WORCESTER POLYTECHNIC INSTITUTE
ROBOTICS ENGINEERING PROGRAM

Demining Autonomous System

A Major Qualifying Project

Submitted to the faculty of
WORCESTER POLYTECHNIC INSTITUTE
In partial fulfillment of the requirements for the
Degree of Bachelor of Science

Submitted By
Karl Ehlers
Benen ElShakhs
Eleanor Foltan
Jessica McKenna
Joseph Niski
Adam Santos
Matthew Schmitt
Andrew VanOsten

Submitted To
Craig Putnam
Nicholas Bertozzi

Abstract

The use of PMN-1 anti-personnel landmines, along with poor documentation and scale of their deployment, has led to a humanitarian crisis. Current methods of landmine removal, including military-grade equipment, trained animals, and manual deminers, are dangerous, ineffective, and expensive. This project built on the work of previous MQPs to develop a more robust rover for autonomous landmine detection and a more accurate drone for landmine detonation. Additionally, the team developed an easy-to-use application that combines the two systems and provides access to non-technical users.

Acknowledgements

We would like to thank our advisors Professor Nicholas Bertozzi and Professor Craig Putnam for their continued support of this project. They worked tirelessly to ensure we had the resources we needed to make this project happen, both on campus and remotely. We would also like to thank Katherine Crighton for her assistance with the organization of our project and all of our parts ordering and shipment. Finally, we would like to thank Nicholas Lanotte of the previous Demining Rover MQP for his assistance at the start of our project; without him we would not have been able to have as successful of a start to this project.

Table of Contents

List of Figures

Table of Tables

Chapter 1: Introduction

PMN-1 landmines are a type of anti-personnel blast landmine developed in the 1950's by the Soviet Union. These landmines are more dangerous than other landmines due to the fact that they contain 0.53 pounds of TNT, compared to approximately 0.11 pounds of TNT seen in other landmines of similar size. When detonated, these landmines cause significant damage to the legs and feet of people nearby, but intentionally maim these people rather than kill them. Today there are an estimated 110 million landmines that remain buried across the globe, which still pose a threat [1].

Many of the landmines still buried across the globe are no longer in areas of conflict; rather, they are in communities recovering from conflict. Many people living in these post-conflict areas cannot afford to have landmines removed professionally. As a result, community members either have to risk their own lives to remove the landmines by hand, or they have to live with the risk that someone they know could be hurt or killed as a result of landmines. The vast majority of people killed by landmines are civilians living in communities near minefields. Children are at extremely high risk for death or injury from landmines because they have not learned what to look for and how to avoid hidden landmines. These communities need a cost-effective and safe way to remove landmines, so that their community members can live safely near these areas. In this case, the long-term goal of clearing these landmine fields could be to use them as farmlands or as an expansion of their community.

Our project looks to develop a cost-effective robotic solution to the world's landmine crisis. We hope to develop a system of robots that can safely locate and detonate landmines without the need for human intervention. Specifically, our solution would consist of a rover and a drone working in tandem to eliminate the threat of landmines. The rover would navigate through the minefield and use a metal detecting coil to find landmines. Once the rover has marked all the landmines, a drone would be sent into the minefield with payloads that would be dropped onto each of the landmines. Once the drone and rover finish their tasks, they would return to where they started.

Chapter 2: Background

2.1 Humanitarian Landmine Crisis

2.1.1 History of Landmine Use

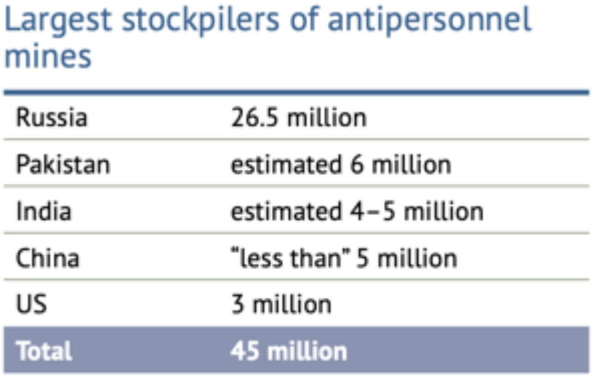
The development of landmines can be traced back to ancient weapons including spike traps and explosive powder trails used in 14th century Italy. However, the first recognized victim of a modern landmine was an American Civil War Union Soldier who stepped on a modified artillery shell close to Yorktown on May 4th, 1862 [2]. The use of landmines rapidly expanded after the Civil War, and their use was prominent during World War II, the Vietnam War, the Korean War, the First Gulf War, and the Cold War [3]. By 1945, it was estimated that 2.5% of US troop casualties and 20.7% of tank losses could be attributed to landmines [2]. Today, landmines remaining from these conflicts, in particular anti-personnel landmines, continue to threaten large civilian populations across the globe.

The application of landmines has also evolved over time. Anti-personnel landmines were initially used to protect larger anti-tank landmines from destruction. They were used defensively and planted around anti-tank landmines or around borders or military camps. However, in more modern conflicts, anti-personnel landmines have been weaponized to target, hinder, and redirect enemy soldiers. Anti-personnel mines are designed to maim, not kill, their victims, a tactic that is based on the reasoning that armies utilize more resources on injured soldiers than on the dead. As the use of anti-personnel landmines became more widespread, armies transitioned from planting well-documented, well laid out minefields to utilizing air drops to randomly deliver mines [3]. The practice of hastily deploying landmines during wartime resulted in the locations of many minefields being lost or left undocumented once the conflict is over. As a result, roughly 20,000 people are killed each year due to landmines, and the vast majority of them are civilians in areas once facing conflict [4]. Today an estimated 110 million landmines remain buried across the globe [1].

2.1.2 Current Humanitarian Crisis

The use of anti-personnel landmines, along with the poor documentation and scale of their deployment, has led to a humanitarian crisis. Landmines remain in the ground, active, until removed, and work indiscriminately against their victims; a landmine does not know the difference between a child or a soldier. In fact, it is estimated that 70-85% of landmine victims are civilians. In many countries where landmines were deployed, anti-personnel mines remain in civilian areas, hindering economic growth, making land unavailable, and preventing the free movement of people and goods [5].

Due to these threats, in 1997 the Convention on the Prohibition of the Use, Stockpiling, Production and Transfer of Anti-Personnel Mines and on Their Destruction (Mine Ban Convention) was adopted. Under the convention 150 countries committed to working towards the regulation and destruction of anti-personnel landmines. Since 1997, 40 million stockpiles of landmines have been destroyed, and an increased emphasis has been placed on support for victims of landmines [6]. Even still, as we can see in Figure 1, there are still millions more landmines sitting in stockpiles around the world. Organizations like the International Campaign to Ban Landmines (ICBL) have been formed to support the mission of the Mine Ban Convention. The ICBL is a coalition of non-government agencies from 100 countries in support of a landmine free world and focuses on providing assistance for victims of anti-personnel landmines. The ICBL claims that global anti-personnel mine clearance can be completed by the year 2025 [6]. However, landmine clearance is not an easy task due to cost, danger to participants, and the ineffectiveness of current methods [7]. The ICBL estimates that in the years 2012 - 2016, 97 square kilometers of mined area has been cleared, resulting in the destruction of only 1,137,082 anti-personnel landmines [8].



Largest stockpilers of antipersonnel mines

Russia	26.5 million
Pakistan	estimated 6 million
India	estimated 4–5 million
China	“less than” 5 million
US	3 million
Total	**45 million**

Figure 1 Antipersonnel Landmine Numbers Around the World (International Campaign to Ban Landmines, 2018)

Though the Mine Ban Treaty has begun a long-term effort towards a mine-free world, the new deployment of landmines threatens forward progress. In 2016, the Landmine and Cluster Munition Monitor, the ICBL’s research and reporting branch, reported the highest number of victims of landmines since 1999. In 2017, 7,239 people were reported to have been victims of landmines - 2,793 of those victims were killed, and 87% were civilians. The proliferation of the landmine crisis can in-part be attributed to the non-participation of many countries in the Mine Ban Treaty. In 2017, the government of Myanmar was found to have deployed new antipersonnel landmines. Myanmar remains one of the states that have yet to sign the Mine Ban Treaty - they are accompanied by countries including the United States, Russia, China, India, and Saudi Arabia. Other non-state armed groups, such as the Islamic State, were also reported to have deployed landmines in eight countries in 2017 - Afghanistan, Columbia, India, Myanmar, Pakistan, Thailand, and Yemen [9].

Landmines have a detrimental socio-economic effect on post-conflict communities where they remain. As indiscriminate weapons, landmines are an inhumane weapon. As of November 2018, over 56 countries have identified a threat of antipersonnel landmines. These countries can be found on four out of seven continents: Africa, Asia, Europe, and South America, and 85% of the total landmine-related casualties each year occur in Egypt, Angola, and Iran [1]. Their existence in communities hinders the free movement of people and goods, prevents the use of land for farming, and prevents access to clean water [5]. Within these countries, many of the areas impacted are poor communities who do not have the funds necessary to purchase the industrial landmine removal equipment on the market today, without the help of international organizations. As a result, many of these communities are unable to develop further due to the threat of landmines in nearby fields; it is the risk of landmines that stops the development of new communities or the inability to farm large areas of land on the outskirts of these communities [5].

Victims of landmines suffer long-term physical and psychological damage. A case-study conducted in Jaffna, Sri Lanka that surveyed 67 victims of landmines found that 68% of the male victims and 93% of the female victims had PTSD related to their incidents. Eighty percent of those surveyed had a related anxiety disorder, and 73% qualified for diagnosis with major depression [10]. In poor countries, the existence of landmines can also threaten the availability of humanitarian aid to communities. Landmines can take the primary caregiver of a family away, proliferating poverty [5]. The continued presence and use of landmines throughout the world threaten the wellbeing of the global community and causes the socio-economic landmine crisis.

2.2 Landmines and Their Removal

2.2.1 Types of Landmines

Landmines are often split into two large categories, anti-personnel mines and anti-tank mines. The difference between the two is that anti-personnel landmines are designed for use on people and anti-tank landmines are designed for use on tanks; this means that though they overarchingly share the same design, anti-tank landmines require more pressure to detonate and have a higher quantity of explosives inside.

The target of this project is the PMN-1 Landmine because it is one of the most widely used landmines and is detectable using metal detectors. These mines have been in use since the 1950s. They are 4.4 inches in diameter, they weigh about 1 pound, use a metal trigger, and contain 0.53 pounds of TNT. PMN-1 mines are triggered by a mass of 12.7 pounds or greater on top of the mine. These mines are particularly deadly because of the higher amount of TNT they contain - 0.53 pounds compared to the 0.11 pounds seen in other landmines of similar size.

2.2.2 How Landmines Work

Landmines are designed to be used in military combat to disable any person or vehicle from passing through a specified area. Generally, landmines are placed in groups called minefields that are designed to cover a large surface area and prevent passage through that area. They are hidden on or just below the surface of the ground and are activated by either pressure, tripwire, remote detonation, or time delays.

PMN-1 landmines are classified as a blast mine, meaning they are buried just beneath the surface of the ground and are triggered by someone stepping on the pressure plate. The main goal of these mines is to destroy objects in close proximity, such as a foot or leg, and break them into small fragments rather than kill someone.

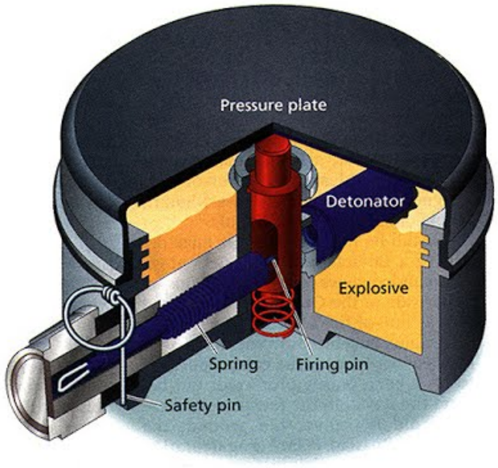


Figure 2 PMN-1 Landmine Schematic (Snyman, 2010)

To arm the landmine, soldiers simply pull out the safety pin on the outside of the landmine, as seen in the schematic in Figure 2 PMN-1 Landmine Schematic (Snyman, 2010), and place it in the desired location. From the time the pin is pulled, the user has 60 seconds until the mine is armed. This time delay is long enough for the soldier to place the mine, however once it is placed it very quickly becomes armed. After it is armed, the only thing preventing the landmine's detonation is the weak spring located within the mine. This spring holds the pressure plate up, but only takes about 12.76 pounds of force applied to the plate to detonate it.

2.2.3 Manual Demining Methods

Manual demining is the practice in which an operator scans the field with a metal detector, investigating signals with a prodding rod, and carefully uncovering and removing landmines [7]. Manual demining was first practiced after World War II, however the first humanitarian efforts around manual demining were carried out in Afghanistan in 1988. These efforts were then expanded to Cambodia, Angola and Mozambique, and today there are 42 global manual demining programs [2]. Though effective, manual demining is dangerous to the operator and slow. It is estimated that only one in 1000 signals investigated by manual demining operators is actually a landmine. False signals are frequently generated due the presence of other metal scrap, such as shrapnel, in the minefield [7]. For these reasons, it is estimated that manual deminers clear land at a rate of only 15-20 square meters per day per person [2].

Though a preferred method of humanitarian demining efforts, manual demining also has many operational flaws. There is no international standard for reporting or recording the success of manual demining efforts and no method of calculating the cost or effectiveness of clearing. Manual demining teams struggle to fill management positions and correctly manage "skill fade," the concern that deminers will become complacent and commit errors in areas that are already perceived to be low risk [2]. However, manual demining does offer economic benefits to poor communities threatened by landmines, as deminers are frequently recruited from the local population [11]. Despite this benefit, and its high effectiveness, manual demining remains a slow, dangerous, and largely unregulated process for the removal of landmines.

2.2.4 Animal Demining Methods

Animal demining methods offer some benefits that manual demining methods do not. Dogs are commonly used to detect explosive residue in minefields and are said to be 10,000 times more sensitive to residue than man-made sensors. In addition to dogs, rats, bees, and bacteria have been utilized for identifying the locations of mines. Animal demining methods are regarded as reliable and fast, however not always effective. For example, dogs can be easily confused by the presence of multiple sources of explosive residue in an environment [7].

Some rats have been trained using a technique called click training. Rats are initially trained with a clicking sound when they find food. Eventually, the rats hear clicks when they find a tea ball infuser with the scent of TNT, positively reinforcing the idea that finding this smell is a landmine. [12] Figure 3 shows a representation of this process.

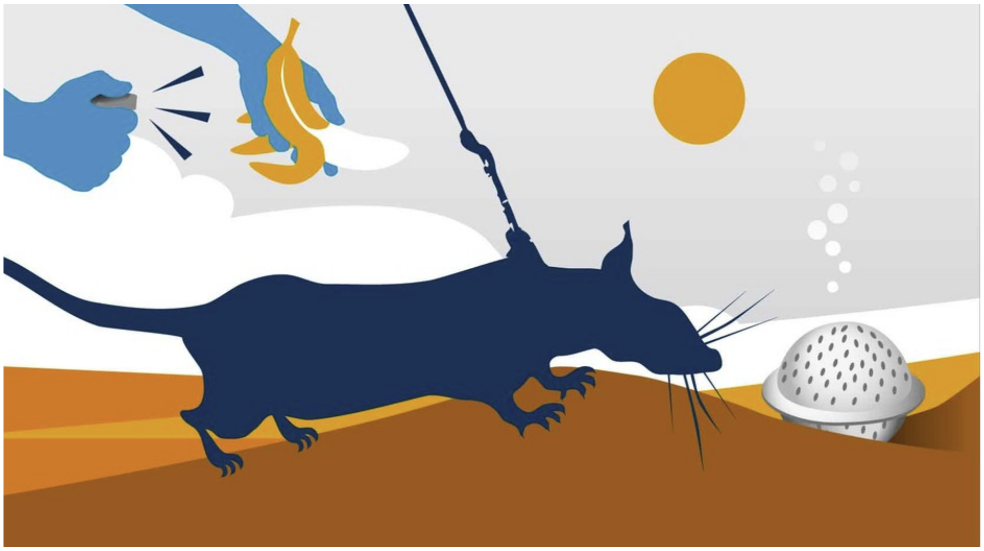


Figure 3 Animal Demining Method Example [12]

2.2.5 Mechanical Methods

The final demining method, mechanical demining, is commonly used by militaries to clear paths through minefields. This process involves heavy machinery that can dig into the fields themselves and push mines out of the way or destroy them. Though effective for this application, mechanical demining often fails to meet humanitarian standards for mine removal. Plows or heavy machinery cannot reach the 99.6% clearance guarantee needed to declare an affected area safe [11]. Commonly these machines are heavily armored and are equipped with either plows or flails. The vehicles are designed for one of two reasons: to purposefully detonate buried landmines, or to push them out of the way and create a clear path through a minefield.

Both instances of implementing this method leave the area heavily damaged. They often do not address the full minefield and can leave live mines buried underground or shift them around. Thus, these tactics do not work well in dense coverage [7]. By only clearing a path, rather than deactivating all the mines in the field, these methods have a low safety rating, and are being phased out for more advanced and safer methods.

2.3 Previous Projects

2.3.1 De-Mining with UAVs

During the 2015-16 academic year, WPI's series of demining projects was begun by a three-member MQP team, with a project titled *De-Mining with UAVs* [13]. This team kept the concept of a full demining system in mind while focusing their efforts entirely on the design, fabrication, and testing of a UAV which could identify and detonate a previously located landmine. The project focused on PMN-1 antipersonnel landmines and operated under the assumptions that targets were at known GPS coordinates and had been marked visually to enable more accurate localization than could otherwise be achieved using GPS.

Planning to detonate mines by dropping the payload (a bag full of sand) from an altitude of 20 ft, it was determined that the drone could likely escape damage by accelerating horizontally immediately after release. Before beginning work on the drone, the team needed to determine the weight of the payload. They created an instrumented test landmine for measurement of impact forces, that would allow them to determine how much force was dispersed by payloads of varying sizes. This device consisted of a 4" diameter PVC tube outer housing with a strain gauge assembly sandwiched between two pieces of plywood and a custom circuit to filter and amplify the signal. Using this, they were able to determine a target payload weight of 0.33 kg that would be dropped from 20ft above the landmine. After experimenting with a missile-like design, it was decided that the system would simply drop small sandbags. A series of trap-door-like mechanisms were designed and built to allow for the release of each payload by firing a solenoid. Shrapnel from the detonation of a real mine could damage a UAV, therefore the characteristics of a PMN-1's "shrapnel cone" were modeled.

After defining a set of constraints, the team selected a DJI Spreading Wings S1000 as their aircraft of choice. This system was retrofitted with a 3DR Pixhawk flight controller to allow for fully autonomous flight via high-level direction from an onboard Raspberry Pi. A downward-facing camera, complete with a carefully selected lens, was attached to the Raspberry Pi. To allow for easy detection using this camera system, the team selected their visual target marker as an 8.5-inch black square outline on a larger white background. An algorithm was designed to recognize and locate this marker within a camera snapshot. In concept, the drone would fly to the provided latitude and longitude, then, using the camera data and a simplistic control system, center itself over the visually identified target before releasing a payload.

In the end, this system worked relatively well; it was able to locate a target, home in on it, and drop a payload when the camera positioning system detected less than 0.5 m of radial error from the target [13]. In addition to proposing the future creation of the other half of the overarching system—a mine detecting rover—the team recommended further improvements to the localization algorithm and payload bay and suggested that a better visual target could be chosen.

2.3.2 Autonomous Landmine Detecting Rover

The original goal of the project was to create a comprehensive, low-cost system capable of semi-autonomously detecting and detonating landmines in a specified area. The goals of the 2018-19 MQP were for the rover to reliably sense anti-personnel landmines up to a depth of 3 inches, implement obstacle avoidance, implement autonomous navigation and landmine marking, and create a user interface (UI) to direct the operation of the rover [14].

The 2018-19 MQP team made developments to the suite of sensors on the rover for detecting landmines and obstacles. They used a Surf Pulse Induction metal detector kit and FLIR Lepton infrared camera to detect mines. The team considered using a ground penetrating radar but rejected it due to the high costs associated with the system. In the end, they chose the MaxBotics EZ1 ultrasonic and Scanse outdoor LIDAR for obstacle detection. Along with these sensors, HC-SR04 ultrasonic sensors were chosen to keep the coil parallel to the ground. For managing these systems, they used an Arduino Mega and Uno to handle signals from the metal detector. Finally, a Raspberry Pi was used for the robot's main controller.

The team also investigated possible improvements to the rover's drivetrain. The platform used for the rover was a Clearpath A100 Husky. The team decided the A100's drivetrain was not adequate for the task. A better drivetrain would be a rocker-bogie suspension capable of driving over six-inch obstacles and a 30-degree slope. They also recommended the tires be replaced with non-pneumatic tires to avoid the risk of sharp objects disabling the rover. However, a chassis change, drivetrain modification, or tire replacement was decided to be outside the budget for the project.

The team decided on using a four-bar mechanism to lift and lower the sensor platform. Previous development of the rover included a four-bar, but due to the modifications to the sensor platform, the original four-bar was no longer adequate to actuate the platform. They used MATLAB and Microsoft Excel to generate and filter appropriate link lengths. Polycarbonate was chosen as the link material for its resistance to bending and deformation. Dimensions for the thickness and width of the links were chosen based on deflection analysis. The team considered five motors to drive the four-bar, and selected the Tetrix TorqueNADO for its torque, size and price. Following torque requirement analysis, a 20:1 worm gear from Tetrix was found to be the most appropriate. An MDD10A motor controller was chosen, with a maximum of 30 volts and 30 amps. However, because the motor would only require an estimated 8 amps under normal conditions, a 10-amp fuse was wired between the battery and the controller to avoid damage to the controller in the case of a stall or short circuit. A Hall effect encoder on the motor shaft was used to track the location of the metal detector. Issues arose driving the links directly from the motors and the transmission system was redesigned to use a winch to actuate the four-bar rather than directly driving the links.

The metal detector needed to sweep back and forth in front of the rover to clear a path wide enough for the rover to travel along. A slider mechanism was designed to be mounted at the end of the four-bar to move the metal detector side to side ahead of the rover. The design for the slider mechanism used a rack and pinion driven by a stepper motor. The rack and pinion were 3D printed and a MATLAB script was used to calculate shear forces to ensure that the teeth would not shear under normal operation. Limit switches were installed at either end of the slider to stop the motor from driving off the end of the rack and provide feedback to the rover when the slider mechanism had reach either end of the rack. Two slider shafts were installed above and below the rack and pinion to hold the coupler in place and control the amount of bending possible.

Robot Operating System was used to run the robot. The network included three computers; the base station laptop, a Raspberry Pi, and the rover's on-board computer. Each system was used to communicate with a different portion of the software; the Raspberry Pi communicated with the sensor platform and controlled the slider and four-bar, while the base station displayed images, stored minefield map data, and commanded the navigation algorithm, and the on board computer handled the communication between these systems.

Google Maps API was chosen for the map definition protocol. An HTML file was written to allow a user to manipulate a map and add coordinates of a polygon to define an area to be searched. An internet connection is required to run this program. A navigation protocol was developed to sweep the polygon defined by the user using a greedy breadth-first search algorithm. The protocol creates a map of 20x20 cm nodes in the polygon to be explored. The robot then follows the algorithm to explore the area, logging obstacles and returns home after all nodes have been explored.

2.4 Drone Fundamentals

2.4.1 Physics of Multirotors

Multirotors, a type of Unmanned Aerial Vehicle (UAV, or "drone"), employ a symmetrical set of upward-facing propellers (rotors) in order to achieve flight. The most common multirotor configuration uses four-rotors arranged in a square or rectangle and is generally called a quadrotor or quadcopter. Larger vehicles that are designed to lift heavier payloads often have six or eight rotors. In the vast majority of drones, propellers are driven by brushless DC motors, which are, in turn, controlled by a flight controller via an electronic speed controller (ESC).

Neglecting external disturbances such as wind, a multirotor can takeoff and control its vertical acceleration, and therefore its altitude, by running all motors at the same speed. Pitch and roll can be controlled by changing the speed of motors on the opposite sides of the drone. For example, a forward pitch can be achieved by increasing the speed of all motors to the rear of the center of gravity (CG) and decreasing the speed of all motors forward of the CG, causing an imbalance of lifting forces, and a lean towards the front. With this pitch achieved, the forces applied by the propellers are directed "up" relative to the body of the drone, which now has both upward and forward components relative to a level orientation, meaning that the aircraft will begin to accelerate forwards. Similarly, this type of differential in motor speeds can be used to move the drone to the back, left, and right.

With these properties in hand, the drone can move in all three spatial dimensions, and rotate about its front-back, and left-right axes, leaving only yaw (rotation about a vertical axis). Yaw can be achieved by an increase or decrease of the speed of propellers that are diagonally opposite each other. For example, the quadcopter pictured in Figure 4 can yaw to the left by increasing the speed of the front-left and back-right motors. These two rotors spin clockwise due to the torque applied to the propeller by the motor. There is, of course, an equal and opposite torque applied to the airframe. Normally this torque is counteracted by the propellers that rotate in the opposite direction, but this property can be manipulated to produce a yaw. In our example, an equivalent decrease in speed can be applied to the front-right and back-left motors, resulting in a yaw to the left, and a net-zero vertical force.

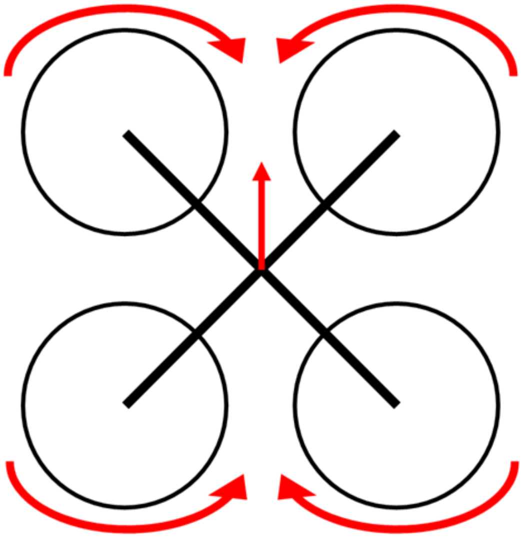


Figure 4 Example Quadcopter, Top View, with Front and Propeller Rotation Directions Marked

2.4.2 Programmable Flight Controllers

While there are many commercial flight controllers available online, only a subset of them allow for users to alter the code present on the devices. This ability greatly enhances the potential implementations of a given flight controller, as customized behavior can be programmed directly into a drone's onboard computer.

There is a thriving open-source community for flight controllers, and part of this community released an open source hardware project known as ArduCopter in 2010. The original hardware model has since been abandoned but the ArduPilot software suite is still freely available online and widely used by the DIY and professional community. Advances in recent years have expanded support for communication protocols aimed at utilizing more powerful external computers through wireless communication and other services such as ROS.

A number of flight controllers are commercially available that support ArduPilot and PX4. One of the current and most popular open-source controllers is the Pixhawk, originally developed in part by the designers of ArduCopter and PX4. The Pixhawk comes with 32-bit processors, an array of sensors and ports for programming and control, such as GPS, wireless telemetry and interfaces for a Raspberry Pi to enable additional capability.

2.4.3 Drone Legality in Worcester

As drone flying (both commercially and recreationally) gets more common, it necessitates further rules and regulations to ensure the safety of everyone involved. The Federal Aviation Association (FAA), has developed a series of rules for model aircraft and drones that residents must abide by in Massachusetts [15].

Drone flying can fall into one of three categories: hobbyist, commercial pilot, or government employee. Because this project is purely academic, it falls under the hobbyist flight laws until it turns into a specific business venture [16].

As a hobbyist, there are a few rules that must be followed to comply with federal regulations. The rules are as follows:

1. Register your drone or UAS with the FAA
2. Fly for hobby or recreational purposes only
3. Follow a community-based set of safety guidelines
4. Fly your drone within visual line-of-sight
5. Give way to manned aircraft
6. Provide prior notification to the airport and air traffic control tower, if one is present, when flying within 5 miles of an airport
7. Fly a drone or UAS that weigh no more than 55 lbs. unless certified by a community-based organization [16]

If this venture should become something of commercial significance, and the designation of flight switches from hobbyist to commercial pilot, then a similar set of rules exists with the addition of a few things. This change adds on a competency test that the pilot of the drone must take and pass in order to fly the drone. It also adds a separate document of rules that the pilot must be familiar with called the “Part 107 Guidelines.” [16]

2.5 Differential GPS

Differential GPS (dGPS) is a system that utilizes a known-stationary “base station” GPS module to calculate correction values which are then used to increase the accuracy of moving GPS modules. Generally, a wireless link is used to transmit the correction values in real-time. When implemented properly, a dGPS system can provide absolute position estimates with an accuracy of a few centimeters. In past iterations of this project, the Rover utilized dGPS to navigate through its map. One of the observations the previous team made was that when powering up the Rover and base station, both dGPS units took a significant amount of time to connect, approximately 30 minutes. [14] This may have been caused by a limited sky-view of the GPS antennas. With another project that used an A100 Husky, they experienced errors up to 1.6

meters off a base test. [17] With proper configuration and usage, these modules can provide much better results; this is a goal that will be strived for in this project.

Chapter 3: Methodology

The following sections detail the methodology taken by the 2019-2020 Demining Autonomous MQP team, with intermediate results. The methodology is arranged to cover upgrades to the Mine Detection system, Mine Detonation System, and the overall System Integration.

3.1 Mine Detection

3.1.1 Project Scope

This iteration of the Demining Autonomous System Project was, from the rover perspective, mainly focused on updating the mechanical rover system to be more robust. While this project also ran in the previous (2018–2019) academic year, many of the mechanical components were 3D printed, causing much of the mechanical structure to be weak with low tolerances and stiffness. This year, the main goal was to fortify the mechanical system and reduce the complexity of the overall system. While our focus was not the software or electrical components, these needed to be updated as well to fit the new mechanical system design.

The principle subsystems of the previous rover design included a four-bar mechanism, a sliding mechanism, and a sensor head. The sensor head incorporated a marking mechanism, pivot mechanism, and metal detector with roll, pitch, yaw compliance. For this MQP the team considered several designs when planning the redesign of the system, including rigidly mounting the metal detector to the chassis, a vertical linear sliding mechanism, and replacing the sliding mechanism with a bar-type metal detector, The team decided to keep the basic overall geometry of the previous four-bar design and slider mechanism, as it was deemed to fulfill the project requirements. This let the team focus on improving the design versus spending the time to create a new mechanism from scratch. Further, the team decided to reduce and remove complexity in the sensor head for a more streamlined system.

In terms of the parameters for use, the project will not consider minefields that closely border cliffs, walls, or other objects that would not allow the rover to slightly overstep the minefield's perimeter in order to turn around. The rover is not equipped to deal with large cliffs within the work area, or steep slopes over a 30% grade, due to the drivetrain's gearbox constraints.

3.1.2 Initial Design Considerations

3.1.2.1 Clearpath Husky A100

The Husky A100 is an off-the-shelf, 110lb robotic drive base intended to be used in projects that require a basic drivetrain capable of navigating outdoor terrain. It contains two CIM motors coupled to gearboxes that then distribute power via toothed belts to the six wheels on the base. The drivetrain has built in encoders, giving the robot the ability to perform simple dead reckoning maneuvers. The Husky takes velocity or position commands through a serial connection to a Linux PC. Although it is designed to traverse outdoor terrain, the Husky has only a couple inches of ground clearance and with only about 160 watts of power, it has trouble traversing steep slopes. It is also relatively slow moving, with a max speed of 1 meter per second. To attach accessories, the top plate of the robot is ⅜” thick aluminum with a grid of tapped holes, allowing changes without modifying the robot. This drive train was initially chosen by the 2018-2019 MQP team because it is a versatile base that can be made to support nearly any design they could come up with. Its default compatibility with ROS allowed them to utilize many software libraries to eliminate complicated work they would have to do otherwise—odometry, communication with a base station, etc.

During initial testing, the team found that although the robot was advertised to be designed for outdoor, off-road activities, it struggled to do anything but drive straight on flat ground. The six pneumatic tires had too much grip on dirt and grass, and when turning, created so much scrub that the robot couldn’t turn in place reliably. Very wide turns could be made. The team also attempted to drive the robot up some slopes, however if they were more than about 15 degrees, the robot would stall. The team discussed many options to address this, such as further gearing down of the rover, but the consensus was that the team would need custom gearboxes to accept more, or different, motors to increase the overall power of the drivetrain. One option was to add more CIM motors which are cheap but heavy, and the other was to utilize brushless motors from hobby R/C applications, which are much lighter and more efficient.

One of the features of the rover that made it difficult to utilize with the team’s design is the fact that the robot’s wheels are all on the same plane. Normally, when designing a wheeled drivetrain, wheel scrub is considered, and some attempt is made to minimize it. Cars do not have wheel scrub, as they have differentials and their front wheels can change their heading to steer itself in a different direction. The Husky turns by driving one side faster than the other. The wheels slide sideways, as their heading does not match the arc they should be traveling in, and friction saps away much of the overall drivetrain power. One way to combat this is to lower the center wheels, allowing the robot to tip forwards or backwards, only contacting the ground with four wheels at a time. The effective wheelbase shortens greatly and reduces scrub. As the rover had no provisions to this effect, the team attempted to mimic a center wheel drop by pressurizing the tires on the center wheels to their limit and depressurizing the outer wheels.

3.1.2.2 Metal Detection

Metal detectors work by using a coil of wire to induce a magnetic field, which is typically known as a “Very Low Frequency” (VLF) detector. There are two coils at the end of a metal detector. Alternating current is run through one coil called the transmitter coil, creating a changing magnetic field. When metal is present near the coil, a current is induced in the metal. The metal in turn creates its own magnetic field, which is detected by the receiver coil. When an abnormality in the magnetic field influences the current through the receiver coil, a circuit detects it and beeps. Smaller coils can be used to optimize detection of smaller pieces of metal, at the cost of ground coverage. It can also result in lower signal strength from smaller radii of coils [18]. An alternative to the very low frequency metal detector above is the “Pulse Induction” (PI) detector which, rather than sending constant AC through the coil, sends one short pulse and listens for resulting eddy currents. The advantage of PI detectors is that they are influenced much less by mineralization, which can significantly impair functioning of VLF detectors. Other methods of mine detection include Ground-Penetrating Radar (GPR). However, GPRs are comparatively much more expensive, and the current commonly available systems involve exerting a force on the ground with the sensing head, which the team viewed as a risk of detonating the mines when trying to detect them.

After evaluating all of the available options for mine detection described above, the team chose to use a metal detecting loop because of the cost and the available knowledge from work and tests conducted by previous MQP teams. The metal detection device on the front of the robot is a handmade coil, connected to a commercial metal detector circuit board that processes the signal from the coil. The effective range of the metal detector varies based on the location of metal within the metal detecting ring, and it also varies according to distance from the ring, both of which are exemplified in Figure 5. In order to generate the figure, the team tested the metal detector based on measurements along the radius of the ring, as well as at differing heights, starting with contacting a piece of metal to the plastic case of the ring itself, then moving it slowly outwards in each direction, noting when it stopped being detectable. The maximum reasonable distance for detection before the coil begins to provide false readings is roughly 6.5 inches away from the sensor head. This varies based on the mass of metal to be detected, but given the current sensitivity of the metal detector, the metal detecting system should be able to find a standard issue PNM-1 landmine at its average depth of 15 cm (~5.9 inches). The team tested the metal detector using a small steel washer, with a mass of around 5 grams, which is less than the mass of metal found within a PMN-1 landmine. Since said landmines are only around 2 inches in height, and the metal portion of the mine sits near the center, the metal detecting ring should be well-equipped to locate the increased amount of metal at such depths. In fact, as seen in Figure 5, this metal detector can detect less than the amount of metal in a PMN-1 landmine in a 0.5-inch cone at a distance of 6 inches away. [19] In this figure you can see several disks generated based on their specific distance from the metal detector. Inside each of these disks are the exact locations that the metal will trigger the detector. For example, at a height of 3 inches, there is a detecting range that sits within a disk that is 1 inch wide. The outer radius of this disk sits 1 inch from the inside of the metal detector ring. This range is true both above and below the metal detector ring, so the range has been demonstrated above the ring for easier visualization.

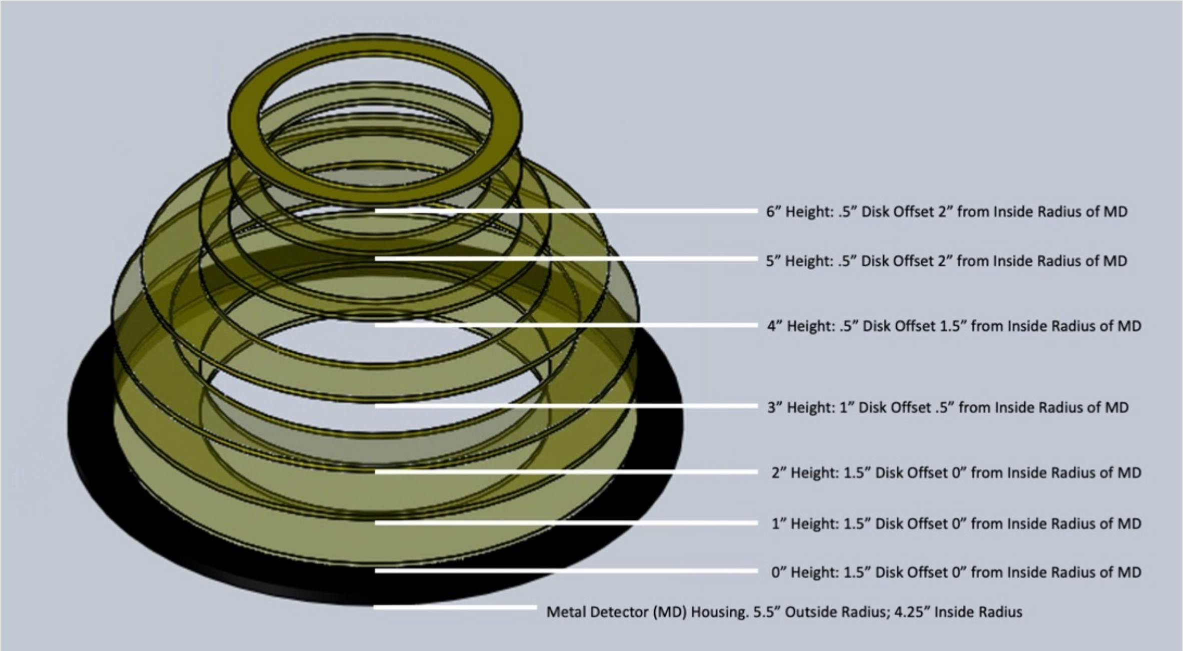

6" Height: .5" Disk Offset 2" from Inside Radius of MD
5" Height: .5" Disk Offset 2" from Inside Radius of MD
4" Height: .5" Disk Offset 1.5" from Inside Radius of MD
3" Height: 1" Disk Offset .5" from Inside Radius of MD
2" Height: 1.5" Disk Offset 0" from Inside Radius of MD
1" Height: 1.5" Disk Offset 0" from Inside Radius of MD
0" Height: 1.5" Disk Offset 0" from Inside Radius of MD
Metal Detector (MD) Housing. 5.5" Outside Radius; 4.25" Inside Radius

Figure 5 Metal Detection at Differing Heights

In order to properly implement the metal detection system however, the rover needs a way to keep it level to the ground no matter where it is on any given hill, which is why the team has redesigned the mechanical alignment system as will be discussed in the following section. Since there is an exact architecture of the detectable zone coming from the metal detector, if the ring is not perfectly level there has to be further calculations and trigonometry to pinpoint the mine location. However, this also means that the metal detection ring may have to be reoriented once a mine is found, independent of the search algorithm. As can be seen, the center of the coil is ineffective, producing a circular dead zone of varying radii. For this reason, it is necessary to move the ring over the detected mine in both the forward and lateral directions in order to find the outer bounds of the mine and pinpoint its location for the drone to detonate it. Although the metal detector is calibrated, its output is binary; there is no way to quantify the point where metal is most strongly detected. To deal with this, the sensor head is swept across the suspected mine, recording the output of the metal detector at certain intervals. Repeating this process multiple times, driving forward slightly between each, a 2-dimensional point cloud is produced. The centroid of this group of points approximates the center of the mine. Transferring this information back to the main computer, an estimate of the suspected mine's location can be calculated based on the position and heading of the rover, and the configuration of the four-bar and slider.

3.1.3 Visual Marking Methods

3.1.3.1 Marking

While the team was undetermined on the adequacy of the differential GPS modules as a method of location for the drone, efforts were undertaken to ensure a viable marking system was incorporated onto the rover. Using this system, the rover would physically mark the location of each mine it detected, similar to systems in previous MQPs. The previous team's design had the visual marking system and the metal detection system close to the ground on the same arm, but since the visual marking system used a metal spray paint can that would interfere with the metal detector's signal, they designed a rotating sensor head so that both systems could be used close to the ground without interfering with one another. However, the team this year hoped to reduce the complexity of the marking system by removing the necessity for rotating the sensor head. In order to do this, the Team determined the most viable option would be to relocate the paint to the chassis of the rover and utilize a tubing and spray system to deliver the paint to the appropriate location. This solution also offered the added benefit of further removing metal components from the range of the metal detector coil.

3.1.3.2 Verification

In order to first investigate the feasibility of relocating the marking system, the team investigated several consumer paint-marking systems. Most common systems—for example Wagner's FLEXiO 4000 [20]—while readily available, would place either electrical systems or metal too close to the sensor head, resulting in interference with the metal detection system. The team also found a simple spray-can extension system, which replaces the head of the spray paint can with a length of tube and nozzle [21]. In the pursuit of a cost-effective and simple solution, in which the metal nozzle could easily be replaced with plastic parts, the team decided to evaluate this simple system first.

A simple test was conducted with a can of spray-paint to determine whether or not the system would yield enough pressure to spray with no clogging and would produce an adequately visible mark. After promising results from this test, the team decided to move forward with this system. However, after dGPS testing proved that an accurate enough hover could be achieved by the drone to hit the landmine with the payload, it was determined that the camera vision system was no longer needed. This also eliminated the need for the marking system on the rover, which was removed to further simplify the design.

3.1.4 Mechanical Design

A main focus of this year's project was to improve upon the mechanical design of the 2018–2019 MQP. The team chose to maintain the use of a four-bar mechanism to raise and lower a horizontal linear slider unit, in order to generate motion along two axes for the metal detector. The rover's third degree of freedom comes from the drivetrain. With this three-axis motion, the rover can move the coil across a minefield of varying elevations—the z axis travel allows the robot to raise and lower the coil to create acceptable approach and departure angles to clear slopes. Figure 6 shows the completed Solidworks CAD design for the Rover's mechanical design.

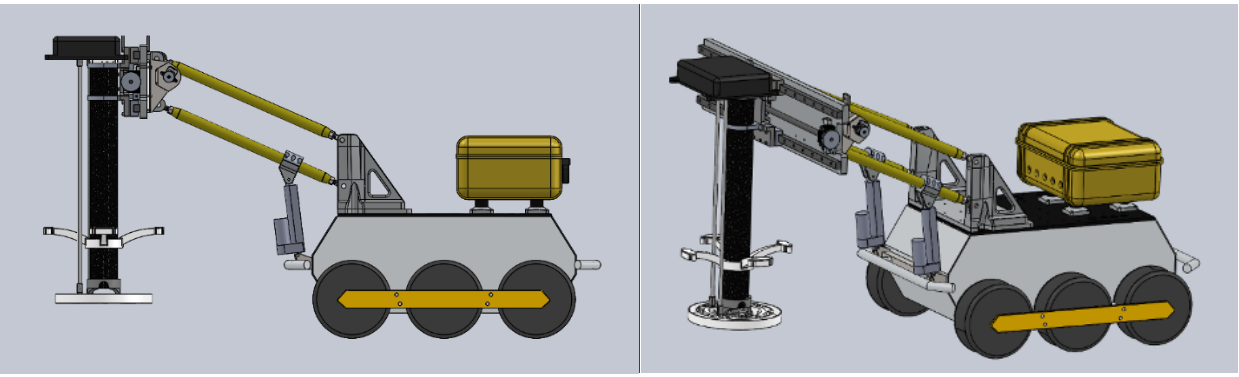


Figure 6 Full Rover CAD

3.1.4.1 Four-Bar Redesign

The main issues with the previous design of the four-bar mechanism were mainly those of poor manufacturing: for example, the construction was governed by material availability, which came at the compromise of suboptimal mechanical design. Most of the four-bar structure was made with donated materials, such as polycarbonate bars and 8020 extrusion.

Using plastic bars in a parallel link configuration meant that there was no provision for handling side load, as plastic has a very low modulus of elasticity. The previous team made some attempts to mitigate this, by connecting the links together with more beams, but this failed to completely solve the side-load issue.

Powering the four-bar up and down was another challenge the team took a different approach to solve. The previous year's team attempted to direct drive the links with gearmotors, but soon switched to a winch that pulled a cable to lift from the middle of the four-bar. The problem with that mechanism stemmed from the left and right halves of the four-bar only being coupled through plastic bars. The winch would not always lift evenly, causing the mechanism to sag to the left or to the right based on the position of the slider mechanism.

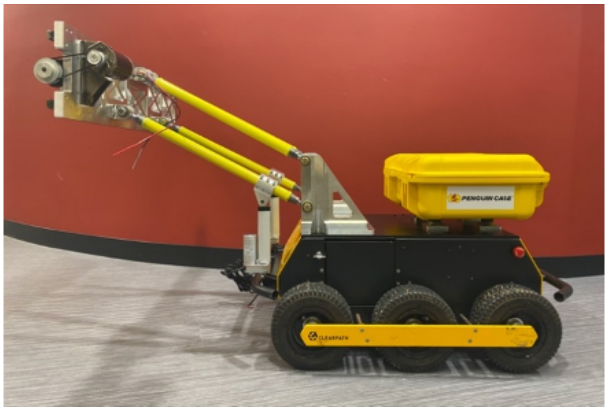
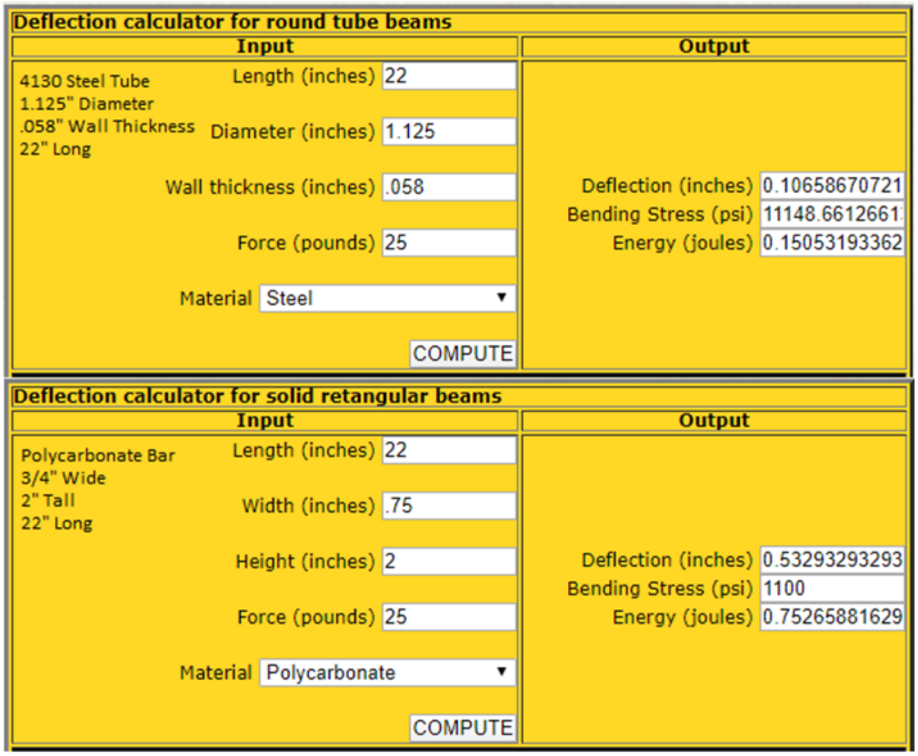


Figure 7 Assembled Four-Bar Mechanism

The new four-bar mechanism improves upon the previous model by switching to steel tube links with spherical ball joint ends. This design is shown in Figure 7. The ball joints have a better weight capacity and coefficient of friction than the previous plastic-on-metal bearing interface. The link lengths can also be adjusted to compensate for manufacturing tolerance and to purposefully tweak geometry. 4130 steel tube, albeit extremely thin-walled (.058") outperforms the deflection of the polycarbonate by nearly a factor of five. The full CAD design for the rover can be seen in Figure 6, and the assembled four-bar design can be seen above in Figure 7. As shown in Figure 8, the team used an online beam deflection calculator early on in our design process to decide what material and shape the team wanted to use for our links to minimize deflection. The previous plastic links flexed under the weight of the sensor arm and slider assemblies and would wobble left and right as the robot moved. The team decided to use steel tubing for the links, as it was very quick and easy to cut them to length and weld on attachment points for the Heim joints. Based on the calculator's output, the team settled on this size of tube based on the factor of safety over the original link - the plastic link deflects .532" vs the steel link at .106, a 5x reduction in deflection. The calculator handles each beam as a cantilever, and applies load at the far end, resulting in deflection. It is not an accurate design tool, as our 4-bar links are not loaded in this fashion but allowed us to pick a more suitable material very early on to mold our design around.



Deflection calculator for round tube beams

Input

4130 Steel Tube
1.125" Diameter
.058" Wall Thickness
22" Long

Length (inches) 22

Diameter (inches) 1.125

Wall thickness (inches) .058

Force (pounds) 25

Material Steel ▾

COMPUTE

Output

Deflection (inches) 0.10658670721

Bending Stress (psi) 11148.6612661

Energy (joules) 0.15053193362

Deflection calculator for solid retangular beams

Input

Polycarbonate Bar
3/4" Wide
2" Tall
22" Long

Length (inches) 22

Width (inches) .75

Height (inches) 2

Force (pounds) 25

Material Polycarbonate ▾

COMPUTE

Output

Deflection (inches) 0.53293293293

Bending Stress (psi) 1100

Energy (joules) 0.75265881629

Figure 8: Comparison of new steel vs. previous polycarbonate four-bar link deflection with 25lb load

Another improvement in the design of the four-bar comes from the triangulated top links, shown in Figure 9. With the ball joints, the team needed to design a fully kinematically constrained system. The previous team was able to get away with running shafts through their links to use as pivots, however our spherical joints do not resist any such side-load in this fashion. Utilizing the range of motion provided by these new joints, the team was able to skew the top links toward each other and still get the out-of-plane motion required for the four-bar. Because the top two links and their mounts now form a triangle, any side load is now converted into compressive and tensile forces, which beams are exceptionally good at dealing with. To keep actuation simple, the bottom links

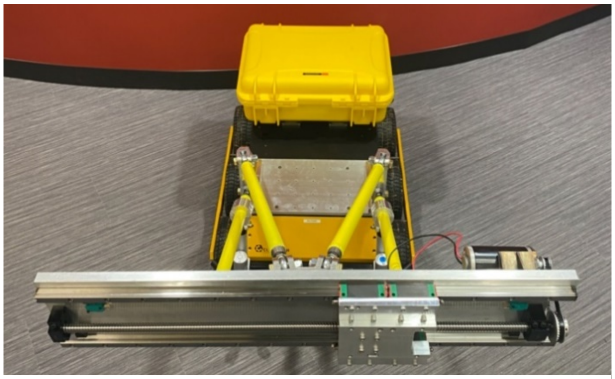


Figure 9 Triangulated Top Links

were left parallel. This results in the four-bar being under constrained, and the tool plate can twist. However, the addition of our linear actuators ensures that the two bottom links stay in the same plane, and the system becomes completely constrained.

The next major focus was actuation. The previous winch system had an unintended side effect of allowing the four-bar to flop up as there was no downward force besides gravity to match the winch's pull. The team settled on using linear actuators with an internal leadscrew and nut system. As a result, the only up-and-down slop of the four bar comes from the backlash in these leadscrews and nuts.

Calculations were performed to assist the team in choosing an appropriate linear actuator to lift the four-bar mechanism. The FBDs of the System were drawn (shown in

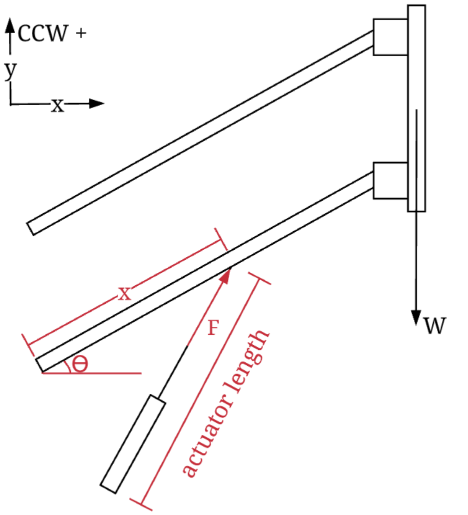


Figure 10 Block Diagram of Linear Actuator Mounting

Appendix C: Four Bar Analysis), and a MATLAB script was produced that calculated the force that was necessary on the linear actuator 'F' vs it's mounting distance 'x' along the bottom link, shown in Figure 10.

The resulting graphs are shown in Figure 11, showing the force that must be applied to lift the sliding mechanism and sensor head. The calculation was completed for four starting angles of the four-bar mechanism – 'θ' in Figure 10.

Figure 11 shows that for a mounting location of 5 inches from the bottom joint, the force required to lift the four-bar is under 200 pounds for all analyzed angles. The mounting distance 'x' can be correlated to the actuator length necessary to achieve that mounting location. For a range of 22.5 – 60 degrees as evaluated Figure 11, an actuator mounted at 5 inches from the base pivot is required to vary its length between 9.38 – 12 inches. It will experience forces between 128.5 –168.9 lbs as it moves through the 22.5 - 60-degree range. This was used to examine available linear actuators, and ensure that for each actuator's length, the exerted force was sufficient to lift the four-bar mechanism: over 200 pounds. In the end, the team chose a 4" stroke linear actuator that changes between 8-12 inches in length, able to lift 330 pounds [22].

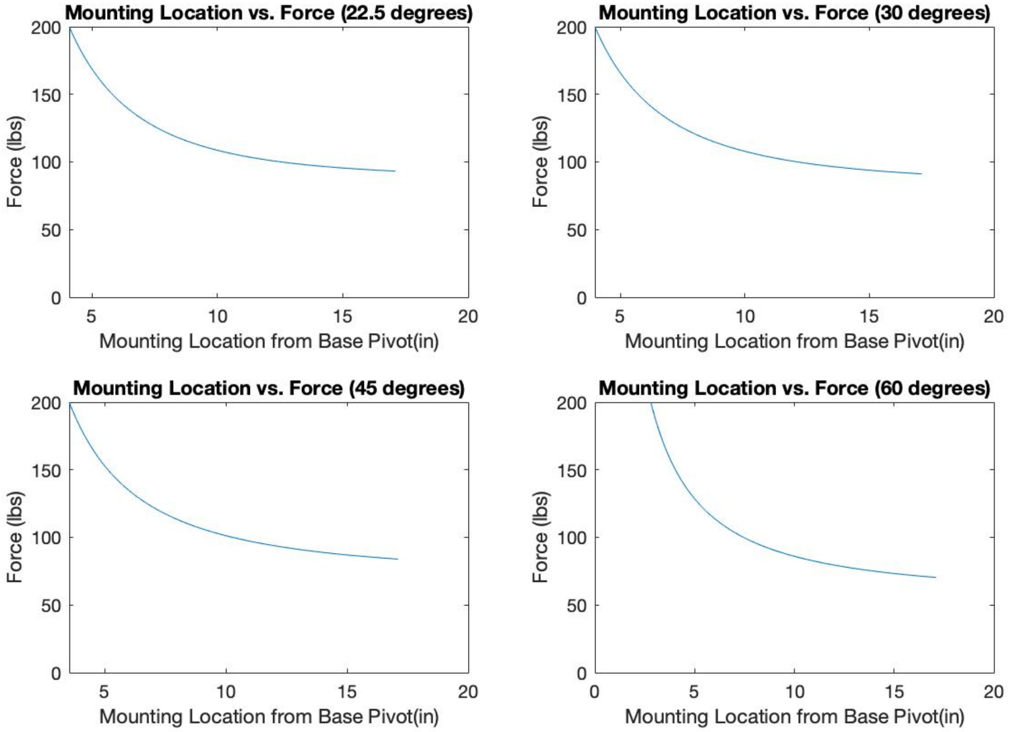


Figure 11 Linear Actuator Mounting Location v. Force

3.1.4.2 Slider Redesign

In order to clear the rover's path of mines ahead of the wheels, the metal-detecting coil needs to move back and forth in front of the rover to sweep the area for mines. The previous project achieved the sweeping motion of the coil with two end-mounted linear rails to guide the motion, and a 3D printed rack and pinion gear to actuate the entire sensor head. There were several issues with this design that contributed to excessive flexibility in the sensor head. One issue was bending in the end-mounted linear rails. Being over 24 inches long and less than an inch in diameter, they bent significantly under minor loading. The other main issue was tolerances in the 3D printed components. Small gaps and tolerance issues intrinsic to 3D printed parts of the quality used in the slider and all the way down to the sensor head stacked up, resulting in shakiness of the sensor head.

In order to avoid these two main issues in the redesign of the slider mechanism, the team machined all load bearing parts from aluminum and used surface-mounted linear rails. Figure 12 shows the new sliding mechanism design and end result. By machining aluminum, the team was able to more carefully control tolerances, and more accurately calculate deflections of individual parts under expected loading. The team used surface-mounted linear rails instead of end-mounted rails to almost entirely eliminate bending in the rails themselves. The rails were mounted with M5 screws at 60mm intervals. A comparison of the stiffnesses can be seen below.

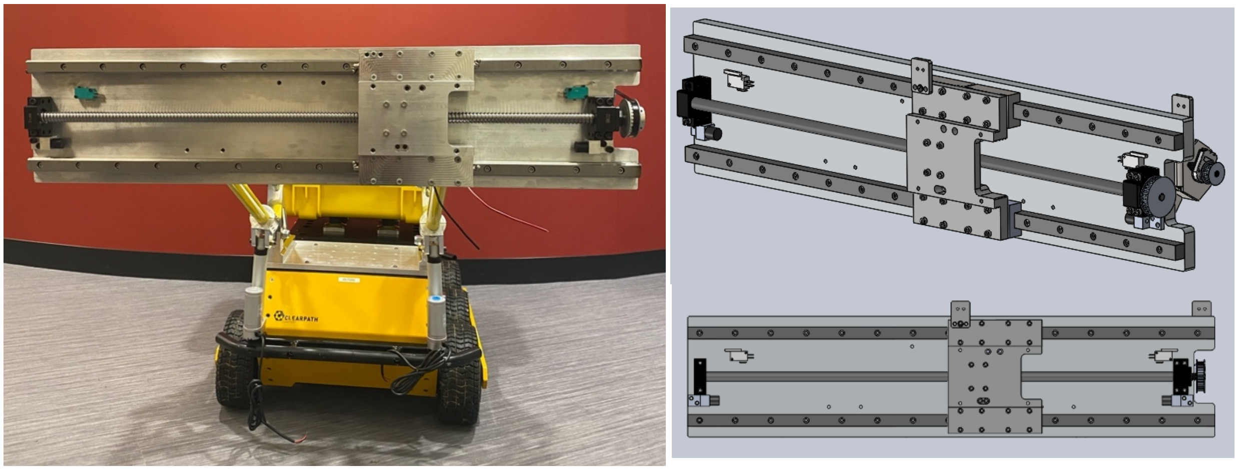


Figure 12 Rover Sliding Mechanism

For actuation, the team decided on a ball screw and DC motor with an encoder to achieve smooth, controllable motion from end to end. The final assembly is shown in Figure 12. Calculations, shown in Figure 13 to compare the stiffness coefficient of the previous MQP's linear rails to this year's rails show an improvement in the new design by about five orders of magnitude, essentially eliminating bending in the linear rails which in the previous MQP had been a significant source of deflection in the sensor head . A simple analysis shows that if a 5lb force is applied horizontally to the sensor head, deflection due solely to bending in the linear rails will be 0.25 inches with the old rails, and 1.06*10^{-5} inches with the new ones.

$$k = \frac{192EI}{L^3}$$

$$I_C = \frac{\pi d^4}{64} \qquad I_R = \frac{bh^3}{12}$$

Carbon Steel $E = 29.3 * 10^6\, psi\ @\ 70^o$ [23]

Previous MQP Linear Rails 2019 MQP Linear Rails

$$d = 12mm = 0.47in$$
$$L = 30in$$
$$I_C = \frac{\pi(0.47in)^4}{64} = \frac{0.1533}{64} in^4$$
$$I_C = 0.0024\ in^4$$
$$k = \frac{192\left(29.3 * 10^6 \frac{lbf}{in^2}\right) 0.0024\ in^4}{(30\ in)^3}$$
$$k = 500.05 \frac{lbf}{in}$$

$$b = 17.5mm = 0.689in$$
$$h = 20mm = 0.787in$$
$$L = 60mm = 2.36in$$
$$I_R = \frac{0.689in(0.787in)^3}{12} = 0.0280\ in^4$$
$$k = \frac{192\left(29.3 * 10^6 \frac{lbf}{in^2}\right) 0.0280\ in^4}{(2.36\ in)^3}$$
$$k = 1.20 * 10^7 \frac{lbf}{in}$$

Figure 13 Calculation of Stiffness Comparison

The linear screw system is driven on one end by a DC CIM motor connected with a belt and pulley system. The team began motor and transmission selection by deciding on a target linear speed of 4in/s, and maximum pushing force of 20lb to push through reasonably small obstacles without doing damage to the rover or risking injury to persons. With a 5mm lead screw, 4 in/s corresponds to 1220 RPM on the lead screw. With this in mind, the team needed a powerful motor capable of reasonably fast speeds. The Vex CIM motor with a no-load speed of 5310 RPM, stall torque of 343 oz*in, and price of $32.99, was selected.

A timing belt and pulley system made the most sense geometrically for a transmission. To select the transmission ratio, analysis was conducted on the necessary torque and speed of the lead screw. The full analysis for the motor selection, transmission ratio, and center-to-center distance (CCD) of the transmission pulleys can be found in Appendix E: Slider Motor and Transmission. The chosen transmission ratio was 2:1. This was the ideal transmission to reach the targets of 4in/s and a 20lb push with the Vex CIM motor. With the high-resolution encoder chosen, the team can very accurately calculate the position of the sensor head anywhere along its range of motion. This is important for accurately locating the landmines in the path of the robot.

One consideration necessitated by the new sliding mechanism was counterweight. The new sliding mechanism is more rigid and robust but also weighs significantly more than the old one, and put the new center for mass slightly forward of the front axle. The solution recommended but not yet implemented by the team is to add 80 pounds of counterweight fixed to the back of the rover to move the center of mass back to a reasonable position. Calculations supporting this decision can be seen in Appendix F.

3.1.4.3 Coil Mounting, Roll, Pitch

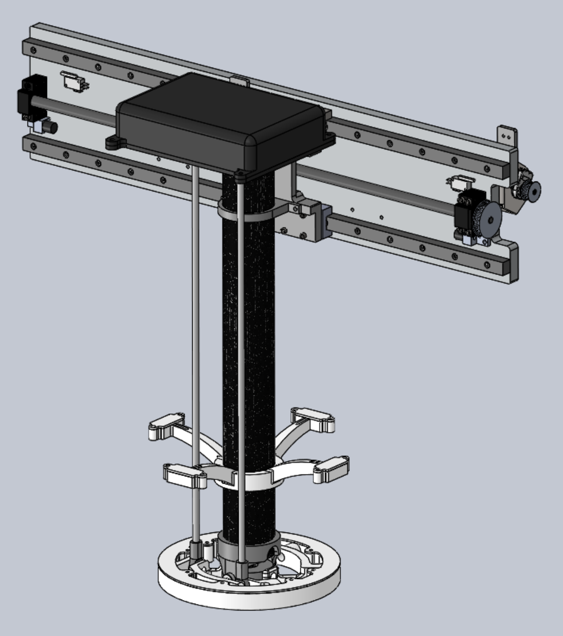


Figure 14 Sensor Arm Mounted to Sliding Mechanism

3D printed components made up the majority of the sensor arm from the previous project. This enabled a very complex design in which the magnetic coil and a spray paint can could be swapped to spray a target directly on the ground where a mine was detected. The coil also was able to pitch to follow the terrain. Unfortunately, due to this complexity, there were many issues, similar to those found with the four-bar. The team decided to simplify this component of the robot to the bare minimum to limit issues.

The sensor tube is an off the shelf carbon fiber tube, which provides excellent stiffness for its weight. It is clamped to the tool plate on the slider system. The coil is mounted at the bottom of the tube, on a custom wrist joint that provides two more degrees of freedom - roll and pitch. Two R/C servos are installed at the top of the tube and use plastic tie rods and spherical joints to move the coil. This all-plastic-and-composites construction will not interfere with metal detection.

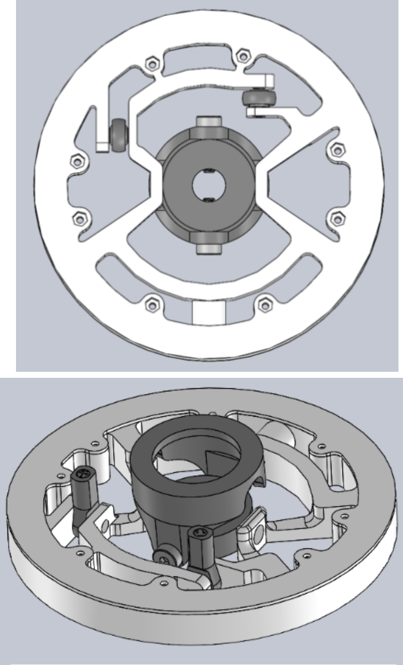


Figure 15 Sensor Coil Version 2

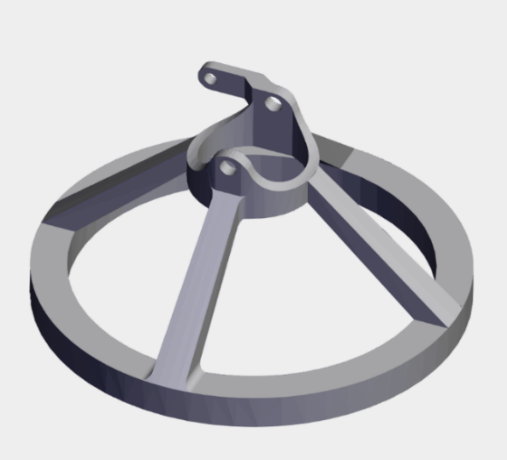


Figure 16 Sensor Coil Version 1

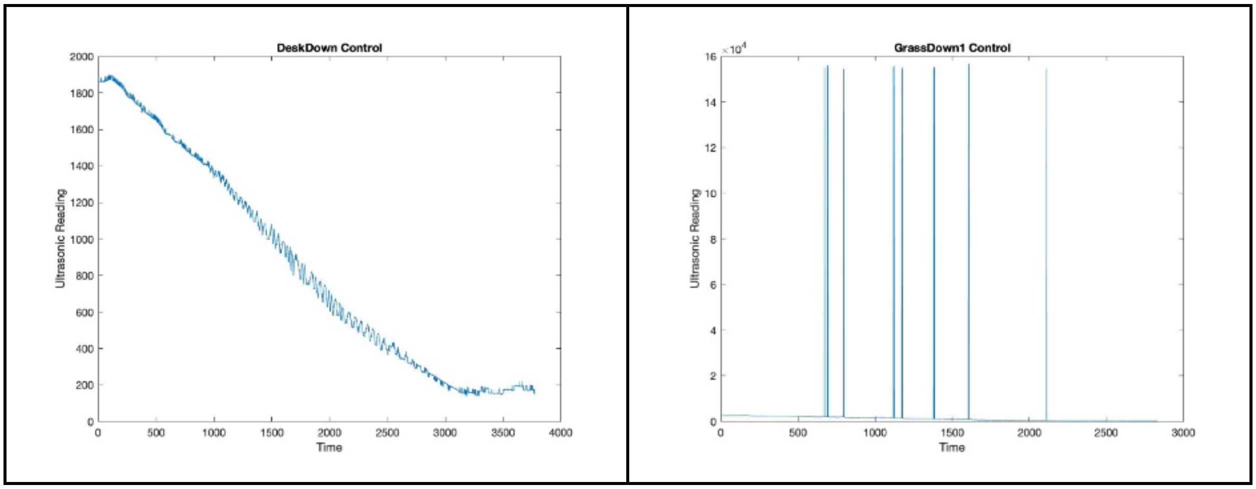
The first iteration of the Sensor Coil is shown in Figure 16. This iteration focused on defining the hinging mechanism, and how the roll and pitch articulation would work. The second iteration of the sensor coil, shown in Figure 15, resized the mechanism, and moved the pivot point down into the center of the coil itself. This allows for simpler control of the roll and pitch of the mechanism. With this design the team was able to achieve around 30 degrees of articulation in each cardinal direction. The mounting points for the nylon tie rods were each moved to attach directly to the sensor head. An electronics box was designed to mount to the top of the carbon fiber tube, to house the servo motors responsible for the tie-rod actuation. The box was also designed to house various other electronics that needed to move with the sensor head, including the metal-detector circuit board, and Arduino Uno responsible for processing the metal-detector signal. In addition, a clamping mount for the ultrasonic sensors was designed to keep them out of the vertical range of the metal detector coil. The team designed the mount to accommodate 4 ultrasonic sensors. The entire system assembly can be seen in Figure 19.

3.1.5 Electrical Design

In addition to reworking the mechanical design of the rover system, it became apparent to the team that improvement could also be made by redesigning the electrical design of the rover. The team focused on effective use of actuators and sensors, and clean and organized routing and documentation of wires. The team's analysis of the previous electrical design, coupled with the expectation of re-writing the majority of the rover's codebase, led to a large overall change and improvement in system architecture.

3.1.5.1 Ground Detection Sensors

In order to implement the roll-pitch mechanism for the sensor-head, the team needed to address issues faced by previous MQPs around detecting distance relative to irregular or "soft" surfaces such as grass. In previous MQPs it was reported that ultrasonic sensors were inaccurate, or erroneously returned no distance when over grass. In order to reproduce these errors and understand the problems faced by the previous MQP, the team first ran two sets of simple ultrasonic tests over grass. In the first set of tests, raw ultrasonic readings were measured while moving the sensor towards the ground from a height of around 1.5 feet. This test was conducted twice - once over a hard control surface and once over grass. In the second set of tests, raw ultrasonic readings were measured while moving the sensor horizontally at a height of roughly 5.5 inches over a hard control surface and grass. The results of these tests are shown in Figure 17 and **Figure 18** respectively. Both tests required the implementation of a moving average filter to successfully compare results. Implementation of the filter and plotting of results were performed in MATLAB.



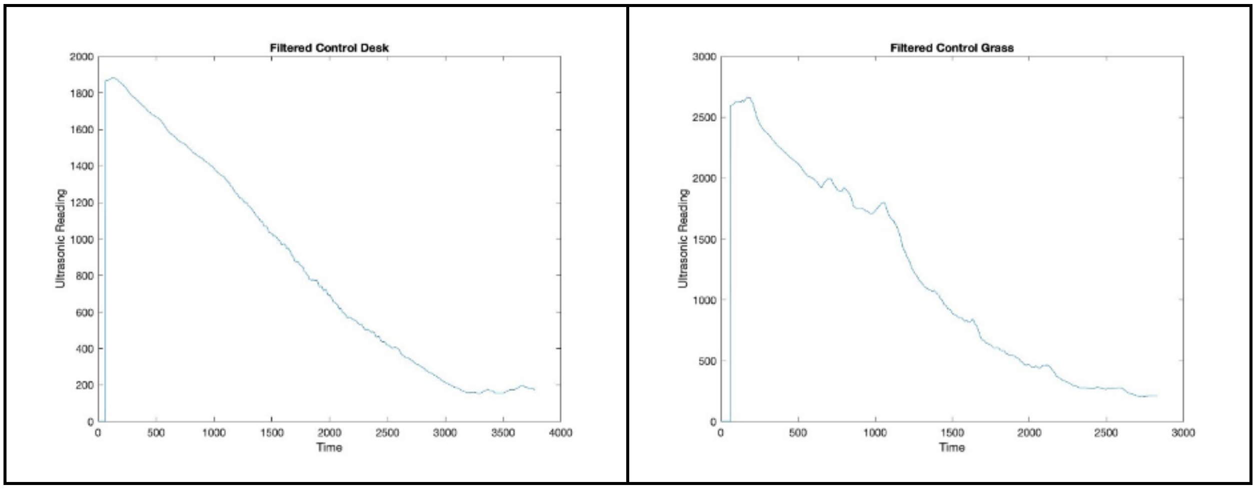


Figure 17 Vertical Motion Ultrasonic Test

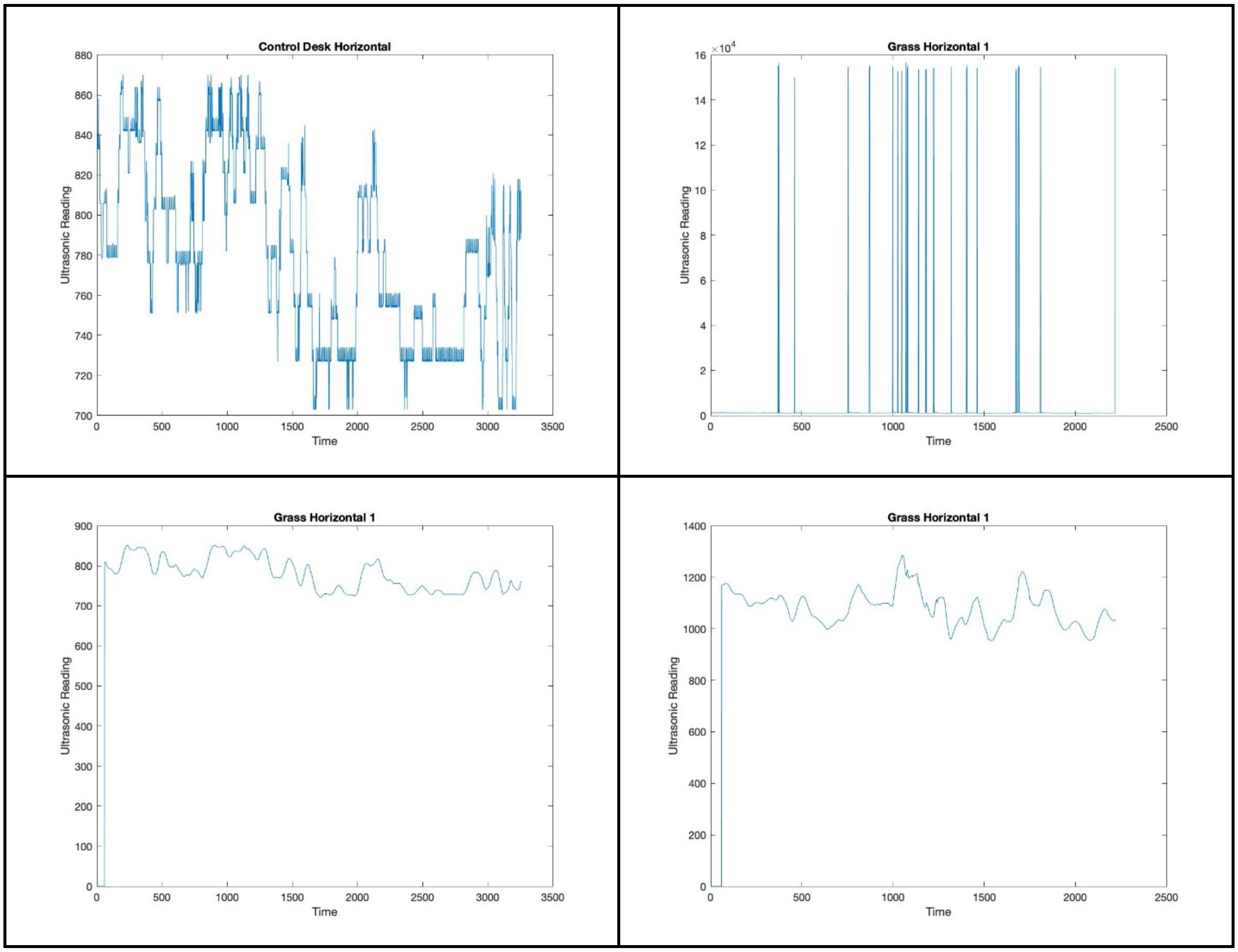


Figure 18 Horizontal Motion Ultrasonic Test

These tests showed some level of success in the implementation of ultrasonic sensors over grass. The team then examined the mechanical placement of the ultrasonic sensors. In order to ensure that the ultrasonic sensors would not interfere with the metal detecting coil, they were moved to a 3D printed mount that would place them in the four-cardinal directions. This is shown in Figure 19. The housing can be adjusted up or down along the carbon fiber tube depending on the range of the metal detecting coil implemented in the sensor head.

The filtered distance measurements reported by the ultrasonic sensors will be used to control the orientation of the sensor head. Because of the chosen orientation of the sensors, the pitch and roll axes are independent, and can be considered separately. For example, the distance measurements from the front and rear sensors can be used to approximate the pitch of the ground under the sensor head. The head is driven to an equivalent angle to maintain proximity to the ground. The process is equivalent for the roll axis. Towards the same goal, the linear actuators which control the four-bar mechanism—and therefore the height of the sensor head—are controlled by the ultrasonic sensor readings. Because the ultrasonic sensors are horizontally equidistant from the center of the sensor head, a simplistic estimate of the desired height is the average of the four distance measurements, plus the desired separation of the sensor head from the ground. The latter provides a buffer to keep the sensor head from contacting the ground in case of erroneous readings or unseen debris.

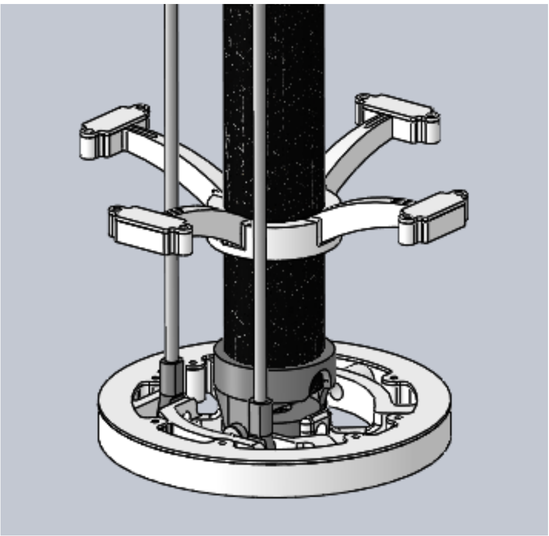


Figure 19 Ultrasonic Mounting Housing

3.1.5.2 Electrical System Redesign

Upon inheriting the electrical system from the stopping point of the previous MQP, the team decided that a true revitalization had to occur for the electrical system. Since the end goal of the

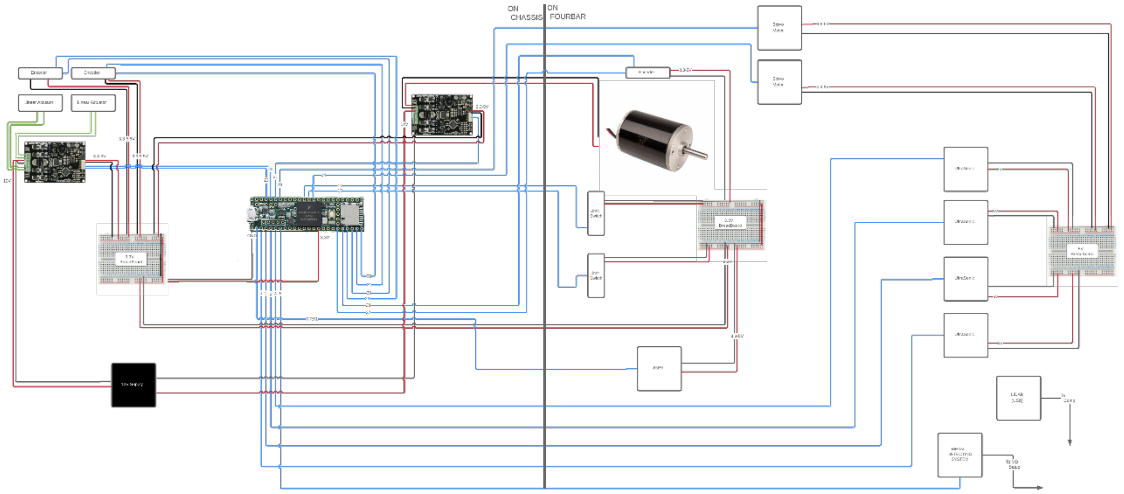


Figure 20 Rover Wiring Diagram

project is to have a professional grade product to apply in locations with active mines, the effort put into the wiring and powering of the robot had to appropriately reflect this.

The first step in the electrical redesign was to simplify the electrical board set up that was housed within the box of the rover. In order to complete this task, the electrical diagram shown in Figure 20 was devised as an aid to model the robot's electrical system. The team sought to reduce the number of redundant boards and communication steps within the electrical box, instead opting for a single Teensy 3.5 development board. This means that all inputs and outputs relied on the interface on the Teensy, rather than the combination of the Arduino Mega and Raspberry Pi alongside other boards that were previously installed in the electrical box. The two electronic boards that remained were two motor controllers that could handle the voltage and current necessary to run the linear actuators and the CIM motor. This meant a more contained and organized interior to the electrical box. The different boards will be lined up in specific locations that have been laser etched into a mounting plate to sit on shock mounts within the case, ensuring that the boards cannot move during rover operation, and no wire connections can break loose from their boards.

In addition to this simplification, the team needed to edit the system so that the wiring could be more easily followed and traced alongside the documentation generated. To achieve this goal, several different labels were printed out to be affixed to the different wires within the case, detailing the port that the wire goes to, its purpose (power, ground, or signal), and the wire's destination on the boards inside the box. In order for the wires to neatly come from one place within the box, the team decided to drill holes in the side of the box opposite the handle in order to affix waterproof connections. For this to work, the team had to segregate the wires into different categories and use connection ports at the end of these clusters of wires according to the gauge requirements and current draw from each wire. This simplifies the process, and protects the wires on the outside of the case, allowing each to travel only to the edge of the case, ensuring there will be no tension on the cables within the case, as it will be anchored on the box rather than out towards the end of the machinery on the four-bar.

Finally, the team also opted to reduce the electronics at the top of the metal detecting arm, consolidating the processes run there, and instead only using the space to process the output from the metal detector in order to send that information back into the electrical box. This was made simpler with a sturdier housing for the electronics, including a design that improved water resistance, and shields the wiring from rainfall. Since the new design of the robot has a very robust system of sliding the metal detector, the team could also mount cable chains to the top plate, and neatly route the wiring through the chain so as to never pinch in any mechanics of the rover, and to neatly disappear into the other wiring running along the arms of the lift. From there the team neatly stowed the wiring at different points of the rover, and wrapped the wires together using cable braid that tightens on the clusters of wires. These braids do not break out until the aforementioned connection ports are reached, allowing a cleaner and less intrusive wire design along the entirety of the rover.

3.1.6 Software Update

3.1.6.1 Software Architecture Changes

Under the electrical design detailed in the preceding section, processing has been split between the rover's onboard computer and two microcontrollers. Communication with the base station, mapping, path planning, navigation, control of the drive motors, and sensor inputs from the LIDAR and differential GPS modules are handled directly by the computer. This division was made because these are all tasks which require heavy computation, direct access to the system's ROS network, or USB communications. These tasks were performed by a series of ROS nodes written in C++ and Python. Low-level tasks of controlling the four-bar, slider, and sensor-head roll-pitch mechanisms (including implementation of the associated ultrasonic sensors) are written in C++ and run on a Teensy microcontroller. This communicates with the main computer via USB, through which the "scanning" motion is triggered. Finally, a dedicated Arduino Uno is used to monitor the metal detector, relaying information back to the Teensy. This division of the computational workload is outlined in Figure 21. It has been noted that the Arduino is somewhat unnecessary; its tasks could be offloaded to the Teensy. However, this shift would significantly increase the complexity of the Teensy software. As this hardware simplification task is relatively unimportant when compared to the larger tasks of the project, the Arduino remains.

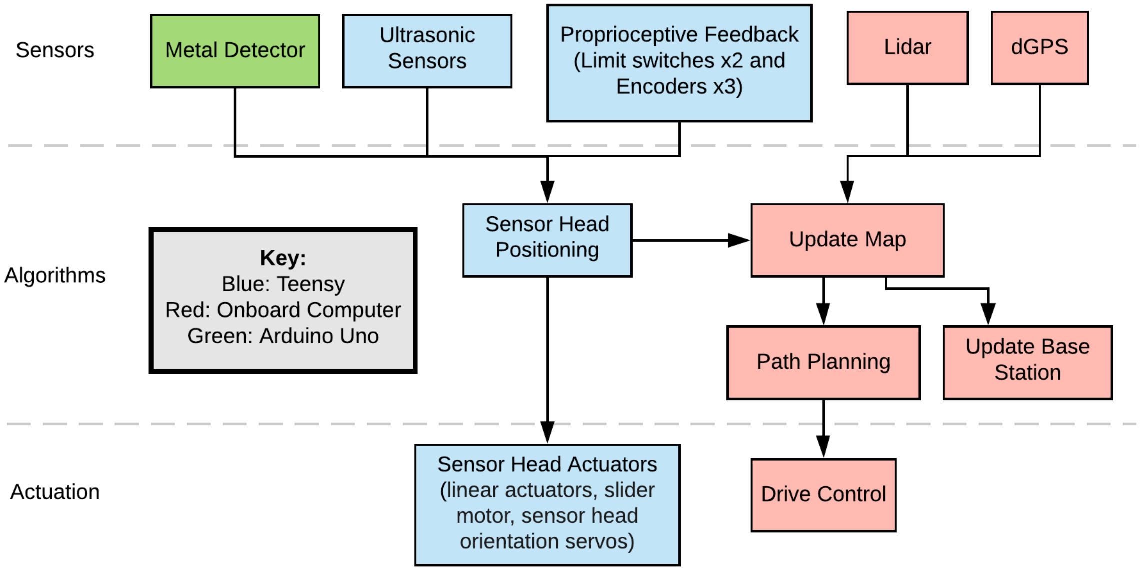


Figure 21 Rover software, sensing, and actuation

During operation, the Teensy acts semi-autonomously, but it is controlled at a high level by the computer. For example, the control loops that maintain desired positioning or motion of the four-bar and slider mechanisms run constantly based on sensor inputs to the microcontroller, but the command to "scan" is handed down from the computer as this action must be coordinated with driving. During such a scan, the slider is moved from one end to the other, while using readings from the ultrasonic sensors to adjust the height and orientation of the sensor head. If a metal object is sensed, the computer is notified, and an estimate of the suspected mine's location is generated.

Software on the computer consists of a series of ROS nodes. The readLidar and readGPS nodes handle USB communication with the lidar and differential GPS modules respectively. Each of these nodes publishes the gathered data to a corresponding ROS topic. These topics are subscribed to by the maintainMap node, which uses the incoming data to curate an occupancy grid representing the minefield. A local copy of the map—within the node—is updated every time new information is received. This map is made available to other nodes via a ROS service. The pathPlanning node calls this service each time a new path is generated. Thanks to the distributed nature of ROS, a node on the base station can similarly call the map service to obtain a copy which can be displayed to the user. The final piece of software on the rover computer is the navigation node, which transfers a pre-planned path into driving controls for the motors.

Software on the Teensy is fairly straightforward, as described in the figure above. Under normal operation, the slider mechanism is driven back and forth, sweeping the sensor head over the ground in front of the rover. As this happens, real-time feedback from the downward-facing ultrasonic sensors is used to control the sensor head's height via the four-bar linear actuators, and orientation via the pitch and roll servos. By this method, the sensor head maintains proximity to the ground as it scans. If metal is detected, the Arduino Uno drives an output pin high, triggering a hardware interrupt on the Teensy and stopping the scan. At this point, the Teensy must notify the rover's onboard computer that it should stop driving immediately. With the rover halted, the Teensy can move the slider to find what it believes to be the center of the mine. During this process, the Teensy can send requests to the computer to perform slight forward or backward movements of the entire rover. This allows for a better two-dimensional estimate of the center of the suspected mine. Given the binary output of the metal detector, this estimate is taken to be the mean position of all local points where metal is sensed. With a position identified, the Teensy relays this information to the onboard computer, which can calculate the global coordinates of the suspected mine.

Due to the COVID-19 pandemic and subsequent inability to access hardware, the development of most of this software was deemed futile at this time. The readLidar and readGPS nodes could not be developed without the corresponding hardware modules, and the any significant development of downstream nodes would be based on speculation. In its final form, files exist for the aforementioned nodes, but they are simply placeholders to represent the structure of the final ROS package. Analogously, the Teensy-based code has been left as only a fragment of the planned implementation. Significant further code development will be necessary to attain an operational system.

3.1.6.2 Path Planning Algorithm

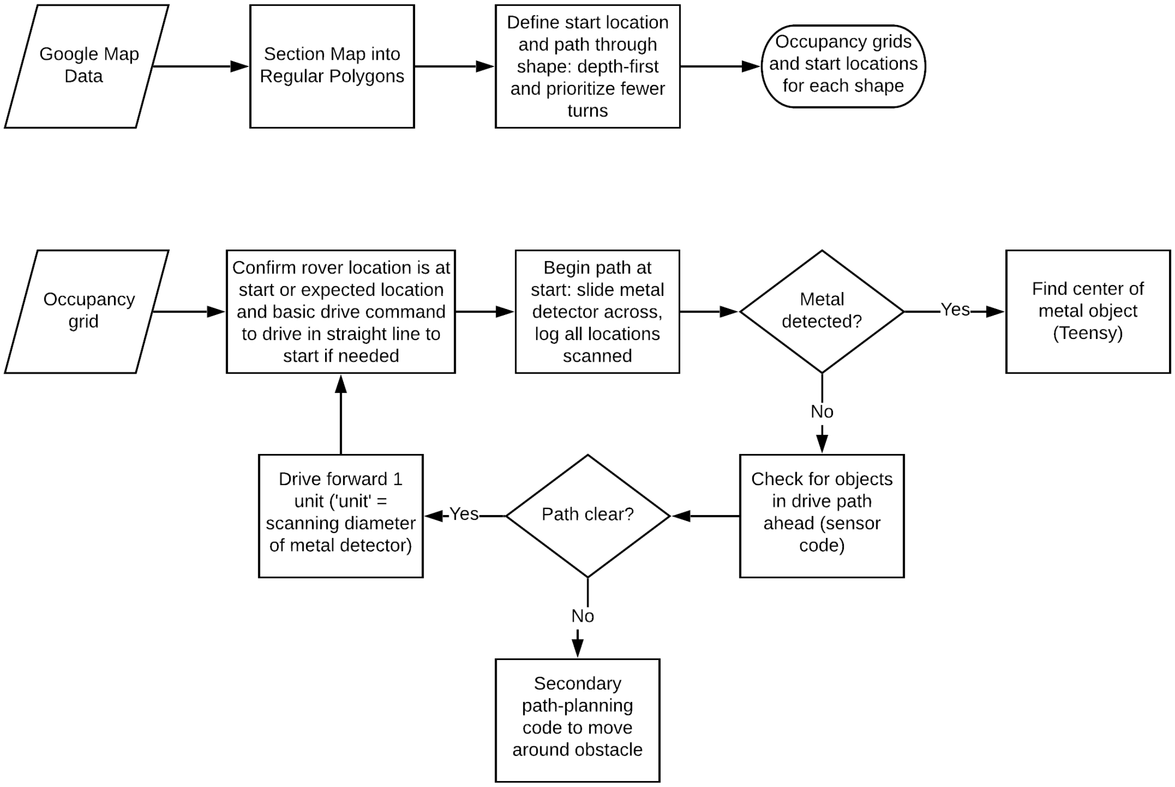

Google Map Data
Section Map into Regular Polygons
Define start location and path through shape: depth-first and prioritize fewer turns
Occupancy grids and start locations for each shape
Occupancy grid
Confirm rover location is at start or expected location and basic drive command to drive in straight line to start if needed
Begin path at start: slide metal detector across, log all locations scanned
Metal detected?
Yes
Find center of metal object (Teensy)
No
Check for objects in drive path ahead (sensor code)
Path clear?
Yes
Drive forward 1 unit ('unit' = scanning diameter of metal detector)
No
Secondary path-planning code to move around obstacle

Figure 22 Flowchart of the rover path planning algorithm

Our system utilizes two path planning approaches to control movements: a primary path for unimpeded rover movements, and a second which utilizes onboard LIDAR to navigate around obstacles within the minefield. The primary path is generated from the raw data output by Google Maps API. The map data consists of an array of latitude/longitude coordinates that describe the map area's bounds. This shape is analyzed for any concave areas in the vertical (y) direction and, if present, divides the map's area into a series of convex polygons. This can be seen in a high-level flowchart in Figure 22.

The convex polygon(s) are then converted into one or more occupancy grids, as seen in Figure 23. The length of a single grid unit is slightly smaller than the diameter of the metal detector's sensing cone at ground level, allowing for slightly overlapping scanning of each grid

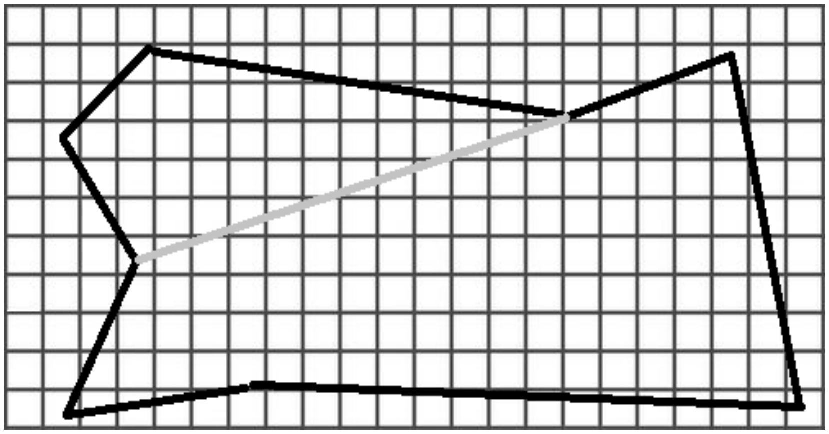


Figure 23 Example of Polygon Splitting Technique

coordinate. In transforming the GPS's 'world' map to an occupancy grid, the polygon's lowest x coordinate is used to define the [0,0] index and the y=0 axis of the array, which is used as the starting location for rover operations. The rover begins at the coordinates listed this newly defined origin.

The rover proceeds to perform a full sweep along the y-axis, sensing along the length of the far left marking each space in the grid as having been scanned. After sweeping, the rover will move forward one grid space and repeat this process. Once reaching the end of the axis, the rover moves to either the highest or lowest y value for that x value, depending on the direction the rover came from. The rover adjusts y values if necessary, moves over to the next x value and turns 180 degrees before repeating the process again. Once a polygon is completely scanned, the robot uses A* to find a path to the start of subsequent polygons. This process can be seen in Figure 24.

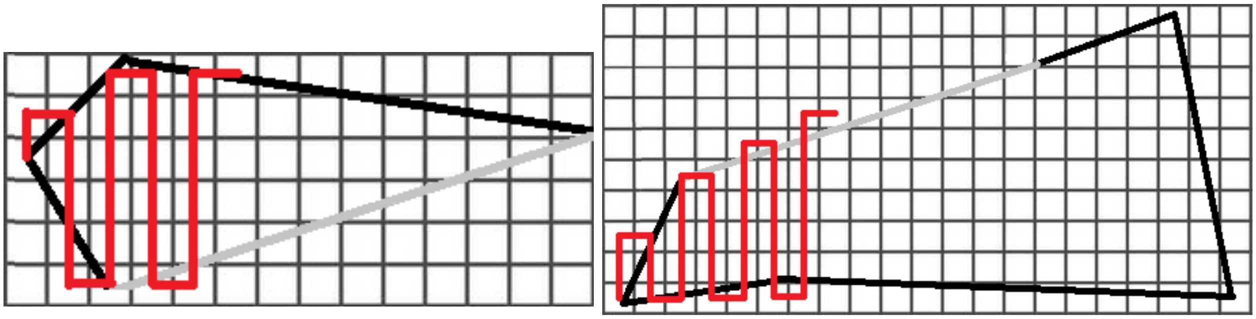


Figure 24 Demonstration of Minefield Path Generation

This path planning algorithm runs as the primary driver of rover movement unless a mine or other obstacle is presented. If either event occurs, the primary algorithm defers to secondary algorithms for homing in on and marking any metal detected and avoiding obstacles. These algorithms update the occupancy grid and the primary algorithm will procedurally continue its path by sensing its way around an obstacle (or using a predefined arc to move around a mine), back-tracking into previously scanned areas as necessary to move around the obstacle or mine.

3.1.6.3 Obstacle Avoidance

The initial plans for this project included obstacle avoidance capabilities for the rover. When an obstacle is detected by the lidar sensor, it should be automatically added to the map. The appearance of this obstacle must trigger a re-planning of the path using the algorithm defined above. With this new path, the rover can continue as usual, inherently avoiding all obstacles and mines. This code is heavily dependent on the rest of the ROS nodes, and therefore was not written due to barriers introduced by the COVID-19 pandemic.

3.2 Aerial Mine Detonation

3.2.1 Project Scope

This iteration of the Demining Autonomous System Project is, in terms of the drone, focused on overhauling the existing drone system and integrating it with the base station and rover portions of the system. In doing this, the software written by the previous team and the sensors they used will need to be updated since drone technology has improved since 2014. The biggest change to the system is converting from traditional GPS navigation to navigation via differential GPS. This provides centimeter-level accuracy but will require a complete update to the way in which the drone currently processes information. Since the team will now have actual landmine locations from the rover, it is necessary to develop an algorithm that will create an efficient path through the landmines that also accounts for reloading after the drone runs out of payloads to drop.

The team will not be considering potential issues such as trees and other objects at or above 20ft (our operating altitude). This also includes topographical differences such as hills that the drone could run into during flight. By assuming that the terrain of the minefield is relatively flat and free of exceedingly tall obstacles, the team could focus on updating the dropping mechanism, electrical systems, and software for use with a new suite of sensors.

3.2.2 Initial Design Considerations

3.2.2.1 DJI S1000

The DJI S1000 “Spreading Wings” octocopter, released in 2014, is a high-end industrial drone built for stability and high lift capacity. It has a maximum takeoff weight of 11.0kgs, and the drone itself weighs only 4.2kg (without a battery). This allows for a significant amount of additional weight to be carried, such as large cameras or a larger battery to extend the typical 15-minute flight time. The vehicle provides a simple, robust platform that allows for customization by the end user, although the modifications necessary for this project are likely more extensive than any anticipated by the designers. The S1000 was chosen as the basis for this project simply because it was available for use through WPI’s Robotics Program. That being said, it fits the constraints of this project well and it provides an excellent foundation for further development.

3.2.2.2 Pixhawk Flight Controller

Pixhawk is a family of open-hardware autopilot boards designed to control autonomous vehicles. Previously, the stock flight controller on WPI’s DJI S1000 had been replaced by a Pixhawk 1 flight controller. Although this module was adequate in early testing, it was eventually determined that the module’s internal compass was no longer sufficient for autonomous flight. It was decided that, rather than attempting to add a new external compass to an old flight controller, the board would simple be replaced by a Pixhawk 4. The new controller is updated with a faster processor and significantly better internal sensors. In addition, it was purchased as a package including the Pixhawk 4 and a modern external compass module which could be mounted away from the electrical noise that is prevalent close to the drone's high-current electronics.

The Pixhawk, running the open-source ArduPilot Copter firmware, acts as the drone's flight controller—reading sensor data, computing position and attitude estimates, and facilitating the control loops that keep the drone stable while airborne. While it is quite excellent at performing these actions, it is not sufficient for control of the high-level autonomous tasks that are required for this project. To solve this problem, the flight controller is paired with a "companion computer"—a second microcontroller which handles high-level control and communications (discussed further in 3.2.4 Flight Control System Architecture).

3.2.3 Mechanical Changes

3.2.3.1 Payload Considerations

The original payload was essentially a sandbag; a clump of sand wrapped in plastic wrap. This was effective for past projects because sand is easy to find, has a small chance of creating shrapnel, and makes for an easy-to-create payload. However, there were still some issues with it. When created, it is tough to ensure that the shape is uniform to reduce drag while falling. In addition, sand around the world is different, so while loading this, users can unintentionally add rocks or anything that could result in shrapnel.

This year, the team tested with water balloons. There were several benefits to using these over sandbags. First, water balloons always maintain their shape while falling. The rubber of the balloon keeps the interior together and it does not warp a significant amount while falling. Second, when the balloon hits the ground, the water spreads out evenly over a small area, which can help distribute pressure if the payload is ever dropped off center. In addition, water is the same around the world, so when filling water balloons, there is no chance of creating shrapnel and the team can guarantee the size of the water balloons (within a certain margin) based off the common density of water.

The team conducted tests with sand in a water balloon, to see if utilizing this design of a casing could still work with sand. However, the balloon ripped very quickly after trying to put sand into it. Plus, the team was unable to get enough sand into the balloon for it to be useful. Lastly, it did not have a uniform shape, so it would create drag and potentially throw the payload off course while falling. From here, the team tried filing the balloons with water. Unlike the bags of sand, there was a uniform shape and density throughout.

3.2.3.2 Dropping Mechanism

The initial dropping mechanism from past projects was a set of 3D printed tubes that were each controlled by a single actuation solenoid. There were several downsides to this design, stemming from its inherently simple nature. First, each cylinder could only be used once. This required the drone to be reloaded every four drops, which led to many cycles in and out of the minefield, increasing the overall system operation time. Next, the four cylinders were placed in different locations underneath the drone so the sandbags would drop from different locations. This needed to be accounted for when the drone was homing in on the drop target and was different after each drop. Since the previous team did not account for this discrepancy, their overall system accuracy was less than it could have been.

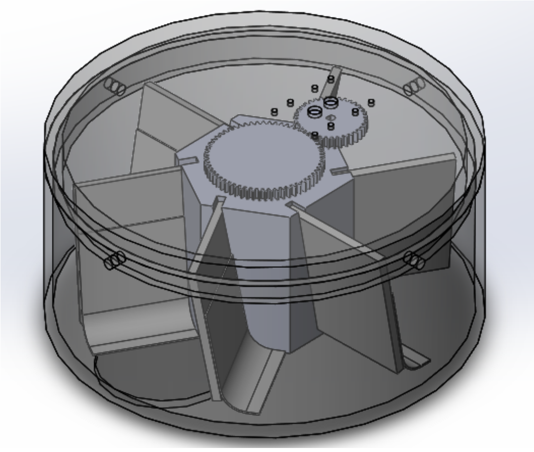


Figure 25 New Dropping Mechanism Design

The new design that was developed worked to solve these issues, and can be seen in Figure 25. It is a rotary style mechanism that can hold up to six payloads, specifically water balloons. The first issue it tackled is the dropping location. Instead of having six different dropping locations, there was a single place for each payload to drop from. This way, the offset from the dGPS coordinate to the payload dropping point would never change. Next, this system was created to be easy to reload. In the past system, users had to hold in a solenoid while adding a new payload, which was tough to do. This system uses four locking pins to allow the top of the mechanism to be removed, which allows users to easily load the new payloads.

Other important improvements of this new mechanism involved the actuation of the system. When dropping the different payloads, the inside will turn, moving the payload over the drop spot. The payload would fall into a holding chamber, where it could be released similar to the previous dropping mechanism. This could be done at any point during the flight because once loaded into the holding chamber, the payload could be dropped with just a solenoid actuation. This would be reset with a motor to close the hatch and a limit switch to sense when it was closed. The motor would then move out of the way so when the next payload was ready to be dropped, the solenoid could fire again and allow the payload to fall. This removes any source of timing error with the rotation of the payloads.

One concern that was discussed in the design process was the non-uniformly distributed weight of the balloons. When a single balloon was dropped, it would affect the center of gravity of the drone. However, due to the Pixhawk's ability to balance itself, the likelihood of a small water balloon impacting the overall system stability is minimal. Even still, field testing will determine the degree to which shifts in payload impacts the system stability.

3.2.4 Flight Control System Architecture

3.2.4.1 Communication protocol for base station and drone

Communication between the base station and the drone is accomplished by a pair of 500mW 915 MHz Holybro Telemetry Radios running open-source SiK firmware. This wireless link simulates a wired serial connection between the two systems. A custom packet-based protocol has been designed and implemented allowing for simple, yet robust communication. Toward this goal, each packet contains only a start byte, a message-type byte, a message-length byte, a variable number of data bytes (the count of which is defined by the message-length byte), and a parity byte for confirmation of message integrity. The parity byte is simply the bitwise XOR (exclusive-or) of all of the bytes of a message, excluding, of course, the parity byte itself. While this is a relatively weak integrity check (there exist only 256 unique parity bytes under this rule), this scheme was chosen because it is simple and easy to calculate on a resource-limited system such as the drone's onboard microcontroller. The specifics of this protocol are detailed in Appendix B: Custom Serial Protocol.

3.2.4.2 Flight Control Logic Onboard the Drone

In order to receive, parse, and react to the commands sent from the base station, the drone has a Teensy 3.5 microcontroller onboard. This controller was chosen because it is quite fast (120 MHz clock frequency) and it has six hardware serial ports. Multiple of these ports are necessary for communication with the base station and flight controller. Additionally, the Teensy's hardware floating point unit allows for rapid computation using latitude and longitude values—something that is very necessary for this application. The software running on this board coordinates all high-level functions of the drone, communicating with both the base station and the flight controller. This code was written in C++ and runs directly on the hardware with no operating system. In general, this subsystem handles the setup of the drone's peripherals, ensures that all portions of the drone are operational, and controls the flightpath of the drone while it is in the air. If, at any point, the abort signal is received from the base station, the controller stops any current operation and puts the flight controller into a "return-to-launch" state, in which the drone will fly back to the takeoff point and land.

3.2.4.3 MAVLink: Communication with the Flight Controller

On a lower level, the drone is controlled by a Pixhawk 4 flight controller, which handles the control loops that keep the drone stable and at its assigned location. The Pixhawk is commanded directly from the Teensy over a serial port using the MAVLink 2 protocol. MAVLink is a packet-based serial protocol designed to control drones from external hardware (either directly from a base station, or, as in this case, through a "companion computer"). Its C++ implementation is extremely lightweight, so it is ideal for use on resource-limited systems such as our microcontroller. Each packet has an associated message ID number, which tells the receiver what the packet contains, and therefore how to parse it.

The ArduPilot Copter firmware running on the Pixhawk has been set up to communicate via MAVLink on one of its serial ports, which is connected directly to the Teensy. The bidirectional connection allows the Teensy to receive acknowledgement messages from the Pixhawk, confirming that its transmissions are received successfully. To send the drone to a specific set of coordinates, the Teensy simply packs up and sends the appropriate MAVLink message over the serial port. All of the low-level flight control is handled by the Pixhawk, leaving the Teensy free to handle high-level coordination of the system.

3.2.5 Electrical Updates

3.2.5.1 High-Level Electrical Overview

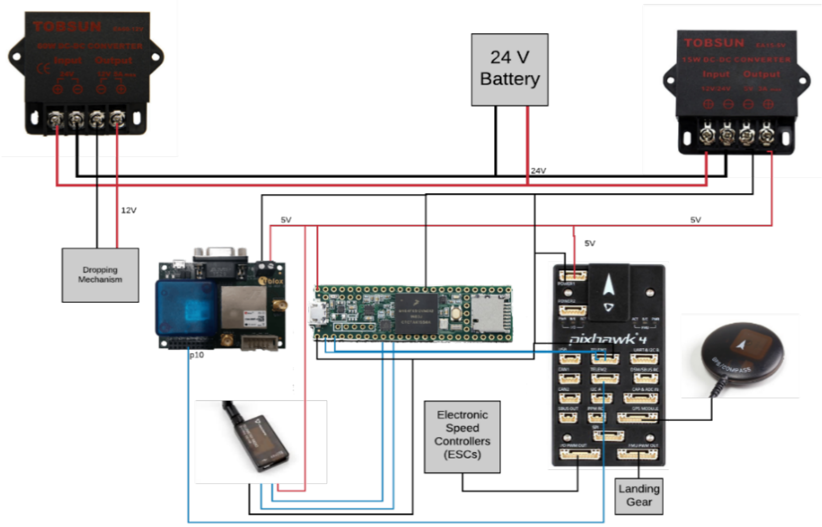


Figure 26 Drone Wiring Schematic

To implement the planned control architecture, the electrical assembly depicted in Figure 26 has been implemented on the drone. Twin voltage regulators are used to reduce the drone's 24 V battery voltage to a 5 V supply for logic circuits, and a 12 V supply for the dropping mechanism's stepper motor. A serial port on the Teensy has been routed to the Pixhawk's "Telem 1" port, which has been configured as a MAVLink interface. Similarly, the "Telem 2" port has been configured to accept GPS information coming from the dGPS board. Another serial port on the Teensy is connected to the SiK radio for communication with the base station. The Pixhawk is directly wired to both the RC receiver (for manual control from the ground), the electronic speed controllers (ESCs) which drive the motors, and the external GPS/compass module.

3.2.6 Software Updates

The software that was written for the drone's onboard microcontroller was very different from the code written for the previous iteration of the project. Because the old project used a Raspberry Pi rather than a microcontroller, a convenient Python library was used to communicate with the flight controller. The new system allows for much deeper control of the system at the expense of being more complex to design. This tight grip on specifics has allowed the software to be tuned to the exact specifications of the project while maintaining maximum efficiency.

3.2.6.1 Base Station Communication

The first major component of the drone's software handles all communication with the base station. This code implements both the sending and receiving process involved in implementing the protocol detailed in Appendix B: Custom Serial Protocol. Sent messages are buffered in the microcontroller's hardware serial buffer, allowing the sending function to be non-blocking. The receiving process is a bit long, as all bytes of a message must be received over a relatively slow serial connection before the message can be parsed. If the receive function simply waited for the end of the message, the rest of a program would grind to a halt. This problem is solved using a state machine. One call of the function simply receives all available bytes and processes them using the state machine, never blocking or idling. The only requirement of this structuring is that the function must be called quite often in order to continually receive any messages that may be incoming. The rest of the functions in this file are simply conveniences for packing and parsing messages.

3.2.6.2 Path Planning

One of the main reasons for needing this communication link between base station and drone is for the locations of mines to be passed to the vehicle. The path planning for the drone is performed on the base station, so it is assumed that received mine location information arrives in the order that will eventually be flown. The flight path of the drone is computed as follows: it will first fly to and detonate the closest unexploded mine, called the “primary” mine. From there, it will attempt to detonate the five mines which are closest to the primary mine, in order of their proximity to the primary mine. At this point, the drone will have exhausted its six payloads, and it will return to the home location to land. If there are more mines in the minefield, then the process repeats after a landing to pick up additional payloads. In this way, mines are grouped in sets of six, beginning with those which are relatively close to the home location and eventually proceeding to those that are further away.

In addition to the coordinates of a mine, each of these mine-location messages contains an escape location—the point that the drone should fly to immediately after releasing a payload over the corresponding mine. These escape locations are calculated such that they will bring the drone out of harm’s way, while flying the direction of the next mine on the list. The exception to this rule happens when the drone is releasing its final payload before flying back to the home location. In this case, the escape point has been calculated to lead the drone in the direction of the home location.

3.2.6.3 Flight Controller Communication

The next major section of the drone’s software facilitates the MAVLink protocol for communication with the Pixhawk flight controller. The official MAVLink C library is used to handle most of the low-level functionality; this code acts as a “wrapper” for that library, simplifying and streamlining the commands that are needed for this system. These include arming, disarming, taking off, setting a target navigation waypoint, returning to the home location (and landing), and several other setup-related commands. These functions greatly simplify and standardize the main structure of the code.

3.2.6.4 Main Program Structure

The main execution of the program begins by running through a sizeable setup process. This procedure begins by waiting for the base station and Pixhawk to become active and ready to operate. After that, the code execution enters a state machine, where the following actions are performed in sequence. Each of these steps must complete before progress advances.

1. Initialize a stream of GPS metadata updates from the Pixhawk (necessary to know when the dGPS has a sufficient position fix)
2. Wait for the dGPS to obtain a position fix
3. Command the Pixhawk to reset its home location to the current position
4. Initialize a stream of GPS position updates from the Pixhawk (necessary to know when the drone has reached a target waypoint)
5. Request the newly reset home location from the drone, and wait for it to be received
6. Send the home location to the base station
7. Wait for, and subsequently receive the path planned by the base station

These processes must be carried out in a state machine because the program must “simultaneously” be communicating bidirectionally with both the base station and the Pixhawk. In reality, the microcontroller only has a single processor on which to execute commands, but this code structure allows for multiple time-intensive tasks to be handled concurrently.

After this setup process is complete, the program advances to the main execution of its task: flying to and attempting to detonate mines. Once again, event sequencing is coordinated by a state machine to allow for communication with other devices in the pipeline. The system defaults to a disarmed state, where it is simply on the ground, inactive. When a takeoff command is received, it proceeds to command the Pixhawk (or, more precisely, the ArduPilot firmware) to arm and takeoff. The altitude of the vehicle is monitored, and the takeoff is considered complete when the estimated altitude above the takeoff point falls within 5% of the target altitude. At this point, the flight controller is commanded to fly to the first set of mine coordinates. When the drone’s position is sufficiently close to the mark, a payload is released, and the escape sequence is triggered. The drone will fly immediately to the escape point associated with that mine, before moving on to the next mine and repeating the process. When the drone runs out of payloads to drop, it moves into a “return home” case, where it flies back to the home location and lands, returning to the disarmed state. At this point, if a payload has been dropped on all known mines, the drone will be fully deactivated. Otherwise, it will wait for another takeoff command. If the abort command is received at any point during the flight, the state machine will immediately enter the “return home” case.

3.2.6.5 Integration of dGPS with the Pixhawk

Early in the project, preliminary research showed that the dGPS board would be outputting more information than ArduPilot knew how to parse, thereby rendering the flight controller unable to maintain a GPS fix. Because of this assumption, early designs had the dGPS data sent directly into the microcontroller, rather than into the flight controller. In theory, the microcontroller would skim through the data, pick out the necessary NMEA strings, and pass them onto the flight controller by writing them to a serial port which would be connected to the flight controller’s GPS port. In this way, the microcontroller would act as a filter while being invisible to the flight controller. Initial testing of the dGPS boards revealed that with careful configuration of both the dGPS and ArduPilot, the two could be linked directly, without the need for the microcontroller in the middle. This reduced the complexity of the system in both hardware and software.

3.2.7 Flight Controller and Compass Upgrade

During early fully autonomous flight tests, it was noted that the internal compass in the Pixhawk flight controller could not be calibrated successfully. Without an accurate compass, the drone was unable to maintain a hovering state or navigate at all. It was eventually decided that, rather than trying to purchase and attach an external compass or IMU to the system, it was in the best interest of the project to replace the aging flight controller all together. There was concern with respect to the accuracy of the internal accelerometers and gyroscopes, and the reliability of the controller itself. A Pixhawk 4 was purchased to fill this need, along with the accompanying external GPS/compass module. The new hardware was mounted to the drone and calibrated. Subsequent flight tests showed significant improvement in flight stability over the previous controller. Notwithstanding wind gusts, the drone appeared almost perfectly still while hovering.

3.3 System Integration

3.3.1 User Interface

In order to address the system level goal of designing an application-based solution for the base station, the team began by defining the desired flow of operation of the system. Shown in Figure 27, a sequence diagram of system events was made at a high-level of abstraction. The sequence diagram defines the base station, rover, and drone objects, and the actions taken by the Actor, or user of the system. First the user turns on the base station and sets up the application, which prompts all necessary calibrations for the rover and drone systems. Then, the user defines the minefield through the BaseStation application using the GoogleMaps API designed by the previous year’s MQP. The base station application then takes care of deploying the rover on its mission and reporting the results to the user. After this is complete, the user is asked to approve the data

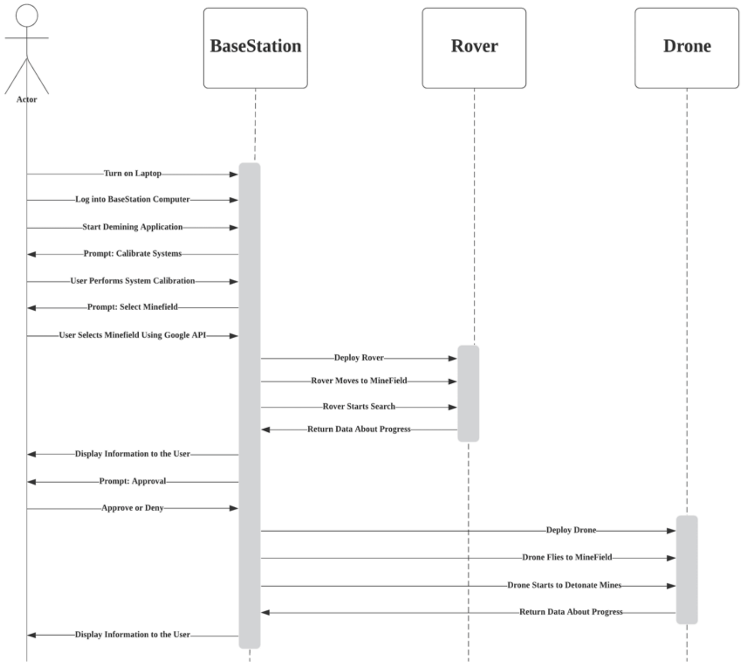
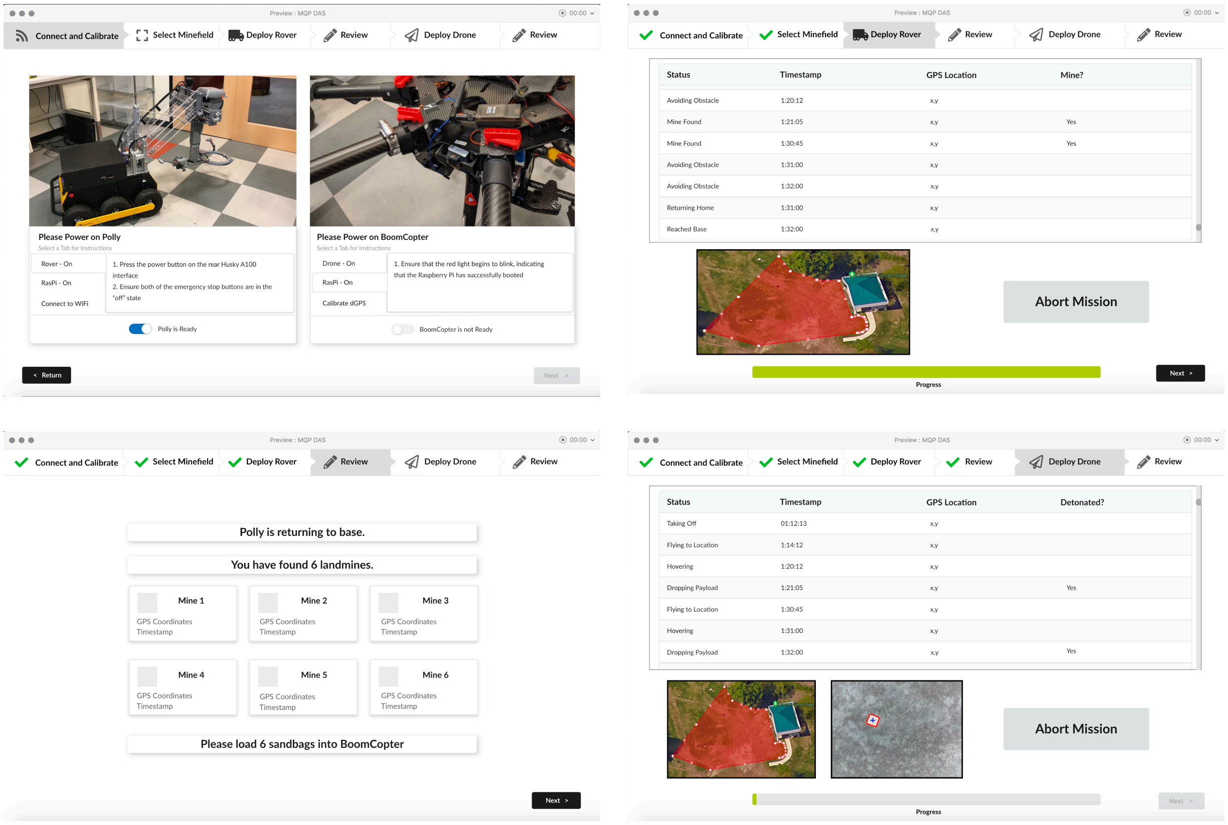

Actor
BaseStation
Rover
Drone
Turn on Laptop
Log into BaseStation Computer
Start Demining Application
Prompt: Calibrate Systems
User Performs System Calibration
Prompt: Select Minefield
User Selects Minefield Using Google API
Deploy Rover
Rover Moves to MineField
Rover Starts Search
Return Data About Progress
Display Information to the User
Prompt: Approval
Approve or Deny
Deploy Drone
Drone Flies to MineField
Drone Starts to Detonate Mines
Return Data About Progress
Display Information to the User

Figure 27 Sequence Diagram of Events

before the BaseStation then deploys the drone on its mission. Results of this mission will also be reported to the user.

Following the determination of the sequence of events, the team used Adobe XD to design a potential user interface for the application. Images from this “mockup” are shown in Figure 28. This exercise allowed the team to better scope and define the flow of the application, as well as set a more measurable goal for the final version of the application. The formal application User Interface was implemented in JavaFX, utilizing Scenebuilder.



Please Power on Polly

Select a Tab for Instructions

Rover - On

RasPi - On

Connect to WiFi

1. Press the power button on the rear Husky A100 interface
2. Ensure both of the emergency stop buttons are in the “off” state

Polly is Ready

Please Power on BoomCopter

Select a Tab for Instructions

Drone - On

RasPi - On

Calibrate dGPS

1. Ensure that the red light begins to blink, indicating that the Raspberry Pi has successfully booted

BoomCopter is not Ready

Status	Timestamp	GPS Location	Mine?
Avoiding Obstacle	1:20:12	x,y	
Mine Found	1:21:05	x,y	Yes
Mine Found	1:30:45	x,y	Yes
Avoiding Obstacle	1:31:00	x,y	
Avoiding Obstacle	1:32:00	x,y	
Returning Home	1:31:00	x,y	
Reached Base	1:32:00	x,y	

Polly is returning to base.

You have found 6 landmines.

Mine 1
GPS Coordinates
Timestamp

Mine 2
GPS Coordinates
Timestamp

Mine 3
GPS Coordinates
Timestamp

Mine 4
GPS Coordinates
Timestamp

Mine 5
GPS Coordinates
Timestamp

Mine 6
GPS Coordinates
Timestamp

Please load 6 sandbags into BoomCopter

Status	Timestamp	GPS Location	Detonated?
Taking Off	01:12:13	x,y	
Flying to Location	1:14:12	x,y	
Hovering	1:20:12	x,y	
Dropping Payload	1:21:05	x,y	Yes
Flying to Location	1:30:45	x,y	
Hovering	1:31:00	x,y	
Dropping Payload	1:32:00	x,y	Yes

Figure 28 Sample Application Designed in Adobe XD

3.3.2 Communication Structure

In addition to hosting the user interface, the base station coordinates all communication between rover and drone. The high-level communication structure is detailed in Figure 29.

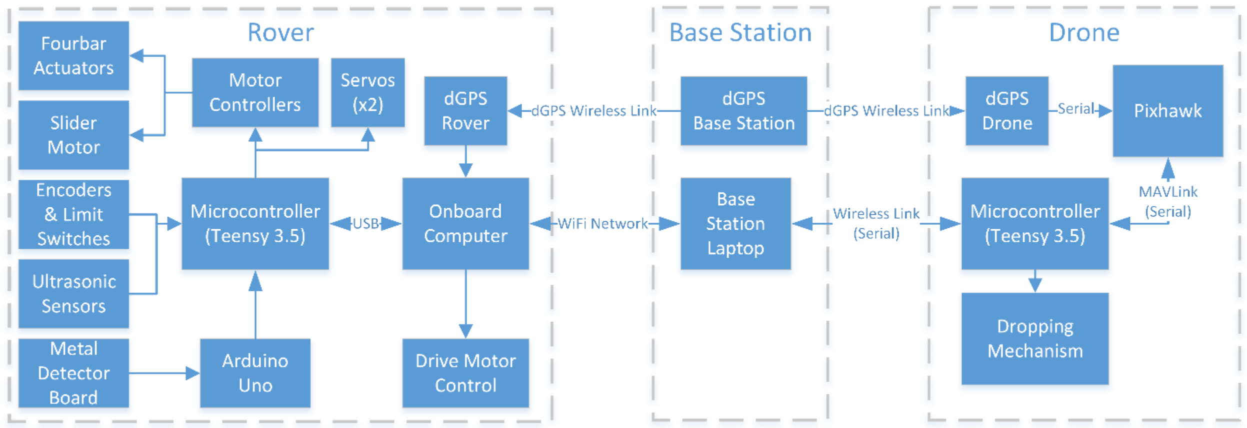


Figure 29 Outline of System Communication Structure

A WiFi network allows the rover's ROS network to reach the base station. This system is practical for the rover but is too complex and power-hungry to be a realistic option for the drone. The base station acts as an intermediary, collecting data from the rover, and relaying it when necessary to the drone using a custom serial packet protocol on a wireless link. The base station also contains a differential GPS unit, which communicates with corresponding units on both the drone and the rover.

3.3.3 Application Design

The final application for the use of the Autonomous Demining System was implemented in Java. This interface interacted with ROS, which in turn communicated with the rover and drone. When the application was first designed, the Team laid out a simple UML Class Diagram shown in Figure 30 to guide the development progress.

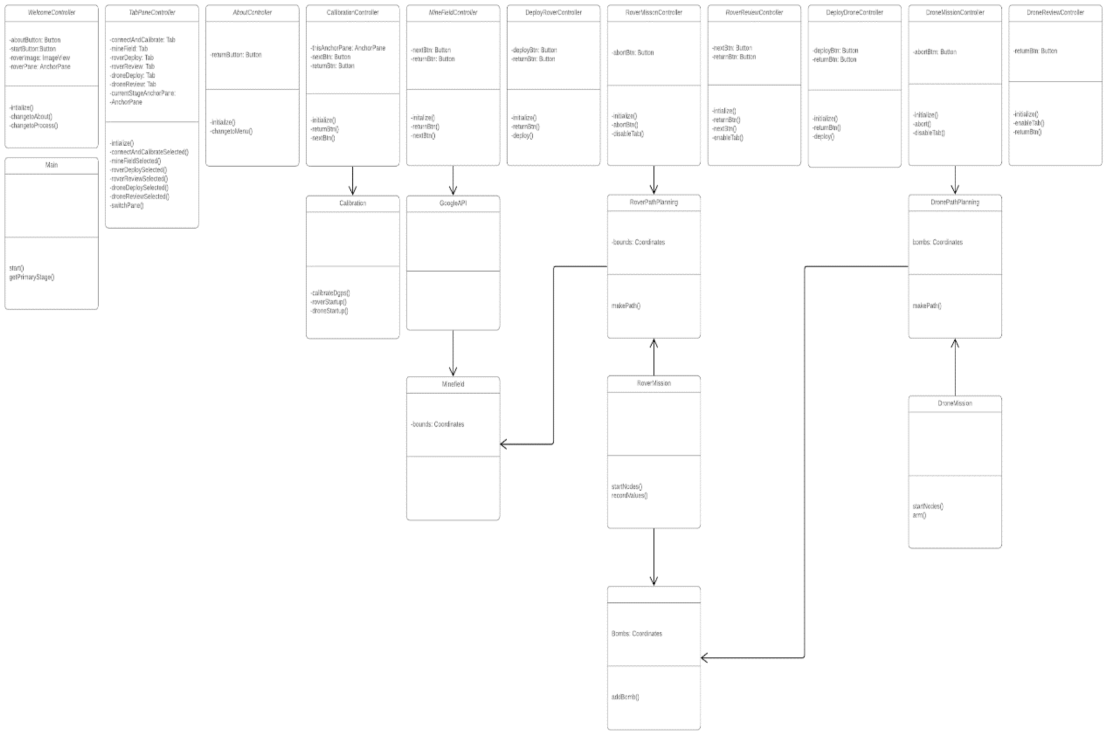


Figure 30 Initial Application Class Diagram

3.3.4 dGPS Navigation

One of the goals the team set as part of our project is to develop a more accurate method for locating the landmine. The previous MQP used GPS to get to the relative area of the landmine, then used a camera to locate the visual marker placed onto the ground above the landmine by the rover. The system iteratively moved towards the marker, as seen in the camera image. When the estimated position error fell below 0.5 meters, a payload release was triggered [13]. The current team quickly determined that, with the size of payloads being used and the accuracy of the system as documented, a more accurate localization method would be necessary. In order to have a system that would realistically function, an accuracy of a few of centimeters would be needed.

The developed system uses differential GPS to achieve this desired accuracy. Using standard GPS, the base station has a rough estimate of its own position, with an accuracy on the order of a few meters. Using differential corrections transmitted from the base station, the modules mounted on the drone and rover each have a global estimate of their respective positions, which are highly accurate relative to the base station position. In good conditions, these position estimates are accurate to a few centimeters. Because both of these units share a base station, they share this accuracy in a local sense—the two modules agree with each other with centimeter-level accuracy. On a global scale, positions of suspected mines are only known with an accuracy of a few meters. However, the shared base station module allows for centimeter-level accuracy locally.

3.3.4.1 Replicating Previous GPS Data

Our first test was to try and understand the accuracy of the system as the former team had it set up. For this the team placed a GPS module in the field and determined its absolute location. The team then manually input this number into ArduPilot as a waypoint. What the team found was that directly after takeoff, the drone circled around the initial starting position a few times before flying towards the given waypoint. Once it got to the waypoint, it repeated the process of circling around the point before lowering over the given point. Much later in the project, it was determined that this circling behavior was due to a bad compass calibration. This test appeared to have an accuracy of around one meter, which was the result of the combined inaccuracy of the given waypoint point, the drone's GPS module, and the ArduPilot command having a certain acceptable error tolerance. The use of differential GPS allows for the minimization of nearly all the observed error.

3.3.4.2 Testing dGPS Accuracy with Two Modules

From here the team decided to test a dGPS unit to see if it would be more accurate than the current GPS and camera method. To be more accurate than the previous system, the dGPS would have to find a point with less than half a meter accuracy, ideally accurate within a couple of centimeters. Our first test involved us taking the two dGPS modules (the one that would live on the drone and the one for the base station) to Institute Park to test its accuracy and ability to communicate over long distances. Using the relative position messages receiver (located in UBX-NAV-RELPOSNED) the team was able to compare the relative distance between the two dGPS modules. The team compared the relative distances from this to Google Earth measurements of the same distance and with these calculations achieved were within half a meter of the actual distance. The error could be human error from picking points on Google Earth. Table 1 shows the data from the team's first trial.

Table 1 Data from Test 1 of the Differential GPS unit

Trial		dGPS (m)	Raw Map Distance	Maps (m)	% Error
Short Range	N	5.8832	6.3487 m @ 23.09 degrees	5.8401	0.73
	E	2.8623		2.9258	-2.22
Long Range	N	-15.947	28.834 m @ 237.10 degrees	-15.6597	1.80

	E	-23.359		-24.206	3.62

This test taught us how quickly the dGPS units can be set up and get working, since the team found that only the base station needs to have a GPS location given to it at startup. In an area filled with notable landmarks such as Institute Park, the team was able to give the dGPS an absolute location from Google Maps rather than wait for it to find itself through the survey-in process. At the end of both processes there would be absolute accuracy within a meter, but the survey-in process takes significantly longer.

3.3.4.3 Testing dGPS Accuracy with Three Modules

During this first test the team found that the biggest inconsistencies in our data came from marking the map on Google Earth rather than getting exact measurements while there in person. To combat this, the team decided to do a second test with two dGPS units representing the drone and the rover and one dGPS unit representing the base station. In this test the three modules were set up next to each other to act as the robots being configured at the base. Then the team first sent the "rover" module out to the field to a predetermined location and recorded the dGPS relative position seen in Table 2. The team then returned that module to home base and sent the "drone" module out to the same physical location. Once this module found its fixed location, the team recorded that information and then compared the results. This test, produced the following results:

Table 2 Data from Test 2 of the Differential GPS Units

		Latitude (º)	Longitude (º)	Error (cm)
Test 1	Board 1	-71.80475150	42.27551483	1.7360649
	Board 2	-71.80475150	42.27551533	
Test 2	Board 1	-71.80475150	42.27551500	0.5902621
	Board 2	-71.80475150	42.27551517	

This data shows us that differential GPS can be extremely accurate when used to determine the relative position between two points. One thing the team noticed from this test was the difference in the time it took for the dGPS modules to determine their fixed location. When there was one “rover” module and one “base” module, the swap from determining the location to homing in on its fixed location happened in less than a minute and once it happened, it stayed fixed pretty consistently while it was moved. Once there were two dGPS modules in the field, it took each of them an extra minute or so to home in on their fixed location. The team believes this is because both modules are trying to take the same data from the same base module, meaning that they are competing for a reliable signal in some way. With this test the team also realized that if the system-level application was set up in such a way, there should only ever be one dGPS module on in addition to the base module, meaning it would take even less time for the fixed location sync to occur.

3.3.5 Testing Equipment

3.3.5.1 Test Landmine

A mock landmine was created by the previous demining drone MQP to serve their testing needs, however this device had no documentation and was missing components, so the interior electronics were replaced with a new strain gauge assembly and the internal structure was modified for increased precision and reliability. This setup implements a 10 kg rated parallel beam load cell with a full Wheatstone bridge powered and monitored by an Arduino Micro.

The Arduino is programmed to log the force value and corresponding timestamp of all forces over the activation force of a PNM-1 mine, group those readings by time, and calculate the total impulse from a group of forces. This reading is logged and output over serial by sending ‘t’ over serial to the Arduino. The calibration curve is shown in Figure 31.

The landmine’s calibration shows that it becomes less sensitive at weights above 2.2kg, and as a result the calibration curve for values above that point flattens out. A possible reason for this is that the internals of the landmine may begin flexing at this weight.

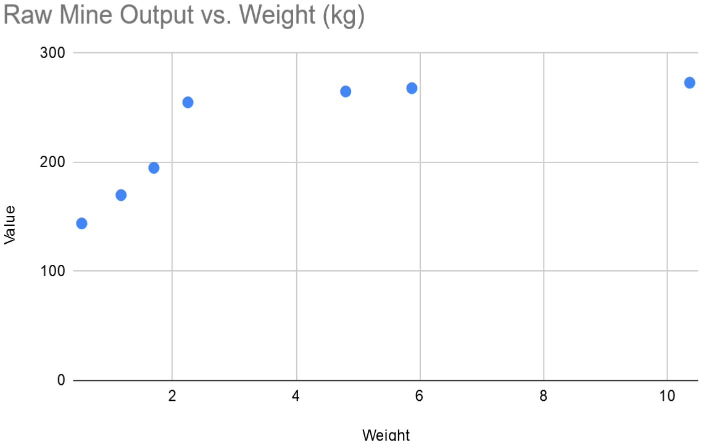


Figure 31 Graph of Test Landmine AnalogRead() Output vs Weight Placed on Mine

3.3.5.2 Payload targeting/delivering

The payload design for detonating our model PNM-1 landmine has three basic requirements: it must create a force greater to 50N to overcome the force of the internal spring, generate a total impulse greater than 50 Ns, and it must fall in a predictable trajectory with minimal deflection due to air and weather conditions.

Due to laws regarding drones and outfitting them with weapons of any kind, the payload must be delivered as a falling weight, rather than dropping a small explosive or shooting at the mine's location from above with a device such as a potato cannon.

The falling time, force, and impact of a falling object is given by the following equations:

$$falling\ time\ (t_f) = \sqrt{\frac{2 * height}{g}}$$

$$total\ velocity\ change\ (v_f) = t_f * g + v_{rb}$$

$$average\ force\ (F) = mass * (g + t_{stop} * v_f)$$

$$total\ impulse\ (J) = F * t_{stop}$$

't_{stop}' Represents the Time for the Payload to Stop and Rebound Once Hitting the Ground
'v_{rb}' Represents the Change in Velocity Due to Rebound
'g' is the Acceleration Due to Gravity

Two unknown factors play into the total impulse generated by our payloads. The time that the payload takes to come to a full stop as it hits the ground and the amount of velocity generated on rebound are variable, depending on how soft the ground is and the material of payload in use.

As a result of these variables, the team looked to explore the requirements of creating an impact that would be sufficient to detonate a mine. This was done using a MATLAB script to plot the expected impulse for different a range of payload height and weight values with an easily modifiable variable for the time it takes for the payload to come to a stop. This script is viewable in its entirety in Appendix G and the resulting graph can be seen in Figure 32.

The graph plots a range of payload weights and release heights, and aids in finding a balance of weight vs height. The S1000 has significant lifting ability, and as a result it is feasible to carry six separate payloads up to 1kg each on every deployment. Increasing the mass allows us to drop the payload at a lower altitude, increasing the accuracy of the trajectory. Drop height brings the drone closer to the mine's detonation as well, so there is a lower limit on this value so that the drone has adequate time to evade the blast.

The script does not fully model the typical waveform of an impact and simply calculates average force and impulse to generate its results. It also does not check to ensure that all impulse calculated was generated at a force at or above 50 N, as the forces generated tend to be very large (average of 200+ N) for a very short window of time, leaving little room for sub-50 N forces to substantially affect the final impulse value.

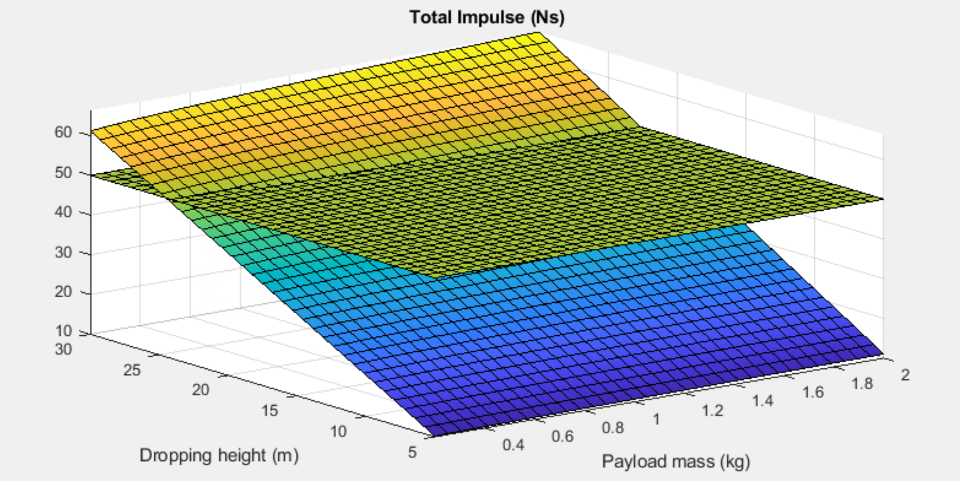


Figure 32 Graph Comparing Effect on Mass and Drop Height on 50 Ns Impulse Generation

The script also substantially overestimates the required weight and height values. More specifically, the script provides baseline impulse values for an impact with absolutely no rebound effects, as these are not able to be calculated directly and may vary significantly across soil types. A rebound effect causes a much larger change in momentum, and thus a larger impulse. This factor proved to be very significant, as our small-scale tests with our model landmine showed greater-than 50 Ns impulse at far lower values (6 m and 0.23 kg) than what is suggested by the script. This type of error was acceptable, given that that the Demining Autonomous System in its completed form would want to have a guaranteed detonation regardless of any soil or payload dynamics.

This graph shows that, even though the drone is capable of carrying a significant amount of weight, the drop height is the most important variable in the total impulse generated. Increasing drop height reduces accuracy due to an increase in time for air currents to affect the payload's trajectory, but increasing the height gives us a far more significant impulse increase than what is possible through altering the payload weight.

Table 3 shows the results of preliminary manual drop tests. These tests show that, based on the test mine data, a 20 ft drop height is more than enough to activate the mine.

Table 3 Test Landmine Drop Test Results

Dropping Height	Test Landmine Activated?	Water Balloon Broken?
8ft	No	No
10 ft	No	No
12 ft	Yes	No
14 ft	Yes	Yes

16 ft	Yes	Yes

Chapter 4: Results

4.1 Rover

4.1.1 Mechanical Stability Testing

One of the goals of the project was to stiffen up the sensor arm assembly. The previous year’s implementation had quite a bit of slop in real life – as the rover drove on uneven ground, it would wobble. This mainly came from the materials selection and joint design. The complex, 3D printed assembly was naturally flexible as it was plastic, and the joints for rotating the spray paint can in place of the magnetic pickup coil contributed to the problem. The new, simplified design replaced this with a carbon fiber composite tube. Carbon fiber is extremely stiff and lightweight. The team made this selection based on simple modulus of elasticity charts – carbon fiber has one of the highest stiffness ratings of non-metal materials, and the team’s basic simulations in SOLIDWORKS support this decision. Unfortunately, the simulation of the 3D printed arm is not fully accurate due to it only considering material stresses - it assumes the arm is homogenous. The carbon tube in this simulation flexes only 1mm, compared to the 3D printed assembly’s 8mm of deflection. The deflection of these the 2018-2019 arm compared to the 2019-2020 arm can be seen in Figure 33 and Figure 34.

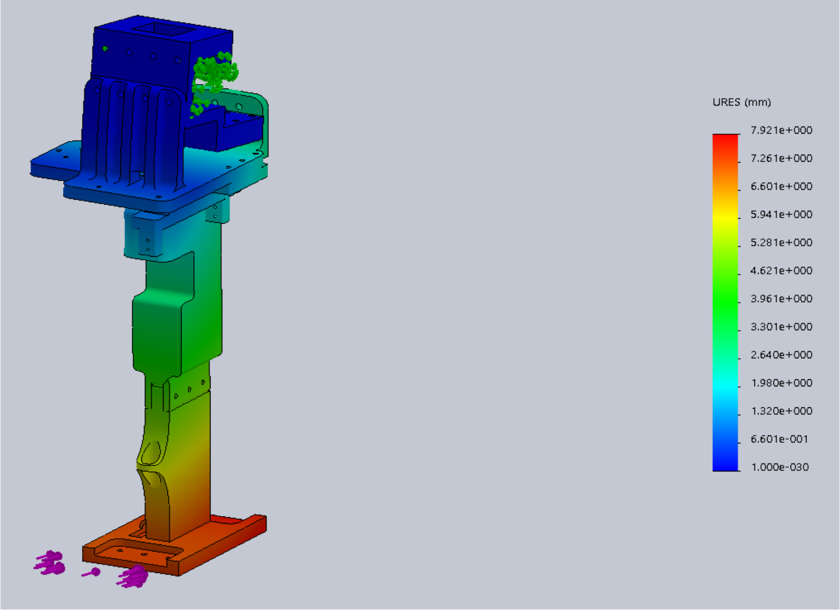


Figure 33 Deflection Analysis of 2018-2019 Rover Arm

The team also solved the flexing in the round, end-supported linear rails of the previous design by incorporating a flat plate with square linear rails connected by bolted joints every few inches. These rails are used in precision machine tools and have near zero-clearance between rolling elements, rail and carriage blocks. This results in stiff, smooth, and precise motion. As it is supported along the length, there is no bending as seen in the previous design.

Finally, the team's new implementation of the four bar with steel links and Heim joints along with screw-powered linear actuations displayed more rigidity than the previous design. One solved problem was there was no preload downwards aside from gravity on the previous design – the linear actuators constrain up and down motion so there is no unintended movement aside from backlash in the screw and nut, and material flex. One side effect of the under-constrained design is that the slider plate was seen to rotate around its center slightly, possibly due to the aforementioned backlash in the linear actuators, or some element in the assembly bending slightly.

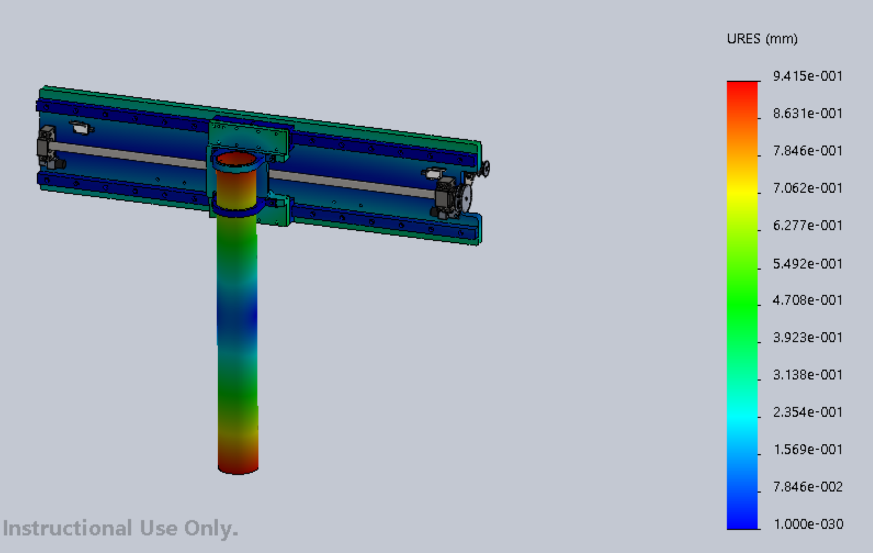


Figure 34 Deflection Analysis of the 2019-2020 Rover Arm

4.1.2 Electrical Rework Capabilities/Simplification

In order to make the rover closer to a finished product and add an increase in professionalism to the system as a whole, the team needed to make substantial improvements to the electrical system. For that to occur, there were several areas of the electrical system of the rover that were modified and upgraded around the rover.

The board structure on the rover at the end of the previous MQP was full of redundant and complicated boards in an arrangement that was difficult to follow. The team had to make it so that another person could go in and understand what was happening within the electronics box, which meant consolidating the structure and reorganizing the appearance. This was accomplished by routing all the wires through a single Teensy board. The Teensy has many more interrupt-capable pins, meaning that all sensors that operated using interrupts (like the encoders) could be wired through one board. The only remaining part of the previous set up is the Arduino Uno that still exists to parse and pass along the data from the metal detector on the front of the rover. This means that everything could be neatly stored within the new Nanuk Case that was purchased and mounted on the rear of the rover.

On the topic of physical structure, the new case was mounted using shock mounts, with the intention of mounting a separate board structure unit on additional shock mounts within it to harbor all the electronics in a specific orientation and order. The goal of this procedure was to make the Rover more resilient to weather and terrain related complications such as the rain or particularly bumpy fields. To further improve the durability of the system, the team decided to use weatherproof connector ports into the case (this preserves the “waterproof” integrity of the Nanuk case). Using these connectors also ensured that there was an anchor point for all wires before they enter the case. This means that none of the wires inside the case – said wiring all connected to fragile boards and more delicate solder points - would ever be in tension. In addition, it makes it easier to group the wires together from the rover and more clearly label where everything is coming from/ going to.

In an overall effort for all of this to build on the professional appearance of the robot, the team developed a system of labeling, along with a more subtle wire organization strategy. Though both were begun, neither were able to be completely implemented due to the COVID-19 pandemic. However, the intention of the labelling system was to have every wire have a label on it somewhere neatly stored out of eyesight, but easily accessible. These labels included the location of the wire, the purpose of the wire, and the current/voltage draw when applicable. This is very helpful in a system that is open to modifications since it allows the user to successfully reassemble anything that they might need to move around. In addition to that, the team created documentation to go with the wiring, so that it can be referenced without delving into the electronics of the rover, should anyone want to understand the intricacies of the electrical system without having to disassemble anything.

4.1.3 Testing Against Acceptance Criteria

The team’s original goals for this project were outlined in the statement of work that was developed at the start of the project. While these goals changed slightly as the project developed, the base goals are comparable. Below are the base goals outlined within that statement of work for the rover.

1. Redesign the lifting mechanism so that it will raise or lower the mine-detecting sensor and deflects less than ¼” in the x-y plane when pressed with less than 5lb of force at the tooltip/end effector

   The final design only had 1mm of deflection in the x-y axis when pressed with a 5lb force as demonstrated in 4.1.1 Mechanical Stability Testing.

2. Get the onboard LIDAR to work such that it can generate point cloud data for obstacle avoidance and be utilized in the Rover obstacle avoidance system which will be able to avoid trees and rocks

   Due to the COVID-19 pandemic, the team was not able to get new sensors on the rover to the point where they could be tested as part of the system. The team did outline a framework for how the LIDAR could be implemented into the existing system.

3. Replace the onboard router and computer system and remake the electronics box so that is made from a more robust material, constructed of a water-proof (pelican box style) box, all outside connections made with waterproof style connectors, has well-organized,

labeled, accessible contents, and the wiring within the rover is labeled so it is easy to understand

The electronics box was updated from the previous wooden box to a pelican-style waterproof container that had waterproof connectors going into it. While the team was not able to complete the wiring inside the box due to the COVID-19 pandemic, the wires on the front of the rover were labeled and carefully organized going into the box. A wiring schematic was made to help further explain how each system is connected and extensive wiring plans were made for the future project team to make it easier for them to complete the wiring of the rover.

4. Get the ultrasonic sensors to work on grass while preventing the mine detection sensor from colliding with objects < 6 inches in height (objects the rover will traverse, not avoid)

   Testing was done with the ultrasonic sensors to develop a filter that would be able to reduce the noise caused by grass when working in the field. This filter was then applied to the ultrasonic sensors, which allowed them to detect the ground, regardless of the height of the grass in the area.

4.2 Drone

4.2.1 Autonomous Field Testing

4.2.1.1 Manual Flight Testing

The earliest tests of the drone were simple manual flights performed in September 2019. These involved no autonomy, with all control being provided by a human pilot. These tests were deemed necessary for two reasons:

1. To verify the flightworthiness of the aging aircraft
2. To train team members to fly the aircraft “by hand” so that they could serve as safety pilots for later autonomous testing

The learning curve was steep, and it took a few weeks of research and experimentation to get the system functioning as desired. Eventually, the aircraft could be armed, disarmed, and controlled via remote. The drone was taken to an open area, and the first flights were performed. These tests were very simple and lasted only a few seconds each. They consisted of a takeoff, a brief hover, and a controlled landing. The success of these tests proved that the system would provide a good base for further development.

4.2.1.2 Early Autonomous Flights

To evaluate the abilities of the Pixhawk flight controller and its ArduCopter firmware, more tests were held in late October 2019 to experiment with autonomous flight. These tests used the Pixhawk’s internal memory to store pre-planned flight paths. Though no objective results were sought, these would be an important proof-of-concept for the progress of the project. Takeoff-and-landing, and point-to-point navigation tasks were loaded to the flight controller. By flipping a switch on the remote control, the safety pilot could trigger the start of the "mission". These tests were very successful, and proved the drone's ability to takeoff, land, and navigate autonomously using ArduCopter's built-in routines.

4.2.1.3 Autonomous Flights with Updated Architecture

By this point in the project (early February 2020), the new Teensy-based control architecture had been fully implemented and tested as much as was possible indoors. Though the drone had untested software for the entire demining routine, the first test was a simple takeoff and landing to prove the new design. When given the "takeoff" signal from the base station computer, the drone autonomously took off, and reached its target altitude. At that point, rather than landing as intended, it maintained altitude and began to spiral outwards, before flying off in a straight line. After a few seconds, the safely pilot manually took control of the aircraft and landed it uneventfully.

By default, ArduCopter keeps very detailed flight logs. These were scoured to find a reason behind the errant flight. Eventually, it was determined that the Pixhawk's compass was not appropriately calibrated. A functioning compass is vital for maintaining position, as the system must know its heading in order to correct for the global errors reported by the GPS. With incorrect heading data, the position controller can quickly diverge, as observed in this test. In previous flights, an external compass had been used. This module was built into the standard GPS receiver, which was removed when the differential GPS was added. Though the Pixhawk has a compass in its internal inertial measurement unit (IMU), it was not calibrated correctly. Later attempts at calibrating this compass failed, which lead to the decision to purchase an updated Pixhawk 4 with better sensors and a new external compass module.

4.2.1.4 Autonomous Flights with Updated Flight Controller

By late February 2020, the new flight controller had been set up and fitted to the drone. After verifying the new configuration as much as possible indoors, the drone was taken to an open area for flight testing. To verify the functionality of the hardware, a manual flight was performed first. A quick and uneventful takeoff, hover, and landing showed that the system was performing as expected. For the next flight, the drone's Teensy microcontroller was programmed to takeoff, hover at the target altitude, then land autonomously. No human input would be necessary other than a start command from the base-station computer. After an unstable takeoff, the drone reached the target altitude and hovered indefinitely. A "return-to-launch" command was sent by the safety pilot, and the drone landed itself successfully.

After a brief investigation, a bug was found in the code which prohibited the system from automatically entering its landing sequence. As for the unstable takeoff, the result was never fully determined. Badly tuned PID loops were suspected by the team, but a reduction in the corresponding P gain value did not fix the issue in later tests. This problem should be investigated in the future to ensure smooth takeoffs.

After fixing the bug in the code, another takeoff-and-land test was performed. This test went as planned—apart from the remaining unstable-takeoff issue—confirming the validity of the bug-fix.

With basic functionality mostly working as expected, an additional test was planned. The base station computer was given the coordinates of two hypothetical mines near the drone's takeoff point. When triggered, the base station planned a flight path, transferred this wirelessly to the drone, and sent a takeoff command which would be relayed to the flight controller by the Teensy. The drone took off, flew to each of the provided locations, returned to the takeoff point, and landed successfully. This test was viewed as a resounding success for the system—all necessary functionality was working, except for the rough takeoffs. The only remaining test would be to run the system with a larger number of mines to simulate the need for a "reload" of payloads. For lack of time and concern over the takeoff issue, this test was planned for another day.

4.2.1.5 Final Flight Test—Updated PID Tuning

A few days later, another basic takeoff-and-land test was performed. With this test the team was attempting to improve the stability of the drone on takeoff. The ArduCopter firmware includes a distinct PID controller for positional (horizontal) error. The proportional ("P") gain was reduced with the hope of decreasing the urgency of the drone as it attempted to correct its position on takeoff. Upon takeoff, the drone performed very similarly to previous tests. After reaching target altitude, the drone veered to one side and became unstable. The drone then rolled to one side and fell to the ground. Upon recovery, the drone's main carbon fiber frame had been sheared off in multiple places, and several less-critical components had been damaged. The power-distribution circuit board had been damaged, and the power cable for all 5V and 3.3V electronics had been pulled from its crimp connector.

Initially, the modified PID gains were suspected as the cause of the crash. However, examination of the ArduCopter logs showed that the Pixhawk had lost power soon after reaching the target altitude. It is presumed that the disconnected power cable was a result of the violent takeoff rather than the crash, and that the resulting loss of power—which would have shut down the Pixhawk, Teensy, wireless link, and GPS module—led to the crash. Because the motors' electronic speed controllers are connected directly to the battery, the motors presumably kept spinning, which would explain why the drone drifted sideways and lost stability, rather than simply falling like a stone.

Some of the replacement parts required to fix the drone could not be obtained due to general unavailability as well as complications caused by COVID-19. The team made plans to manufacture the necessary parts, but this was rendered impossible by an online-only D term.

4.2.2 Electrical Systems Update

One major portion of the drone upgrades was the integration of the differential GPS and updating the onboard sensors to the newest models. Figure 35 shows what the electrical system looked like at the start of our project, which highlights the accomplishments of the 2015-2016 team.

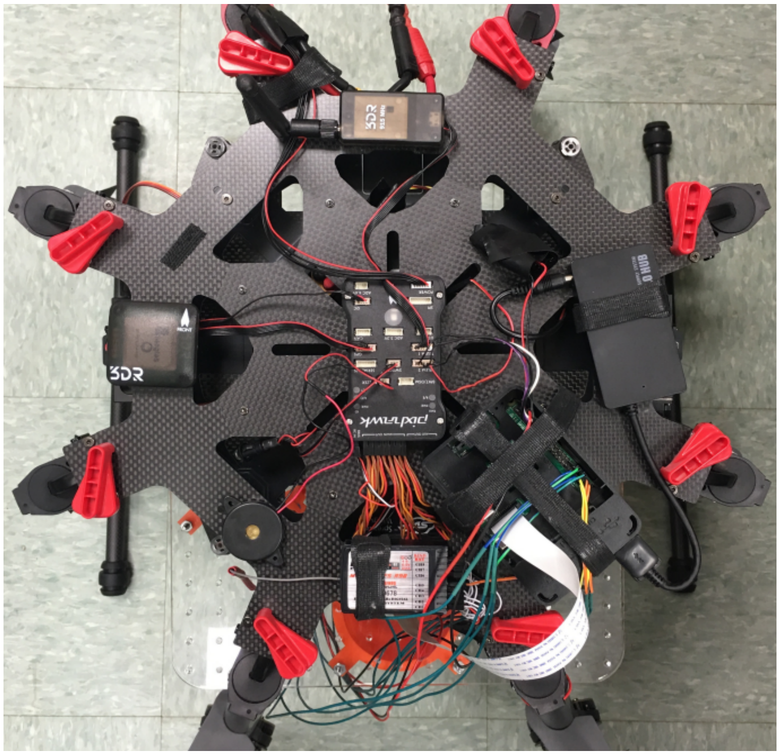


Figure 35 Drone Electrical System From 2015-2016 MQP

From their work, the team replaced their Pixhawk (black box at the center of the drone) with the newest version and replaced their GPS module (black square on the left) with a differential GPS module and external compass. As the team updated the software on board, they realized it made sense to replace the existing Raspberry Pi with a Teensy microcontroller that could control the dGPS data and communication with the base station. Finally, the team worked to reorganize the voltage regulation throughout the drone to limit the number of excess modules onboard. These changes were first mapped out, as seen in Figure 36 to ensure the team had all the necessary components running at the correct voltage.

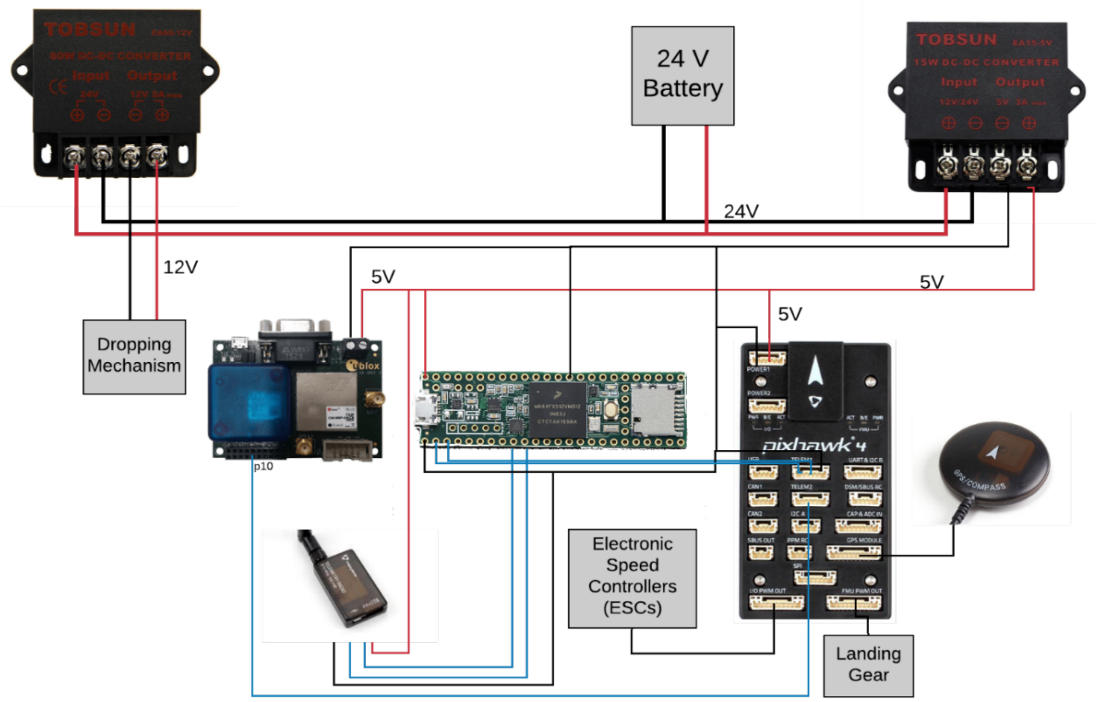


Figure 36 Proposed Electrical System for 2019-2020 MQP

After completing a system walkthrough and ensuring all modules would be able to run at the voltages they needed, the team redesigned the drone's electrical system in accordance with the plans they made in the schematic. They also added labels to each wire coming from the drone to ensure it was easier to understand. These changes can be fully seen in Figure 37.

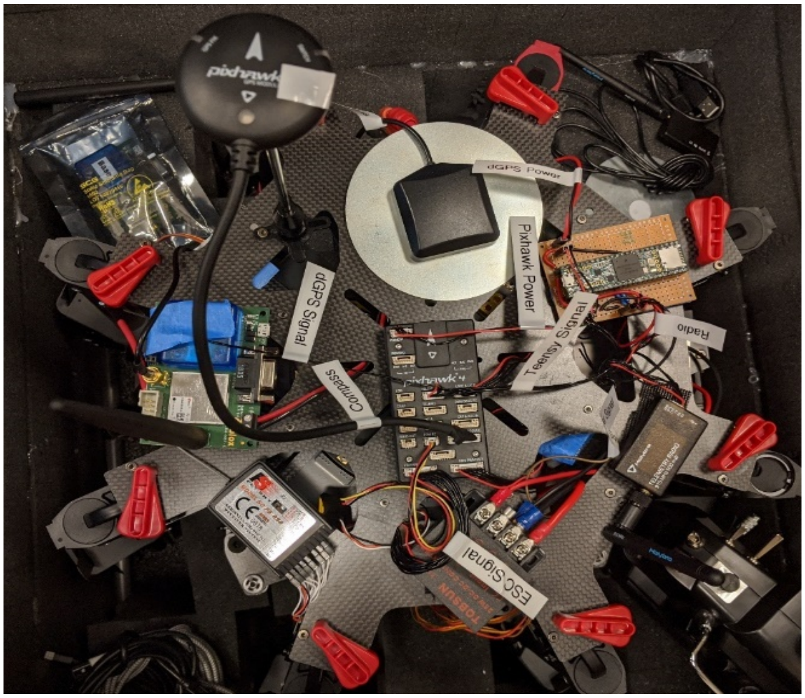


Figure 37 Electrical System from the 2019-2020 MQP

Our Teensy sat on a prototyping circuit board (seen in orange on the right) that organized all the wires coming to and from the Teensy and secured them to the drone so nothing would come loose during flight. Future iterations of this project could investigate creating a custom PCB for the drone that is modeled after our prototyped circuit board.

4.2.3 Testing Against Previous MQP

While the team could not complete a full system test, it is still possible to compare portions of our results with that of the previous MQP. The first area to test is pre-drop accuracy, which is when the drone is given a GPS location of a landmine and the drone is hovering over that location, but before the drone drops a projectile. The 2015-2016 MQP used GPS and a camera to achieve an accuracy of +/- 0.5 meters from the visual marker on the ground, which was often seen as a light circular motion around the given point [13]. In contrast, the 2019-2020 MQP used differential GPS to achieve +/- 0.02 meters from the given GPS point. While the 2019-2020 MQP team could not test the accuracy of dropping from the dropping mechanism, it is still possible to compare the abilities of the previous dropping mechanism to the current mechanism. Both mechanisms were designed to hold ~0.5lb payloads that would be able to detonate a landmine from 6 meters, but the 2019-2020 mechanism was designed to hold 6 payloads with a single dropping point whereas the 2015-2016 mechanism was designed to hold only 4 payloads with a unique dropping point for each payload. Figure 38 shows the one of the four payload bays of the dropping mechanism designed by the 2015-2016 MQP team. One of the major flaws of their design was the fact that the solenoid used to open the door on the payload bay would stick out into the bay, causing payloads to often get stuck and tear on the metal end of the solenoid [13]. In the 2019-2020 dropping mechanism design, the team worked to eliminate this problem as well as simplify the dGPS calculations by having a single payload drop point and a rotating barrel to supply payloads to the drop location. To limit the concern for tearing payloads seen in the previous MQP, this team added curved edges to the flaps that rotate the payloads, resulting in

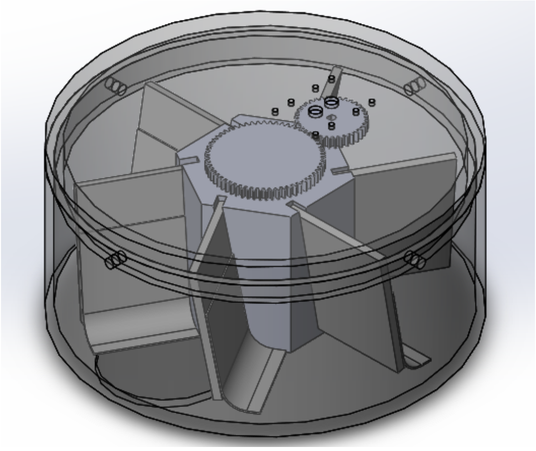


Figure 39 Dropping Mechanism, 2019-2020

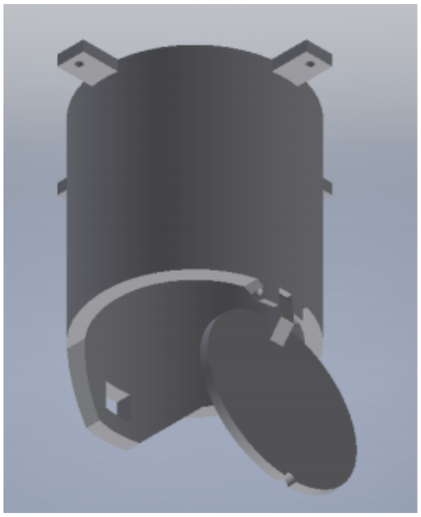


Figure 38 Dropping Mechanism, 2015-2016

payloads rolling through the barrel rather than sliding, which caused unnecessary friction with the outside of the payload. This new design can be seen in Figure 39.

4.2.4 Testing Against Acceptance Criteria

The team's original goals for this project were outlined in the statement of work the team developed at the start of the project. While these goals changed slightly as their work developed, the work done in the 2019-2020 MQP can be shown against these criteria. Below are the base goals outlined within that statement of work.

1. Get the drone to identify the marker from 20 feet from the ground, with at least 75% consistency. (Stretch goal: 90% consistency)

   The team removed the camera feature implemented by the 2015-2016 MQP at the beginning of B term when they determined differential GPS would be accurate enough on its own. To this degree, the drone is still able to find a mine at 20 feet above the ground, so this goal was completed.

2. Develop a payload deployment system that can hold four 3-pound sandbags and deploy them one at a time over a marked location

   The team redesigned the mechanism to hold six payloads, which is more than they originally accounted for. Also, they redesigned the payloads to be water balloons and found that a 0.5 lb water balloon dropped from 20 feet was enough to detonate the mine.

3. Drop payload within 15 inches of the actual marker (Stretch goal: within 5 inches of the actual marker)

   While the team was unable to test the full system to determine the true system accuracy, they were still able to test the pre-drop accuracy which was 0.79 inches (2 cm). The true accuracy will be something slightly higher than this value, but the pre-drop accuracy shows promise for being able to have a true system accuracy of less than 15 inches.

4.3 Base Station

4.3.1 Final User Interface Design

The final implementation of the user interface application was achieved by replicating the concept that was developed in AdobeXD in JavaFX. Figure 40, Figure 41, Figure 42, and Figure 43 show various pages of the completed application. This design emphasizes the steps of the system operation, first providing the user with calibration and setup information for both the rover and the drone, then requiring them to confirm they have followed the correct setup steps. Then, the user is able to select minefield boundaries. The Rover and Drone each have a deployment page, a mission page that can display progress updates, and a review page to display the number and location of each mine.

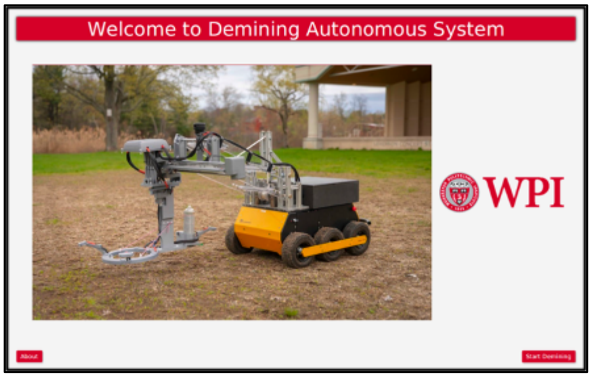
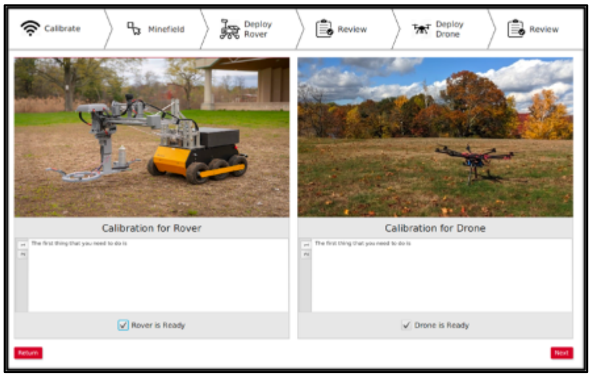


Figure 40 User Interface Welcome Page



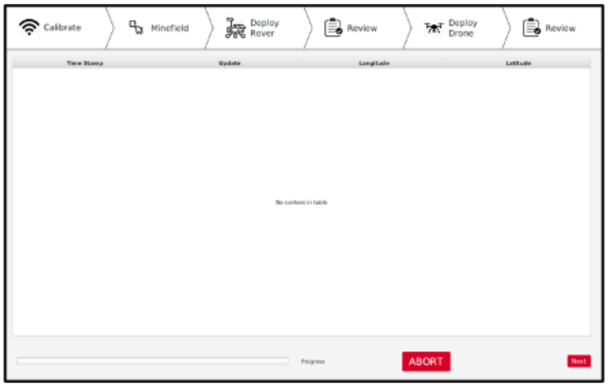
The first thing that you need to do is

☑ Rover is Ready

The first thing that you need to do is

✓ Drone is Ready

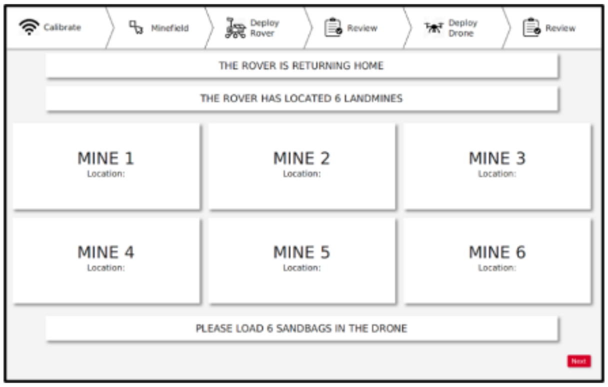
Figure 41 User Interface Calibration Page



Time Stamp	Update	Longitude	Latitude

No content in table

Figure 42 User Interface Mission Page



THE ROVER IS RETURNING HOME

THE ROVER HAS LOCATED 6 LANDMINES

MINE 1
Location:

MINE 2
Location:

MINE 3
Location:

MINE 4
Location:

MINE 5
Location:

MINE 6
Location:

PLEASE LOAD 6 SANDBAGS IN THE DRONE

Figure 43 User Interface Rover Review Page

Due to the limitations of D-Term which prevented the full integration and development of a back-end system architecture for the User Interface, the work of the 2020 Team focused on creating a strong basis for future MQP teams. The original class-diagram, included in the Methodology section, was revised as shown in Figure 44. The classes shown in green were implemented by the 2020 MQP Team. The classes remaining in white are system integration classes that will need to be implemented at a later date based on the 2020 System Architecture Plan.

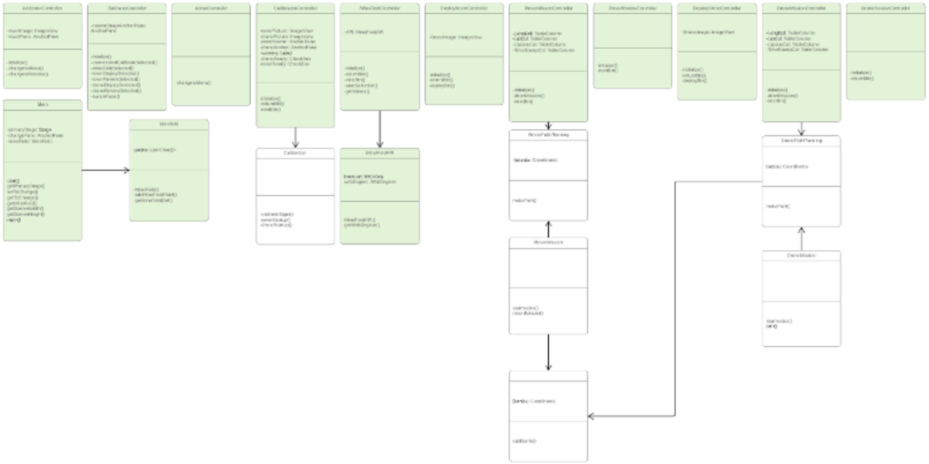
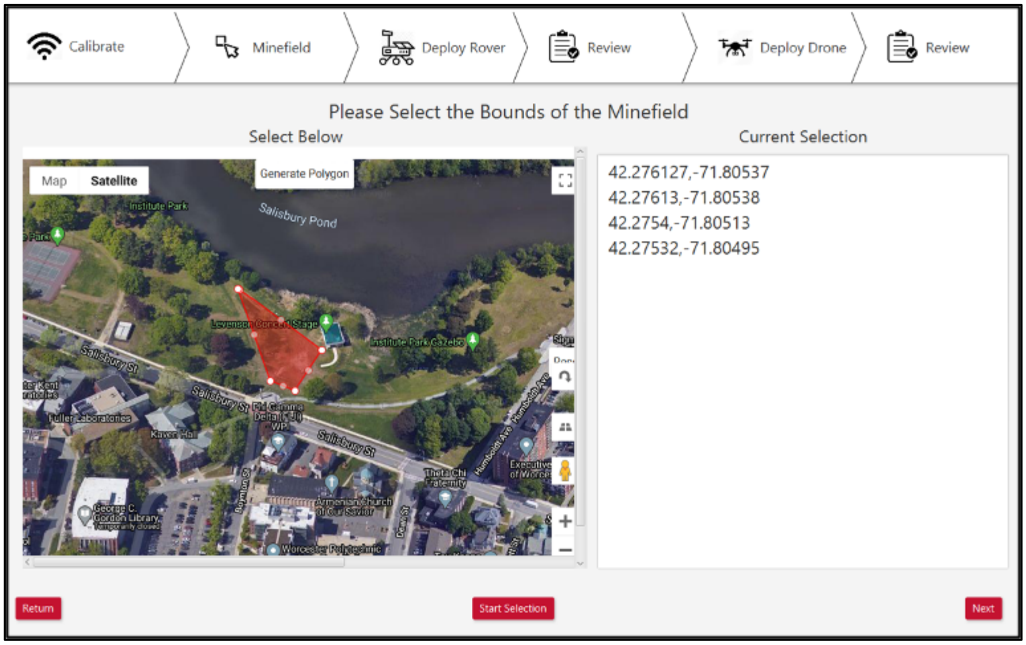


Figure 44 Final User Interface Class Diagram

A major portion of the user interface design was its integration with a Google Maps API that was designed by the 2019 MQP team. The API was given its own class in the User Interface's Java program and was loaded directly into the JavaFX Scene. This is shown in Figure 45. A *MineField* class developed in the user interface stores the minefield boundaries generated in the API for later use. This integration allows the user to more easily visualize the minefield and removes a large technical barrier of working with the demining system. Due to the COVID-19 pandemic, the team was unable to verify the accuracy of working with the embedded API in the application. Future teams should verify the transmission of accurate information from the API to the user interface and to the Rover.



Please Select the Bounds of the Minefield

Select Below

Current Selection

42.276127,-71.80537
42.27613,-71.80538
42.2754,-71.80513
42.27532,-71.80495

Figure 45 Google Maps API Integration

4.3.2 Base Station System Architecture

At the completion of the project, several of the constituent software components of the base station existed, though they had not been integrated. In addition to the user interface, a C++ program had been written to communicate with and control the drone. Additionally, this code performs path planning for the drone. This program was fully functional but would require modification to be integrated with the full system. Namely, the code would need to be restructured into a C++ ROS node such that it can accept commands from the user interface and relay them to the drone via the wireless link.

In addition to the UI and drone communication ROS node, the base station will require a ROS node to coordinate the transfer of suspected mine locations from the rover to the drone via the base station. Presumably, these coordinates would be passed from the rover to the base station by ROS messages as they are discovered. They would be stored there while the rover operates, then used to plan the drone's path once the rover has cleared the field. This flow of information would secure any suspected mine locations, even if the rover is compromised.

Chapter 5: Conclusion/Discussion

5.1 Rover Discussion

5.1.1 Husky Drivetrain Limitations

The most significant drawback to the Clearpath Husky A100 platform was the underperforming drivetrain. The team initially had high hopes as it looked like a capable drivetrain, with six large pneumatic wheels, but after the test drive in Institute Park, the team discovered the hidden problem. Our proposed solution was to integrate higher power motors into the system, which could have theoretically been a drop-in replacement and the ROS layer would not notice the difference. Professors Putnam and Bertozzi brought to our attention during C term the availability of a heavy-duty drivetrain with tank treads and a much higher payload capacity. This drivetrain was advertised by the professors as having no traction and power issues like the Husky but raised concerns about its increased weight and possible higher ground pressure. As a large portion of work had been done with the intention of using the Husky, the team declined to shift to the new platform. However, there is certainly a possibility of a future team using components designed for the Husky and retrofitting them for this new drivetrain. The question for that team would be the amount of work required to come up with an entirely new design, versus finishing the leftover work of upgrading the Husky drivetrain and systems integration.

5.2 Drone Discussion

5.2.1 Sourcing of Parts

Since the drone the team used was initially released in January 2014 and that model of DJI drone is no longer stocked, the team has struggled to find replacement parts as the drone gets damaged. The team recommends the purchase of a new drone or the assembly of a custom drone with easy to source parts. If future teams are looking to purchase a new drone, the one the current team recommends at this point is the DJI Matrice 600 Pro (via DJI website) or the newest model of professional grade drone from DJI. Before purchasing a new drone, ensure that there are multiple ways to source parts as they break. Outside of the manufacturer's website, the team found eBay and hobby shops to be an ideal way to find reliable parts, but only if the parts are still being manufactured. Once the original drone leaves production, it becomes extremely difficult to source custom parts for any drone. Another option for a drone could be to leverage a drone design similar to the design of the SnowDrone MQP done this year. While that MQP will be continued in future years, the new Demining MQP team could reference their CAD and calculations to build a similar drone for the project's purposes

5.2.2 World Events Impact on Project Work

The impact of COVID-19 and other world events extended far beyond the closure of campus for D term. As early as October, the team was struggling to source parts from China due to the ongoing trade wars between the US and China. This was a heightened issue for the team since DJI is a Chinese company and all official replacement parts and sensors associated with the DJI S1000 are produced exclusively in China. Once the team started flight tests for the drone, they needed more replacement parts for minor damages during testing. This became increasingly difficult as the cost of parts and shipping increased, stressing their preset MQP budget. As they moved into January, the timeline for receiving shipments from China increased dramatically as a result of COVID-19. For any given part, it could take upwards of 2-3 weeks for them to receive orders. At the end of February, the team went to perform a flight test and had a major crash with the drone, resulting in major damages to the main frame of the drone. In order to repair it, the team needed to order a new center frame and arm bracket, but with COVID-19 in full swing in China, the costs were much higher than expected and the shipping times were at almost 4 weeks, with a warning that shipments may take even longer than that. Losing a month of progress during 7-week terms is almost impossible, so the team investigated making these parts ourselves. Before they could get to making these parts themselves, the school closed its doors to students and moved all classes online. As a result, the drone remains disassembled with a broken center plate, with no replacement parts ordered. Since the DJI S1000 is already an older model, the parts were difficult to source without the events the team saw during their year on the project, so a future team may find it nearly impossible to source these parts once they begin.

5.3 Future Work for the MQP

5.3.1 Multi-Rover System

In the current system, the critical task of the mission is the rover traversing the entire minefield in search of landmines. The time allotted to the rover's minefield traversal is the majority of the time in a mission. To expedite this process, a future MQP could experiment with adding additional rovers that can search the minefield together. Ideally the rovers would know the location of each other, and they could all be monitored from the base station. When the rovers would path plan, they would be able to develop the most efficient path through the field that allows them to all work together to cover the field in the least amount of time while minimizing the amount of overlap in the areas they cover. By adding additional rovers, the entire mission time can be reduced dramatically.

5.3.2 Drone Navigation and Flight

5.3.2.1 Active Altitude Control

In its current state, the drone's altitude estimate—as maintained by the Pixhawk flight controller—is measured relative to the takeoff point. The altitude used by the flight code on the

Teensy board is exactly this. Inherent in this design is an assumption that the ground will always be flat within the minefield. If the ground slopes up from the takeoff point, then the drone may be too low for the dropped payload to set off the mine. In the opposite case, the drone may unwittingly be too high for the payload to hit its mark accurately. Ideally, the drone should maintain a constant altitude *relative to the ground directly below it.* Some type of ranging sensor may be necessary, though this may be tricky to implement as the drone pitches and rolls.

5.3.2.2 Obstacle Avoidance

While obstacle avoidance is relatively straightforward in the assumed-two-dimensional space of the rover, it is a much larger task in the context of the drone. The current platform assumes that the minefield contains no trees or other obstacles near the operating altitude. In the real world, this may be a very constraining assumption. The system could be improved greatly by the addition of the ability to actively avoid unknown obstacles during flight. This problem is large enough that it may be worthy of a separate project all-together. This would likely involve some type of 3D scanning (such as lidar or stereo cameras) in the direction of flight.

5.3.3 Mine Detonation Verification

With the current sensor combination, a mine could be safely detonated, but there is nothing to confirm that the dropped payload actually detonated the landmine. To ensure that the landmine was detonated, future project work could include adding a microphone to the drone that would listen for the explosion of the landmine (which is just a large soundwave). With the addition of the microphone, the software loop could be improved on to add a verification step where, if the drone did not hear the explosion, it would fly back and drop another payload on a landmine to ensure that all of the mines are actually detonated. It would be difficult to test this feature without having a live minefield, since it would be difficult to properly test this microphone feature in the team's previous testing field (Institute Park) due to concerns from the residents living in the area.

5.4 Conclusion

Right now, there are 110 million anti-personnel landmines buried around the globe that are making land unavailable, hindering economic growth, and preventing the free movement of people and goods, especially in developing areas of the world [1]. Due to the fact that current demining methods either involve bringing in military demining machines (which are expensive and ruin the surrounding lands) or involves using humans or animals to manually search for mines (which is often ineffective and dangerous), the team's goal was to develop a low-cost robotic system that would be able to safely remove landmines from a given minefield. Building on the work of previous MQP teams, the 2019-2020 Demining Autonomous System team worked to update both the rover and drone subsystems, then combine the two of them into a fully functioning system with an easy to use base station application. This system would allow for any non-technical user to easily set up and use this autonomous demining solution.

The goal at the start of this year was to fully combine the two systems to the point where they could be used without technical knowledge. While the team did not fully meet this goal due to world events such as the US/China trade war and the global COVID-19 pandemic hindering progress on the project, the team still made decent progress towards this goal. When the final term of the project shifted online, the team shifted their goals to ensure they could leave the next team in a good starting place. Since they were unable to do the final drone field tests, finish the wiring or software for the rover, or do a full system test, the team spent their online D term writing a comprehensive user guide listing how to set up and use each system and what fixes needed to be made before the first mission. They also worked to outline what needed to be written in the rover code to make it functional and wrote the portions of the code that could be tested without the physical rover. In the end, the team was able to complete most of the objectives outlined at the start of the project, and for the goals that couldn't be met due to online classes, the team outlined resources that the next team would need to complete those goals.

In the end, the rover, drone, and base station each had thorough updates. The rover was reconstructed to be more structurally sound and updated with new sensors. The drone was given a new suite of sensors, with the major swap being from standard GPS to differential GPS that allowed for the accuracy to be increased from 1 meter to 2 cm. New communication protocols were developed by the team to increase the amount of information that was shown to the base station and allowed for information from the rover to be sent to the drone, allowing both systems to know where the landmines on the field were located. On the base station, a new system application was developed to make setting up the rover and drone easier and more user friendly. This application also allowed the user to track the progress of the mission in an easy to understand format. The team is looking forward to seeing how a future team can take the work they did and move it forward to a complete autonomous system prepared to work in a live minefield.

References

1] "Facts about landmines," Minesweepers: Towards a Landmine Free World, [Online]. Available: https://www.landminefree.org/2017/index.php/support/facts-about-landmines. [Accessed 1 October 2019].

2] Geneva International Centre for Humanitarian Demining, "1. History, Summary and Conclusions of a Study of Manual Mine Clearance," in *A Study of Manual Landmine Clearance*, Geneva, Switzerland, Geneva International Centre for Humanitarian Demining, 2005.

3] "A History of Landmines," International Campaign to Ban Landmines, [Online]. Available: http://www.icbl.org/en-gb/problem/a-history-of-landmines.aspx. [Accessed 7 October 2019].

4] UNICEF, "Children and Landmines: A Deadly Legacy," [Online]. Available: https://www.unicef.org/french/protection/files/Landmines_Factsheet_04_LTR_HD.pdf. [Accessed 10 October 2019].

5] "Arguments for the Ban," International Campaign to Ban Landmines, [Online]. Available: http://www.icbl.org/en-gb/problem/arguments-for-the-ban.aspx. [Accessed 7 October 2019].

6] "Landmines," United Nations Office For Disarmament Affairs, [Online]. Available: https://www.un.org/disarmament/convarms/landmines/. [Accessed 7 October 2019].

7] M. Habib, "Mine Clearance Techniques and Technologies for Effective Humanitarian Demining," *Journal of Conventional Weapons Destruction,* vol. 6, no. 1, 2002.

8] "International Campaign to Ban Landmines," International Campaign to Ban Landmines, [Online]. Available: http://www.icbl.org/en-gb/home.aspx. [Accessed 7 October 2019].

9] International Campaign to Ban Landmines, "Landmine Monitor 2018," International Campaign to Ban Landmines, 2018.

10] H. R. Gunaratnam, S. Gunaratnam and D. Somasundaram, "The psychosocial effects of landmines in Jaffna," *Medicine, Conflict and Survival,* no. 3, pp. 223-234, 2007.

11] T. Chun, G. W. H. Lye and B. S. C. Weng, "Introduction to Mine Clearing Technology," Defence Science and Technology Agency, Singapore.

12] APOPO, "MDR Training," APOPO, [Online]. Available: https://www.apopo.org/en/what-we-do/detecting-landmines-and-explosives/how-we-do-it/mdr-training. [Accessed 9 October 2019].

13] A. Lockman, G. Tighe and T. Haydon, "De-Mining with UAVs," Worcester Polytechnic Institute, Worcester, MA, USA, 2016.

14] B. P. Wagner, D. J. Wensley, D. J. Arnold and N. J. Lanotte, "Autonomous Landmine Detection Rover," Worcester Polytechnic Institute, Worcester, MA, USA, 2019.

15] UAV Coach, "Drone Laws in Massachusetts," 2019. [Online]. Available: https://uavcoach.com/drone-laws-massachusetts/. [Accessed 9 October 2019].

16] Federal Aviation Administration, "Unmanned Aircraft Systems (UAS)," Federal Aviation Administration, 10 September 2019. [Online]. Available: https://www.faa.gov/uas/. [Accessed 9 October 2019].

17] A. U. Cookson, M. Landergan and S. T. O'Neil, "Wolfgang: An Autonomous Mobile Robot for Outdoor Navigation," Worcester Polytechnic Institute, Worcester, MA, USA, 2019.

18] D. Johnson, "Fisher Research Labs," Fisher, 2009. [Online]. Available: http://www.fisherlab.com/hobby/faqs.htm. [Accessed 8 April 2020].

19] B. Casey and T. Rocks, *Landmine Detection Rover,* Worcester, MA: WPI.

20] "Wagner Flexio 4000 Sprayer," Wagner, 2020. [Online]. Available: https://www.wagnerspraytech.com/products/paint-sprayers/flexio-4000-paint-sprayer/. [Accessed 15 December 2020].

21] "Amazon.com," 2020. [Online]. Available: https://www.amazon.com/JCT-ECO-Fluid-Spray-Extension/dp/B07X4JYG2S. [Accessed 2020].

22] colorfulemall, "2"-18" Inch Stroke Linear Actuator 1500N/330lbs Pound Max Lift 12V DC Motor," [Online]. Available: https://www.ebay.com/itm/2-18-Inch-Stroke-Linear-Actuator-1500N-330lbs-Pound-Max-Lift-12V-Volt-DC-Motor/112887896341?_trkparms=ispr%3D1&hash=item1a48a49915:m:mdfXAjchkXMgJEmUIpXpOCQ&enc=AQAEAAACQBPxNw%2BVj6nta7CKEs3N0qWKsL%2Fl%2BD7I%2BWwCSrKM2E8eBPHgZ2hh4. [Accessed 26 January 2020].

23] The Engineering ToolBox, "Young's Modulus - Tensile and Yield Strength for common Materials," [Online]. Available: https://www.engineeringtoolbox.com/young-modulus-d_417.html. [Accessed 25 March 2020].

24] I. M. Synman and T. van Dyk, "The imparted impulse of a PMN anti-personnel mine," in *National Ballistics Symposium of South Africa*, Overberg Test Range, Western Cape, South Africa, 2010.

25] I. M. Snyman, "A schematic diagram of the PMN anti-personnel mine [New Scientist]," ResearchGate, September 2010. [Online]. Available: Izak Marius Snyman. [Accessed April 2020].

Appendix A: Authorship

The writing responsibilities were divided as follows:

Section	Author(s)
Abstract	Eleanor Foltan
Chapter 1: Introduction	Jessica McKenna
Chapter 2: Background	
2.1 Humanitarian Landmine Crisis	Eleanor Foltan and Jessica McKenna
2.2 Landmines and Their Removal	Eleanor Foltan, Jessica McKenna, Joseph Niski, and Matt Schmitt
2.3 Previous Projects	Karl Ehlers, Benen ElShakhs, and Andrew VanOsten
2.4 Drone Fundamentals	Eleanor Foltan, Jessica McKenna, and Andrew VanOsten
2.5 Differential GPS	Andrew VanOsten
Chapter 3: Methodology	
3.1 Mine Detection	Karl Ehlers, Benen ElShakhs, Eleanor Foltan, Jessica McKenna, Adam Santos, Mathew Schmitt, and Andrew VanOsten
3.2 Aerial Mine Detonation	Jessica McKenna, Joseph Niski, Adam Santos, and Andrew VanOsten
3.3 System Integration	Eleanor Foltan, Jessica McKenna, Adam Santos, and Andrew VanOsten
Chapter 4: Results	
4.1 Rover	Eleanor Foltan
4.2 Drone	Jessica McKenna and Andrew VanOsten
4.3 Base Station	Eleanor Foltan and Andrew VanOsten
Chapter 5: Conclusion/Discussion	
5.1 Rover Discussion	Karl Ehlers
5.2 Drone Discussion	Jessica McKenna and Andrew VanOsten
5.3 Future Work for the MQP	Jessica McKenna and Andrew VanOsten
5.4 Conclusion	Jessica McKenna
References	
Appendices	

Appendix B: Custom Serial Protocol	Andrew VanOsten
Appendix C: Four Bar Analysis	Eleanor Foltan
Appendix D: Linear Slide Analysis	Eleanor Foltan and Benen ElShakhs
Appendix E: Slider Motor and Transmission	Benen ElShakhs
Appendix G: Payload Force and Impulse Graph Matlab Script	Adam Santos
Appendix H: DAS Media	Jessica McKenna

Appendix B: Custom Serial Protocol

This appendix describes the details of the custom serial communication protocol that was designed and used for communication between the base station and the drone. The protocol is loosely based on the concepts of the MAVLink protocol (which is used to communicate with the flight controller). This convention was designed to be as simplistic as possible, while still providing the necessary functionality.

B.1 Packet Structure

Each packet is structured as follows.

Table 4 Custom Packet Structure

Portion of packet	**Number of bytes**	**Purpose**
Constant start byte (0xFD)	1	Signals the start of a message
Message type	1	Defines the purpose of the message (detailed below)
Data length	1	Number of data bytes to expect
Data bytes	Variable: [0,128]	Message data
Parity	1	Bitwise XOR of all preceding bytes

B.2 Message Types

The following listing contains all valid message types and details their structure. The parenthetical number represents the ID number associated with each type.

HEARTBEAT (#0)

This message informs the receiver that the sender is active. It passes a single data byte, communicating the system status. This status is a member of the following enumeration.

Table 5 Heartbeat Message System States

System State	**Description**
SYS_STATE_UNINIT	System is uninitialized/state is unknown
SYS_STATE_SETUP	System is initializing
SYS_STATE_STANDBY	System is ready for flight
SYS_STATE_ACTIVE	System is armed and in flight

SYS_STATE_EMERGENCY	System is in an emergency state (i.e. aborting the mission)

STATUS (#1)

This message sends a text string, mainly used for informing the user. It contains a variable number of data bytes, up to 128.

MINEFIELD (#2)

This message provides information to the drone about the minefield in a general sense. It contains two data bytes, representing a single 16-bit unsigned integer which described the number of mines in the minefield.

Table 6 Minefield Message Byte Structure

Byte	Purpose
#1	Least significant byte of mine count (bits 0:7)
#2	Most significant byte of mine count (bits 8:15)

MINE (#3)

This message communicates specific information about a single mine. It contains 10 bytes of data, representing a 16-bit index value, and two 32-bit values representing the latitude and longitude of the mine. Lat/lon values are represented at 32-bit unsigned integers, with units of degrees*10^{-7}, that is, the decimal point of each value must be shifted 7 places to the left to achieve an accurate floating-point representation of the values.

Table 7 Mine Message Byte Structure

Byte	Purpose
#1	Bits 0:7 of mine index (mine_id)
#2	Bits 8:15 of mine index (mine_id)
#3	Bits 0:7 of latitude value
#4	Bits 8:15 of latitude value
#5	Bits 16:23 of latitude value
#6	Bits 24:31 of latitude value
#7	Bits 0:7 of longitude value

#8	Bits 8:15 of longitude value
#9	Bits 16:23 of longitude value
#10	Bits 24:31 of longitude value

TAKEOFF (#4)

This message is a signal from the base station to the drone to takeoff and begin/resume the mission. Used to trigger the beginning of a run, or to resume a run after additional payloads have been reloaded. The message contains no data bytes.

ABORT (#5)

This message is a signal from the base station to the drone to abort the current mission. Upon receipt, the drone will return to the takeoff point and land. The message contains no associated data bytes.

ACK (#6)

This message is an acknowledgement of the receipt of a packet. Used to confirm to the base station that an individual packet was received by the drone. It contains a single data byte, which contains the message type (from this list) that is being acknowledged.

MSG_HOME (#7)

This message communicates the drone's home location (latitude and longitude) back to the base station. This is necessary for the drone's path planning on the base station.

Table 8 Home Message Byte Structure

Byte	**Purpose**
#1	Bits 0:7 of latitude value
#2	Bits 8:15 of latitude value
#3	Bits 16:23 of latitude value
#4	Bits 24:31 of latitude value
#5	Bits 0:7 of longitude value
#6	Bits 8:15 of longitude value
#7	Bits 16:23 of longitude value

#8	Bits 24:31 of longitude value

Appendix C: Four Bar Analysis

This appendix further details the calculations that were conducted to analyze the Four Bar mechanism as a linkage. The FBDs are shown below and were used for various tasks including selecting a linear actuator and analyzing stiffness.

C.1 Free Body Diagrams

By analyzing the four-bar mechanism, the team was able to determine the force 'F' that would need to be applied by a linear actuator to lift the linkage. The team modeled forces at all pin joints of the linear actuator ('A','B','C', and 'D'), as well as the force applied at the pin joint that mounts the linear actuator to the chassis of the rover ('E'). 'W' represents the combined weight of the sliding mechanism and sensor platform. The weight of the individual links and actuators have been neglected in this analysis.

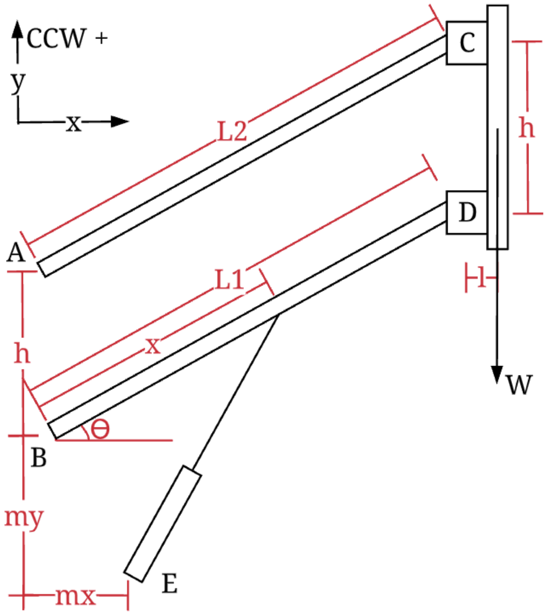


Figure 46 Whole Body Diagram

Because the length of the linear actuator, and the force applied depends on the angle of the four-bar and the mounting distance along the bottom-most link, the analysis was conducted with variables 'θ' and 'x' respectively. MATLAB was used to calculate the numerical force 'F' values for all possible 'x' values at four different 'θ' four-bar angles. This is detailed in 3.1.4.1 Four-Bar Redesign.

Figure 46 shows the four-bar mechanism and the sliding mechanism's mounting plate. The system is simplified to assume the entire sliding mechanism weighs 50 pounds, and its weight acts from the center of the sliding mechanism's mounting plate. In this diagram, the linear actuator used to move the four-bar mechanism is mounted at point 'E' to the rover chassis, and at distance 'x' from the bottom-most joint of the four-bar. The values of the constants are shown below:

$h = 5\ in, l = 1.25\ in, L2 = 17.82\ in, L1 = 17.13\ in, m_x = 4.9608\ in, m_y = 7.4589\ in$

$W = 50\ lbs$

Figure 47 shows the sliding mechanism’s mounting plate and the components of the forces at pin joints C and D on the plate. Static equilibrium equations are below:

$$\Sigma F_x = 0 = C_x + D_x$$
$$\Sigma F_y = 0 = -C_y - W + D_y$$
$$\Sigma M_D = 0 = -W(l) - C_x(h)$$

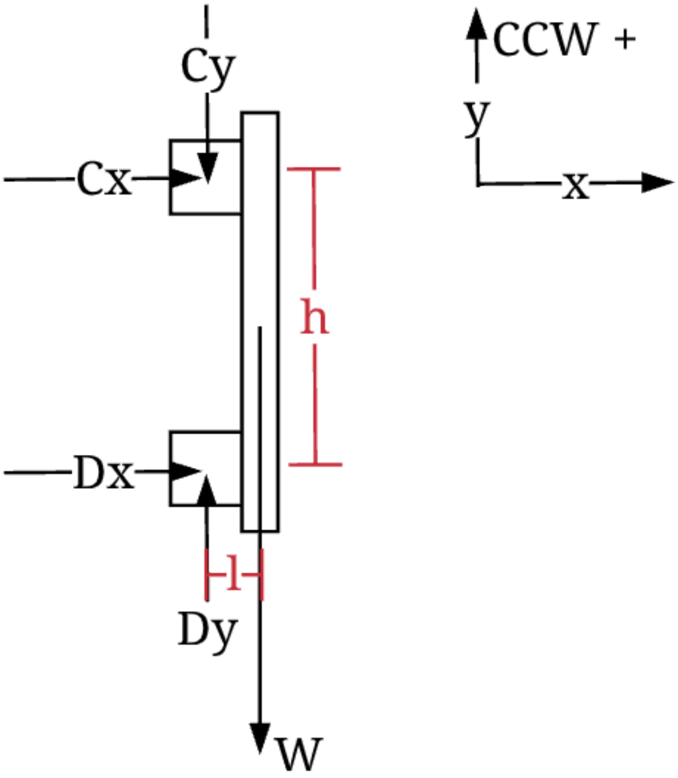


Figure 47 Link CD FBD

Figure 48 shows the link AC and the components of the forces at pin joints A and C on the link. Static equilibrium equations are below:

$$\Sigma \mathrm{F}_{\mathrm{x}} = 0 = -C_x + A_x$$
$$\Sigma F_y = 0 = C_y + A_y$$
$$\Sigma M_A = C_x L_2 \sin(\theta) + C_y L_2 \cos(\theta)$$

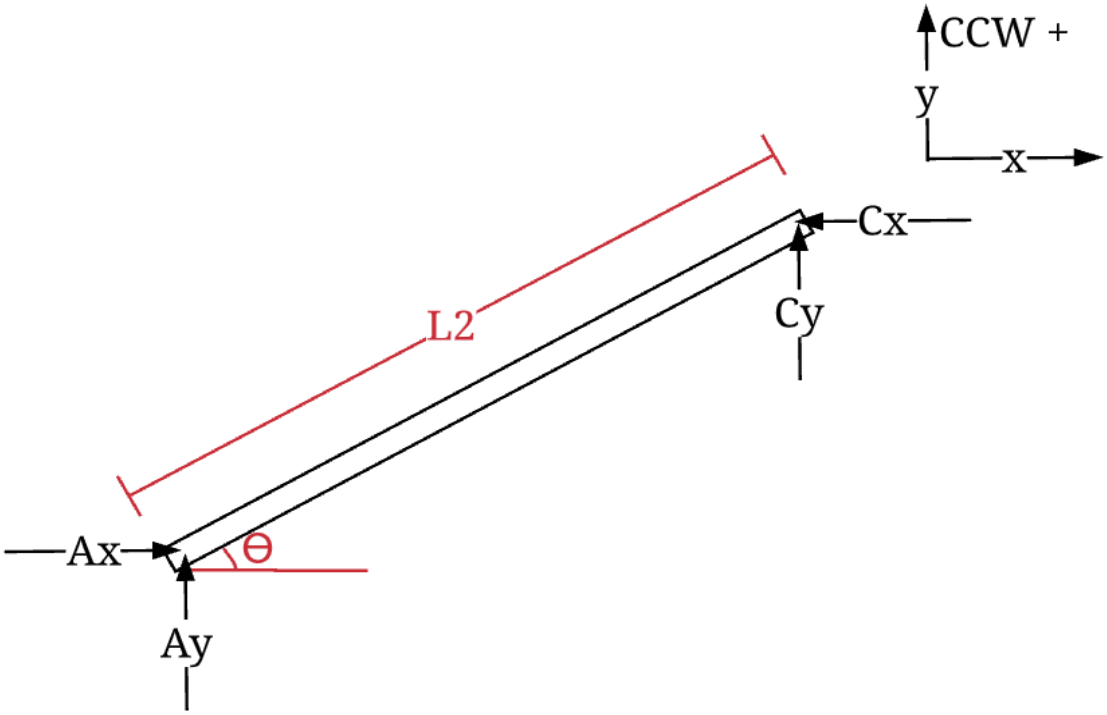


Figure 48 Link AC FBD

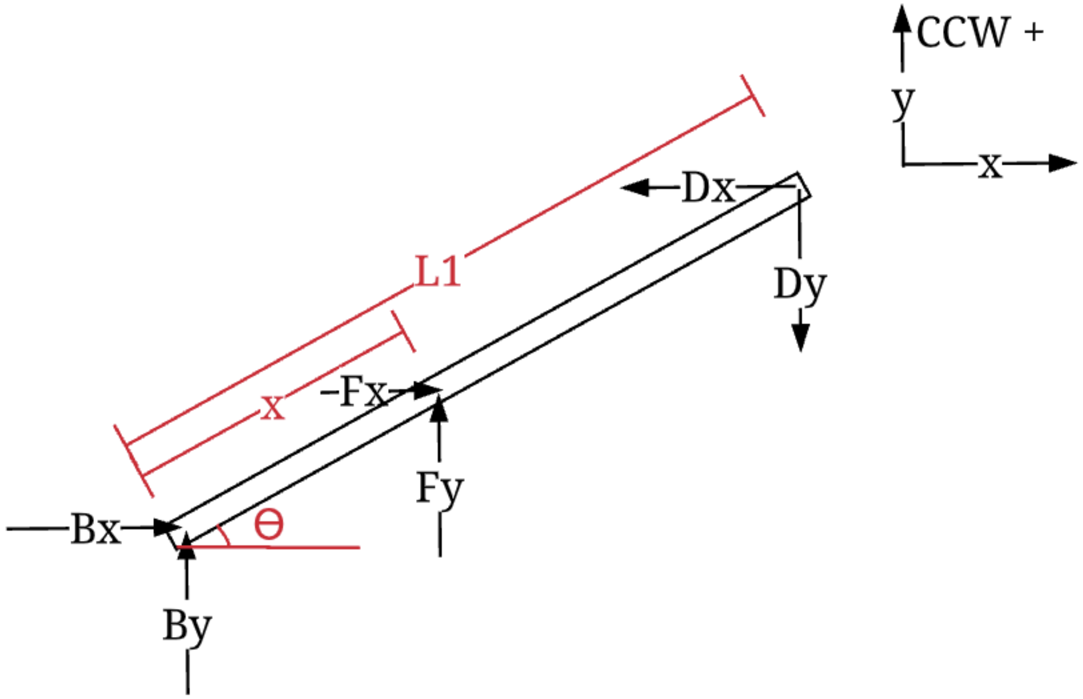


Figure 49 Link BD FBD

Figure 49 shows the link BD and the components of the forces at pin joints B and D on the link. Static equilibrium equations are below:

$$\Sigma F_x = 0 = -D_x + B_x + F_x$$
$$\Sigma F_y = 0 = B_y - D_y + F_y$$
$$\Sigma M_B = F_y x cos(\theta) - F_x x sin(\theta) - D_y L_1 \cos(\theta) + D_x L_1 \sin(\theta)$$

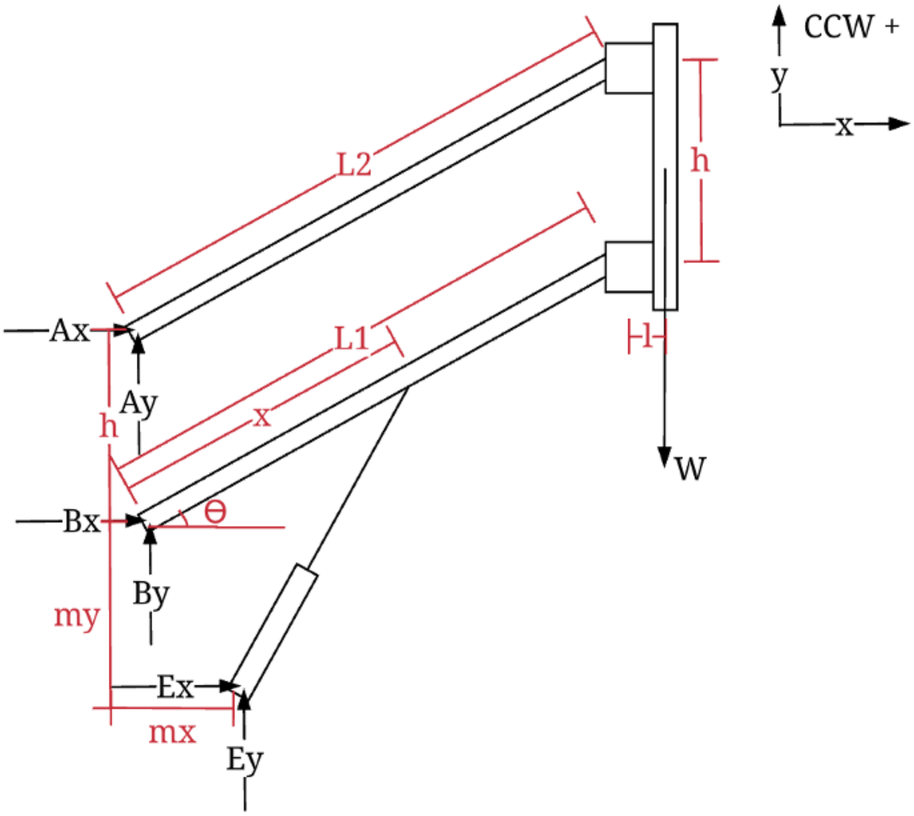


Figure 50 Whole Body FBD

Figure 50 shows a Free Body Diagram of all forces acting on the four-bar from the chassis, including the forces at the pin joint that mounts the linear actuator at point 'E'. Static equilibrium equations are shown below:

$$\Sigma F_x = 0 = A_x + B_x + E_x$$
$$\Sigma F_y = 0 = -W + A_y + B_y + E_y$$
$$\Sigma M_B = -W(L_1\cos(\theta) + l) + E_y m_x + E_x m_y - A_x(h)$$

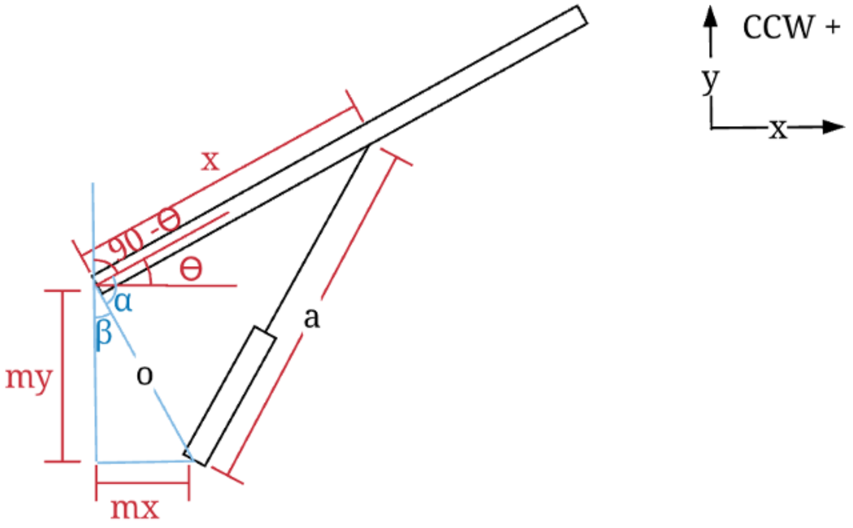


Figure 51 Relation between 'x' and 'a'

Figure 51 shows the geometric relation between the mounting location 'x' of the linear actuator along the bottom link of the four-bar, and the total length 'a' of the linear actuator. This conversion is necessary to determine what length of linear actuator can achieve the mounting location analyzed.

$$\beta = \tan^{\{-1\}}(\frac{m_x}{m_y})$$

$$\alpha = 180 - (90 - \theta) - \beta$$

$$o = \sqrt{m_y^2 + m_x^2}$$

$$a = \sqrt{o^2 + x^2 - 2oxcos(\alpha)}$$

A MATLAB script was made to generate plots of mounting distance 'x' vs. force 'F' necessary for the linear actuators to apply. These plots are shown in Figure 11. The same MATLAB script was used to calculate the unknown forces in pounds at each evaluated angle, with a mounting distance $x = 5in$:

$$\theta = 22.5^o$$

Unknown Forces (lbs)

A_x = -12.5	$A_y = -5.18$	$B_x = 18.66$	$B_y = -113.57$	$C_x = -12.5$	$C_y = 5.18$
$D_x = 12.5$	$D_y = 55.18$	$F_x = -6.16$	$F_y = 168.75$	$E_x = -6.16$	$E_y = 168.75$

$$\theta = 30^o$$

Unknown Forces (lbs)

A_x = -12.5	$A_y = -7.22$	$B_x = 22.98$	$B_y = -108.02$	$C_x = -12.5$	$C_y = 7.22$
$D_x = 12.5$	$D_y = 57.22$	$F_x = -10.48$	$F_y = 165.24$	$E_x = -10.48$	$E_y = 165.24$

$$\theta = 45^o$$

Unknown Forces (lbs)

$A_x = -12.5$ $A_y = -12.49$ $B_x = 32.145$ $B_y = -89.18$ $C_x = -12.5$ $C_y = 12.49$

$D_x = 12.5$ $D_y = 62.49$ $F_x = -19.65$ $F_y = 151.67$ $E_x = -19.645$ $E_y = 151.67$

$$\theta = 60^o$$

Unknown Forces (lbs)

$A_x = -12.5$ $A_y = -21.65$ $B_x = 38.76$ $B_y = -54.16$ $C_x = -12.5$ $C_y = 21.65$

$D_x = 12.5$ $D_y = 71.65$ $F_x = -26.26$ $F_y = 125.81$ $E_x = -26.26$ $E_y = 125.81$

Appendix D: FBD Analysis of Linear Slides

This appendix further details the calculations that were conducted to analyze the force transferred to the linear slides by a horizontal force applied to the end of the Carbon Fiber tube. The FBDs are shown below.

Analyzing a force applied at the bottom of the carbon fiber tube simulates what might occur if the Rover were to run into an object in its path. A weight of 5 lbs., indicated by force ‘F’ in the Free Body Diagrams was chosen to correspond with Solidworks simulations, and to act as a common analysis weight for structural rigidity. Calculating the force applied on the linear rails allows us to understand what types of strain they may be experiencing.

The problem below is a statically indeterminate problem. As a result, several assumptions were made to simplify the system.

1. All bodies are rigid and are rigidly connected to each other.
2. Assume that no screw connection introduces a preload into the system.
3. Assume the mechanism is assembled and manufactured to specification.
4. Treat the system as two entities: the carbon fiber tube, and the entirety of the sliding mechanism carriage, bearings, and tube clamps.
5. Assume that the vertical force exerted at the carbon-fiber tube’s connection with the clamps is shared between the two clamps. i.e. $F_{CTY} = -F_{CBY}$

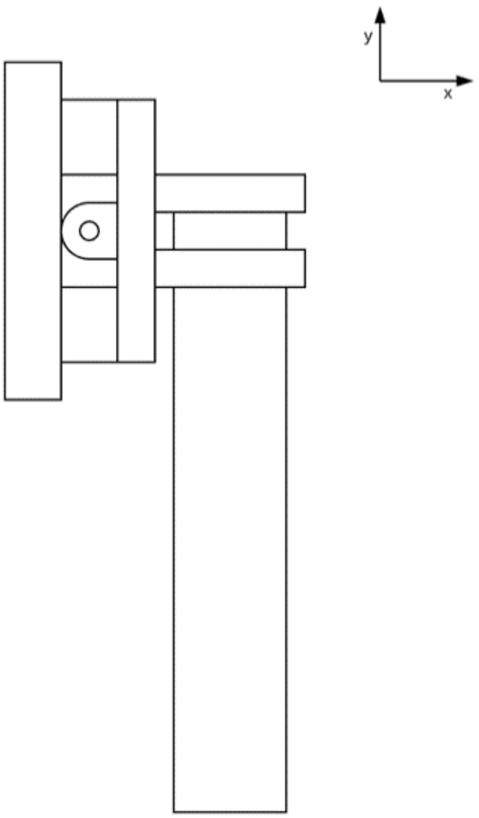


Figure 52 Block Diagram of Carbon Fiber Tube and Slider Carriage

Figure 52 is a block diagram of the Carbon Fiber tube that supports the sensor arm, the sliding mechanism carriage, and the sliding mechanism's mounting plate. Values for the analysis are listed below:

$r = 1.5"$,
h_1=3.52", h_2= 23.75", w= 2.95", d_b=0.8", L_{bT}=2.875", L_{bB}=2.915" , L_{cT}=1.74",
L_{cB}= 1.78", L_{COM}= 0.6", h_c=22.01", $W_{Carriage}$=7.28 lbs, W_{Tube}=0.9087 lbs

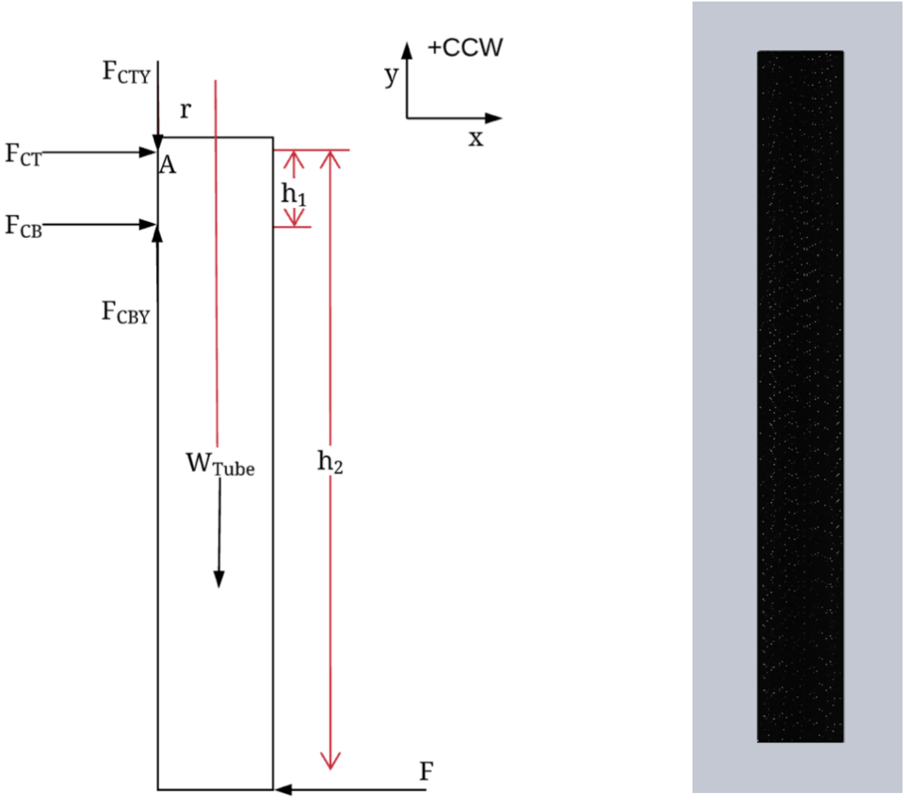


Figure 53 FBD of Carbon Fiber Tube with Applied Force F

Figure 53 shows the Carbon Fiber tube, and the forces ‘F_{CT}’, ‘F_{CTY}’, ‘F_{CB}’, and ‘F_{CBY}’ applied on it by the two clamps that hold it to the sliding mechanism carriage. The Force ‘F’ is applied in the -x direction at the end of the tube. This would correspond towards a force acting in the direction of the rover’s chassis from in front of the robot. Static equilibrium equations are shown below:

$$\Sigma F_x = 0 = F_{CT} + F_{CB} - F$$

$$\Sigma F_y = 0 = -W_{Tube} - F_{CTY} + F_{CBY}$$

$$\Sigma M_A = 0 = h_1 F_{CB} - h_2 F - r W_{Tube}$$

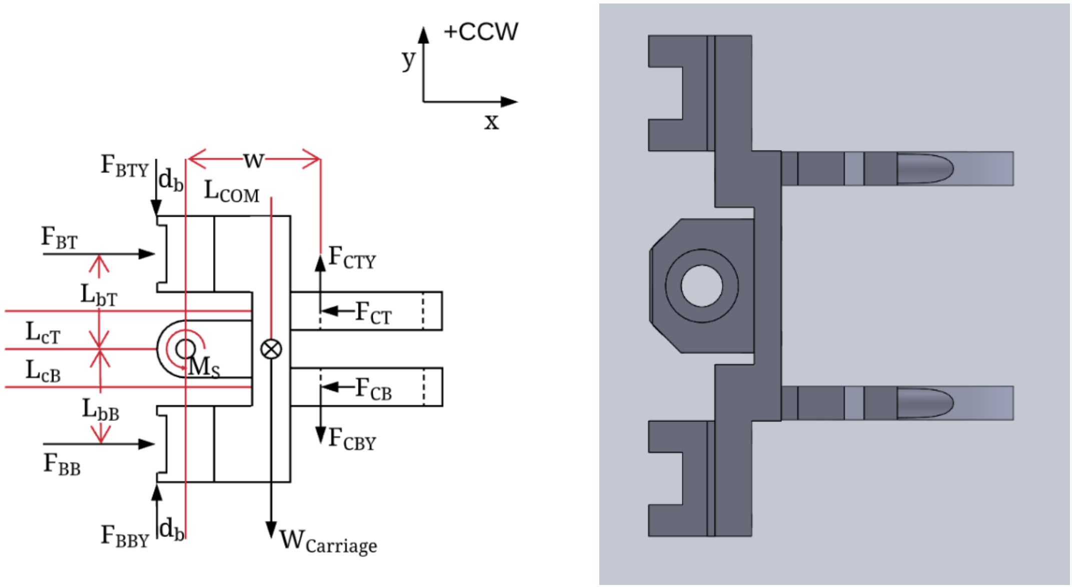


Figure 54 FBD of Slider Carriage

Figure 54 shows the slider carriage, ball-screw and nut, the two tube clamps that hold the Carbon Fiber tube, and the linear rail bearings. These were treated as a solid mass for the purposes of the analysis with a weight '$W_{Carriage}$'. The forces 'F_{BT}', 'F_{BTY}', 'F_{BB}', and 'F_{BBY}' are used to represent the components of the forces acting between the bearings and the linear rails. The static equilibrium equations are below:

$$\Sigma F_x = 0 = F_{BT} + F_{BB} - F_{CT} - F_{CB}$$

$$\Sigma F_y = 0 = F_{BBY} - F_{BTY} - W_{Carriage} + F_{CTY} - F_{CBY}$$

$$\begin{aligned}\Sigma M_S = 0 = M_s - L_{bT}F_{BT} + d_bF_{BTY} + wF_{CTY} + L_{cT}F_{CT} - L_{cB}F_{CB} - wF_{CBY} - L_{COM}W_{Carriage} \\ - d_bF_{BBY} + L_{cB}F_{BB}\end{aligned}$$

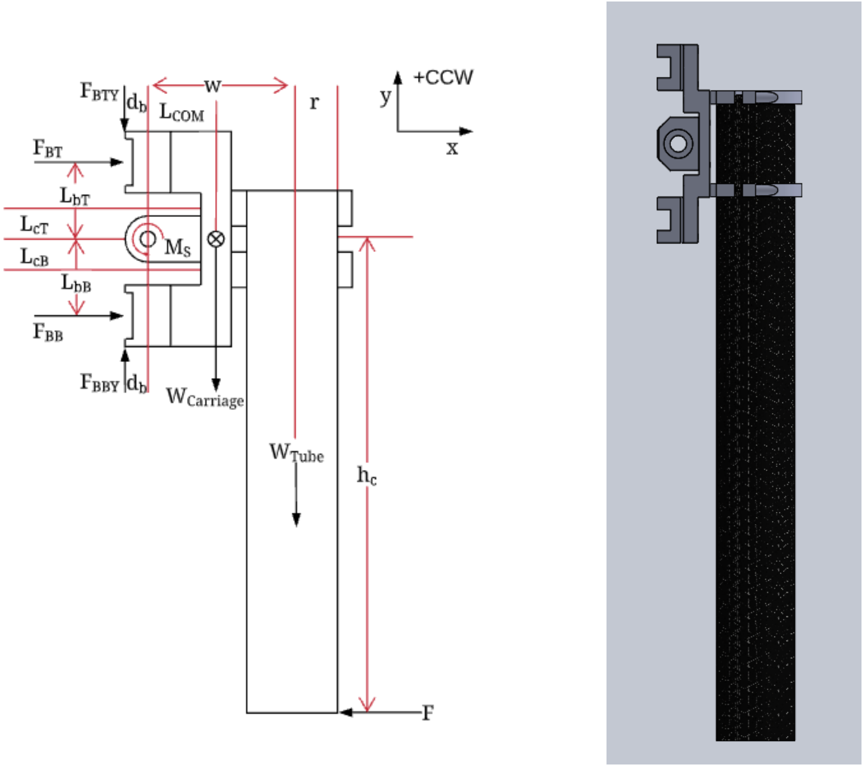


Figure 55 FBD of Carbon Fiber Tube and Carriage with Applied Force F

Figure 55 is a free body diagram of both the carriage and the carbon fiber tube. External forces on the bearings, and the applied force 'F' are shown. Static equilibrium equations are below:

$$\Sigma F_x = 0 = F_{BT} + F_{BB} - F$$

$$\Sigma F_y = 0 = F_{BBY} - F_{BTY} - W_{Carriage} - W_{Tube}$$

$$\Sigma M_A = 0 = Ms - L_{bT}F_{BT} + d_bF_{BTY} - L_{COM}W_{Carriage} - d_bF_{BBY} + L_{bB}F_{BB} - wW_{Tube} - h_cF$$

Results:

MATLAB was utilized to perform the numerical calculations. The results are shown below.

$$M_s = 0$$

$$F_{BT} = -19.0739\ lbs$$

$$F_{BTY} = 14.1813\ lbs$$

$$F_{BB} = 24.0739\ lbs$$

$$F_{BBY} = 22.37\ lbs$$

$$F_{CT} = -29.123\ lbs$$

$$F_{CTY} = -0.45436\ lbs$$

$$F_{CB} = 34.123\ lbs$$

$F_{CBY} = 0.45436\ lbs$

Appendix E: Slider Motor and Transmission

Calculations for selection of motor and transmission ratio.
In a ball screw assembly where

- F_{ax} = axial load [N]
- l = lead = 0.005m
- η_1 = efficiency factor = 0.9
- F_{pr} = preload force [N] note: assumed to be 0.08*(load capacity) = 612N
- BCD = ball circle diameter = 0.016m
- W_{car} = carriage weight ≈ 15lb = 66.7N
- μ_{ln} = friction coefficient of linear rails ≈ 0.004

The following equations apply

Driving Torque: $T_d = \frac{F_{ax} \cdot l}{2\pi \cdot \eta_1} = \frac{0.005m}{(0.9) \cdot 2\pi} F_{ax} = 8.842 \cdot 10^{-4} F_{ax}$

Drag Torque: $T_p = \frac{0.05}{\sqrt{\frac{l}{\pi \cdot BCD}}} \cdot \frac{F_{pr} \cdot l}{2\pi} = \frac{0.05}{\sqrt{\frac{0.005m}{\pi \cdot 0.016m}}} \cdot \frac{F_{pr} \cdot 0.005m}{2\pi} = 1.262 \cdot 10^{-4} F_{pr}$

Total Driving Torque: $T_{tot} = T_d + T_p = (8.842 \cdot 10^{-4}) F_{ax} + (1.262 \cdot 10^{-4}) F_{pr}$

$T_{tot} = (8.842 \cdot 10^{-4}) F_{ax} + 0.0772 N \cdot m$

$F_{ax} = \mu_{ln} W_{car} + F_{push}$ where F_{push} is the designed force with which the carriage should be pushed (against resistance)

$T_{tot} = T_{motor}(r_{trans})(\eta_{trans})$ where r = ratio and *n* = efficiency ≈ 0.9

The team decided the carriage should be run under normal conditions at a speed of about 4 in/s or 1220 RPM, and encountering resistance the carriage should push with a force no greater than 20lb in order to avoid damage to the rover or injury to persons. Therefore, the minimum torque the team wants is the $T_{totFree}$ when F_{push} = 0lb and the maximum torque should be $T_{totLoad}$ when F_{push} = 20lb or 88.96N

$$F_{axFree} = (0.004)(66.72N) + 0N = 0.267N$$
$$T_{totFree} = (8.842 \cdot 10^{-4})(0.267N) + 0.0772N \cdot m = 0.0775N \cdot m$$
$$T_{totFree} = 10.97 oz \cdot in$$
$$F_{axLoad} = (0.004)(66.72N) + 88.96N = 89.23N$$
$$T_{totLoad} = (8.842 \cdot 10^{-4})(89.23N) + 0.0772N \cdot m = 0.1561N \cdot m$$
$$T_{totLoad} = 22.11 oz \cdot in$$

Considering several motors, the Vex CIM Motor delivered a high enough max speed for a very reasonable cost of $32.99. The motor specifications are as follows.

- No-load speed = 5310 RPM, 2.7A
- Stall torque = 343.4 oz*in, 133A

Speed function: $RPM(T) = 5310 - 15.46T[oz \cdot in]$
Current function: $\mathrm{I}(T) = 2.7 + 0.387T[oz \cdot in]$

$$RPM\ of\ screw, no\ load = \left[5310 - 15.46\left(\frac{10.97}{r_{trans} \cdot \eta_{trans}}\right)\right]/r_{trans}$$

$$\text{Current drawn, max load} = 2.7 + 0.387\left(\frac{22.11}{r_{trans} \cdot \eta_{trans}}\right)$$

As seen above, using a 4:1 transmission with η = 0.9, the no-load speed of the screw is 1316 RPM, slightly above the 1220 target. The team decided instead to use a 2:1 transmission to allow us to increase the speed if desired, and to account for any unforeseen effects on speed. The ball screw used was very inexpensive but had almost no documentation. The preload force, for example, which was assumed to be 612N, was unknown and could only have been determined by precise testing. Using a 2:1, the team could control the speed within a favorable range with a motor controller, and upon encountering resistance, the team would ensure the motor would stay on the rising slope of the power curve. With a 2:1 transmission, the screw's no-load speed is 2608 RPM or a theoretical maximum linear speed of 8.5 in/s. The current needed to be limited to 7.5A to ensure a maximum push of 20lb.

Source: (Ball screw torque calculations n.d.)

Calculations for selection of belt length, pulleys, and transmission center to center distance.

Belt Circumference: $C = \frac{\pi D_{p30}}{2} + \frac{\pi D_{p15}}{2} + 2l$

- l_{inst} = Installation length = $\sqrt{\left(\frac{D_{p30}-D_{p15}}{2}\right)^2 + CCD^2}$
- l_{adj} = Adjusted length = $\sqrt{\left(\frac{D_{p30}-D_{p15}}{2}\right)^2 + CCD^2 + 0.5^2}$
- D_{p30} = 30 Tooth Pitch Diameter = $1.91''$
- D_{p15} = 15 Tooth Pitch Diameter = $0.955''$

Adjusted Circ.: $C_{adj} = 13 = 4.5 + 2\sqrt{0.5 + CCD^2}$, solved for CCD yields 4.19"
Installation Circ.: $C_{inst} = 13 = 4.5 + 2\sqrt{0.5^2 + 4.19^2} = 12.94''$ leaving about 1/16" slack on a 13" belt for installation

CCD = 4.19"
C_{belt} = 13"

Appendix F: Rover CG and Axle Weight Calculations

Once the four-bar was installed, the rover was front heavy and it became necessary to add counterweight to the back of the rover to keep it from tipping. We temporarily added weight to the electronics box but recommend to a future team that a counterweight of 80lb be permanently and rigidly mounted to the rear of the rover.

Initial weight	$W_i = 175\ lb$
Counterweight	$W_{CW} = 80\ lb$
Final weight	$W_f = W_i + W_{CW} = 255\ lb$
Rear axle position	$x_1 = 0\ in$
Mid axle position	$x_2 = 10\ in$
Forward axle position	$x_3 = 20\ in$
Initial CG position	$x_{CGi} = 20.92\ in$
Counterweight position	$x_{CW} = 0\ in$
Final CG position	$x_{CGf} = \frac{W_i x_{CGi}}{W_f} = 14.36\ in$

The weight on each axle is statically indeterminate, but taking the compression of the tires into account, it is possible to solve for the normal force each axle applies to the rover chassis.

$$N_1 = -W\left[\frac{-x_2^2 + x_2 x_{CG} - x_3^2 + x_3 x_{CG}}{2(x_2^2 - x_2 x_3 + x_3^2)}\right] = 29.44\ lb$$

$$N_2 = W\left[\frac{x_3^2 - x_2 x_3 + x_{CG}(2x_2 - x_3)}{2(x_2^2 - x_2 x_3 + x_3^2)}\right] = 85.00\ lb$$

$$N_3 = W\left[\frac{x_2^2 - x_2 x_3 - x_2 x_{CG} + 2x_3 x_{CG}}{2(x_2^2 - x_2 x_3 + x_3^2)}\right] = 140.59\ lb$$

Appendix G: Payload Force and Impulse Graph Matlab Script

```
% Rough calculator for estimating payload impact force and impulse values
%for different height and mass parameters.
% ------------------------------------------------
% -Independent vars: height (h), mass (m), intial velocity (vi=0)
%
% -TBD/problem variables:
$     -Stopping_time (tstop = 0.2), can't find source on a good value for
$this, it may also change with ground type and should be verified
$experimentally
%     -The velocity caused by rebound effects, assumed as 0 here for a
%baseline impulse regardless of ground or payload. This is value difficult to
%predict but can substantially increase the force and impulse values
%
% -Dependent: drop_time -> vf -> force -> impulse
%
% drop time (td)= sqrt(2*h/g)
% vf = td * g
% Average Force = F = m * a = m * (g + vf*tstop)    //minimum force for
%overcoming spring is 50N
% Total Impulse: J = F * tstop

syms h;
syms m;
vi = 0;
g = 9.8;
tstop = 0.2;
td = sqrt(2*h/g);
vf = td*g;
a = (g + vf*tstop);
F = m * a;
J = F * tstop;

figure(1);
f_plot = fsurf(F, [.2 2 5 30]);
title('Average Impact Force (N)');

figure(2)
j_plot = fsurf(J, [.2 2 5 30]);
hold on;
fsurf(50);
title('Total Impulse (Ns)');
ylabel('Dropping height (m)');
xlabel('Payload mass (kg)');
```

Appendix H: DAS Media

Team Media:

Google Photos Drive for MQP: https://photos.app.goo.gl/AVy2h7REwGQzNiQ89

WPI Press Release: https://www.wpi.edu/news/search-and-destroy-wpi-students-build-robotic-system-find-and-safely-detonate-landmines

WPI Press Release: https://www.wpi.edu/news/search-and-destroy-wpi-students-build-robotic-system-find-and-safely-detonate-landmines

Media Coverage:

BBC News: https://www.bbc.com/news/av/technology-50045250/rover-drone-combo-hope-to-spot-and-destroy-landmines

Boston 25 News: https://www.boston25news.com/news/local-students-build-robot-that-finds-safely-detonates-hidden-landmines/992398318/

Digital Trends: https://www.digitaltrends.com/cool-tech/drone-and-rover-wpi-landmine-project/

Robotics Business Review: https://www.roboticsbusinessreview.com/research/wpi-students-combine-robots-drones-to-search-and-destroy-land-mines/

Worcester News Tonight: https://www.youtube.com/watch?v=18-YdCYUu3M