
            ZeroSensorAndThrottle();

            /* loop forever */
            while (true)
            {
                //Set the state of operation
                SetState();


                Loop10Ms();
                UpdateGamepad(ref _btns,ref _axis);
                currentPos = UpdateCurrentPos(currentPos, _axis[_RJX], _axis[_RJX], _axis[_LJY]);
                lineLengths = CalcLineLength(currentPos, anchorA, anchorB, anchorC);
                    //if (_gamepad.GetConnectionStatus() == CTRE.UsbDeviceConnection.Connected) //
check if gamepad is plugged in OR....
                if (_gamepad.GetButton(_LT)) // check if left trigger button is held down.
                {
                    /* then enable motor outputs*/
```

```
                CTRE.Watchdog.Feed();
            }

            /* print signals to Output window */
            Instrument();

            /* 10ms loop */
            Thread.Sleep(10);
        }
    }
    /*Zero the sensor and zero the throttle.*/
    void ZeroSensorAndThrottle()
    {
        _talon1.SetPosition(0); /* start our position at zero, this example uses relative positions */
        _targetPosition = 0;
        /* zero throttle */
        _talon1.SetControlMode(CTRE.TalonSrx.ControlMode.kPercentVbus);
        _talon1.Set(0);
            Thread.Sleep(100); /* wait a bit to make sure the Setposition() above takes effect before
sampling */
    }
    void EnableClosedLoop()
    {
        /* user has let go of the stick, lets closed-loop whereever we happen to be */
        _talon1.SetVoltageRampRate(0); /* V per sec */
        _talon1.SetControlMode(CTRE.TalonSrx.ControlMode.kPosition);
        _talon1.Set(_targetPosition);
    }
    void Loop10Ms()
    {
        /* get all the buttons */
        UpdateGamepad(ref _btns, ref _axis);

        /* get the left y stick, invert so forward is positive */
        float leftY = _gamepad.GetAxis(1);
        Deadband(ref leftY);

        /* debounce the transition from nonzero => zero axis */
        float filteredY = leftY;
```

```
if (filteredY != 0)
{
    /* put in a ramp to prevent the user from flipping their mechanism */
    _talon1.SetVoltageRampRate(12.0f); /* V per sec */
    /* directly control the output */
    _talon1.SetControlMode(CTRE.TalonSrx.ControlMode.kPercentVbus);
    _talon1.Set(filteredY);
}
else if (_talon1.GetControlMode() == CTRE.TalonSrx.ControlMode.kPercentVbus)
{
    _targetPosition = _talon1.GetPosition();


    /* user has let go of the stick, lets closed-loop whereever we happen to be */
    EnableClosedLoop();
}

/* if a button is pressed while stick is let go, servo position */
if (filteredY == 0)
{
    if (_btns[1])
    {
        _targetPosition = _talon1.GetPosition(); /* twenty rotations forward */
        EnableClosedLoop();
    }
    else if (_btns[4])
    {
        _targetPosition = +5.0f; /* twenty rotations forward */
        EnableClosedLoop();
    }
    else if (_btns[2])
    {
        _targetPosition = -5.0f; /* twenty rotations reverese */
        EnableClosedLoop();
    }
}

/* copy btns => btnsLast */
```

```
    System.Array.Copy(_btns, _btnsLast, _btns.Length);
}

/*If value is within 10% of center, clear it.*/
void Deadband(ref float value)
{
    if (value < -0.10)
    {
        /* outside of deadband */
    }
    else if (value > +0.10)
    {
        /* outside of deadband */
    }
    else
    {
        /* within 10% so zero it */
        value = 0;
    }
}

/** throw all the gamepad buttons into an array */
void UpdateGamepad(ref bool[] btns, ref float[] axis)
{
    for (uint i = 1; i < btns.Length; ++i)
        btns[i] = _gamepad.GetButton(i);
    for (uint j = 0; j < axis.Length; ++j)
        axis[j] = _gamepad.GetAxis(j);
}

/** occasionally builds a line and prints to output window */
void Instrument()
{
    if (--_timeToPrint <= 0)
    {
        _timeToPrint = 20;
        _sb.Clear();
        _sb.Append(currentPos[0]);
        _sb.Append(", ");
```

```
        _sb.Append(currentPos[1]);
        _sb.Append(", ");
        _sb.Append(currentPos[2]);
        _sb.Append(" :: ");
        _sb.Append(lineLengths[0]);
        _sb.Append(", ");
        _sb.Append(lineLengths[1]);
        _sb.Append(", ");
        _sb.Append(lineLengths[2]);
        _sb.Append(" :: ");
        _sb.Append(_axis[_RJX]);
        _sb.Append(", ");
        _sb.Append(_axis[_RJY]);
        _sb.Append(", ");
        _sb.Append(_axis[_LJY]);
        Debug.Print(_sb.ToString());
    }
}

/** Update current position*/
float[] UpdateCurrentPos(float[] currentPos, float x, float y, float z)
{
    float [] newpos = { currentPos[0] + x, currentPos[1] + y, currentPos[2] + z };
    return newpos;
}

float ArrayMax(float[] a)
{
    float max = 0;
    for (int i = 0; i < a.Length; i++)
    {
        max = (float) Math.Max(a[i], max);
    }

    return max;
}
float[] ArrayAbs(float[] a)
{
    float[] abs = {0};
```

```
    for (int i = 0; i < a.Length; i++)
    {
        abs[i] = (float) Math.Abs(a[i]);
    }
    return abs;
}
float[] Normalize(float[] a)
{
    float[] abs = ArrayAbs(a);
    float max = ArrayMax(abs);
    float[] Normalized = { 0 };
    for (int i = 0; i < a.Length; i++)
    {
        Normalized[i] = (a[i]/max);
    }
    return Normalized;
}
bool CheckSaturation(float[] a)
{
    bool sat = false;
    for (int i = 0; i < a.Length; i++)
    {
        if(a[i] > 1)
        {
            sat = true;
        }
    }
    return sat;
}

/** Calculate line lengths between anchors and current position*/
float[] CalcLineLength(float[] currentPos, float[] anchorA, float[] anchorB, float[] anchorC)
{

        float[] LineLengths = {(float) Math.Sqrt(Math.Pow(((double) anchorA[0] - (double) currentPos[0]), 2) + Math.Pow(((double) anchorA[1] - (double) currentPos[1]), 2) + Math.Pow(((double) anchorA[2] - (double) currentPos[2]), 2)),
```

```
                        (float) Math.Sqrt(Math.Pow(((double) anchorB[0] - (double) currentPos[0]),
2) + Math.Pow(((double) anchorB[1] - (double) currentPos[1]), 2) + Math.Pow(((double) anchorB[2]
- (double) currentPos[2]), 2)),
                        (float) Math.Sqrt(Math.Pow(((double) anchorC[0] - (double) currentPos[0]),
2) + Math.Pow(((double) anchorC[1] - (double) currentPos[1]), 2) + Math.Pow(((double) anchorC[2]
- (double) currentPos[2]), 2))};
        return LineLengths;
    }
    /** calculate target velocities*/
     float[] VelocityTranspose(float[] sticks, float[] currentPos, float[] anchorA, float[] anchorB, float[]
anchorC)
    {
        lineLengths = CalcLineLength(currentPos, anchorA, anchorB, anchorC);
          float[] target = { ((anchorA[0] - currentPos[0]) * sticks[0] + (anchorA[1] - currentPos[1]) *
sticks[1] + (anchorA[2] - currentPos[2]) * sticks[2]) / lineLengths[0],
                           ((anchorB[0] - currentPos[0]) * sticks[0] + (anchorB[1] - currentPos[1]) *
sticks[1] + (anchorB[2] - currentPos[2]) * sticks[2]) / lineLengths[1],
                           ((anchorC[0] - currentPos[0]) * sticks[0] + (anchorC[1] - currentPos[1]) *
sticks[1] + (anchorC[2] - currentPos[2]) * sticks[2]) / lineLengths[2] };
        if (CheckSaturation(target))
        {
            Normalize(target);
        }
        return target;
    }

    void InitTalons()
    {
        /**set talons to brake mode*/
        _talon1.ConfigNeutralMode(CTRE.TalonSrx.NeutralMode.Brake);
        _talon2.ConfigNeutralMode(CTRE.TalonSrx.NeutralMode.Brake);
        _talon3.ConfigNeutralMode(CTRE.TalonSrx.NeutralMode.Brake);

        /** Select Mag encoder */
        _talon1.SetFeedbackDevice(CTRE.TalonSrx.FeedbackDevice.CtreMagEncoder_Relative);
        _talon2.SetFeedbackDevice(CTRE.TalonSrx.FeedbackDevice.CtreMagEncoder_Relative);
        _talon3.SetFeedbackDevice(CTRE.TalonSrx.FeedbackDevice.CtreMagEncoder_Relative);

        /** flip sensor direction*/
```

```
    _talon1.SetSensorDirection(false);
    _talon2.SetSensorDirection(false);
    _talon3.SetSensorDirection(false);

    /** Set Current Limit*/
    _talon1.SetCurrentLimit(10);
    _talon2.SetCurrentLimit(10);
    _talon3.SetCurrentLimit(10);

    /** Enable Current Limit*/
    _talon1.EnableCurrentLimit(true);
    _talon2.EnableCurrentLimit(true);
    _talon3.EnableCurrentLimit(true);

    /** set closed loop gains in slot0 */
    _talon1.SetPID(0, 0.2f, 0.0f, 0.0f, 0.0f);
    _talon2.SetPID(0, 0.2f, 0.0f, 0.0f, 0.0f);
    _talon3.SetPID(0, 0.2f, 0.0f, 0.0f, 0.0f);

    /** use slot0 for closed-looping */
    _talon1.SelectProfileSlot(0);
    _talon2.SelectProfileSlot(0);
    _talon3.SelectProfileSlot(0);

    /** set the peak and nominal outputs, 12V means full */
    _talon1.ConfigNominalOutputVoltage(+0.0f, -0.0f);
    _talon1.ConfigPeakOutputVoltage(+12.0f, -12.0f);

    /** how much error is allowed?  This defaults to 0. */
    _talon1.SetAllowableClosedLoopErr(0, 0);
    _talon2.SetAllowableClosedLoopErr(0, 0);
    _talon3.SetAllowableClosedLoopErr(0, 0);
}
void StatusTalon(CTRE.TalonSrx talon)
{
    CTRE.TalonSrx _talon = talon;
    _sb.AppendLine("Status Talon");
    _sb.Append(_talon.GetDeviceNumber());
    _sb.Append(": ");
```

```
            _sb.Append(_talon.GetBusVoltage());
            _sb.Append("V, ");
            _sb.Append(_talon.GetOutputCurrent());
            _sb.Append("A, ");
            Debug.Print(_sb.ToString());
        }
        void SetState()
        {
            if ((_gamepad.GetConnectionStatus() == CTRE.UsbDeviceConnection.Connected))
            {
                _State = State.Enabled;
                CTRE.Watchdog.Feed();
            }

        }


    }
}
```