University of Central Florida

Department of Electrical & Computer Engineering

EEL4914

Senior Design 1: Group D

Soft Vine Robot

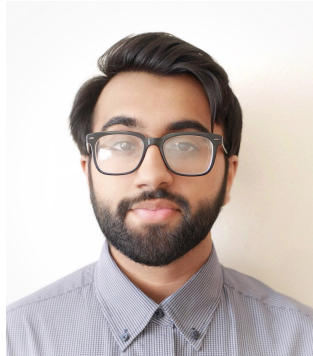**Senior Design I**
Final Documentation

Dr. Samuel Richie and Dr. Lei Wei

April 26, 2022

Team Members

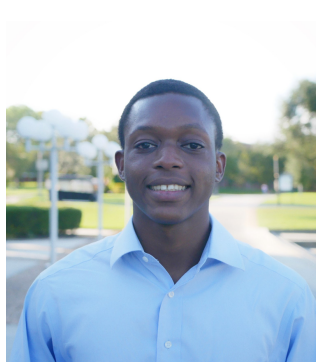**Juan Battaglia**
Electrical Engineering

Muhammad Gudaro
Electrical Engineering

Kevin Abreu-Aguila
Computer Engineering

Abdul-Malik Mustapha
Electrical Engineering

Table of Contents

List of Figures

List of Tables

1 Executive Summary

From the beginning of time, mankind has found innovative ways to complete tasks with less effort. As we transition into the technological age, there has been a significant increase and demand for robots to replace human labor completely. However, what if robots can completely mimic the movements of other animal species? This idea brought about the future of soft robotics. 'Soft' robotics can be defined as designing, controlling, and fabricating soft, flexible bodies, instead of using conventional rigid materials, thus mimicking both land and aquatic animals such as worms, fishes, and octopi. This field of study has numerous benefits and applications. It can be applied in areas ranging from medicine to botany to archeology.

The main goal of our project is to create a soft, autonomous robot using flexible materials such as silicone and eco flex that will be programmed to navigate its way through an obstacle course of rough terrain, narrow spaces, and elevations. As a stretch goal, we plan on implementing a gripper tool that will be used to pick up the object easily. A RunCam camera will be installed at the tip of the robot which will be able to send visual footage to the user who can operate it manually using a controller. We plan to incorporate a manual option for the user to control the robot's movement using a joystick. An accelerometer and gyroscope will both be used to measure the speed and orientation of the soft robot. The maximum length of the soft robot is twice as long as the length of the obstacle course and the diameter will be closer to five inches so it can navigate its way through hard-to-reach places. The robot will be able to move through pneumatic control and a pressure sensor that controls its orientation. For our microcontroller, we will be implementing a Raspberry Pi and use robot vision to program the robot to navigate effectively through the obstacle course. Our budget for this project ranges from $1400 to $1700.

Our final project documentation is divided into thirteen sections. In the first and second sections, we discuss our objectives, motivation, goals, and requirement specifications for this project. In the third section, we discuss the related standards and constraints that may hinder the success of the project. In the fourth to the seventh section, we will be talking about relevant research and technology that currently exists in the field of soft robotics, the mechanical design and electronics, and the power design of the robot. From sections 8 to 13, we discuss the software design of the robot and how we want to provide a user-friendly interface to allow the user to control the robot manually. We will also be discussing our prototyping process and how we improved our current design based on our initial prototype. Also, we will explain our testing environment and how we plan on implementing our project in reality. In the final section, we will talk about how we allocated our budget and the milestones we need to reach to complete the project successfully. We will be also giving an overall summary of what our project entails.

2 Project Description

2.1 - Project Motivation and Goals

Ever since the conception of robotics, the objective was to always mimic human movements or complete tasks that are difficult to do, reducing the need of humans altogether. However, what if nature, along with animal movement and composition, were the inspiration for robotics? There are various animals and species capable of moving and doing things in ways no human can do, which brings us to the idea of "soft robotics." Soft robotics is a growing area of research that deals with robotics designed with compliant materials, such as fabric, silicone, or other flexible material, along with flexible electronics, instead of the rigid material that is conventionally used. Using this material can improve safety, allow greater flexibility, and make it easier to access hard-to-reach areas. This new technology could have a variety of great applications in medicine, construction, disaster relief, or archeology. Moreover, our project aims to study and showcase the potential of soft robotics by bringing to life a soft robot capable of reaching hard-to-reach places.

The goal of our project would be to design a soft vine robot using flexible materials that would be programmed to navigate the obstacle course. Using pneumatically controlled tubing, the robot would attempt to navigate through a series of turns, tight areas, sticky surfaces, liquids, rough terrain, and elevations to retrieve a small object at the end. To grab the object, a gripping system will be implemented on one end of the vine robot, which would allow the robot to navigate the obstacle course autonomously.

We also plan to implement an application software where users can easily interact with the worm using joystick controllers. In addition, we will be using OpenCV as an image processing platform open-source library. It will be used to provide an image feed to the camera, which provides visual feedback on the vine robot's environment. To move the worm robot, we will be using three motor encoders as actuators to help guide the worm through the obstacle course. An air pump will be used to inflate the worm and use the solenoid valves and relay to provide accurate control based on negative feedback to correct error signals or any degree of randomness when completing the obstacle course.

Furthermore, as this is a relatively new technology, there isn't any available input from customers or current marketing analysis on competitive products utilizing soft vine robots. There is, however, research being conducted on how soft robotics can be used in the form of a worm-like design, similar to what we are aiming for in this project. Vine Robots is a soft robotics project supported by

Stanford Applied Nanophotonics and the IRiS lab of the University of Notre Dame. Their robot, built with flexible material, mimics a vine by retracting and growing with pressurized pneumatics. The robot can also autonomously avoid obstacles with pre-programmed haptic controls. This project serves as one of our inspirations to pursue this idea.(**Figure 2.101**)
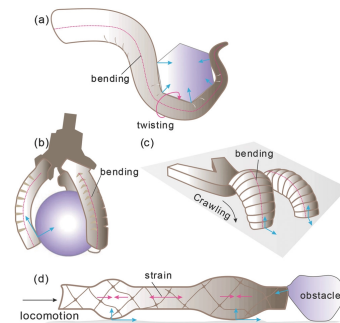
Figure 2.101 - Worm and gripper illustration
(reprinted with permission from Hongbo Wang)

2.2 - Objectives

The objective of this project is to design and build a soft autonomous robot that has the ability to navigate through narrow spaces, rough and elevated terrains and wet surfaces through a dynamic pneumatic system. We will use valves within the artificial muscles to properly control how much air flows in and out of the soft body to produce an efficient locomotive system within the obstacle course. We will be using either Arduino or Raspberry Pi as the main microcontroller unit and we will be using Python to effectively program the robot to move through obstacles within the obstacle course. We plan on using both lipo batteries rated at 12V and 2.6Ah as our main power source for the robot to move efficiently through the obstacles. Using these solar panels as a stretch goal initially served as a motive to consume less non-renewable energy to help reduce global warming. We also plan to use an application software where the user has the opportunity to manually control the robot to navigate through the obstacle course effectively. As the robot navigates through the obstacle course, the user will be getting real-time visual footage through a camera. Furthermore, suppose this robot surpasses our expectations, in that case, this device can be applied in archaeological excavations where the robot can navigate through deep and narrow tunnels and provide real-time visual feedback. This robot can also discover new species in aquatic biosystems. Also, this can be applied in the medical field to help doctors properly perform invasive surgeries because this device will be able to navigate through hard-to-reach places using its soft, flexible body.

This project will address significant issues surrounding how animals interact within their ecosystems because we can use the novelty of soft robots technology to mimic animal movements and recreate them with more real-life movements. They could also be applied to wearable technology such as prosthetics for individuals with disabilities. The soft robot will have the capability to expand twice as much with the help of compressed air applied to it at multiple time intervals.(**Figure 2.201**) Here is a picture of a vine robot that shows how the vine triples in length based on how much air is compressed through the flexible body.

Figure 2.201- Vine robot elongating twice as much as its original length
(reprinted with permission from vinerobots.org)

The robot will have an internal roller that helps it retract to its original formation. A camera will be installed at the tip of the robot to provide in-depth visual realization feedback of the robot's environment and how effectively it uses path planning to determine the optimal route through obstacles within the obstacle course. The Raspberry Pi will be connected to the camera receiver, which will provide instant feedback on the visual depth of the robot's environment. The camera will be provided with night vision capabilities designed for manual steering. We plan on using a gyroscope that helps to detect the deviation from the obstacle course's original location to navigate through the obstacles at maximum accuracy properly. A stretch goal will be to include grippers to successfully pick up a relatively large object and run through the obstacle for a second time. These grippers will be made of a soft, flexible material made from elastomer or silicone which will be controlled using pressurized air.

2.3 - Requirement Specifications

During this section, we will look at the main requirements that we are striving to meet by the end of the project. These requirements will keep us on track for both our research and design portion as well as for our manufacturing period. As engineers, we abide by these rules as we would have to comply with any requirements set by customers in the industry. Besides the requirements, there is also a list of features that we want our design to include and be able to perform. Features are not all required but give a good description of what our robot will be able to do. Furthermore, we also include some of the constraints that this design will have. These constraints are natural to the engineering design process and there are always possibilities that we may encounter more as we test our design.

4

2.3.1 - Requirements

Robot:
- Shall house a camera, gyroscope, and accelerometer to relay information to the user and be controlled autonomously.
- The body shall be made of flexible material capable of elongating to twice the required length of the obstacle course(20 feet maximum) using pneumatic pressure.
- The body diameter shall be 5 inches to allow for navigating tight areas and easier turning
- Shall turn using artificial muscles that expand using pneumatic pressure and shall have a maximum turning radius of 1.5 feet.
- Shall retract using an internal roller that inverts the base tubing into itself
- Shall be able to grip objects of 4cm diameter with a pneumatically controlled soft gripper.
- The batteries used will be lipo (Lithium-polymer) rated at 12V, 2.6Ah
- Shall extend from the tip, and the tail shall hold the body of the vine, the electronics, and the pneumatics.

- Shall contain a Fluid Control Board that dissipates and distributes the air pressure to all the parts dependent on pressure.

User Interface:
- Users shall be able to select if the robot will be manually driven through a series of rough terrains and obstacles or if it will autonomously follow a user-provided light.
- Users shall be able to see in real-time what the light sensors and camera are reporting back to the robot.

Obstacle Course:
- The obstacle course shall be able to be completed using a maximum of 10 feet of robot length.
- The obstacle course shall be designed using a combination of turns, tight areas, sticky surfaces, rough terrain, and elevations.

2.3.2 - Feature Specifications

- Batteries should be at 12V and 2.6Ah.
- The camera for manual steering should have night vision capabilities or a flashlight.
- The gyroscope will have a range of 450 degree/s to measure and maintain the angular velocity of the robot and assist with steering.
- Accelerometer range of (+/-) 2g to record the distance traveled.
- A robotic gripper at the head of the robot to grab objects should be capable of 5 psi of gripping strength.

- Material for the robot should be flexible, tendon-like tubes that expand upon being applied pressure.
- The pressure sensor allows the arm to apply as much as (2 - 2.5) psi for grabbing objects.
- The gripper will have a radius of (4 inches) for its range of motion in all directions.
- The gripper should have at least 2 points of contact.
- Robot should use a Fluid Control Board to regulate pressure flowing into the flexible tendons and the gripper.
- Users should be able to control the pressure distributed to the robot to control its growth rate and speed by using the pressure sensor.
- The user should be able to manually turn on and off the air pump using a switch.
- Raspberry Pi board should be used to get sensor feedback.
- The obstacle course should be at least 5 feet long and made mostly out of cardboard.
- The obstacle course will include dead-ends for all possible sides and obstacles for the robot.
- The body of the robot will be developed using (eco flex and silicon elastomer) as well as the artificial muscles.
- The robot body should grow at a maximum rate of 2 in/s.

2.3.3 - Feature Constraints

Mobility: While the robot is made for ease of access and mobility throughout different terrains, carrying the pressure tank or dispenser might not be as easy to carry or move when deploying the robot in different environments. Both the body of the robot and the base that houses the pneumatics and electronics will likely be heavy and voluminous. We expect the entire system to have a weight of close to 50 pounds.

Gripper & Sensors: The robot has limited space for the features that the team wants to implement. Since the nature of the bot indicates that the head expands, the robot needs to include a cap at its head, so the sensors, camera, and arm grip do not fall off as it expands.

Retraction: The body of the robot allows it to navigate numerous terrains and through or around obstacles because of the pressure being applied. However, when retracting the vine, there is a mobility constraint in narrow spaces, which is a key feature of these vine robot projects. Implementing the artificial muscles, sensors & gripper, and internal roller for retracting will be challenging. We will also have to build an internal roller so that the robot does not coil when retracting since this can be damaging to the environment as it is unwanted behavior.

Accessibility: One of the biggest advantages of using this type of robotic structure is accessing tight spaces, which is a feature that will be limited to any space that can fit the head of the worm containing the gripper. Normally, other worm designs

6

are able to fit through extremely small spaces since the force provided by the pressure will be greater than the area we want to go through.

2.4- House of Quality Analysis

The house of quality is a quality matrix used by engineers to decide how feasible and realistic the product is and if it can live up to the consumer's needs. In order to build a successful project that can be applied in the real world, we need to tailor our specifications to what the consumer desires. Based on our house of matrix figures, our engineering requirements are dimensions, processing time, weight, power consumption, production cost, and accuracy, while our marketing requirements are durability, ease of use and maintenance, reliability, and cost. Collecting all this data, we organized it in a table format that symbolizes the relationship between the engineering specifications and consumers' needs. (**see Figure 2.401**)

Dimensons and durability have a positive correlation because a wider robot will allow the internal components to fit in properly and also give the robot a lower center of gravity which will prevent tilting through the rough obstacles. The weight and durability of the robot have a strong positive correlation because the heavier the robot will make the robot sturdier. In addition, the processing time and cost will have a negative correlation because it will require more sophisticated electronic parts to have the robot finish the obstacle course at a faster rate. The power consumption and overall cost will have a strong positive correlation because if we need our robot to generate more speed, it will require more power to charge the batteries within a short period of time which will require looking into more advanced power architectures that can provide these requirements.

Moreover, the accuracy of the user controlling the robot will have a strong positive correlation with the reliability of the robot long-term because this will determine how useful the soft robot will be over a long period of time. However, the weight and the convenience of the use of the robot will have a negative correlation because it will require more software and hardware complexity if more components are added to the soft robot. In addition, weight and portability will have a strong negative correlation because the heavier the soft robot, the more cumbersome it will be to transport it from one location to another. The power consumption of the soft robot should be at most 200 watts. This will provide a considerable amount of power to the lipo batteries.

Dimensions of the soft robot will be limited to five feet in diameter and ten feet in length.The processing time of the robot will be at most 150 frames per second. The weight of the robot will be less than fifty pounds. Our production cost is estimated to be five hundred dollars. The accuracy of the robot's orientation through the obstacle course will be within one inch. This will help increase the validity of our project and create a concrete plan of action of how we decide to push our project into the market based on optimal cost and efficiency.

7

Our project plan includes reaching out to professionals in the medical, archeological, and botanical fields who will benefit tremendously from our finished version of the project. These consumers will test each marketing requirement which will determine how feasible and practical our project can be in the real world. Our legend below shows the symbols we used to describe the relationship between the engineering and marketing requirements. Positive and negative polarities are designed to show either a direct or inverse relationship between the engineering and marketing requirements, respectively. Another example will be easy to maintain and production cost. Easy to maintain has a positive polarity because that will be a benefit to the user, while production cost can be inconvenient to the engineers who are supplying the parts.
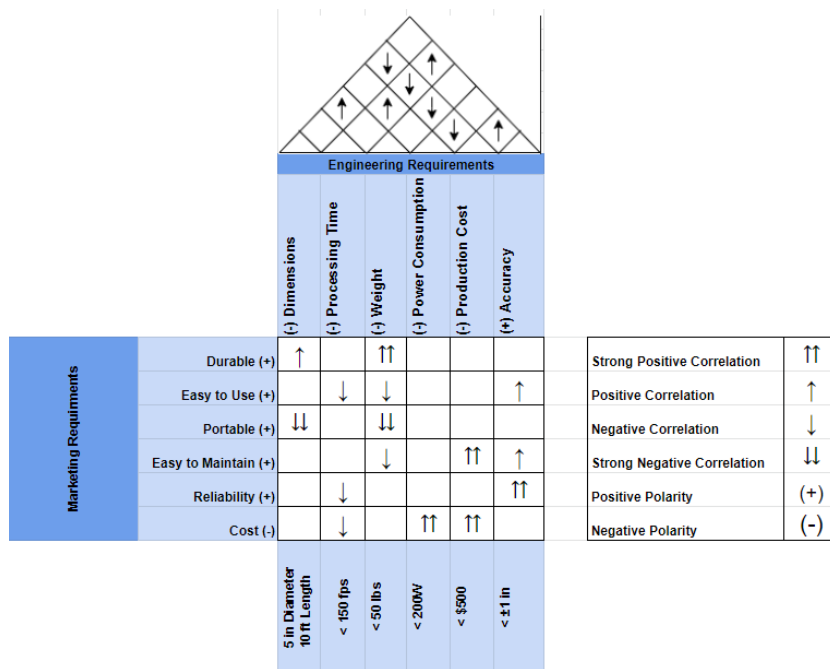
Figure 2.401 - House of Quality Analysis

3 Related Standards and Realistic Design Constraints

In this section, we talked about standards related specifically to our project and how we reference the standards of the electrical components we used. We also brainstormed realistic design constraints that have made this project more realistic and feasible to accomplish by the end of this year. We also discussed the health, economic and political constraints associated with making this project a success,

3.1 - Related Standards

Standards related to this project are not always obligatory as they only show proper procedures that are recommended to develop efficient and optimized processes. They can be quite flexible depending on different circumstances during world trade. Some standards are regarded as codes which are enforced by the law and mandatory for all industries to apply diligently.

Standards are implemented in this project to ensure safety checks are made ince the robot starts to expand through the obstacle course. The standard ANSI/RIA R15.06 allows us to provide risk assessment. A risk assessment is outlined in ANSI B11 Series Standards for Industrial Machinery, ANSI/RIA R15.06-2012 Safety Standards for Industrial Robots, and the National Fire Protection Association (NFPA) 79-2015 Electrical Standard for Industrial Machinery. The main goal of a task-based risk assessment is to identify hazards associated with machinery or robots. This requires an on-site visit by a risk assessment professional who audits and assigns each machine a risk rating based on three considerations: Severity of Injury, Exposure Frequency, and Avoidance Likelihood, which produces a Risk Level. Today's risk assessment specialists use software-based tools that can make the process quicker than working through a pen-and-paper risk assessment form.

USB 3.0 Standards: First developed in 2007 and mainly showed what was considered a new feature back then which was the SuperSpeed bus, which provides a fourth transfer mode. It gives data a transfer rate of 4.8 Gbit/s. The USB 3.0 has a raw data throughput of 4 Gbit/s, but its data transfer rate is 3.2 Gbit/s or 4 GBytes/s. This standard is also backward compatible with USB 2.0. It contains 4 of the SuperSpeed wires, 2 wires for DP/DM for USB 2.0 as well as 3 wires for Vbus/GND. USB 3.0 does not use polling and its power control can be configured at multiple link levels, and it has a maximum power supply to the bus of 900 mA. We use this USB to make a connection between our microcontroller and the Raspberry Pi mainly. There are other uses for the USB connection like connecting I/O devices to the Raspberry Pi so the user can interact with the User Interface, but we want to highlight the connection that is dedicated to data transfer within the system.

I2C Standards: I2C bus is a de facto world standard that is now widely used in electronics. It is highly versatile and so it is included in multiple control architectures such as System Management Bus (SMBus), Power Management Bus (PMBus), Intelligent Platform Management Interface (IPMI), Display Data Channel (DDC), and Advanced Telecom Computing Architecture (ATCA). I2C is a bidirectional, 2-wire bus for efficient inter-IC control. It only requires 2 lines; a serial data line (SDA) and a Serial Clock Line (SCL). All devices that are connected using I2C can be software controlled and can act as controller-transmitter or controller-receiver, which is how we will implement it in this project. Furthermore, I2C is serial, 8-bit oriented, and can make transfers up to 100 kbit/s in standard mode and up to 3.4 Mbit/s in High-speed mode, which is its fastest for bidirectional data transfer. We take advantage of the speed of I2C buses and rely on it heavily for the data transfer from sensors to microcontrollers and to motors and servos.

3.1.1 - Design Impact of Relevant Standards

Our soft vine robot project must comply with all related standards associated with this project. These standards will ensure our design process is reliable and conforms to all the legal constraints that may occur during the timeline of the project. There are several technical standards related to applying this invention to performing surgery on patients. It will not be feasible and we do not plan on implementing any work that falls under human testing (due to uncertainty and a high degree of randomness), however, we will be paying attention to the standards and guidelines related to the design of the soft robot. We are going to avoid testing the efficiency of the robot in delicate aquatic ecosystems where they might be a threat to endangered species presently accommodating the area. Since soft robotics is still new and growing in the industry, there are no existing standards related to how a soft robot should be manufactured under certain circumstances. On the other hand, we will be applying the ANSI r15.06-1999 which is a standard based on the industrial manufacturing and integration of industrial robots and their overall safety use in the industry. These standards include performing hazard identification and risk assessments such as controlling the speed of the air pump using a motor controller and valves to minimize the degree of randomness once the worm expands through the obstacle course. Also, according to standard 5.3 of our soft worm robot will be firmly secured with a cap on the outer tip of the worm robot. This will allow the robot to have all the hardware components firmly attached to it and avoid parts being launched in random areas once the robot pulls back and contracts to its base.

3.2 - Realistic Design Constraints

Although soft robots allow us to do a variety of things not previously possible by conventional robotics, soft robots do carry their subsequent drawbacks and constraints. Soft robots excel in not damaging delicate environments and being

able to fit in tight spaces thanks to their rubber and pneumatic composition, however, it is these same elements in soft robots that introduce randomness, and thus constrain the ability of soft robots. Their constraints lie in their predictability, controllability, precision, and compatibility with other hardware.

Air is a gas and is thus inherently chaotic. This poses an issue for soft robotics since they are controlled pneumatically (although you can find the occasional hydraulically controlled soft robot). Not only does the air introduce chaos to soft robots, but their soft exterior also introduces randomness into the mix. Soft materials, such as rubber can move in many different ways since they are not bounded by axes of rotation like traditional robots. However, it can be difficult to predict the movement of said materials thanks to their freedom in movement. Predicting movement is a difficult thing for soft robots and there are even specialized software like SOFA (Simulation Open Framework Architecture) that are used to predict how a soft robot will move once it is built.

Thanks to the unpredictability of soft robotic movement, trying to control their movement is similar to trying to grab a bar of soap. When you grab a bar of soap, you have to be careful not to grab it too hard or too soft because a bar of soap is so slippery that it will react to any slight unevenness in pressure and slide right out of your hands. One approach that I like to use in the shower is to grab the bar of soap in such a way that the places it can go are limited. By limiting the place the bar of soap can go I am reducing its unpredictable nature of it. This bar of soap analogy can easily be applied to soft robots. For example, to control the contractions of soft robots, you will see people using various kinds of molds designed to restrict the movement of air and force air to the places necessary to make the mold actuate in the desired manner. Molds have to be designed in such a way that there is only one likely manner in which the air will cause them to actuate. If there are any inconsistencies in the mold, the air will randomly fill those imperfections in the mold and cause an undesired result. Imagine a rubber glove; When you inflate a rubber glove you can observe that the palm of the glove expands much more rapidly than the fingers, and if you inflate it too much the glove will quickly be morphed into a shape that was not intended by the manufacturer. This same phenomenon can happen with soft robotic molds. Molds are only meant to have a certain amount of pressure in them before they start to behave in undesired and unpredictable ways. The more pneumatic pressure there is in a mold, the more chaotic the air inside of it becomes, and the more volatile its shape of it gets. This is a principle taken from thermodynamics. When air is pressurized, its temperature increases,- and the movement of the atoms increases because they are excited by the newly introduced thermal energy.

When it comes to vine robots, we are looking to control the direction in which they grow. There are several ways to do this, and the chief among these methods are pneumatically controlled actuators that are called artificial muscles. These artificial muscles are long, inflatable, rubber tubes that attach to the vine robot. Once you introduce pneumatic pressure into them they will expand and thus turn the robot's

body in the opposite direction in which they are placed. Multiple artificial muscles can be combined to produce complex directional motion in these vine robots. These artificial muscles can be integrated either inside or outside of the robot, and each method of integration has advantages and subsequent drawbacks. Integration of the artificial muscles inside of the robot allows it to turn at a sharp angle since the axis of rotation is inside of the robot's body as opposed to outside of the robot's body.(**Figure 3.201**) However, when the robot turns it is permanently committed to turning in that direction until the vine robot is retracted and redeployed. This kind of controlled movement is perfect for an obstacle course since the robot can turn in multiple directions and can do so at a sharp angle. The other kind of integration of the artificial muscles is outside of the vine robot's body. This kind of integration allows the robot to change the direction in which it turns, but it can only turn in one direction and it does not turn quite as sharp as internally integrated artificial muscles since the axis of rotation is now on the outside of the robot
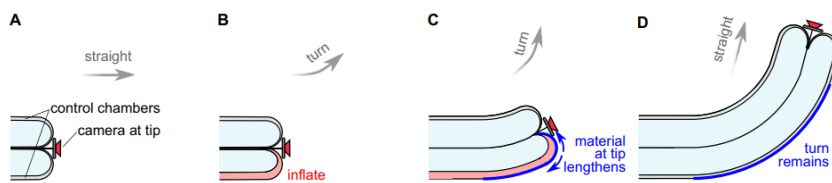
Figure 3.201- Turning Vine Robot
(reprinted with permission from vinerobots.org)

Vine robots also have a very hard time retracting once they have expanded. If you try to retract a vine robot by pulling on a string connected to its tip, it is very likely that the robot will buckle unless it has not turned at all in its elongation. To allow a vine robot with a complex shape to retract without buckling, researchers have designed an internal roller which allows the robots to retract from the tip by slowly rolling the material back in on itself.

Another unique constraint related to soft robots is just how hard it is to get the robot to comply with other hardware. Attaching sensors to the rubber exterior of the robot is very difficult to do with any degree of secure attachment, and it can be even harder to route cables to a power source or to separate communications hardware. One solution that researchers have developed is the implementation of an outer cap to the vine robot's tip. This cap routes its wires through the body of the robot and allows the user to have a hard surface on the end of the robot that is perfect for sensors, actuators, or any other mechanical devices that need a more firm foundation to be implemented. On this cap the users can place a gripper, a set of robot vision sensors, a flashlight, a camera, and many other things. There are even magnetically attached tips that can be quickly interchanged thanks to their modularity. This allows the robot to have a variety of functions. Having a tip on the end is a huge part of vine robots because the tip gives the robot's a lot more versatility apart from just growing and retracting. The combination of the internal

12

roller and a cap on the tip of the robot is very common as it allows the robot to do two critical tasks which are elongation and retraction for redeployment or correction of movement and a task function which the cap can carry out such as gripping an object.

Another important constraint of vine robots and soft robotics, in general, is the lack of precision associated with them. You can imagine that hard, metallic robots can be more precise thanks to their components not having a lot of variation associated with them. However, in pneumatically or hydraulically controlled soft robots, this is not the case. Since the rubber exterior of soft robots has a lot more hive than a metal component, it is subject to a lot more variation, and thus it is not precise.

3.2.1 - Economic and Time Constraints

Economic and time constraints are usually the biggest constraints engineers face when designing or conducting research on a new project. Likewise, we are faced with these obstacles since all the team members are current university students. We are the sole financers for this project, and so we have to treat the budget with care. We also plan to reach out to university professors to see if they would be interested in sponsoring our project and allowing us to use their labs, but moving forward, we can only rely on what we can each contribute. Therefore, parts selection such as material, sensors, and the camera have been accommodated to reflect what we can currently afford while making sure that it is up to par with our design standards.

There are many technologies that have been used to create similar projects. We had to weigh the pros and cons of said technologies to come up with the best affordable and feasible solution for us. I mention feasible because most of the research that has been done on the soft vine robot area has been over a large period of time with sufficient resources to rely on trial and error methods. We will conduct specific component testing to make sure that we do not have to resort to trial and error and end up wasting resources.

Another great obstacle, and probably the greatest of them all, is time management. Time being a huge constraint is to be expected of any project as just with classes, customers also have deadlines for engineers to meet. During the design portion of the project, we spend a great majority of the time doing research and making sure that we are making the right decisions and selecting the best components. This process probably takes from two to three months, at the end of which, we start elaborating the design with a clear idea and goals in mind. The remaining time is used to actually work on the project, and even then, the time dedicated to solely working on the project is scarce. All team members are students and therefore have other responsibilities alongside this project so meeting times are harder to arrange due to conflicting schedules. This constraint was expected and to make sure we are moving at a steady pace we are trying to conduct weekly meetings and biweekly meetings towards the end of the project design. We are going to set realistic goals to be achieved in an agile methodology where we have weekly

sprints to make sure that we keep making progress. On top of this, maintaining constant communication via chat is crucial for any casual updates.

3.2.2 - Environmental, Social, and Political Constraints

Asides from economic and time constraints, there have been environmental, social, and political constraints when designing and conducting research on this project. During the design project, we wanted to design a soft robot that mimics the mannerism of a worm or snake because they have the unique ability to maneuver obstacles in their path. However, since we have not had previous experience with designing and building soft robots, we were very ambitious in creating a maze for our worm robot to navigate through. For example, we wanted the robot to go deep underground and go past sticky and watery surfaces. However, this put a constraint on how realistic the project is so we decided to focus on three main maneuvering tactics and obstacles. We have also encountered some social constraints during the brainstorming section of this project since this project was what we came up with. It was a bit difficult going along with the project since we have not encountered the realm of soft robotics so we hardly found reference files to work with. Although this gave us a unique perspective on how to approach certain challenges within the project, our advisors gave us some realistic comments on how feasible the project can be. Fortunately for us, we were able to minimize the features we had currently such as removing solar panels and lidar sensors. As for political constraints, there have been certain issues with ordering parts from foreign countries such as China, and how there can be potential issues as we are currently experiencing the tension between Russia and Ukraine which has put Russia in a tight condition where they cannot export goods to other countries in the world. Regarding the social constraints, this type of project is still new in the industry and does not have a lot of existing models that can be applied to the project.

3.2.3 - Ethical, Health, and Safety Constraints

In terms of ethical constraints, it can be seen as controversial for soft robotics to be applied for human locomotion. When designing the robot, the degree of randomness of the vine robot has been known to be notoriously high which has been a safety issue especially since we plan on installing hardware components at the tip of it. We deliberated on this part of the project a lot and did some research. We eventually found a research paper that discussed the technique of attaching a cap to the tip of the vine to prevent buckling which is the automatic retraction of the vine robot once it has completed the obstacle course.

We also wanted to include photoresistors in order to detect light. We planned on having around six of them around the cap on the head of the vine's body. However, upon learning the toxicity of the material, we have decided to move in a different direction. Photoresistors use materials that are poisonous and have been banned in Europe which would limit our market expandability. So we are instead using

phototransistors to achieve this goal of detecting light and avoid this health and environmental risk.

3.2.4 - Manufacturability and Sustainability Constraints

There is another great constraint that we face that while it may not seem as important now, we will definitely feel the negative effects of it, and that is not having a manufacturing facility. Unlike most vine robot research, we do not have all the required tools readily available. We have to consider this as it may affect both our time and budget as we have to buy or rent tools for development.

Currently, we are restricted to working in the UCF Innovation Lab and using only the tools they can provide. It is a good resource but it will be highly inconvenient at times since they might not have everything we need. Furthermore, we are likely not going to be the only team that will be using the Innovation Lab which can make accessing tools a bit harder. A solution to this problem will depend on our ability to find a professor that would like to sponsor us and let us use their labs if they have one available. Working with a professor can help us save time and resources and even provide assistance in the form of mentorship from the professor themselves or any present grad students.

Finding the parts can be yet another difficulty in itself. Of course, we will try to select the best possible materials and components that are both useful and easy to acquire. However, there is a shortage of materials due to the backed-up supply chain issues caused by the global pandemic. I believe it is fair to say that is an unpredictable scenario whether materials will become unavailable in the near future or if stores will have their shelves restored by then. The only way to avoid this is by conducting thorough research and selecting parts that are currently available.

In terms of sustainability, we cannot attest that the materials we are using will be completely compatible with the textiles we plan to use for the soft body. Sensors need to be flexible enough and chips and electronics need to be well stored inside the body of the vine. The purpose of this robot is for it to be able to navigate different terrains and tight spaces so we need to make sure that the textile we use is durable and none of the electronics gets damaged during the navigation.

4 Research & Background Information

Planning, researching and eventually designing a product is all a part of the engineering process that we must dedicate a substantial amount of time to. To do so, the team members must have a clear understanding of the goals and overall objective of the product. We have looked at various designs with many features and properties which not all have worked out for us and some that required us to make sacrifices in features. Furthermore, during the research aspect of this project, we redesigned our initial idea as well as redefined our project objective so that it better aligns with the specifications. In this section, we will detail what the aforementioned research entails. We dive into some of the preexisting products, studies, and components that will make up this robot. This will provide a better insight into the thought process behind the selection of materials and component design.

4.1 - Existing Similar Projects and Products

When starting any project, it is critical to research similar products and projects. Furthermore, as this is a relatively new technology, there isn't any available input from customers or current marketing analysis on competitive products utilizing soft vine robots. There is, however, research being conducted on how soft robotics can be used in the form of a vine-like design, similar to what we are aiming for in this project. The following section will discuss similar projects and research, as these designs serve as the inspiration for our soft robot.

4.1.1 - Vine Robots

https://www.vinerobots.org/

Vine Robots is a soft robotics project supported by Stanford Applied Nanophotonics and the IRiS lab of the University of Notre Dame. Their robot, built with low-cost flexible material, mimics a vine by retracting and growing with pressurized pneumatics. The robot can also autonomously avoid obstacles with pre-programmed haptic controls. It can be used to navigate difficult environments, tight areas, and hard-to-reach places. Similar to a root, the robot has a base, from which it can grow and expand up to one hundred times its length. Furthermore, applications of this robot include search and rescue, deployable structures, and medical procedures.(**Figure 4.1.101**)
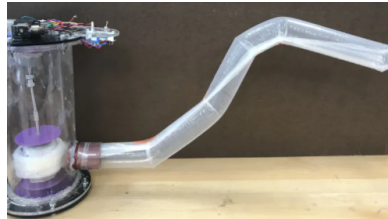
Figure 4.1.101 - Vine Robot
(reprinted with permission from vinerobots.org)

The vine robot, as seen in the figure above, can be both pre-programmed to follow a specific path or can be manually steered to navigate an area. Its base holds a tubular roll of plastic which turns and extends the plastic with pressurized pneumatics, as the pressure pushes out the plastic material with actuation. The roll of plastic is also motorized, allowing it to retract growth. Overall, the vine robot is one of the biggest inspirations for our project; similar to the vine robot, we hope to incorporate a manual mode and autonomous mode into our design.

4.1.2 - MIT Autonomous Earthworm Robot

https://news.mit.edu/2012/autonomous-earthworm-robot-0810

Researchers at Massachusetts Institute of Technology, Harvard University, and Seoul National University have developed a soft autonomous robot that moves on surfaces by contracting parts of its body, similar to that of an earthworm. The worm is only a few inches long and can be used to navigate difficult and rough terrain, along with breaching tight spaces that no one human could navigate through. Furthermore, the robot is quite flexible, resilient, and does not break easily. The material consists of artificial muscle developed from wire made of nickel and titanium; this design allows stretching and contracting of the material with heat. Moreover, the wire is wound around a tube, which creates segments, similar to real earthworms. A current is then applied to the segments, causing the robot to move forward.(**Figure 4.1.201**)

Figure 4.1.201 - MIT Robot
(reprinted with permission from vinerobots.org)

Within the tubing in the worm, as seen above, there is a small battery and circuit board that generate a current to heat the wire at certain segments. This heating of the segments allows for the movement of the worm. The robot uses algorithms to control the heating and cooling of the segments, which allows the worm to move in various different patterns.

4.1.3 - GE Autonomous Tunneling Worm

https://www.ge.com/news/press-releases/fatbergs-beware-ge-research-demonstrates-autonomous-pipe-worm-robot-that-can

General Electric Research's robotics team developed a soft robot, similar to that of an earthworm, that autonomously tunnels and navigates around underground obstacles, tight spaces, and small rocks. It can function quite reliably in rugged terrains and can also dig underground. GE demonstrated this robot by having it tunnel underground at GE's research campus at a distance similar to current trenchless technologies. Furthermore, this robot is powered by fluidic artificial muscles that are able to squeeze through small rocks and obstacles.

Furthermore, GE has expanded on this initial research by developing an autonomous "pipe-worm" robot that can demolish solid waste deposits. This design incorporates cockroach-like whiskers to the current fluid-powered muscles that allow it to have the flexibility and advanced perception capabilities. This robot can demolish solid waste deposits, known as "fatbergs," which are currently a problem in many municipal sewer systems. Along with solid waste removal, the robot can also map oil and gas pipelines or municipal water systems. These underground systems can be monitored, inspected, or repaired with this autonomous soft worm technology. Furthermore, during GE's demonstration of the robot, the worm was able to traverse over 100 meters of pipe while navigating through turns, liquids, and diameter and altitude changes, all while not disturbing normal operations. The robot is also able to detect the orientation of the pipe it is currently within. Overall, this project also serves as an inspiration for our current design. We hope to incorporate a method for our worm to detect where it needs to turn and its current orientation.

4.1.4 - Harvard University Soft Robotics Research

https://gmwgroup.harvard.edu/soft-robotics

Harvard University's Whitesides Research Group is actively conducting research on soft robotics. The group has researched various ways on how soft robotics can be made and how they can be used. One example of their current research is how elastomers can be used for soft robotics through actuation. They have developed a soft robotic gripper made of elastomer that can be used to grip an uncooked chicken egg. A tube is used on the gripper to provide pressurized air for actuation, which allows for opening and closing the grip.(**Figure 4.1.401a & b**)

Figure 4.1.401a - Harvard Soft Robotics Gripper
(reprinted with permission from vinerobots.org)

Furthermore, the research has also developed systems that combine soft and hard materials for robotics. They developed a hybrid robotic system that uses a wheeled robot and a four-legged quadruped soft robot. The wheeled hard robot is utilized for rapid travel over flat terrain, while the soft robot can be used for gripping and retrieving objects. The research group demonstrated this technology by having the robot attempt to retrieve an iPod Nano object from the center of a room, which it was successfully able to do. We hope to test our robot's functionality by having it retrieve an object at the end of the obstacle course.
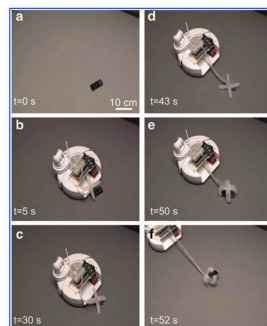
Figure 4.1.401b - Harvard Soft Robotics Soft/Hard Robot
(reprinted with permission from vinerobots.org)

4.2 - Materials and Development Procedures

In this section we go over different materials that have been previously used to design similar projects. We compare these materials to ensure that we use what is necessary for our target environment and goals. We also discuss many development procedures relating to growth and navigation, as well as some of the methods used for mounting tools at the head of the robot. This is relevant to our material and parts selection since the methodologies we use will directly affect what we can use so no component gets damaged.

4.2.1 - Materials and Manufacturing

The material of the main body of the vine is a special type of material that differentiates this project from the typical robot as it needs to be soft and malleable. The material is required to be inextensible enough so that it can produce eversion as opposed to radial expansion upon pressurization. It also has to be fluid and impermeable and sealed. Different designs will have or require different materials. Terrain navigation will also have a significant effect on the material selection as we do not want a material that will have excessive friction or be prone to possibly deteriorating along the path. The construction of the body is similar in every case and very simple as it only requires a sealed head and an entry point for the pressurized air. Furthermore, we leave a summary of some of the materials and how they compare to each other in Table XX.

Material	Behaviors	Manufacturing Method
Thermoplastics (LDPE, TPU)	Fastest prototyping Material uniformity Low burst pressure	Heat sealing/ preformed
Thermosets (latex, silicone)	Slow prototyping Variable burst pressure Low hysteresis	Casting/ preformed
Thermoplastic-coated fabrics (TPU-coated nylon)	Fast prototyping Moderate burst pressure Good structural characteristics	Heat sealing
Thermoset-coated fabrics (silicone-infused nylon)	Slow prototyping High burst pressure Lowest eversion friction Extensible/ inextensible	Adhesives
Uncoated fabrics (ballistic nylon)	Slow prototyping High structural strength High eversion friction	Sewing with internal bladder Ultrasonic welding

Table 4.2.101 - Materials & Manufacturing

4.2.2 - Actuating Length Change and Direction

Growth: The entire premise of the vine robot is based on the way it grows, bends and finally retract. The growth is induced by the fluid pressure inside the robot that makes the tail, which resides within the body of the robot, to evert at the tip and thus incorporate into the robot body wall while the outer wall stays in place. In other words, the robot grows from the inside out. There are two common ways to store the body of the robot in order to achieve this everting feature. We can create a closed tube of robot body material with a pressure input from the base. The tube

can be inverted on itself and shortened to half its original length while storing the tail straight inside. The other method for storing the body is mostly used when we want to change the length by more than 100%. To do this, the robot tail must be stored in a more compact form or outside the pressurized area of the robot body. However, due to the complexity of this method, it has not been demonstrated, but there have been many examples showing the body stored rolled up on a reel with arbitrary growth lengths. The best implementation observed was where the body was stored in a reel that was inside a pressure chamber, or a base, which was used as a rigid grounding point to attach the body outside walls.

Retraction: As opposed to growing, retracting the body of the vine can be quite complex and in some designs, unachievable. It usually is not as simple as reducing the pressure distributed to the body, unfortunately. To retract the body, we must exert a force on the tail to pull it towards the base while a moderate level of pressure is maintained in the body. There have been research experiments that have accomplished this feat by using a motor that controls the reel that stores the body of the robot, which not only helped with retraction but the growth as well. However, there are more complications that arise when studying the retraction of the body of the robot since, under certain conditions, it is prone to buckling or bending in undesired ways. To eliminate this we can design a retraction device that applies pressure to the tip of the body and thus making buckling no longer a problem. When using this method we have to keep the tension at the tail of the body to a minimum, that is, if it is desired to keep the roller and motor design.

Direction: Steering and everting vine robots presents major design constraints and challenges in the process of manufacturing as well. The design of the vine also dictates how the direction can be handled. There are many characteristics that will affect steerings, such as the length of the body and the radius of the body. Some other design specifications to consider are the number of actions needed for sufficient control, scaling of actuator magnitude, and speed with the length of the body. Most of these designs are application-dependent. However, the general methodology for actuator growth direction is that it works by adding thin, flexible tubes alongside the body which shorten or lengthen in order to cause the vine to steer. Some studies detail some of the ways these flexible tubes can be used to steer the robot.

Distributed Strain Actuation: In this type, the actuator contracts uniformly across its length so that in a single input, the entire body of the robot curves uniformly. This uses soft pneumatic actuators that can extend to the size of the entire body length. This inverse pneumatic artificial muscle (IPAM) works by contracting as tendons do with the muscles of the human body. There are other versions as well that do not require the actuator to contract uniformly throughout the body as well.

Concentrated Strain Actuation: This is an alternative to distributed strain actuation in which the actuation comes straight from the base of the body instead of dispersing evenly throughout. This method has been generally accomplished by having the tendons routed along the surface of the pressurized tube and pulled by

21

DC motors. However, this method decreases local stiffness which in turn means that bending due to tendon actuation will concentrate in a single point, which can furthermore limit the usage of this method.

Tip-localized Strain Actuation: In the previous actuation methods, there is a constraint to the shapes that can be achieved by the robot. Research shows that while independent steering control along the full length of the robot as it grows, we can instead actuate it by coupling the steering to the growth through tip-localized strain actuation. Having mechanical latches hold preloaded strain and can be unlatched when they reach the tip by pressurizing pockets that run along the entire length of the robot.

Preformed Actuations: Unlike the previous methods that create actively created curvatures and steering for the robot, this method consists of having presets of where the actuations will occur. There have been two methods to do this which basically indicate that we can mold the body of the vine itself by heating the thermoplastic material and the other method uses tape at certain positions to bend the body as it grows out of the tail.

Passive Environment steering: This method allows the robot to grow and adapt to its environment as it runs into walls or obstacles. By allowing the robot to easily deform this can be accomplished but it is cautious to notice that this can also cause unwanted behavior in navigation.

4.2.3 - Mounting Sensors and Tools

Many applications require the vine robot to contain sensors and cameras that allow it to be fully or semi-autonomous. This equipment has to be mounted in a way that it can be carried with the body and give feedback on the environment of the robot. There are a couple of locations that have been explored and depending on where they are fixed, they will be fixed to the material of the body of the robot or they may move along with the body itself. Below, we present some of the findings on how the location of sensors are important to the design and the application.

At the tip: This is an important spot to place sensors and cameras as this is the first part of the body that enters an area and can serve for guidance navigation and exploration. Besides sensors and cameras, there is also the possibility to mount other components such as a gripper, which allows for interaction with the environment. However, while it sounds like a convenient place, it can be challenging as this everting robot is always changing and growing from the inside out. There has been research on ways to attach sensors to the tip including cables inside the tail, friction with the wall, and rolling interlocks. All of these methods use both the inside of the pressurized part of the robot and the outside walls. This means that wire handling can be challenging. Another tradeoff of having the sensors at the tip is that they can add weight to the front of the robot and limit the natural ability to move through confined spaces.

Fixed to the wall: This method also puts the sensors in direct contact with the environment. This is well suited for sensors that need to interact with the environment along the entire length of the body. However, everything mounted needs to be flexible enough to be everted and inverted in retraction to avoid corrosion.

Inside the pressurized area: Some items may not need to interact with the environment, in which case this location is perfect for them. This is best for sensors and items that interact with the robot itself instead. This is a possible position for a retraction device that helps retract from the tip without causing buckling. There are many other applications such as wire carrying that are better suited for this location as long as they do not need direct contact with the environment.

Fixed to the tail: The robot tail moves at twice the speed of the tip due to the eversion, which makes this location good for transporting items between the base and the tip. Items fixed to the tail can come in contact with the environment once they reach that point and move to the outside, at which point they can go into the environment or stay stuck to the now outside wall. This protects sensors from any damage until they have to come out for interaction but it has the tradeoff that mounting of the sensor can only be done during manufacturing and it will be fixed so we must know the final length of the robot body itself.

Inside the tail: This location allows us to overcome the shortcoming of fixing payloads to the tail since they are inside the tail but not fixed. While this method contains the same benefits as the previous section, it falls behind on the fact that storing the body material in a reel is impossible because of the need for relative movement between the tail material and the items inside the tail.

4.2.4 - Robot Control and Planning

Controlling vine robots is also unlike controlling most robots that we have previously encountered. The structure of its body provides a new mechanism for the type of movements that it is able to do. Likewise, controlling it also has its challenges when it comes to both robot-level control and autonomous steering. Some of the considerations to keep in mind are the behaviors that can be planned ahead for and also how we are able to have a user operate it remotely. There have been studies that detail their experience with robot-level design, the design of a human interface, and prior planning methods to consider obstacles and environment interaction.

Robot-level control: There are a couple of strategies that have been developed in order to design a successful method for controlling the fundamental movements of the robot. These are related to two major areas which are growth/retraction and steering.

Growth/retraction: To maintain the desired speed at all times, we can have the robot use a stepper motor in the base reel or a back drivable DC motor with an

23

encoder in the base reel which could also be combined with a DC motor in the retraction device at the tip if that is the desired route for retraction. When it comes to the pressure we can opt for a pump connected to the base that is always on with a passive relief valve or a closed-loop pressure regulator that is connected to the base.

Steering Control: We can have a camera at the tip of the robot that sends a command for moving either left, right, or straight that interacts with the pressure commands and has solenoid valves connected to side chambers. We can also have the camera send video feedback only to the user and the user can control and steer the robot using a joystick that interacts with the pneumatic controls.

Planning: Planning is one of the few sections that previous research has either diverged from or indicated its difficulty. Planning proves to be harder since there are lots of constraints that affect the robot when actuating, most of which are environmental. Another key component in the planning for the path and steering of the robot is the actuation design which has an influence on the growth. Due to the nature of the robot and its movement capabilities, planning and modeling are very limited which usually leads most research teams into a trial and error phase.

4.3 - Overall Block Diagram
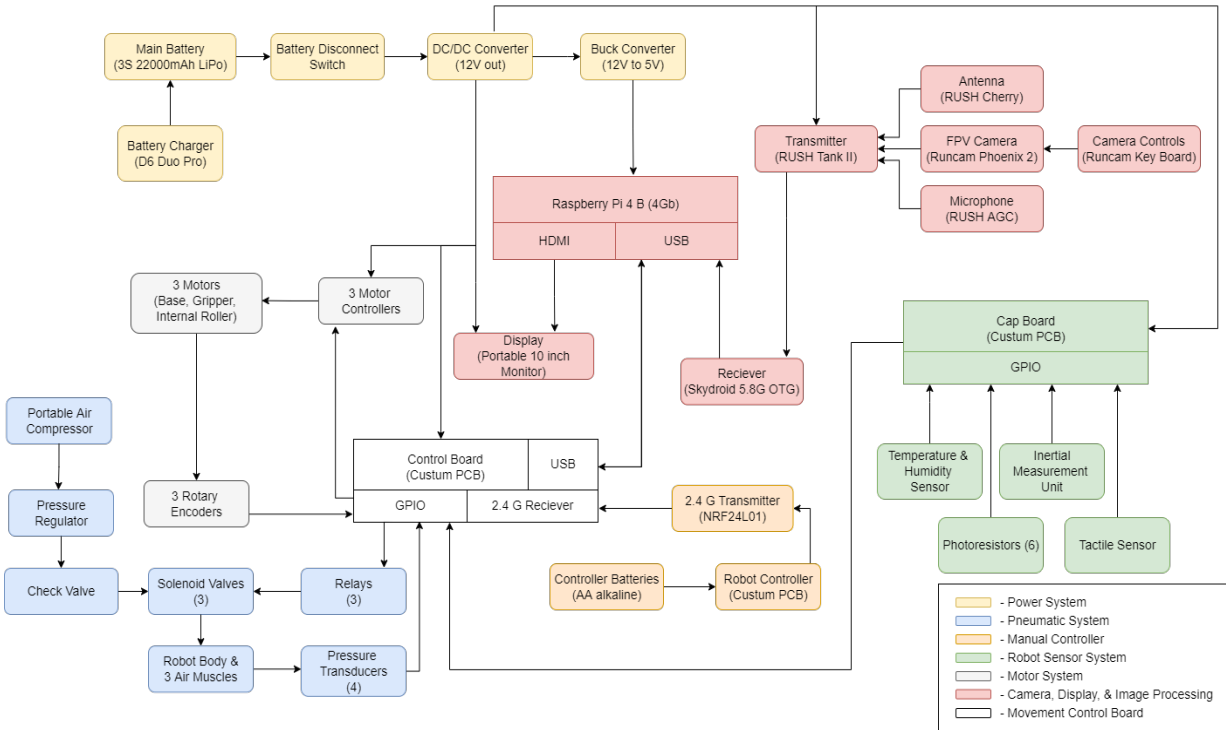
Figure 4.301 - Master Block Diagram

4.4 - Strategic Component and Part Selection

Here we leave in detail how and why we have selected the components for the project. This is a combination of all the research the team undertook and, of course, with great consideration of the team goals and the series of constraints that we have identified previously.

4.4.1 - Parts and Components

Choosing components and parts was a challenging task, given that there is high codependency for most all of the parts. In this section, we cover what are the parts that we will be using to develop this soft robot. There are some components that are going to be custom-made, whether 3D printed or assembled with materials.

The power system is composed of the batteries that we will be using for other components that depend on power. We chose to have a single battery that will be connected to various components throughout the body and head connected with internal wires. The battery we are using is the URUAV 3S LiPo XT90 Battery. This will be connected to a battery disconnect, a DC/DC Converter, and a Buck Converter to be able to reach the Raspberry Pi. This battery will also be connected to the D6 Duo Pro which will be its battery charger.

The components related to the software and image processing are also some parts that will be outsourced. The camera will be the RunCam Phoenix 2 which is a mini FPV camera that is used for drone piloting. The camera provides the required specs for computer vision applications. The camera will be connected to the RUSH AGC microphone for audio input and a RUSH Antenna which are all connected to the RUSH Tank II transmitter. On the other end, we have the Skyrold 5.8G OTG receiver. This receiver will connect to the Raspberry Pi 4 Model B. This Raspberry Pi will act as the main mediator for the user input and output by displaying the data feedback on a computer display.

The sensors in the head will be relaying information to a custom PCB. We want the user to have as much possible relevant information as possible. To do so, we will have, besides the camera, an MLP3115A2 I2C Barometric Pressure/Altitude/Temperature sensor combo so that we can better understand the environment. We also have navigational data incoming from the ADXL345 for acceleration and angular rate.

The user input is mainly received from a custom-made robot controller which will include another custom PCB. We aim to use the NRF 2.4 L01 Transmitter to wirelessly transmit the signal to the PCB that is connected to the Raspberry Pi as a stretch goal, but at the moment we are going to send said information through wires. It would be helpful to have a wireless setup, but it also proves to be difficult as we might require an extra MCU and software accordingly. This information will command how the pneumatic controls will work. These pneumatic controls are also

going to be designed by the team. Some other components that the team will build are the gripper, base, and roller, which will be each connected to a motor.

4.4.2 - Methodologies

The methodology explains the ways we have chosen to build the vine robot. First, we have decided to make the growth and retraction at a constant rate. This relieves the user from having to control how much it needs to grow and when. It will also minimize the opportunity of error in case the user retracts and grows rapidly which would result in loss of air pressure without any progress. To make the growth easier we are going to store the body in a reel inside a pressure chamber which is our base. For retraction, we are designing a retraction device that will go in the head. This device will consist of rollers that will prevent the body of the robot from buckling once we start retraction. When it comes to steering, among the methods presented, we chose to go with tip-localized strain actuation. By coupling at the tip when we turn this lets the body conserve its shape. While this feature is not necessary or required, we believe that it will grant the user to steer without worrying about the rest of the body turning and knocking things over or running against walls and damaging its material. For mounting the sensors and tools we chose the head, or tip of the body, as the location. This is mostly due to the safety that it can provide for the hardware being stored inside the cap. This way we can also guarantee that we see the environment first and receive the most updated data from the sensors. The wiring can be sent from the base to the head using the inside walls of the inverted body.

4.5 - Part Selection Summaries

This section of the report will lay out and detail the parts that have been selected for our project. The parts can be seen in the table below, along with the quantity required.

The goal for the project's budget would be to keep the cost under $250 and to have funding through the university. Most components we will be using will be SMD and soldered onto a flexible PCB, which should keep our costs low. Below is a cost estimate of our project, along with a range of what our total cost is projected to be with the current components.

5 Mechanical Design

Given that we are designing a soft vine robot the mechanical design will be a fair bit different from your traditional robot. We must account for the unique properties of the materials we are using so that we can produce a functioning product. The Mechanical Design will discuss all the mechanical systems of the robot and how they work together.

5.1 - Design Overview

As shown in **Figure 5.101** the mechanical design will take into account several crucial components including an air compressor that will supply all the pneumatic pressure to the soft components, pneumatic controls (a system of solenoid valves, pressure regulator, pressure tubing, and tube fittings), a base that will house the spooled vine robot body and a motor, the robot body which will be a pneumatically controlled rubber tube used for elongation, the artificial muscles which will also be pneumatically controlled rubber tubes used to steer the robot, the internal roller which assists in retracting the robot, the cap which will mount on the tip of the robot and house some electronics, and the two-finger gripper that will mount on the cap of the robot.
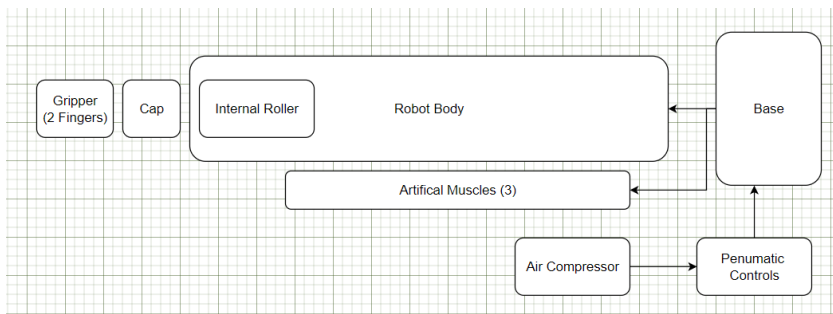
Figure 5.101 - Mechanical Block Diagram

5.2 - Initial Design Architecture

In this section, the theory and objectives of the mechanical components will be depicted. Each section will evaluate different design solutions and will discuss the reasoning behind different design directions within the mechanical components. Without talking about part selection or component construction methods, the designs for the body, steering muscles, tip mount, internal roller, base, and pneumatic controls will be covered.

5.2.1 - Lengthening Through Eversion

The moving parts of the vine robot will be made up of the main body tube and several auxiliary tubes that will assist in steering the robot. It is important for these tubes to be made of a polymer that is flexible enough to allow the robot to move in any direction. There are several options of fabrics and plastics that could potentially hold the desired properties. Plastic tubing can usually be attached together and formed into shapes using heat to seal sections together. Fabrics will usually have to be stitched together. There are also options of plastic-lined fabrics, which could also do the trick, although they may also be a bit heavy. Regardless of what material is picked for the body and steering chambers, the robot must be constructed in a way that preserves the lightweight nature of soft robotics.

The robot should be able to expand in the direction in which it wants to navigate and retract when it is time to explore a new path. In order to expand and retract, the robot will expand from the tip through a process called eversion. First, the thin-walled polymer that makes up the body of the robot will be inverted and then the vessel will evert from the tip and expand when pneumatic pressure is introduced. Eversion of this material with pneumatic pressure allows for substantial elongation at relatively high speeds. The internal pressure forces of the pneumatics force the inverted tubing to evert at the tip while pulling more material through the middle of the tube (**see Fig. 5.2.101**). The robot can continue to pull material from the center of its body so long as there is material available at its base. Ideally, this material can be stored in a spool and dispensed as needed using a motor. One of the benefits of eversion is that because only the tip moves, the robot is not sliding through the environment in a way that a snake or worm would. This means that the robot is not generating any friction, which is important for applications in fragile or delicate environments in which soft robots often find themselves in.
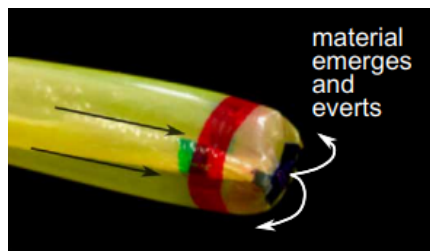
Fig. 5.2.101 Polyethylene tubing eversion
(reprinted with permission from vinerobots.org)

5.2.2 - Reversible vs Non-Reversible Steering

In order to steer the robot steering chambers will be placed on the sides of the robot and will be attached to the main robot body. These steering chambers can either turn the robot in the direction in which they are placed or in the opposite

direction in which they are placed relative to the robot body by expanding with pneumatic pressure. Vine robot steering can be divided up into two camps: reversible steering and nonreversible steering. Reversible steering refers to the ability to change the steered direction of the robot without having to retract the robot. Nonreversible steering refers to steering mechanisms that do not have the ability to change the steered direction of the robot without first retracting said robot. The steering chambers can also be positioned in one of two ways depending on how much directional control the user would like. For creating shapes with the vine robot in two dimensions, two chambers are required in order to steer the robot in either the left or right direction. For creating shapes with the vine robot in three dimensions, three chambers are needed in order to steer the robot left, right, or up by partially inflating the three chambers in different combinations.

In order to achieve nonreversible steering, must have a mechanism for locking its shape as it grows from the tip. This can be achieved in a variety of ways. One way includes having a series of latches that are embedded into the steering control chambers, with each latch crossing pinched material. When the latches are released, the steering chamber they belong to lengthens and the robot is steered in the opposite Non-Reversible releasing of the latches can be activated by pressurizing their respected steering chamber as seen in **Fig. 5.2.201**.
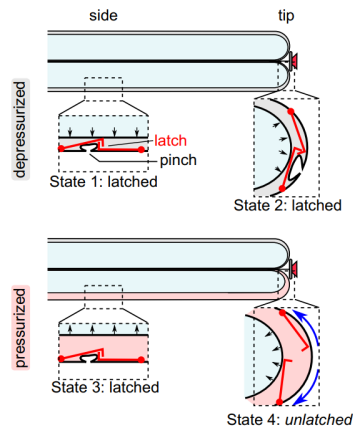
Figure 5.2.201 - Demonstration of latch locking mechanism
(reprinted with permission from vinerobots.org)

As the steering chambers are pressurized the latches at the tip open, but the latches on the sides that are already past the point of expansion will remain closed. When the steering chambers are pressurized, the latches remain closed if they are along an already expanded side, because of the shape of the interlocking, but the latches open if they are at the tip because of the curvature overcoming the interlocking. In the steering chambers that are depressurized, the latches will remain closed at both the already expanded sides and at the tip during extension.

When the steering chambers are depressurized, the pressure from the body chamber keeps the latches closed regardless of whether they are on the already expanded sides or at the tip. Because of the nature of the interlocking latches, the robot will steer in the opposite direction of the steering chamber that is pressurized. As a steering chamber is pressurized and the latches are released in that chamber, the chamber will have more material to expand than the depressurized steering chamber that has material trapped between the latches.

Another form of nonreversible steering includes tendon-like strings running along the sides of the main chamber with locking bodies that grow through guided tubes and lock the shape of the curvature created by pulling the tendons. By varying the lengths of the tendons (see l_e and l_i in **Fig. 5.2.202**), the body chamber of the robot will be curved in the direction of the shortest tendon. Once a curvature is created in the body chamber of the robot, this posture can be maintained by resisting the change of the lengths of the tendons. In order to resist the tendons from changing lengths in a particular section of the robot that is to have its shape locked, two flexible guiding tubes are attached along with the body chamber and over the tendons. These flexible guiding tubes will have shape-locking bodies within them that can expand through eversion, much like the body chamber. In order to preserve the shape locking functionality and prevent slipping, the inside of the guiding tube and the exterior of the shape-locking bodies are coated in high friction material. Using this shape locking mechanism, the shape of the body chamber is held constant even as the tension of the tendons changes. It is important that the tendons can still slide while being held underneath the flexible guiding tubes. This will allow the tendons to continue to change the curvature of future sections of the body chamber. One of the most important advantages that this form of chape locking has compared to the aforementioned latch locking design is that the steering of the robot remains reversible. Since the shape-locking bodies that run along the flexible tubes expand through eversion, they can be retracted in a similar way to how the body chamber can be retracted. The body chamber and the locking bodies can be retracted by pulling on their tail and wrapping the material back into a spool controlled by a motor. Since the locking bodies are within a guiding tube, they will not buckle. Also since the body camber is essentially within a guiding tube as well, it will also not buckle as it is retracted. In order for the body chamber to not buckle, however, it will have to be retracted in small sections and only after the locking bodies of the desired section to be retracted have already been retracted beforehand.
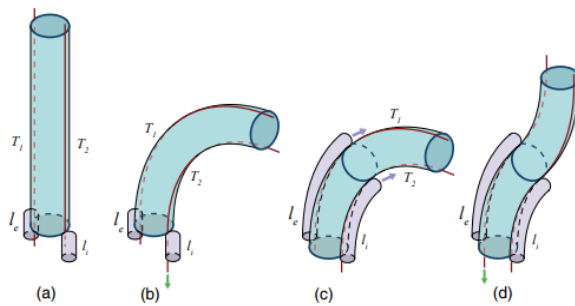
Figure 5.2.202 - Demonstration of tendon and stiff body locking mechanism
(reprinted with permission from vinerobots.org)

The third and most heavily adopted form of steering vine robots is through steering chambers composed of pouch motor actuators. These pouch motor actuators can be manufactured by heat-sealing at regular intervals a series of pouches on the steering chambers, which will cause the chambers to shorten when inflated. These steering chambers can also be attached to the body chamber through heat-sealing as well. As the pouched steering chambers become larger in diameter, the stronger the steering forces will be on the body chamber, and the tighter the turning radius will become. Because of this, the steering chambers should be as large as they can be without interfering with the function or touching the other steering chambers. This is not so much an issue when there are only two steering chambers, however, when three or more steering chambers, they are much more likely to touch and interfere with each other when they are inflated. To create an equal range of motion and steering force, the pouched steering chambers should be equally spaced around the diameter of the body chamber; this will also allow the chambers to be the maximum diameter that they can possibly be. The steering occurs when one of the pouched steering chambers is pressurized. When the series of pouch motors are pressurized, length changes occur along with the body chamber, and the robot will curve in the direction of the pressurized steering chamber as seen in **Fig. 5.2.203**. When using three equally spaced pouched steering chambers along with the body chamber, the robot is given two degrees of freedom in a three-dimensional space, with the third degree of freedom coming from the lengthening of the body chamber as the body chamber is pressurized. Using the steering configuration of three pouched steering chambers allows reversible steering because it does not have a mechanism to lock the curvature of lengthened portions of the body chamber, unlike the aforementioned non-reversible steering methods that include locking latched and shape locking mechanisms. Due to the reversible nature of this steering method, the robot gains the ability to change the curvature of sections of the body chamber that has already been lengthened. The only drawback to this method of reversible steering is that it is not easily controlled. Once the steering chambers are pressurized, they will incrementally change the curvature of the entire body chamber of the robot in the

32

direction of the selected pressurized steering chamber. This essentially means that the robot can only be turned in one direction at a time, and any shape locking would have to occur by constraints posed by the environment surrounding the robot.
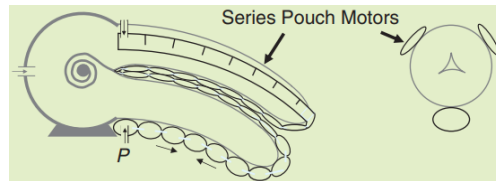
Figure 5.2.203 - Demonstration of series pouch motor actuators
(reprinted with permission from vinerobots.org)

5.2.3 - Steering Control Calculations

If it is assumed that the most popular form of steering a vine robot is to be used, that is steering using three equally spaced series pouch motor steering chambers (**see Fig 5.2.203**), a method only can begin to develop about how to steer the robot with more granular control. In order to control the amount of air pressure inside the body and steering chambers will be increased and decreased using electronically controlled solenoid valves. These solenoid valves have the ability to inflate and deflate the chambers using one inlet for the supplied air pressure and two outlets for the release of air pressure from the actuators (or chambers), and the inflation of the actuators. Pressurization of each of the steering series pouch motor chambers causes the movement of the robot tip toward that steering chamber with displacement proportional to the pressure. The resulting position of the body chamber tip is a superposition of the displacements produced by each steering chamber. The nature in which the superposition of steering chamber pressures controls the tip of the body chamber can be described using the equations below.

$$x = c(p_1 cos(\theta_1) + p_2 cos(\theta_2) + p_3 cos(\theta_3))$$

$$y = c(p_1 sin(\theta_1) + p_2 sin(\theta_2) + p_3 sin(\theta_3))$$

In the above equations the θ's represents the angle at which the steering chambers are places relative to the body chamber (counterclockwise), the p's represent the pressure present in each steering chamber, and the c is a constant that turns the pressure values into position values based on whatever metrics are being used. The x and y represent the coordinates of the tip of the body chamber in the two-dimensional plane not governed by the growth of the robot. Due to the weight and pulling forces of the robot, a reasonable assumption is that the robot will at most be able to steer the length of one meter from the tip at a time. There is some redundancy in the three-chamber steering method in the sense that the pressurized chambers may pull in opposite directions when pressurized simultaneously. For example, if all three chambers are fully pressurized at the

same time, the robot tip x,y coordinates will remain 0,0 because the forces generated by the steering chambers will cancel out. Because of these forces canceling out, it can be assumed that one of the steering chambers will have a pressure close to zero for any desired turn or curve being made. Knowing this it can be assumed p_3 to arbitrarily be zero for any curve being made. From there, using some simple geometry, the following equations can be derived to calculate the steering chamber pressures required for the body chamber tip to arrive at any given x,y coordinate.

$$p_1 = \frac{sin(\theta_3 - \theta_2)}{sin(\theta_2 - \theta_1)}p_3 + \frac{xsin(\theta_2) - ycos(\theta_2)}{dsin(\theta_2 - \theta_1)}$$

$$p_2 = \frac{sin(\theta_3 - \theta_1)}{sin(\theta_1 - \theta_2)}p_3 + \frac{xsin(\theta_1) - ycos(\theta_1)}{dsin(\theta_1 - \theta_2)}$$

$$p_3 = p_3$$

P_3 will be equal to itself because of the redundancy mentioned above. Basically, since there are only two degrees of freedom being controlled by three actuators, they do not all need to be actuated or pressurized at the same time. It is worth noting that the d in the above equations refers to the diameter of the robot body and that the p values should always be positive. To identify the required p values, P_3 is initially set to zero and the values of the other two p values are calculated. When we update our P_3 value until all p values are positive and at least one of them is close to zero.

5.2.4 - Retraction Device

The goal of adding a retraction device on vine robots is to allow retraction without undesired bending or buckling of the robot body. These devices are usually introduced to the tip of the robot because retracting a robot with zero-length (retracting from the tip) results in no buckling or bending during the process. There exist vine robot versions where the motor that drives the reel at the base is not only used to control the release of material as the vine robot lengthens, but also reverses its motion to allow the vine robot to be retracted. This method works well in constrained environments, however, vine robots that are in loosely constrained or open environments tend to bend and buckle in unpredictable ways before shortening. This uncontrolled movement can result in damage to the surrounding environment or damage to the robot itself. When steering chambers, wires, and sensors are introduced to the robot, the buckling and bending become more mission-critical, as different sections of the robot will be retracted at different rates. This will cause the robot to be mangled when redeployed and essentially renders it useless. The buckling is caused by pulling tension that is not directly aligned with the orientation of the tip of the vine robot. One fact that can be exploited when creating a solution to this problem is that vine robots with very short lengths tend not to buckle. Knowing this, a retraction device that directs the pulling tension of

retraction right at the robot's tip can be devised that works in tandem with the motor at the reel to successfully retract the robot without having it buckle. The retraction device at the tip of the robot serves to usher the robot material back into the inverted tubing, and the motor at the reel serves to pull the tail of the robot so that the slack generated by the retraction device is spooled back up in the base and prepared for redeployment. **Fig. 5.2.401** depicts the retraction devices discussed above. The device uses two motors to rotate two rollers in opposite directions in order to feed the robot tailback to the base.
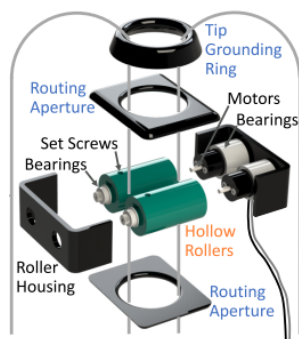
Figure - 5.2.401 Internal Roller Assembly
(reprinted with permission from vinerobots.org)

The device is held inside the tip of the soft robot body using two rings that are coated in low friction tape. And the hollow roller which the motors are used to rotate are coated in high friction tape in order to get a stable grip on the robot tail and pull it back towards the base. The wires that are used to power and control the roller motors are run through the inside of the robot body and will be spooled up with the rest of the robot body chamber and possible steering chambers. While the robot is retracting, the motors rotate in opposite directions to push the robot tailback to the base, and while the robot is lengthening the motors also rotate in opposite directions to push the body chamber material towards the tip of the robot. It is imperative that the motor at the base of the robot and the motors controlling the retraction device are in sync and both push and pull the robot body material at the same speed and at the same time. If the motors are not operated in sync then the internal roller will not remain at the tip of the robot at all times. When the internal roller motors run in the retraction direction, the roller will drive itself up the robot and away from the base until it reaches the tip of the body chamber. When the internal roller motors run in the lengthening direction the internal roller will either remain at the tip of the robot or will drive itself towards the base depending on whether the base motor is driving the tail faster or slower than the internal roller. Whether the chamber is pressurized also plays a role. If the body chamber is pressurized while the roller attempts to retract then, depending on how great the pressure is inside the chamber, the internal roller will either be able to or will not be able to overcome the forces and retract the robot.

35

5.2.5 - Camera Tip Mount

One of the most important components to any robot are the sensors it uses to capture information about the world around it. A very popular sensor option for vine robots is a tip-mounted camera. Vine robots are often used to navigate tight areas where humans are unable to go. Because of this, a camera that can relay visual information back to an operator of a vine robot is incredibly beneficial. Unlike the aforementioned retraction devices, the camera must be mounted outside of the robot body chamber so that it can be used effectively. There are a number of possible methods for external tip mounts that allow a camera to be mounted on vine robots (see **Fig. 5.2.501**), but many have notable drawbacks. A combination of these ideas can be developed that will give the vine robot most of the benefits and reduce the cons.
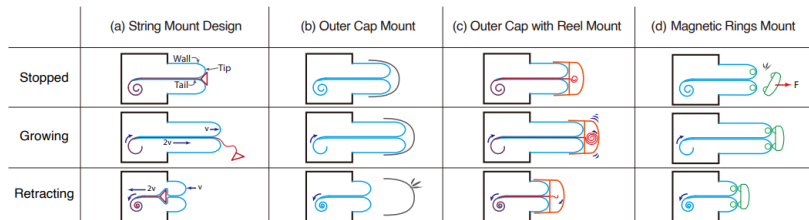
Figure 5.2.501 - Camera Mount Cap Ideas
(reprinted with permission from vinerobots.org)

The string mount design functions by tying a string to the end of a tip mount and using the tail of the vine robot to route wires back to the base of the robot. However, since the robot tail grows at twice the velocity compared to the tip relative to the base the tip mount will end up being ejected from the robot during lengthening or engulfed into the robot during restriction. The only way to overcome this issue is to have the string pulled by a separate motor that forces the string to elongate and retract at the same speed as the tip of the robot. The outer cap mount uses a rigid cap fitted over top of the robot tip. The cap will be pushed along with the robot's tip as the robot grows in length. However, unlike the string mount design, the rigid cap is not connected to the base of the robot, or any part of the robot at all, so when the robot retracts, the rigid cap has a chance of popping out and being left behind. The robot will also be unable to produce any pulling force on the environment due to the mount not being connected to the base. One benefit of this design is that it will not be engulfed when retracting or ejected when lengthening, unlike the string mount. The outer cap with reel mount design is a combination of the two previous tip mounts, and combines the benefits of both designs, but comes with its unique drawbacks. The design contains a motorized reel that spools up the slack generated by the string mount design as the robot grows, which solves the issues created by the string mount design. The outer cap and reel mount design also connect a rigid cap to the string mount which will help secure the rigid cap during retraction and helps the rigid cap produce pulling force, thus eliminating the

issues posed by the rigid cap design. However, this new design poses a new issue. As the length of the robot increases the reel inside the mount grows as it holds more and more string. The reel will eventually become too big to reasonably contain inside of the rigid cap, and thus the robot is limited to how long it can become. The magnetic ring mount design comes with a lot of benefits, but it is not perfect. This mount is unique in that it places components inside the pressurized region of the robot. The design consists of two rings, one inside and one outside of the robot body. The rings have magnetic rollers on them which hold the rings together. As the robot grows and retracts, the material will slide through the low friction rollers, which ensures that the cap can remain on the tip during all robot movement states. One of the downsides to this design is the lack of pulling forces that it can produce. The pulling forces it can withstand are limited to the strength of its roller magnets.

Another possible tip mount design is a rigid cap that attaches to the robot body using a zipper as shown in **Fig. 5.2.502**. The zipper will close as the robot lengthens, and will open as the robot retracts. In order for this to work the robot body must be wrapped in a sheath that will also be reeled or unreeled as the robot retracts and lengthens.
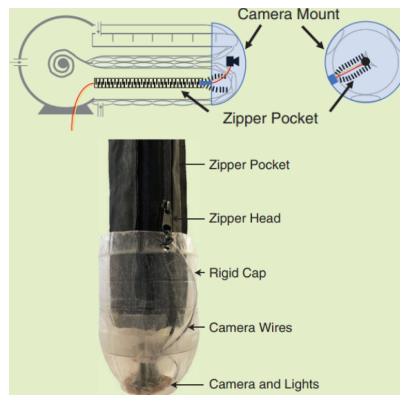
Figure 5.2.502 - Ridgid Cap With Zipper Camera Mount Assembly
(reprinted with permission from vinerobots.org)

In this design, the camera wires are stored in a separate reel at the base of the robot. As the soft robot body lengthens, the wires are pulled along the exterior of the robot body but inside the sheath by the camera cap. The sheath also serves to prevent the wire from getting caught or damaged in the surrounding environment. Using a zipper, the robot body and surrounding sheath lengthen as the body chamber is pressurized. The zipper head attaches to the camera cap so that the zipper begins unzipped and zips up as the robot grows, thus creating a sheath that is always the length of the soft robot body.

37

5.2.6 - Gripper

Being able to manipulate their environment is a key functionality of robots. Adding a manipulator to a soft robot is a tricky task due to the lack of stable material available for the gripper to attach to. However, by leveraging the rigid cap tip mount design, a gripper can be implemented on the tip of a vine robot, increasing its functionality immensely. As seen in **Fig. 5.2.601** a gripper coupled with a retraction device adds very powerful functions to a vine robot. The gripper allows the robot to lengthen and grab an object while exploring a certain path The retraction device then allows the robot to grow and explore a different path and carry the gripped item to a checkpoint.
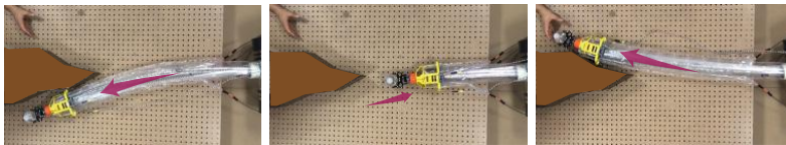
Figure 5.2.601 Gripper Implementation (reprinted with permission from vinerobtos.org)

These grippers can be controlled by a motor attached to the tip mount in much the same way that a camera would be integrated into a tip mount. The wires from the motor would be routed through the worm body or through an outside sheath that surrounds the worm body. The motor would then drive a gear attached to one of the fingers of the gripper which would interlock with another gear that attaches to the second finger of the gripper, and the two-finger gears would drive each other in opposite directions. This would create the opening and closing motions of the gripper depending on the direction that the motor turn in.

5.2.7 - Pneumatic Controls

In order to actuate the body and steering chambers, vine robots use compressed air to pressurize and depressurize their pneumatic actuators. When the body chamber is pressurized it will cause the vine robot to lengthen/grow. This growth can be controlled by balancing the body chamber pressure with the base reel motor turning speed and direction. In order to lengthen the robot, the reel motor must move in a direction that releases the body chamber material and maintains tension on the robot body while the body chamber itself is pressurized in a range large enough to trigger eversion, but small enough as to not cause the chamber to burst. On the other hand, to make the robot retract the pressure must be low enough for the chamber not to burst as the motor shrinks its volume, and the motor must spin the reel in a direction that spools the robot back into the base. In order to accurately control the motion of the reel motor, an encoder can be attached to the motor to read its positioning. If the motor spins faster than the robot is growing,

the robot will become limp and the speed of growth will not be controlled. 10 cm per second of growth speed seems to be a good top speed for most vine robots.

In order to make most vine robots lengthen the air pressure must be at least 10 kPa, and most of the common materials tend to burst at close to 20 kPa. As the pressure inside the chambers increases, the velocity of the body lengthening increases as well. The air pressure to lengthening velocity relationship seems to be monotonic and is described in **Fig. 5.2.701**. One of the most common materials used for vine robot pneumatic chambers is polyethylene, and **Fig. 5.2.701** 80 μm, 2.3 cm radius polyethylene tubing was used to record values. Although different vine robots will have different radiuses, the relationship between pressure and expansion rate follows a very similar relationship.
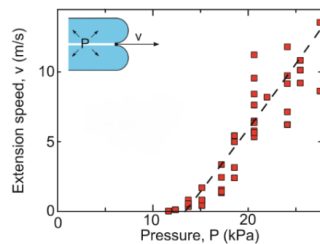
Figure 5.2.701 Robot lengthening rate vs air pressure graph (reprinted with permission from vinerobots.org)

A more robust method of measuring the lengthening rate is to use math to calculate it based on known parameters. The following equation describes the expected velocity of a vine robot in more universal terms.

$$v = \varphi(P - Y)^n$$

In the equation above, v represents the lengthening rate in cm, φ represents the extensibility of the chosen material used, P represents the pressure inside the chamber in kPa, Y represents the yield pressure blow which no lengthening occurs, and n is a power term close to one based on your robot length, diameter, and material thickness. These parameters must be customized to the specific material being used for the vine robot chamber, but it can be useful to have these calculations depending on how much control is needed of the vine robot in question.

The exact implementation of growth control can be different between systems. One method is to use a continuously-running pump with a relief valve to maintain a constant pressure (around 10-20 kPa). This reduces a bit of the complexity of having to calibrate the base motor and air pressure, and instead, it puts all the decision-making on the base motor. While lengthening and retraction, the speeds were controlled by commanding the stepper motor. Care is needed to ensure that

39

the stepper motor does not introduce slack when the robot faces outside forces like obstacles or when steering slows the robot lengthening. Another way to implement the growth control is by using a motor with a pressure regulator to make growth speed control robust to disturbances introduced by the outside environment or disturbances caused by the steering of the robot. By setting the motor to only resist growth and allowing the pressure to back drive the motor up to the desired speed, the speed could be controlled without allowing slack in the tail. Retraction can be accomplished in a similar way, just in the opposite direction, and with aid from the addition of a retraction device. With this device, the motor inputs of the base motor and the retraction device motors must be synchronized and their combined forces must balance the internal pressure. One way of doing this it to have retraction device motors determine the speed of growth or retraction, while the base motor applied the forces necessary to maintain material tension and reel material slack as it developed. It is also worth mentioning that when the robot if faced with resistance it will require more force to continue growing. This relationship can be described in the equation below.

$$F = PA$$

In the above equation F represents force, P represents pneumatic pressure, and A represents the area of the tip of the robot. When there is weight attached to the tip of the robot, such as when a tip mount or retraction device is included in the design, the robot will require more force to lengthen and thus more pressure. This relationship can be observed in **Fig. 5.2.702**. This relationship should be considered when choosing a tip mount and a pneumatic control system.
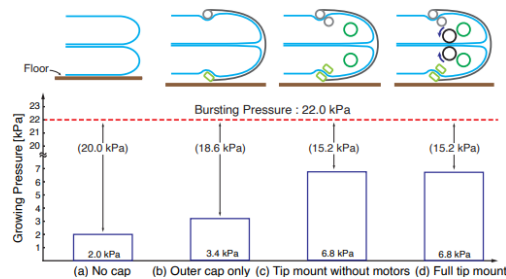
Figure 5.2.702 Robot Growth Pressure Depending vs Tip Mount Design
(reprinted with permission from vinerobots.org)

Another factor that will affect the pressure needed to grow a vine robot is the length of the robot in question. The relationship between the pneumatic pressure required to lengthen a vine robot and its length seems to be a positive monotonic one. That is to say that as the length increases, so does the force and pressure required to lengthen the robot body chamber. This relationship can be described in **Fig. 5.2.703**. In the figure it can be observed that the friction force acting on the robot increases as the length of the robot increase, thus the pressure required to move

40

a robot forward at the same speed increase as the length increases. This phenomenon is due to the vine robot's tails being dragged along the inside of the body chamber and thus producing friction. In the equation found inside of **Fig. 5.2.703** F represents the friction force acting on the robot body, μ represents the friction coefficient, w represents the normal force per cm of robot length, and L represents the length of the robot.
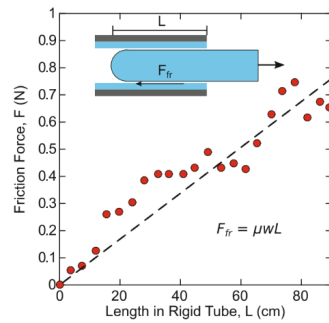
Figure 5.2.703 Friction Force vs Length of Vine Robot (reprinted with permission from vinerobots.org)

When it comes to steering the three steering chambers must be independently controlled using solenoid valves and some kind of pressure regulation. The movement of the robot tip to a position is done by setting the desired pressures of the three series-pouch-motor actuators. The theoretical calculations shown in section **5.2.3** provide a method for calculating the pressure required to have the tip of the robot reach a specific x,y coordinate position. Electronic solenoid valves can be used to selectively inflate the side chambers based on the commands the software commands given to the MCU. Mechanical pressure regulators or electrical pressure regulators can be used to control the air pressure coming out to the compressed air tank and going through the solenoid valves. Electronically controlled pressure regulators tend to be much more expensive than mechanical regulators, but the most cost-effective way of controlling air pressure passing through solenoid valves is to use a PWM signal to control the FETs that drive the solenoid valves. A PWM signal will allow much more granular control of the state of the valves. However, it is still recommended that a manual pressure regulator is used at the compressed air tank outlet.

5.3 - Explicit Design

In this section of the report, the expplic design of the mechanical components will be discussed. The methods and thought process behind part selection will be addressed, and the final designs for the major mechanical systems and components will be showcased and explained. The system must be capable of

navigation and exploration tasks, which means it must be able to grow to a length useful for navigation, pass through small apertures, and support its own body weight when navigating vertically and over gaps in the floor of the environment. All of these requirements will be taken into consideration while designing the mechanical aspects of the project.

5.3.1 - Body Design

In order to control the vine robot with three degrees of freedom we will have one body chamber to control the the z-axis (lengthening the robot), and three steering chambers to control the x & y axes. This is a commonly used vine robot steering configuration. There are many different materials that can be used for the body and steering chambers, all chambers will be constructed using the same material with varying tube diameters. The body chamber will have a diameter twice as large as each of the steering chambers. **Table 5.3.101** shows a quick comparison between the two most popular vine robot materials. These materials are low-density polyethylene (LDPE) and thermoplastic polyurethane coasted ripstop nylon (TPU-Coasted Nylon). Both of these options follow the basic guidelines of vine robot material. Those guidelines are that the soft robot body be made of a nonstretchable material that is flexible enough to be turned inside out at the tip and that is capable of containing pressurized air.

Material	Bursting Pressure	Extension Pressure	Manufacturing Method	Thickness	Form	Cost
LDPE	14 kPa	10 kPa	Heat Sealing	80 μm	Tubing	$.02 /ft^2
TPU-Coated Nylon	21 kPa	10 kPa	Heat Sealing	200 μm	Sheet	$.08 /ft^2

Table 5.3.101 Tubing material Comparison

Since both materials are heat sealable and have similar extension pressures, they can really be used interchangeably in most vine robot scenarios. Although the LDPE is much cheaper, the TPU-Coated Nylon is much more durable and looks a little nicer since it is fabric and not plastic. Since they are both functionally very similar we will use the cheap LDPE for early rapid prototyping, and opt to use the more expensive but more durable TPU-Coated Nylon for the later stages of prototyping and for the final build of the robot.

Regardless of what material we use our robot dimensions will be the same throughout. We would like the robot dimensions to remain the same. Our robot will have a body tube that is 5" in diameter, and three Steering Tubes that are 2.5" each and adhere to the body tube using heavy-duty double-sided tape (as seen in **Fig. 5.3.101**). We will also have our robot extend to be 10' long when fully extended.

42
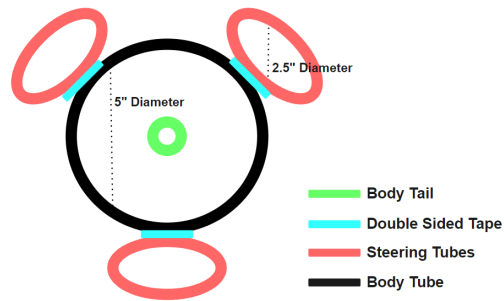
Figure 5.3.101 Robot Cross-Sectional View

Each of the steering tubes must also have a tube fitting attached to them about 2" from the end of their length so that the pneumatic hose will fit on it. Our chosen steering mechanism will be the pouch series motor tubes that shorten the length they are attached to when they are pressurized. The pouches will be made using a heat sealer and will allow room for an air gap on one side of the heat-sealed suture. This form of steering mechanism can be seen in **Fig. 5.3.102.**
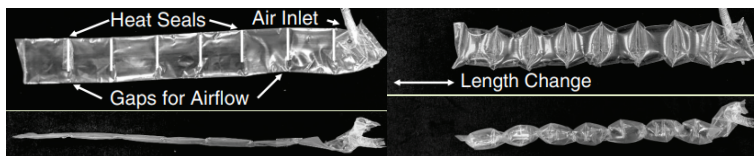
Figure 5.3.102 - Series Pouch Motor Sealing Demonstration
(reprinted with permission from vinerobots.org)

5.3.2 - Base Design

The robot base (**see Fig. 5.3.201**) is a cylindrical pressure vessel that will be made by enclosing a large acrylic cylinder with two end caps. It is used to store the undeployed robot body material on a spool. In order for the robot body to grow to full length and still be retracted back, the distal end of the main body tube will attach to a string the length of the robot body. The string must be tied to the spools in the base and the spool must be driven by a motor. which will allow controlled release of the robot body material during growth and assist with retraction of the robot body material back into the base. The length of the robot base was chosen to contain the motor and spool assembly, and the diameter of the base will be chosen to contain the rolled-up soft robot body. The stepper motor will have an encoder to ensure that the motor movements are accurate. The motor will also be attached to a shaft coupler that will connect to the driving shaft of the spool on which the robot body will be stored upon retraction. On the other end of the base, the shaft will attach to a bearing so that it can smoothly rotate.

43
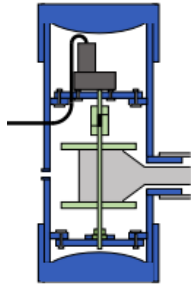
Figure 5.3.201 Robot Base Cross Section
(reprinted with permission from vinerobots.org)

A detailed assembly process of the base can be found in prototyping section **8.2.1**. The acrylic portions of the base must be laser cut using the drawing files found in **Fig. 5.3.202** and the 3D printed parts must be printed using the STL files pictured in **Fig. 5.3.203**. The acrylic parts will mostly make up the shell that houses the base of the robot as well as a second cylinder found inside of the base that will serve as our air tight chamber for routing pressurized air to the robot body tube. The 3D printed parts only make up 2 parts that allow us to maximize the use of acrylic panels.
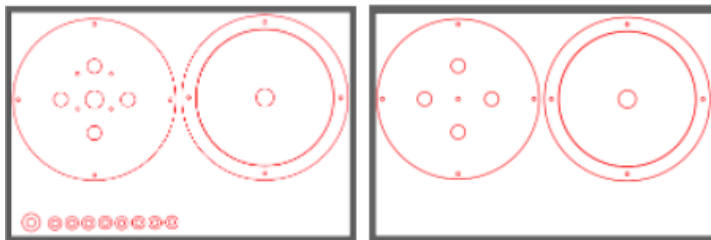
Figure 5.3.202 - Laser Cutter Base Files
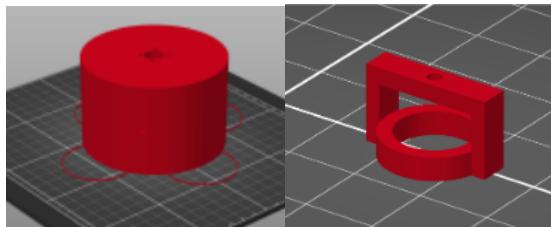
Figure 5.3.203 - Spool and Encoder Mount STL Files

44

5.3.3 - Tip Mount Design

The tip mount design for our vine robot will serve a variety of purposes. Its primary function is to provide a solid surface for mounting tools such as the camera or a gripper. It will also have room in the interior space between the surface or the tip and the tip of the soft body chamber to house a small PCB where the sensors will reside. This tip mount will also encompass an internal roller within it for robust retraction.
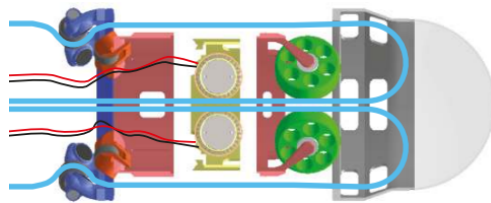
Figure 5.3.301 Tip Mount Design Cros Section
(reprinted with permission from vinerobots.org)

This tip mount design aims to overcome the shortcomings and limitations of all the tip mounts compared in **Fig. 5.2.501**. This tip mount remains on the robot tip during growth and retraction, incorporates a retraction device, and has the ability to pull things in the environment without the cap becoming disattached. This tip mount design consists of three main components: the mounting cap (in white and grey), a retraction device that prevents buckling (in green and yellow), and a rolling interlock that attaches the mounting cap and retraction device together (in blue and red). The mounting cap provides a location for mounting sensors and tools that can be used in the robots surrounding environment. The mounting cap also has three cutouts to accommodate our robot's three series pouch steering chambers and allow those chambers to move and inflate freely. The retraction device combines the use of two motors that turn two high friction rollers and squeeze the tail of the robot back towards the base. There is also a set of freely moving rollers that assist in ushering the robot tail into the motorized retraction rollers. Aside from the motorized rollers, all other sliding components on the tip mount are designed to be as low friction as possible. The rolling interlock consists of three matching sets of rolling magnets placed around the circumference of the robot body. The three sets of magnets are designed to fit right in between the three gaps left by the three series pouch motor steering chambers. Each roller-magnet unit has a passive roller with a disk-shaped magnet on either side. The rollers contact the wall and must slide between the pairs of disk-shaped magnets.

It should be noted that we have been given permission to use this design by Dr. Margarette M. Coad. Dr. Coad was able to provide the CAD files for the tip mount in **Fig. 5.3.301** which she designed as part of her research with vine robots during

her time at Stanford. Our team will use these CAD files for our own tip mount and will edit them to accommodate for any inconsistencies with our design.

5.3.4 - Gripper Design

There are many possible designs for simple two-finger grippers, many of which require gear ratios or sliding mechanisms. The gripper designed in **Fig. 5.3.401** uses a sliding mechanism and two levers to slide the two fingers of the gripper away from each other when the micro servo turns in the clockwise direction, and towards each other when the micro servo turns in the counterclockwise direction.
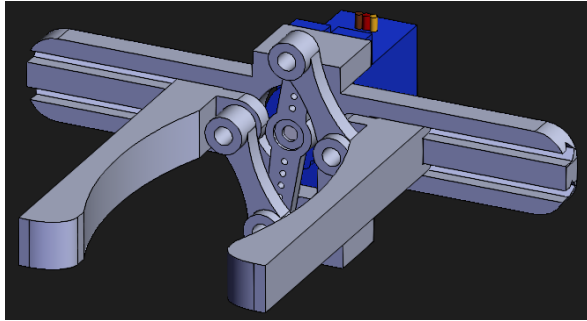
Figure 5.3.401 - Gripper Assembly

It should be noted that this gripper design does not yet have a mechanism for attaching to the tip mount presented in section **5.3.3**. Since the tip mount CAD files were only acquired recently, they have not been modified to accommodate a mounting mechanism for this gripper design. Both the gripper and tip mount CAD files must be edited to include a mounting mechanism. Since the gripper is a stretch goal for this project, the mounting mechanism will be designed if the team agrees to pursue this stretch goal.

5.3.5 - Pneumatic Design

Our robot will feature 3 different "air muscles" (referring to the steering chambers) and the main body, all of which will need to be pressurized in order to make the robot move. We will need to dynamically control the pressure in the steering chambers so that they can turn at a number of different angles, and we will need to dynamically control the pressure in the body so that we can achieve a variety of lengths. To do this we need to be able to both pumps and release the air inside these components in a controlled manner.

First, we will need a way to pressurize and store the air so that we can use it later on. Given that we want our robot to function in the field without any outlets we will need to find an air compressor that is portable and can inflate our robot completely.

46

We plan our robot to be around 10 feet long and 5 inches in diameter. If we assume that the robot will be a perfect cylinder then we find that our volume will be:

$$\pi r^2 * length = \pi(5/2)^2 * (10 * 12) = 2356.2 \ in^3$$

2356.2 cubic inches comes out to around 1.364 cubic feet (10.2 gallons). We also know that we need 20 psi to expand our robot and 10 psi to maintain the robot's shape. Using these measurements we can select a portable air compressor. In the worst-case scenario, we would need to fill 1.364 cubic feet with 20 psi. If we put a time constraint of 30 seconds to do so, then we arrive at our air compressor specs. We need an air compressor that can push around 3 cfm at 20 psi. After browsing the postable air compressor market **Table 5.3.501** compares currently available air compressors.

Product	Air Velocity	Air Pressure	Capacity	Cost
CRAFTSMAN 6-Gallon Single Stage Portable Corded Electric Pancake Air Compressor	2.5 CFM	100 PSI	6 Gallons	$130
MCGRAW 3 Gallon 1/3 HP 110 PSI Oil-Free Pancake Air Compressor	.6 CFM	90 PSI	3 Gallons	$70
DEWALT Pancake Air Compressor, 6 Gallon, 165 PSI (DWFP55126)	2.6 CFM	90 PSI	6 Gallons	$150

Table 5.3.501 Air Compressor Comparisons

Based on the above table we have decided to go with the MCGRAW option. Although it is slightly below our desired specs, it will do the job, and it is very cheap comparative. It is also an item that is locally available which makes it an even more attractive option.

Another useful piece of equipment to have is a pressure regulator. We can use this to control the amount of pressure coming out of a compressed air tank outlet. The pressure regulator will tell us the amount of air in any tank and will also allow us to release that air in a controlled manner. For this component, we would just need any manually controlled pressure regulator that can read up to 30 psi.

What will allow us to electronically control the air pressure going into each pneumatic component is a solenoid valve. A solenoid valve typically takes 12-24V of electricity and can both release or inject air into our pneumatic component (using a 3-port 2-way solenoid valve). And when coupled with electric relays the solenoid valves can be controlled using PWM signals from a microcontroller. In our case,

47

we will control the relays using an ATmega IC on a custom PCB. We will connect each solenoid valve to its own relay (four in total) and then connect the relays to the ATmega control board. When releasing air we must be mindful of where the air is directed. Since the air will likely be at high pressures, we want to direct the air in a direction away from the hardware components. In order to do this, we can connect pneumatic tubing to the release valve of the solenoid valve and then direct the tubes away from the components of the robot. We could also make these tubes quite wide so that they do not make too much noise or come out with as much force, but in doing this we will also allow more air to come out of the release valve when activated. This could pose a problem because if we cannot regulate the amount of air we release then the robot will become very hard to control. Below in **Table 5.3.502,** we compare different solenoid valve options. After comparing parts we have decided to go with the cheapest option (Woljay) since all the parts seem to have similar specs. We do know the most expensive option (Clippard) is widely used so if we run into any problems we will purchase that one instead.

Product	Minimum Pressure	Maximum Pressure	Cost
National Pipe Tapered PV0178	21.76 PSI	116 PSI	$22
Woljay 3V210-08	21.76 PSI	116 PSI	$11
Clippard NME-31NES-D012	20	125	$37

Table 5.3.502 Solenoid Valve Comparison

Another issue we might run into is the injection valves not being able to hold their air in. To prevent air from flowing backward, we can utilize check valves or non-return valves on the tubes connecting to the air injection valve.

Lastly, we will use pressure transducers to sense the pressure inside each pneumatic chamber. These pressure transducers turn pressure into an analog signal that is then passed on to the microcontroller. Using these readings, we can check that we are not going over the pressure limitations that the pneumatic chambers have. We can also check that the chambers have enough pressure in them to do what we are asking them to do (grow, retract, remain in place, etc.). A block diagram with each pneumatic component and how they all connect to each other is shown below in **Fig. 5.3.503**.
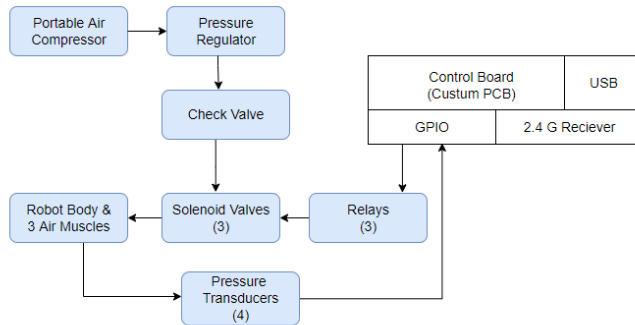
Figure 5.3.503 - Pneumatic Block Diagram

5.4 - Bill of Materials

Part	Description	Quantity	Source	Cost
Heat- sealable fabric	Heat-sealable ripstop TPU-coated nylon fabric	8 yards	Seattle Fabrics	119.6
Through-wall tube fittings	Push-to-Connect Tube Fitting for Air Through-Wall Connector for 1/4" Tube OD	3	McMaster-Carr	16.74
MD-9000 True Tape marker tape	1'' x 3yd Medium-duty marker tape	1 roll	Marker-Tape.com	3.35
Fishing Line (Power Pro)	50lb braided fishing line	1 spool	Amazon	19.99
Sewing thread	All-purpose thread	1	Amazon	5.74
Heat sealer	Flexzion 8" Impulse Sealer 300w	1	Amazon	29.95

Table - 5.4.1 BoM for Robot Body

Part	Description	Quantity	Source	Cost
Large acrylic cylinder	12" OD, 11-3/4" ID *can only be purchased at a minimum length of 2 ft, only 1 ft needed	1*	McMaster Carr	358.36
Laser cut acrylic pieces	Scratch and UV Resistant, 12" x 24" x 1/4"	2	McMaster Carr	63.10

49

Qwik Caps	12'' ID	2	Supply House	110.22
Small plastic cylinder	3" OD, 2-7/8" ID *can only be purchased at a minimum length of 1 ft, only 5 in needed	1*	McMaster Carr	10.88
Air inlet (push to connect tube fitting)	for 1/2" Tube OD x 1/4 NPT Male	1	McMaster Carr	7.58
Flexible Shaft Coupling Iron Hub (5/16'')	with Set Screw, 1-23/32" Overall Length, 1-5/64" OD	1	McMaster Carr	4.03
Flexible Shaft Coupling Iron Hub (¼'')	with Set Screw, 1-23/32" Overall Length, 1-5/64" OD	1	McMaster Carr	4.03
Shaft Coupler Rubber Spider	18000 rpm Buna-N Rubber Spider for 1-5/64" OD	1	McMaster Carr	2.64
Tiny machine screw (for fastening encoder)	M1.6 x 0.35 mm Thread, 6 mm Long	1	McMaster Carr	13.69
Tiny hex nut (for fastening encoder)	Low-Strength, M1.6 x 0.35 mm Thread	1	McMaster Carr	10.04
Hex nuts	1/4"-20 Thread Size	8	McMaster Carr	5.56
Machine screws	1/4"-20 Thread Size, 1" Long (for fastening motor and bearing plates to holder plates)	8	McMaster Carr	9.85
Ball bearing	Sealed, Trade Number R4-2RS, for 1/4" Shaft Diameter	1	McMaster Carr	6.80
Set Screw Shaft Collar	for 1/4" Diameter, Zinc-Plated 1215 Carbon Steel	1	McMaster Carr	1.12
D-Profile Rotary Shaft	D-Profile Ends, 1045 Carbon Steel, 1/4" Diameter, 12" Long	1	McMaster Carr	9.74
Set Screw Hub	1/4" Bore	1	Sparkfun	4.99

	with Hex Drive, 6-32 Thread Size, 1/2" Long	4	[McMaster Carr](#)	9.09
Machine Screws (for hub)				
Fishing Line (Power Pro)	50lb braided fishing line	1	[Amazon](#)	19.99
Rotary Shaft	Black-Oxide 1045 Carbon Steel, 1/4" Diameter, 12" Long	1	[McMaster Carr](#)	4.88
Ball Bearing	Flanged, Shielded with Extended Inner Ring, Number R168-2Z	2	[McMaster Carr](#)	14.70
Set Screw Shaft Collar	for 1/4" Diameter, Zinc-Plated 1215 Carbon Steel	2	[McMaster Carr](#)	2.24

Table - 5.4.2 BoM for Robot Base

Part	Description	Quantity	Source	Cost
Solenoid Valves	Woljay 3V210-08	4	[Amazon](#)	$43.6
Air Compressor	MCGRAW 3 Gallon 1/3 HP 110 PSI Oil-Free Pancake Air Compressor	1	[Harbor Freight](#)	$69.99
Pressure Transducers	Pressure Transducer Sender Sensor for Oil Fuel Air Water,1/8"NPT Thread Stainless Steel (100 PSI)	4	[Amazon](#)	$65.28
Check Valves	AIGNEP Check Valve 82663-04, 1/4" Male Nptf	4	[Global Industrial](#)	$13.68
Pressure tubing	UV-Resistant Firm PVC Tubing ½" and ¼"	Varied	[McMaster Carr](#)	$17.6
Tube fittings	Push-to-Connect Tube Fitting for Air	Varied	[McMaster Carr](#)	$9.8

Table - 5.4.3 BoM for Pneumatics

6 Electronics Design

Proper and careful design of the soft robotics vine is critical to ensure reliable functionality, efficiency, and accuracy of the system. This section will purely focus on the electronics design and development of the system, which includes a microcontroller, motor circuit, pneumatic control circuit, accelerometer circuit, controller circuit, voltage regulators, and a controller circuit with a joystick and buttons. These circuits all work in conjunction to operate the soft robotics vine. All electronics sensors and components that are utilized in the design will be outlined.

6.1 - Design Overview

The figure below depicts an overview of all the electronics and hardware utilized in the soft robotics design.
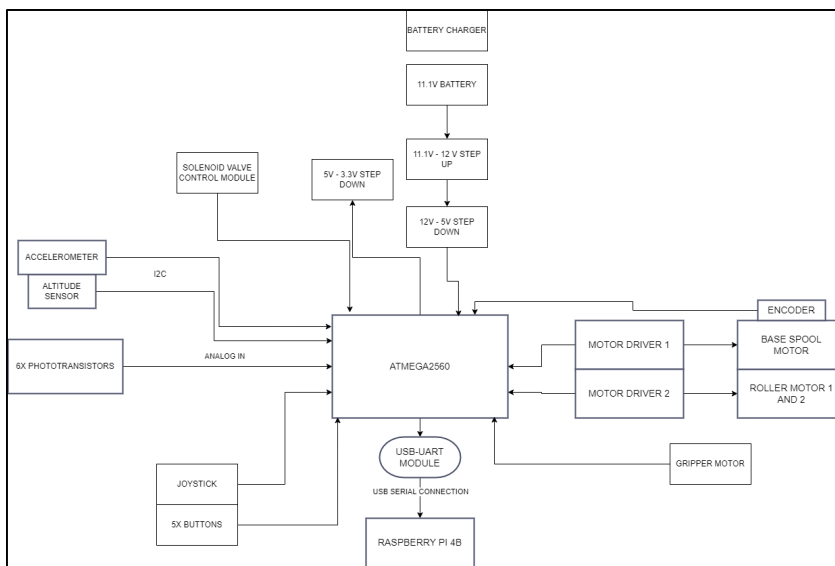
Figure 6.101 - Hardware block diagram

6.1.2 - List of Materials

Below is a list of all the major electronic components used to build the soft vine robot.

- 12V DC Gearmotors
- Motor encoder

- Motor drivers
- Accelerometer
- Altitude Sensor
- Pneumatic pressure sensors
- Lithium Ion Polymer Battery
- Transistors
- Printed Circuit Boards
- Joystick
- Switch
- Buttons
- Passive Components
- Integrated Circuits

6.1.3 - Microcontroller Selection

Our soft robot will include several pieces of hardware such as pneumatics, gyroscope, accelerometer, pressure sensors, and much more. A robust microcontroller is essential to be able to control all the hardware components and low-level movements effectively. The MCU will assist with communicating to all the hardware devices, allowing the vine to expand, retract, navigate autonomously, and read all sensor values. Overall, sufficient research will be needed to develop the architecture of our MCU hardware control system. The following sections will detail which microcontrollers we researched and some of our microcontroller-related design decisions.

There are several great options that can be utilized for our vine robot. The Atmega328p, Atmega2560, and MSP430 were all considered for our design. An in-depth look at each of these microcontrollers will be conducted, along with a comparison of each.

The Raspberry Pi is a great option, which utilizes a Linux architecture with a series of small-single board computers. It has a 1.5 GHz 64-bit quad-core ARM Cortex-A72 processor, with onboard 802.11ac Wi-fi and Bluetooth, along with anywhere from 1 - 8 GB of RAM. The Pi also contains multiple GPIO ports and various communication protocols. The Pi, however, lacks analog inputs, which would make it difficult to read analog values from the pressure sensor, gyroscope, and accelerometer chips without using an analog extension hat for the Pi or designing a custom analog to digital converter.

The ATmega2560 would be a great possible option for this project, as it has good processing power with multiple digital and analog inputs. This microcontroller is frequently used for robotics projects and includes a 16Mhz clock, 256KB flash memory, and 8KB of RAM, along with multiple GPIO ports. The Arduino supports communication protocols such as UART, SPI, and I2C, which would allow us to communicate with our various analog and digital sensors.

Feature	ATmega328	ATmega2560	Raspberry Pi 4B
Memory	32 KB	256 KB	4 GB RAM
Clock Speed	16 MHz	16 MHz	1.5 GHz

Feature	ATmega328	ATmega2560	Raspberry Pi 4B
Operating Voltage	5V	5V	4.75 to 5.25
Digital I/O Pins	14	54	26
Analog Pins	6	16	0
PWM Pins	6	15	4
Operating System	N/A	N/A	Raspbian Linux

Table 6.1.301 - MCU Comparison

As our project will require multiple analog inputs and digital inputs and outputs, we decided to select the Atmega2560 to be used to interface with the motors and sensors of the worm robot. Multiple PWM pins are also needed for motor driver connection and the Atmega2560 provides us with enough to achieve this. Furthermore, the Raspberry Pi will also be used to interface with the camera for use with OpenCV Computer Vision. More detail on the Raspberry Pi usage can be found in Section 7.

6.1.4 - Microcontroller Design

The figure below showcases our Atmega2560 microcontroller design on Autodesk EagleCAD. The design includes the MCU, a 16 MHz resonator, bypass capacitors used for filtering, an LED indicator, a USB DTR port for serial connection, and a header for a UART to USB connection.
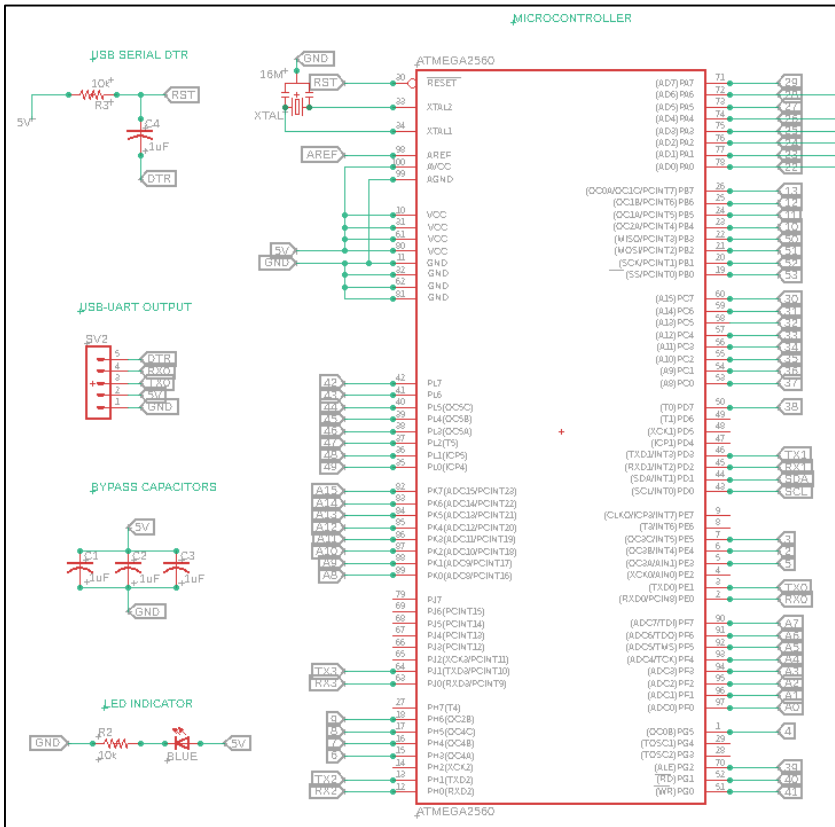
Figure 6.1.401 - Atmega2560 Schematic

6.1.5 ATmega2560 and Raspberry Pi Serial Connection

The Atmega2560 will provide feedback from the sensors and motors to the Raspberry Pi via USB serial. This will be achieved by utilizing an Arduino USB 2 Serial Micro module. This module will connect to the Atmega2560's serial data pins as shown in the figure below. It would then connect via USB to the Raspberry Pi. The module uses an Atmega16U2 programmed as a USB-to-serial converter. The firmware uses standard USB COM drivers. It has 5 pins including RX (receiving data), TX (transmitting data), 5V, ground, and a reset pin.
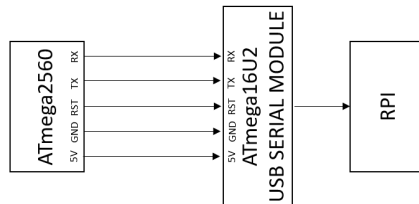
Figure 6.1.501 - Atmega2560 and RPI USB Serial Connection with Atmega16U2

A pin header for the serial port connections will be added on the base control board to connect to the USB-serial module, as shown below.
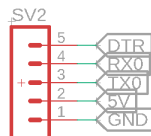
Figure 6.1.502 - Atmega2560 USB-UART Output Header Schematic

6.2 - Motors

Motors are a great electromechanical technology that allows for controlled movement in a circular motion. There are various motors such as DC brushless, AC brushless, stepper, servo, direct drive, linear, and gear motor. For this project, we will be going with a DC gear motor. We decided on going with this type of motor, as a gear motor is a combination of a motor and gearbox, allowing for high torque and controlled lower speed. Gearmotors have numerous advantages over other types of motors. These motors can simplify the step of separate gear and motor designs by combining both into a single unit, thus reducing costs. A motor and gear combination can also prolong the design life of the motor itself and allow for optimum power use by combining both in a single unit.

Furthermore, every motor requires a motor encoder to determine speed and orientation. This allows for control of the motor. Encoders turn the mechanical motion of the motor into an electrical closed-loop feedback signal that can be read by a microcontroller as an analog input, which is achieved by tracking the motor shaft. The following parameters can be determined by a motor encoder: speed, distance, RPM, and position. Furthermore, there are AC motor, servo, stepper, and DC motor encoders. For our design, we will be using a DC motor encoder, as our motor will be a DC gear motor.

This project will include a single motor at the base. This motor will require an encoder, which provides closed-loop feedback signals by tracking the speed and position of the motor shaft with a hall sensor. The direction of spin of the motor is very important in the design of robotics. Clockwise spinning allows the spool, which contains the plastic material of the robot to expand. This rotation allows the robot's body to grow and expand. The counterclockwise spinning causes the spool to shift the plastic material backward, forcing the body to retract. This motor allows for the motion of expanding and retracting. The figure below showcases how the motor direction influences the expansion and retraction of the spool material.
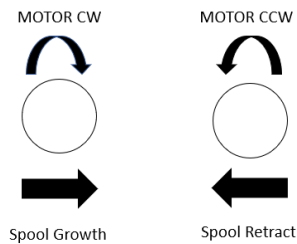
Figure 6.201 - Motor Direction Diagram

Furthermore, the motor speed will remain constant, so the vine's body will grow and retract at a constant speed, which is still yet to be determined after testing out the motor with the spool. Furthermore, the motor will be driven by a motor driver that will be controlled by a switch for growth and retraction.

As discussed in section 5.2.4, two motors will also be utilized for the worm rolling function. They will be utilized to pull the material back in at the tip to prevent buckling. The figure below showcases how the two roller motors will be connected to the motor drivers at the base of the worm.
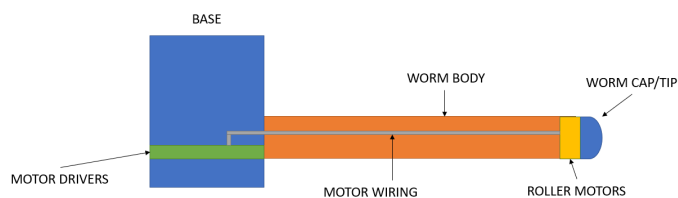
Figure 6.202 - Roller Motor Mechanical Connection

Furthermore, a motor is also needed for the worm gripper, as depicted in **Section 5.2.6.** The motor we will use for this will be an SG90 servo. The wiring

57

for this is depicted below and a pin header will be used on the base control board to interface with the servo.
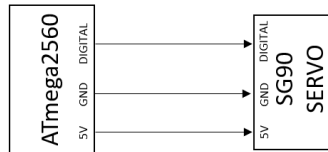
Figure 6.203 - SG90 Servo Motor Connection

6.2.1 - Motor Driver

The base growth and retract motor and two roller motors will utilize two motor drivers to be able to drive the growth and retraction with manual switch control, which will be located on the robot controller. Moreover, a motor driver is needed, as microprocessors operate on lower voltages and currents compared to motors, which can require 12V or 24V for operation; a motor driver contains an integrated circuit that can be used to control the motor. The motor driver will take the low-current signal from the microprocessor and amplify it to control the actual high-powered motor. The driver receives a signal from the microprocessor and transmits that converted signal to the motor, this can cause the motor to run in either clockwise or counterclockwise and can adjust the speed of the motor. Overall, the motor driver acts as an interface between the microcontroller or processor and the motor. We plan on using two motor drivers with two motor outputs each, giving us a total of four possible ways to control motors. One control output would be utilized for the base motor and two for the roller motors. A diagram of how the motor driver will be utilized alongside the microprocessor can be seen below.
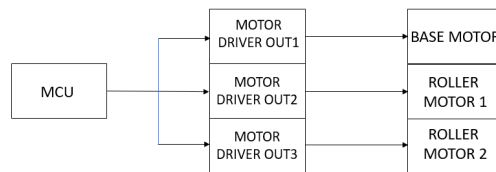
Figure 6.2.101 - MCU-Motor Interaction

The motor drivers in our project will be controlled by the ATmega2560, which sends data back to the Raspberry Pi via serial communication, and a switch will be used to set the motor positions to clockwise or counterclockwise, in order to control the growth and retraction of the robot. The two roller motors will and the base motor will work in conjunction to cause the robot to grow and retract.

58

6.2.2 - Motor Selection

Selecting a motor has a big impact on how the soft robotics worm will grow and retract. The motor is a key element in the design of the base. The size, speed, torque, power, performance, cost, and compatibility are all factors that need to be considered when selecting the appropriate gear motor. We will have to select a gear motor that has enough speed to grow and retract the soft material with the spool. The motor must also have a viable size to hold the spool and material without collapsing. Furthermore, it must also meet the power requirements of our project and must not exceed the power supply voltage of 12V. The table below showcases the various motors we were deciding between when selecting an appropriate base motor for our soft robotics worm.

Motor	Voltage	Speed	Full Load Current	Manufacturer	Price
Compact Square-Face DC Gearmotor	12 VDC	25 RPM	1.3 A	McMaster-Carr	$62.74
Dayton DC Gearmotor	12 VDC	17 RPM	1.4 A	Dayton	$62.49
Parallel Shaft DC Gearmotor	12 VDC	7.9 RPM	1.39	IronHorse	$192

Table 6.2.201 - Base Motor Comparison

After careful consideration, we decided to select a square-face DC gear motor from McMaster-Carr. This motor has a 12V requirement, which will be retrieved from the stepped-up voltage from the 11.1V battery. The motor also has the highest RPM of the 3, which will aid in quick growth and retraction of the spool material. The other gear motors have lower RPM, which would slow down robot growth and retraction. The motor we selected also meets our budget requirements.For the roller motors, we will need smaller motors that can fit right below the cap of the worm, as seen in section 5.2.4. We decided to utilize two 3485 12V 28 RPM gear motors from Pololu corporation.

The motor encoder we will be selecting for use with the base motor will be a 20D mm encoder for metal gear motors from Pololu corporation, which takes a VDC input from 2.7V to 18V. We decided to go with this, as it will sit perfectly flush on the motors we selected. The encoder uses a dual-channel hall effect sensor and two 10-pole magnetic discs that are used to add quadrature encoding to two 20D mm metal gear motors. M1 and M2 are the motor outputs, Vcc and GND are used for encoder power, and OUT A and B are digital signals that are either driven low or pulled to high to Vcc.

59

6.2.3 - Motor Driver Selection

Selecting the right motor driver is just as important as selecting the right motor. Once the right motor is selected, a corresponding motor driver is needed for optimal control of the motor. When selecting the right motor driver, the following need to be considered: motor driver maximum supply voltage, maximum output current, rated power dissipation, load voltage, packaging type, and the number of outputs. All these motor driver specifications must be taken into account depending on the motor and microcontroller requirements. To satisfy these requirements, we selected a Texas Instruments L293 motor driver, which the pinout of can be seen in section 12.4.

The L293 motor driver is a quadruple high current half-H driver. It provides bidirectional drive currents of up to 1A at voltages from 4.5V to 36V. We will be supplying it with 12V, as our motors are rated at this voltage. The L293 can be used to drive inductive loads, such as solenoids, relays, DC motors, and stepper motors. The motor driver IC has two power input pins 'Vcc1' and 'Vcc2.' Vcc1 is used for driving the internal logic circuitry, which should have a 5V input and Vcc2 is the power input for the internal H-bridge, which can range from 4.5V to 36V. They are both grounded to a common ground. The IC has four output terminals that supply power to two motors. 1Y and 2Y, along with 3Y and 3Y, are used to drive two motors. As our project will utilize three motors, we will use two L293 motor drivers within our design. The motor driver IC also has two control pins used to control the speed and to control the direction of the Dc motor. 1A and 2A control spinning directions of motor A, while 3A and 4A control motor B. A logic high or low needs to be applied to either pin to control the motor, as depicted in the table below. The spinning direction is controlled by changing the polarity of the input voltage, which can be accomplished by implementing an H-bridge circuit. This circuit contains four switches with the motor at the center, which creates an H-like arrangement. When selecting two particular switches in the H-bridge circuit, can reverse the polarity. The L293 has an integrated H-bridge circuit within the chip itself, so external H-bridge circuitry is not needed for this design.

1A/3A	2A/4A	Spinning Direction
Low (0)	Low (0)	Motor Off
High (1)	Low (0)	Forward
Low (0)	High (1)	Backward
High (1)	High (1)	Motor Off

Table 6.2.301 - Motor Digital Control

Furthermore, the L293D motor driver has two EN pins that are used to turn on, off and control the speed of both motor A and motor B. The EN pins will be connected to the pulse width modulation (PWM) inputs of the Atmega2560. This allows us to control the speed of the motor. This will let us control how fast the worm will retract and grow. Overall, two L293 motor driver ICs will be utilized in our design. One for the base motor and one for the two roller motors.

6.2.4 - Motor Circuit Block Diagram

The figure below showcases the motor circuit block diagram. The 12V power source will provide power to the motors, while the motor driver will be powered from the Raspberry Pi's 5V source. The encoder, which would be attached to the motor's shaft, will send the signal back to the Raspberry Pi. A toggle switch will also be connected to the Raspberry Pi for control of the motor's position. This switch will allow the soft robotic worm to grow and retract depending on the toggle state.
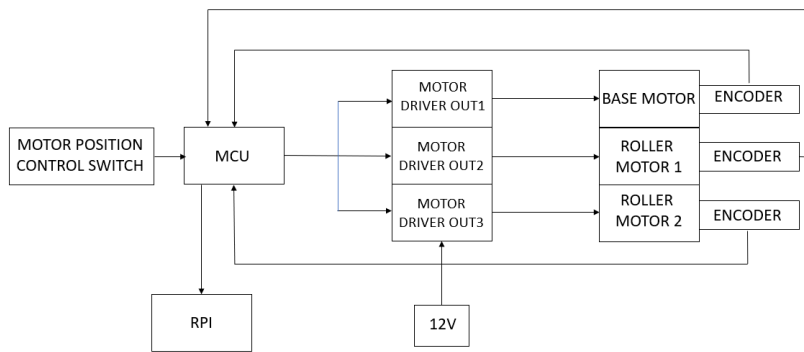
Figure 6.2.401 - Motor Block Diagram

6.2.5 - Motor Circuit Schematic

The figures below showcase our motor circuit schematic with the 12V DC metal gear motor, motor encoder, and L298 motor driver connected to the ATmega2560 microcontroller.
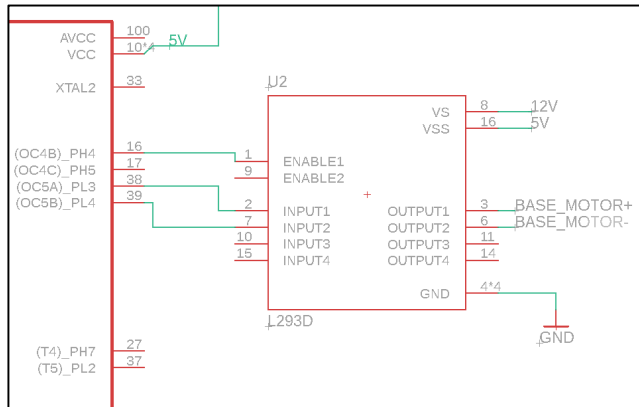
Figure 6.2.501 - Base Motor Circuit Schematic

The first figure above illustrates the L298 motor driver circuit for the base motor. Only 1 motor channel will be used on this motor driver. The ENABLE1 will be connected to the PWM digital output on pin 16 on the Atmega2560, while the INPUT 1 and 2 pins will be connected to digital output pins 46 and 45. VS will be provided 12V as the motor requires 12V and 5V will be provided to VSS to power the internal H-bridge circuit. The figure below showcases the same circuit but for the two internal roller motors. Both motor channels are utilized in this driver.
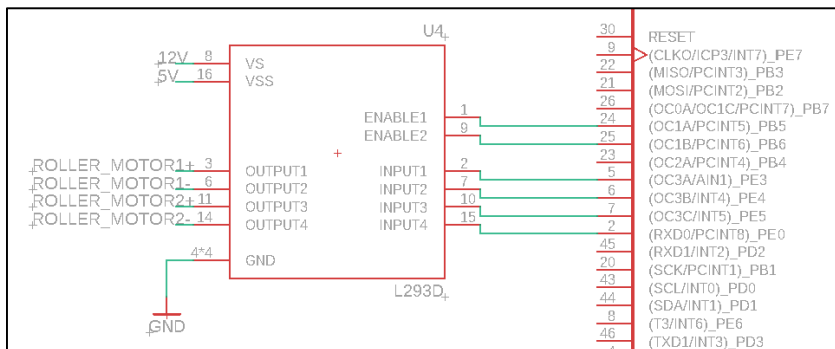
Figure 6.2.502 - Roller Motors Circuit Schematic

ENABLE 1 and 2 are connected to PWM-enabled pins 11 and 12 of the Atmega2560 and INPUT pins 1-4 of the L293D motor driver are connected to digital output pins 0, 4, 5, and 3. Lastly, the motor encoder will be powered with 3.3V from the step-down converter on the base control board and the OUTA and B pins will be connected to interrupt pins PCINT9 and PCINT10, as seen in the

figure below. The pins are driven high with 10k pull-up resistors so that the interrupt for the hall effect sensor can be detected. The encoder will allow us to track the direction of rotation of the spool by counting revolutions forwards and backward.
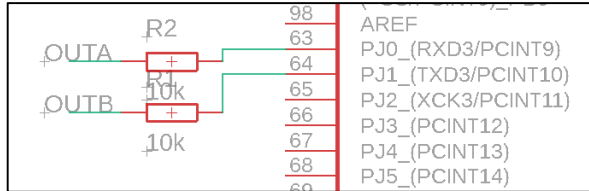
Figure 6.2.503 - Motor Encoder Circuit Schematic

6.2.6 Motor Circuit Parts Selection Summary

Below is the parts selection summary for the motor circuit, which includes all motors, integrated circuits, and passive components used within the design.

Component	Quantity	Use
Compact Square-Face 12VDC Gearmotor	1	Base motor
Metal Gearmotor 20Dx46L mm 12V CB	2	Roller motors
SG90 Servo	1	Gripper motor
Magnetic Encoder	1	Encoder for base motor
L298 Motor Driver 10pcs	1	Motor driver for 3 motors

Table 6.2.601 - Motor Circuit Parts Summary

6.3 - Accelerometer Sensor

Our soft robotics worm will also include an accelerometer which is a sensor that measures triaxial acceleration and angular velocity, which allows for the output of linear acceleration signals on a three-axis space. Accelerometers are generally utilized in things such as tracking and navigation, human activity recognition, and video games or devices that use inertial measurement to detect movement.

For our soft robotics worm, the accelerometer sensor will be used to track the acceleration, angular velocity, and heading; the data will then be sent back to the user. We will be accomplishing this by utilizing the ADXL345BCCZ from Analog Devices Inc., which can be seen in the figure below. The sensor's applications can be for indoor navigation, smart user interfaces, advanced gesture recognition, gaming, virtual reality input devices, displays, and map orientation and browsing.

This ADXL345 is a low-power, small, 3-axis accelerometer with a 13-bit resolution measurement at ± 2, 4, 8, or ±16 gauss. The digital output data from this sensor is formatted as a 16-bit twos complement and is accessible via serial communication such as SPI, which can be utilized as a 3 or 4 wire, or through an I2C digital interface. The ADXL345 measures the static acceleration of gravity in a tilt-sensing application. It can also detect dynamic acceleration that results from motion or sock. The device has an operating voltage of 3.3V and can take an analog supply voltage of anywhere between 2.0V to 3.6V. The sensor also features an eco power-down mode, which brings the current down to 1.9mA for power saving. Overall, this is a robust sensor that provides us with all the measurements that we need for the soft robotics worm and falls within the right power requirements. The table below illustrates all of the sensor's specifications.

Parameter	Value
Operating Voltage	2.0V - 3.6V
Serial Interfaces	I2C, SPI
Acceleration Channels	3
Measurement Mode Current	23 uA
Standby Mode Current	0.1 uA
Resolution	13-bit
Operating Temperature	-40°C to +85°C

Table 6.301 - ADXL345BCCZ Specifications

The purpose of selecting an accelerometer sensor for our soft robotics worm is to send data regarding the worm's orientation, acceleration, and heading back to the user. This data will allow the user to better understand the worm position and speed within the maze. The sensor will be soldered to a custom printed circuit board attached to the tip of the worm. The printed circuit board for the accelerometer sensor will contain a female pin header that will have wires connecting to it that will be running back to the control printed circuit board that will be located at the base. The sensor will relay information directly to an ATmega2560, located on the control board, via I2C. The data from there will then be sent back to the Raspberry Pi via USB serial communication to be displayed to the user on the screen next to a camera from the worm's point of view.

6.3.1 - Accelerometer Sensor Circuit Block Diagram

The block diagram below showcases how the accelerometer sensor will be connected to the ATmega2560 with data being sent to the Raspberry Pi and then that data is displayed on a user interface.
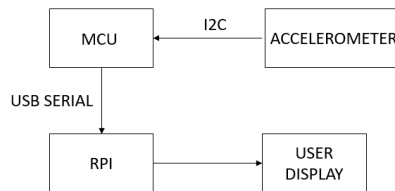
Figure 6.3.101 - ADXL345 Sensor Block Diagram

6.3.2 - Accelerometer Sensor Circuit Schematic

The figure below showcases the ADXL345 sensor circuit schematic design on Autodesk EagleCAD. VDD and VS will be shorted and connected to a 3.3V source from the ADP160AUJZ-3.3-R7 step-down converter, which takes an input anywhere from 2.2V to 5.5V and steps it down to 3.3V. The step-down converter will take the ATmega2560's VCC of 5V and step it down to 3.3V for use with the ADXL345 accelerometer sensor. The I2C SDA and SCL connections are 3.3V signals, which are not compatible with ATmega's SDA and SCL, which are 5V signals. A simple logic level converter was designed to convert the 3.3V I2C signals to 5V signals. This was designed using two BSS138 Logic Level Enhancement Mode Field Effect Transistors and four 10k ohm resistors. The sensor's SDO and SCL pins are pulled high to 3.3V with two 10k ohm pull-up resistors to enable a serial output for I2C. A 0.1uF capacitor was added to the design based on datasheet recommendations for filtering. CS is also driven high as there are no internal pull-up or pull-down resistors for unused pins, which means that there is no known state or default state for the CS. A 10k pull-up resistor was added to CS to drive it high to Vcc.

Figure 6.3.201 - ADXL345 Accelerometer Sensor Circuit Schematic

6.3.3 - Accelerometer Sensor Parts Selection Summary

Below is the parts selection summary for the theADXL345 Accelerometer Sensor circuit, which includes all integrated circuits and passive components used within the design.

Component	Quantity	Link
ADXL345 Accelerometer Sensor	1	ADXL345BCCZ Analog Devices Inc.	Sensors, Transducers	DigiKey
BSS138 Field Effect Transistor	2	BSS138 onsemi / Fairchild	Mouser
10k ohm Resistor	5	RC0805FR-0710KL YAGEO	Resistors	DigiKey
0.1 uF Capacitor	1	C0805C104K5RAC7800 KEMET	Capacitors	DigiKey

Table 6.3.301 - Accelerometer Parts Summary

6.4 - Altitude Sensor

We will also be incorporating an altitude sensor that measures barometric pressure, temperature, and altitude that will be placed within the cap of our soft robotics worm. This sensor will be used to return data back to the user. A barometric pressure sensor measures change in pressure, which also ends up measuring the change in altitude, as altitude changes with pressure. Some applications of this kind of sensor include the following: high accuracy altimetry,

personal electronics altimetry, GPS dead reckoning, GPS enhancement for emergency devices, map assist, navigation, and weather station equipment. Moreover, our soft robotics worm will utilize this sensor to return altitude and temperature data back to the user. The altitude data can allow the user to understand how high or low the worm is when navigating the maze by returning an altitude value. The specific sensor we will be utilizing to accomplish this is the MPL3115A2.

This MPL3115A2 altitude sensor utilizes a MEMS pressure sensor with an I2C interface to provide an accurate reading of pressure, altitude, and temperature data. The sensor accomplishes this by outputting the signals through a high-resolution 24-bit ADC. Its supply voltage is 1.95V to 3.6V, which is internally regulated by an LDO. The pressure reading is a 20-bit measurement in pascals; the altitude reading is a 20-bit measurement in meters; the temperature is a 12-bit measurement in Celsius. The sensor also includes programmable events, autonomous data acquisition, 32-sample FIFO, the ability to log data for up to 12 days via FIFO, and an I2C digital output interface, which operates up to 400 kHz. Furthermore, the altitude calculations are based on the pressure that is measured, the user input of the equivalent sea level pressure and the US standard atmosphere 1976 (NASA) to provide the altitude readings. The altitude is calculated from the pressure by using the equation seen below.

$$h = 44330.77\{1 - (p/p_0)^{0.1902632}\} + OFF\_H \text{ (Register Value)}$$

In the equation above, p0 is the sea level pressure, which is 101326 pascals and h is in meters. The MPL3115A2 utilizes this value. Overall, this is a robust sensor that provides us with all the measurements that we need for the soft robotics worm and falls within the right power requirements. The table below illustrates all of the sensor's specifications.

Parameter	Value
Supply Voltage	1.62V - 3.6V
Operating Voltage	1.95V - 3.6V
Serial Interfaces	I2C
Integrated Current	40 uA
SCL Clock Frequency	400 kHz
Operating Temperature (°C)	-40 - 85

Current Consumption	2 mA

Table 6.401 - Altitude Sensor Specifications

This sensor will be soldered onto a printed circuit board that will be located within the cap of the soft robotic worm. This printed circuit board will include the accelerometer and the altitude sensor. The pins of the sensors will then connect to the analog inputs of the MCU in the base control PCB.

6.4.1 - Altitude Sensor Block Diagram

The block diagram below showcases how the MPL3115A2 Altitude sensor will be connected to the ATmega2560 with data being sent to the Raspberry Pi and then that data is displayed on a user interface.
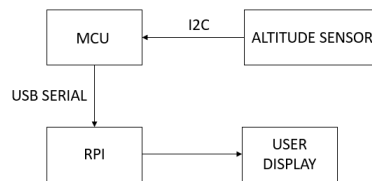
Figure 6.4.101 - Altitude Sensor Block Diagram

6.4.2 - Altitude Sensor Circuit Schematic

The figure below showcases the MPL3115A2 altitude sensor circuit schematic design on Autodesk EagleCAD. VDD and VDDIO will be shorted and connected to a 3.3V source from the ADP160AUJZ-3.3-R7 step-down converter, which takes input from 2.2V to 5.5V and steps it down to 3.3V. The step-down converter will take the ATmega2560's VCC of 5V and step it down to 3.3V for use with the MPL3115A2 altitude sensor. The I2C SDA and SCL connections are 3.3V signals, which are not compatible with ATmega's SDA and SCL, which are 5V signals. A simple logic level converter was designed to convert the 3.3V I2C signals to 5V signals. This was designed using two BSS138 Logic Level Enhancement Mode Field Effect Transistors and four 10k ohm resistors. Various capacitors were also added to the design based on datasheet recommendations for filtering.
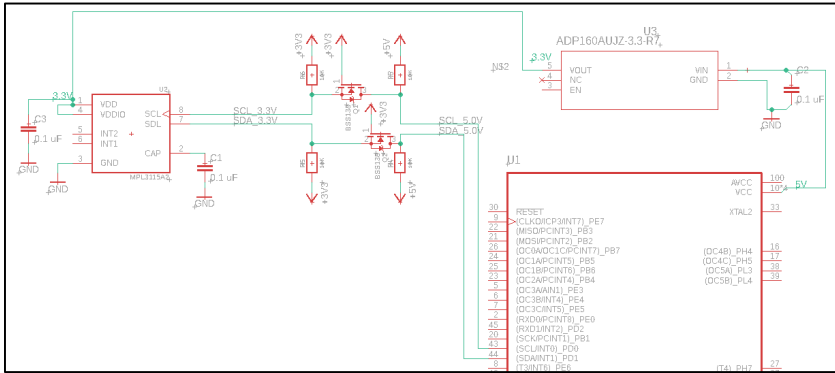
Figure 6.4.201 - Altitude Sensor Circuit Schematic

6.4.3 - Altitude Sensor Parts Selection Summary
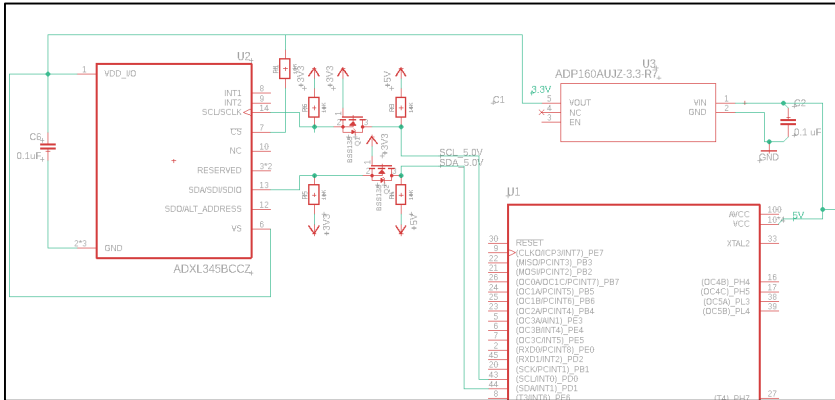
Below is the parts selection summary for the altitude sensor circuit, which includes all integrated circuits and passive components used within the design.

Component	Quantity	Link
MPL3115A2 Altitude Sensor	1	SparkFun Triple Axis Accelerometer Breakout - ADXL335 - SEN-09269 - SparkFun Electronics
BSS138 Field Effect Transistor	2	BSS138 onsemi / Fairchild | Mouser
10k ohm Resistor	4	RC0805FR-0710KL YAGEO | Resistors | DigiKey
0.1 uF Capacitor	2	C0805C104K5RAC7800 KEMET | Capacitors | DigiKey

Table 6.4.301 - Altitude Sensor Parts Summary

6.5 - Phototransistor

The soft robotic worm will also be autonomously navigating toward the nearest light. This can be achieved with OpenCV and phototransistors. Phototransistors are sensors that allow you to detect light. These are small, low-power, and inexpensive devices. They are referred to as CdS cells, as they are made from Cadmium-Sulfide, light-dependent resistors (LDR), and phototransistors. The resistive value of a transistor changes depending on how much light is shining onto the surface of the cell. Furthermore, when exposed to no light, the resistance increases greatly, and when exposed to light, the resistance drops. The transistor that we will be utilizing for our soft robotics worm will be a 161 phototransistor from

69

the Adafruit industries, as seen in the image below. The phototransistor requires a 3.3V - 5V input and a 10k ohm resistor. One end is connected to power and the other end connects to a pull-down resistor to the ground. The other end of the resistor is then connected to the analog input of a microcontroller.

The worm will include six phototransistors located around the cap of the worm. These phototransistors will be used to return data on how much light is located around the field of view of the camera, so whatever the light the camera is unable to detect, the phototransistor will pick up that value, assisting in the autonomous navigation toward a light source. The resistors and analog connections will all be placed on the sensor printed circuit board, which connects to the base control board. The below figures showcase the phototransistor alignment on the worm cap board.
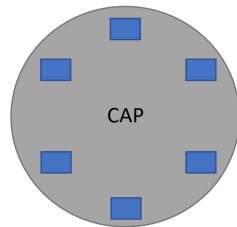
Figure 6.501 - Worm Cap with Phototransistors Depicted in Blue

6.5.1 - Phototransistor Circuit Block Diagram

The block diagram below showcases how the six Adafruit phototransistors will be connected to the ATmega2560 with data being sent to the Raspberry Pi and then that data is displayed on a user interface.
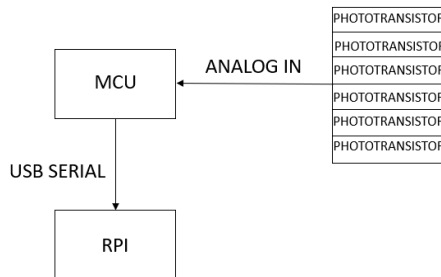
Figure 6.5.101 - Phototransistor Block Diagram

6.5.2 - Phototransistor Sensor Circuit Schematic

The figure below showcases the phototransistor sensor circuit schematic design on Autodesk EagleCAD. The first pin of all the phototransistors are connected to the 5V source from the ATmega256. The second pin is then connected to the analog input and a 10k ohm resistor. The second pin of the resistor is then grounded. All six phototransistors are connected to the analog inputs of the ATmega2560.
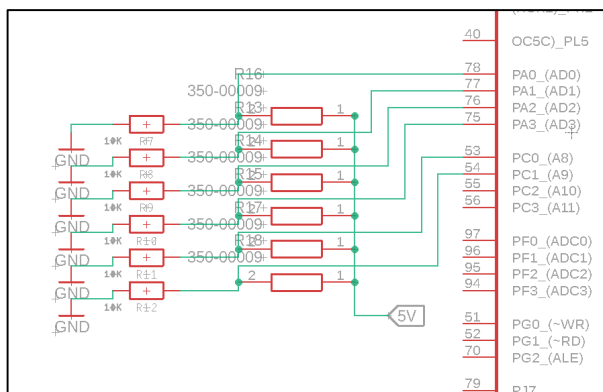
Figure 6.5.201 - Phototransistor Circuit Schematic

6.5.3 - Phototransistor Parts Selection Summary

Below is the parts selection summary for the phototransistor circuit.

Component	Quantity	Link
Phototransistor	6	02-LDR4 NTE Electronics, Inc	Sensors, Transducers
10k ohm Resistor	6	RC0805FR-0710KL YAGEO	Resistors	DigiKey

Table 6.5.301 - Phototransistor Parts Summary

6.6 - Solenoid Valve Control Module

A control module is also needed to control the pneumatic solenoid valves used to inflate and deflate the soft vine robot. A total of four solenoid valves will be used to inflate the robot for steering and base inflation and deflation, as illustrated in Section 5.2.3. The joystick from the controller board in section 6.7 will be used to control the three solenoid valves. Moreover, this would require three ways to drive current to the 12V inputs of the solenoid valves. To achieve this, we will plan to use Darlington transistors to drive current and switch the solenoid valves on and

71

off. The TIP120 from ON Semiconductor is an NPN epitaxial Darlington transistor. The table below showcases its electrical characteristics. This transistor will be utilized like a switch to control the solenoids.

Feature	Value
High DC Current Gain	hFE = 4.0 Adc
Collector-Emitter Sustaining Voltage @100mAdc	VCEO = 60 Vdc
Low Collector-Emitter saturation Voltage	VCE = 2.0 Vdc @ IC = 3.0 Adc VCE = 4.0 Vdc @ IC = 5.0 Adc

Table 6.601 - TIP120 Electrical Characteristics

Solenoid coils also have a very high inductance. When switching them off, a high voltage spike is generated as the magnetic field collapses, which can potentially kill the transistor. To prevent this from occurring, we will be incorporating a diode across the solenoid, with the cathode of the diode connecting to the positive terminal of the solenoid. A 1N4933RLG general purpose 50V 1A diode from onsemi will be used to accomplish this. Furthermore, a 2.2k ohm resistor will be added between the base of the transistor and the digital pin of the Atmega2560.

6.6.1 - Solenoid Valve Control Module Circuit Schematic

The figure below showcases the solenoid valve control module circuit schematic design on Autodesk EagleCAD. The PNEUMATIC- connection will be connected to the negative terminal of the solenoid valve, while the positive terminal will be supplied 12V from the battery step-up convertor. The base of the Darlington transistors is connected to digital input pins 71, 72, 73, and 74 of the Atmega2560. When the digital signal is driven high, the transistor closes with the current input and completes the circuit for the solenoid valve, allowing 12V to flow through.
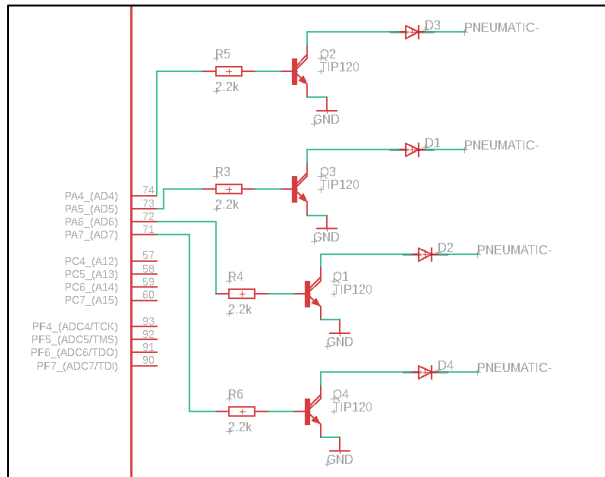
Figure 6.6.101 - Solenoid Valve Control Circuit Schematic

6.6.2 - Solenoid Valve Control Module Parts Selection Summary

Below is the parts selection summary for the phototransistor circuit.

Component	Quantity	Use
TIP120 Darlington Transistor 3 Pack	2	Drive current to Solenoid Valves
2.2k ohm Resistor	4	Allows for MCU digital input/output
1N4933RLG Diode	4	Transistor Protection

Table 6.6.201 - Solenoid Valve Control Module Parts Summary

6.7 - Controller Board

Our project will include a controller board that will be used to wirelessly control the motor and pneumatic relays. This controller board will include a transmitter controller battery and around 5-6 push button switches. The controller batteries will provide power to the robot controller which allows the user to interact with the robot manually in order for the robot to navigate throughout the obstacle course.

6.7.1 - Controller Board Block Diagram

As shown in the block diagram, the main power source of the controller batteries will be AA Alkaline batteries ranging from 3.3V-5V which will be connected to the power supply of the custom PCB controller. The robot controller will have joysticks and push buttons added to the PCB to make the robot user-friendly. The transmitter will send visual and audio signals to the receiver which communicates to the raspberry-pi through SPI (Serial Peripheral Interface). The SPI uses the concept of Master and Slave in which most cases, the Raspberry Pi will be the master and the receiver, a slave. This entails using a serial clock and a select pin to determine which device the master wants to speak to. Also based on the clock frequency, this will help synchronize the transmission of data between the master (Raspberry pi) and the slave device (receiver).
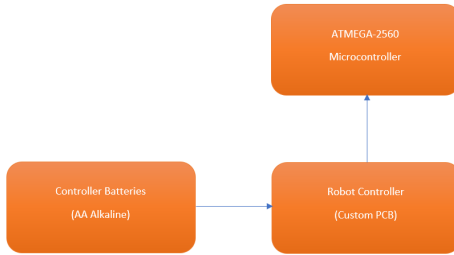
Figure 6.7.101 - Controller Board Block Diagram

6.7.2 - Controller Board Circuit Schematic
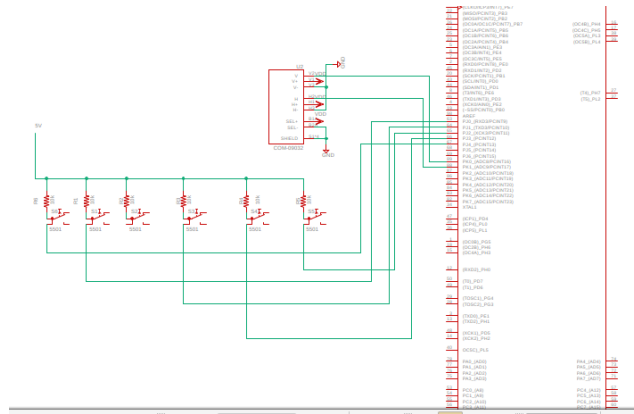
Figure 6.7.201 - Controller Board Circuit Schematic

74

This schematic was designed using the Autodesk Eagle software. As shown in the schematic, the pushbuttons are set to active high because we want the switch to be on once it is pressed. Setting the resistors to 10k ohms was a strategic decision because we wanted to limit the current flowing through the switch and prevent short-circuiting. The joystick module we plan on using is the COM-09032 module. The vertical and horizontal pins will be connected to the analog inputs of the ADC pins of the atmega2560. The controller battery is set to 3.3V as anything higher will result in the transmitter overheating. The five push buttons will be connected to the digital pins of the microcontroller to provide different functions for the udder to interact with when manually controlling the robot worm.

6.7.3 - Controller Board Parts Selection Summary

For our controller, we decide to look at multiple parts to provide a more reliable component for our project. The first part we looked at was COM-09032. This is normally used on Play-Station 2 which is a game console. It has two potentiometers (10k ohms) that provide both vertical and horizontal movement. It also has a select button that actuates once the button is pressed. It is also very convenient for arcade games. The second part we looked at was the Xbox Adaptive Controller. The X-Box Adaptive Controller has multiple USB ports. It has an input voltage of 1.8V. Each port is compliant with the USB 2.0 Full Speed signaling and electrical interface. The USB interface is designed to only operate with USB HID joysticks that report either an HID (human interface device) Joystick or HID Gamepad Usage Page. Input from the USB HID joystick will be normalized and reported as the thumbstick position that corresponds to the respective USB port in use. The final part we looked at was the ThBP001 joystick controller. It has a 2,000,000 life cycle for toggles. This also has multi-directional movement on both axes, center select momentary switch and small packageable size. It has a maximum input voltage of 5V.

Component	Quantity	Link
10k ohm Resistor	5	RC0805FR-0710KL YAGEO	Resistors	DigiKey
COM-09032 Joystick	1	COM-09032 SparkFun Electronics	Switches	DigiKey

Table 6.7.301 - Controller Board Part Selection

6.8 - Power Design

Our power design consists of a 22000mAh Lipo battery rated at 11.1V with a pre-installed BMS (Battery Man, a Hota D6 pro smart charger with multiple USB ports to provide efficient charging. We will also be connecting the battery to a disconnect switch to provide control when there is a situation of overheating or short-circuiting.

The main battery is rated at 11.1V sp we will be using a buck-boost converter to increase the voltage to 12V for our electrical loads and a buck converter to step down the 12V to a much lower voltage so each electrical load can be provided with the right amount of electrical power without short-circuiting.

6.8.1 - Design Overview

Lithium-ion batteries have also been widely used in electric vehicles which provides a faster charge. These applications have provided a sustainable atmosphere for the next generation because there will be less pollution present in the atmosphere. There are also LiPO (lithium polymer) batteries which are similar to lithium-ion batteries, however, what makes them different is that there are polymer separators that act as shutdown separators which help to stop current flow when the battery gets overheated. Regarding our power design, we will be using lipo batteries as our main source of power. Since our robot is mainly battery controlled, we decided to go with a 12V lipo battery rated at 22Ah and a battery charger. This battery will be used to power other electrical loads such as the microcontrollers(Raspberry pi) and the manual controls of the robot worm such as the camera, motor, motor drivers, gyroscope, accelerometer, and pressure sensor. Our raspberry pi is set to operate at 5V, thus we will need a buck converter to step down the 12V voltage in the battery to 5V. Batteries and solar panels have made major breakthroughs in the power industry. With the advancement of lithium-ion batteries, there is a considerably higher energy density than a regular battery.

Component	Power Rating (Ah)
Raspberry Pi 4GB	4.425
RumCam	0.4
Transmitter	3
3 Motors rated at 12V each	5.4
Robot controller	0.975
Receiver	3
Antenna and Microphone	6
Total Estimated Power	22.9

Table 6.8.101 - Power Estimate for the project

6.8.2 - Initial Design Architecture

The figure below depicts our overall block diagram for our power design. We plan on using the 22000mAh battery from Banggood with a pre-installed BMS (Battery Management System) as our main power source for the vine robot. The main battery will be charged using a battery charger known as the D6 Duo Pro Smart Charger. It has multiple USB ports and an input AC voltage of 100-240V and a DC input voltage of 6.5-30V. It has a charging power of 650W and a discharging power of 2x15W. Its cell count lasts from 1-6 series which provides ample capacity for our 3 series battery rated at 22000mAh. We also have a battery disconnect switch
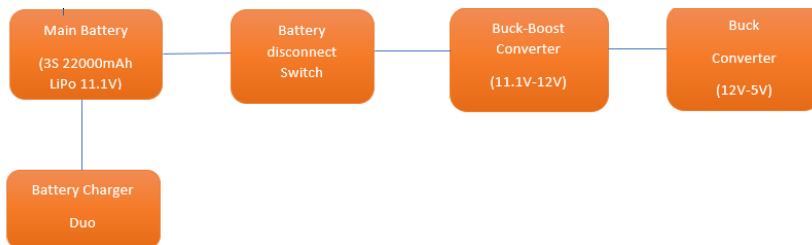
Figure 6.8.201 - Power Design Block Diagram

6.8.2.1 - Electrical Schematic Diagrams

We decided to use a switching regulator instead of a linear voltage regulator. This is because switching regulators provide higher efficiency and lower heat generation. In order for power to be supplied to other low-voltage components, a buck converter(switching regulator) is required to step down our battery voltage (12V) to (5V) raspberry pi and other manual control components. During our design process, we considered a high-efficient power converter with a maximum current output of 5A. We also took a look at a buck-boost converter to step up our battery voltage from 11.1V to 12V. We chose a converter of 93.9% efficiency which will help us provide a consistent voltage throughout the power supply. Figs 7.3.201 and Fig 7.3.202 show the electrical schematic diagrams of two types of buck converters we decided to compare and based on its higher efficiency vs output current, we eventually chose the TPS5652. We plan to use the other buck converter as an alternative if it warrants for us to do so. In Figure 6.8.2.101, the graph shows the rate of efficiency voltage increase generates the highest efficiency of the buck converter.

77
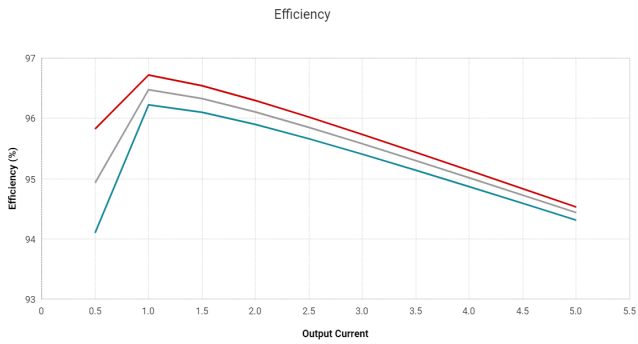
Efficiency

Figure 6.8.2.101 - Efficiency of Buck Converter vs Output Current from Datasheet TPS5652

Design Parameter	Value
Vin min	10V
Vin max	14V
Vout	5V
Iout max	5A

Table 6.8.2.101 - Webench Power Design Parameters for 5V Buck DC-DC Converter
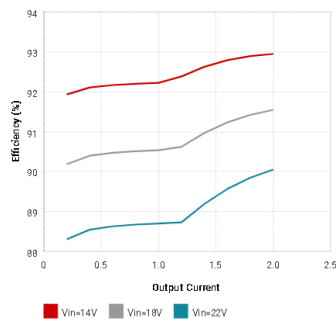
Figure 6.8.2.102 - Efficiency of Buck Converter vs Output Current from Datasheet TPS56637

Design Parameter	Value

Vin min	10V
Vin max	14V
Vout	12V
Iout max	5A

Table 6.8.2.102 - Webench Power Design Parameters for 12V Buck-Boost DC-DC Converter

6.8.3 - Explicit Design

This section explains how we brainstormed the appropriate power converter designs to use, our choice of battery and battery charger for the soft vine robot. Initially, we wanted our robot to be mainly battery-controlled which meant we had to do some research on batteries with a huge power consumption rate. A battery from banggood.com is rated at 22Ah and 11.1V. This was an appropriate power supply because we needed to power the three motor encoders that will be used to expand the worm as it moves through the obstacle course. We will also need to power the motor that controls the internal roller to prevent bucking from taking place when the worm retracts to its original position. Finally, we will need to power the gripper which will be used to pick up different sizes of an object based on weight and dimensions from one specified location to another.

6.8.3.1 - Batteries, Photovoltaics, & Voltage Regulators

Batteries have been the main source of power for many mobile devices and electrical systems. A battery cell can be described as two electrodes(cathode and anode) with an electrolyte between them. In order for the battery to produce electricity, mobile charge carriers called electrons flow from the cathode(negative electrode) to the anode(positive electrode). This constant flow of electrons leads to the batteries generating electrical energy to power the electrical load. There are two main types of batteries: non-rechargeable and rechargeable batteries or commonly known as primary and secondary batteries. Primary batteries cannot be re-used once depleted and some types of batteries include alkaline, dry-cell batteries, galvanic and zinc-carbon batteries. On the other hand, secondary batteries can be used over and over again because the chemical reactions within the battery can be reversed by applying an external voltage. There are several secondary batteries available in the market such as lithium polymer, lithium-ion, nickel-cadmium, alkaline, and lead-acid batteries as shown in the table below.

For our project, we decided lipo(lithium-polymer) batteries will be the most appropriate choice because it is lightweight and portable and it has a long shelf-life. It also has a high energy density which will be a valuable resource for our robot worm to move at a faster rate through the obstacle course.

Battery type	Advantages	Disadvantages
Lithium polymer	Good safety performance Lightweight Better discharge characteristic	Demands special protection Short life-cycle Expensive
Lithium-ion	Higher current and energy density Rechargeable Slower self-discharge	Expensive Short life cycle
Nickel Cadmium	High power density Operates in a wide-temp range	High cost Made from toxic material
Alkaline	Rechargeable Low self-discharge Operates at low temp Eco-friendly raw materials	Bigger in size If leaked, can damage material
Lead Acid	Rechargeable Withstands slow and fast overcharging Lead is recyclable and can be used over again	Slow and inefficiency charging

Table 6.8.3.101 - Battery Types, Advantages, and Disadvantages

Battery balancing is a technique used to improve the cells' charge capacity of a battery pack usually with multiple cells in series. A battery balancer is an electrical component used to perform battery balancing and these devices can be applied in many technological devices today such as laptops, phones, and electric vehicles. There are many scenarios where batteries can become deteriorated or be depleted due to cells in the batteries having different charging capacities. Battery balancing is commonly used for cells in series because cells in parallel naturally balance each other since they are directly connected to each other.

The battery management system (BMS) takes other factors into consideration such as temperature variation, voltage, or current.Usually, when a cell in a battery is fully charged, all discharging must stop. Likewise, when a cell in a battery is fully depleted, discharging must stop as well. This is because if the cells are not charged equally, it could lead to battery failure. There are two types of battery balancing: passive and active. Passive balancing is a technique used to keep the energy within the cells at an equal magnitude by dissipating energy from the most charged cell as heat which can be quite wasteful and ineffective. However, with active balancing, energy is drawn from the most charged cell and distributed to the cells with the least amount of charge stored with the aid of an inductor, capacitor or a DC-DC converter. The principle behind this concept is simple and straightforward. Energy is transferred from a cell with a higher state of charge(SOC) to a cell, using a switching capacitor that is disconnected and connected to a battery with a lower

state of charge. It can also be connected to a DC-DC converter which can be connected across the whole battery pack. Although it is more effective than the passive balancing technique, it is very costly and it also has a complex topology.

For our project, we decided lipo(lithium-polymer) batteries will be the most appropriate choice because it is lightweight and portable and it has a long shelf-life. It also has a high energy density which will be a valuable resource for our robot worm to move at a faster rate through the obstacle course. While providing a power source from the 12V battery, a battery charger will also be required in case we run into an issue of low battery and we also do not want to replace the battery every time it is depleted. This will improve the overall power efficiency of the robot. In addition, using the right type of battery is very crucial for the success of the project because there are certain hardware components of the robot that require specific voltage requirements or else they could get damaged which will lead to an unnecessary increase in costs.

There are different types of charging known as constant voltage, constant current, float charging, and many more. Constant voltage charging simply consists of a DC power supply and a step-down transformer and a rectifier to provide the voltage required for other low voltage electrical components. However, constant current charging varies the voltage applied to the battery to maintain constant current flow and switches off when full voltage is reached in the battery. The c rate is also used to determine how fast a battery can discharge or charge based on its capacity. An induction power charger uses electromagnetic energy through the use of inductors to store energy in the batteries. No electrical contacts are needed for this type of charging which prevents electrocution. Finally, float charging is a method used to maintain a battery by applying a continuous voltage and current at a minimum level till it reaches full charging capacity.

For our project, we will be implementing a constant voltage charging because it is the easiest and most cost-effective way to charge a battery. A constant voltage can return as much as 70% in the first 30 minutes. This type of charging will also be easier to use and implement.

Figure 6.8.3.101 - 22000mAh lipo battery

The battery shown in the figures above will be the one we plan on implementing in our project. Our choice of battery was dependent on reliability, high quality, energy density, battery capacity, discharge rate, cost and battery life. Another battery we decided to implement for our project was the Gens Ace GEA15K3S100E5 which is rated at 11.1V 15000mAh 3S 100C LiPo battery. It has a discharge rate of 100C and weighs about 981G. Taking all this information into consideration. We decided

81

to go with the URUAV battery with a battery capacity of 22000mAh because it provides a stronger power source and does not have a high discharge rate which prevents short-circuiting and overheating. The weight of the battery will not be too significant in our robot because the base of the worm will be stationery and only the flexible worm will be moving through the obstacle.

Description	GEA15K3S100E5 battery	URUAV Battery
Battery Capacity	15000mAh (15Ah)	22000mAh (22Ah)
Number of Cells	3	3
Continuous Discharge Rate	100C	30/60C
Total Weight	981G	1320G
Maximum Charging Current	15000mA	22000mA

Table 6.8.3.102 - Battery comparisons

Figure 6.8.3.102 - Hobbymate D6 Duo Pro

The battery charger shown in the figure above is the Hobbymate D6 Duo Pro Charger. It has dual independent charging ports and an internal independent lithium battery balancer that balances individual cells when the battery is being discharged. It also has overheating and short-circuit protection which is crucial since we will be having a lot of electrical loads that need to be powered. Another battery charger we decided to take a look at was the D6 Duo battery charger

Description	Hobbymate D6 Duo Pro	Hota D6 Pro Smart Charger
AC (Power Rating)	200W	200W
DC (Power Rating)	650W	325W

Table 6.8.3.103 - Battery charger comparisons

Figure 6.8.3.103 - Battery Disconnect Switch

In Table 6.8.3.102, we compare two different battery chargers and picked the component that was most suitable to our specifications. The Hobbymate D6 Duo Pro Charger has a higher power capacity which will serve us better in the long run when we need more power to be generated to move the worm through the obstacle course. The aim of the battery disconnect switch in the figure above is to conveniently shut down the power supply when needed to avoid battery depletion.

Our stretch goal included using solar panels as our main battery charger, we had to do some research on the different types of solar panels that are both cost-effective and readily available to us. A very important factor while choosing our solar panel was the ability for it to use excess heat generated from the sunlight to convert directly to electricity. In addition to our solar panels, batteries will be used as a backup power source in the case of abnormal weather conditions.There are four main types of solar panel cells we looked into which are the monocrystalline, polycrystalline, passive emitter and rear cell (PERC) solar panels, and thin-film solar panels. Monocrystalline cells are made from single pure silicon crystals cut into different wafers. The use of pure silicon is seen as a clear advantage because it makes these panels one of the most space-efficient and long-lasting panels. However, it can be quite expensive due to the use of excess silicon to make one cell. Polycrystalline solar panel cells are made from different silicon fragments and melted into one. Unfortunately, this reduces the efficiency of the solar panels and since they are made from multiple materials, this makes them impure and thus have lower heat tolerance which makes them unable to perform in high-temperature environments. In addition, PERC (Passivated Emitter and Rear) cells are modified versions of monocrystalline cells by having more improved features such as reflecting light back into the cell which increases the rate of absorption of the solar radiation.

Monocrystalline cells are slightly more durable than polycrystalline cells and provide efficiency within the range of 15%-20%. Their high-efficiency rate leads to producing more power per square foot which means these panels can be applied in limited spaces. They also have a longer life expectancy than polycrystalline cells, about 20-25 years. They also perform well in low-light conditions which means high solar radiation will not be required for the panels to perform at their optimal rate. Thin-film solar panel cells are both lightweight and flexible which are very practical and cost-effective to use in our design. However, in the long term, it is not as durable as mono-crystalline. These features make thin-film solar panels a possible choice for our project because we plan on making our robot portable and light because we need it to easily navigate through the objects in the obstacle

course. In addition, solar power technology has advanced over the recent years. There are three types of technology in which solar power is harnessed: photovoltaics which is the use of sunlight to strike electrons in the solar panel cells which generate mobile carriers that conduct electricity (DC power). Also, concentrated solar power provides heat to power wind turbines and is also used to heat up water for heating, ventilation, and air conditioning (HVAC).

Voltage regulators are electrical circuits that are used to step down a higher DC voltage to a lower DC voltage to maintain a constant value. There are two main types of voltage regulators: Linear voltage regulators and switching voltage regulators.

A linear voltage regulator is controlled by a transistor which provides a negative feedback control circuit needed to maintain a constant output voltage. Switching regulators use an inductor, capacitor, diode and a switch to regulate the energy between the input voltage and output voltage.
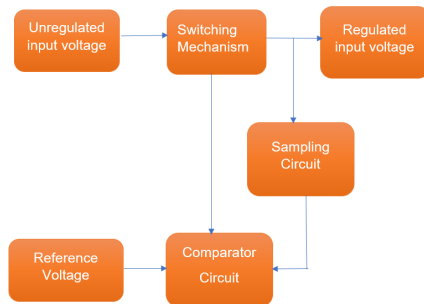
Figure 6.8.3.104 - Linear voltage circuit block diagram

The figure above shows the working principle of the linear voltage regulator. An unregulated input voltage is passed through the circuit and it is converted to a regulated output voltage with the aid of switching mechanisms(i.e transistors). There is also a reference voltage which acts as an error mechanism to determine how far away the output voltage is far from the desired output voltage, similar to a feedback control system.

Switching voltage regulators are composed of many arrangements but the three main types are the buck converter, boost converter, and a buck-boost converter. A buck converter is a DC-DC converter that steps down the voltage from its input node to its output node using a switching mechanism. It simply consists of a transistor, diode, and at least one energy-storage element (inductor or capacitor). A boost converter, on the other hand, steps up the voltage between its input and output voltage node. Finally, a buck-boost converter is used to produce voltages that have a larger or smaller magnitude than the input voltage.
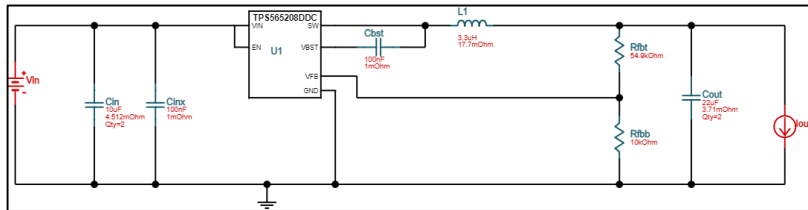
Figure 6.8.3.105 - TPS565208DC Switching Voltage Regulator (12V-5V Buck Converter)
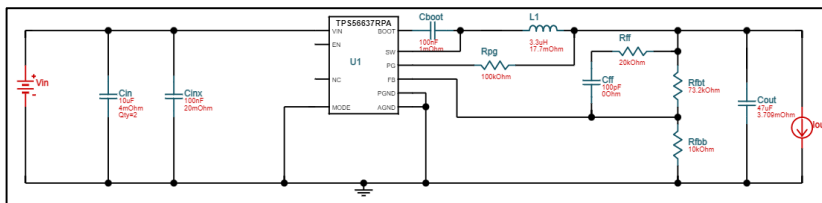
Figure 6.8.3.106 - TPS56637RPA Switching Voltage Regulator (12V-5V Buck Converter)

The figure above is shown as a buck converter stepping down voltage from 12V to 5V. This schematic was designed with the aid of the Webench tool on the texas instrument website. This provides us with an accurate representation of how we plan to implement this circuit on our PCB layout and this circuit aims to provide 5V to our microcontrollers (Raspberry pi and atmega2560). After comparing the two linear regulators, we decided to go with TPS56637RPA due to its higher efficiency. The latter figure shows the bill of materials for the components required to assemble the buck converter.
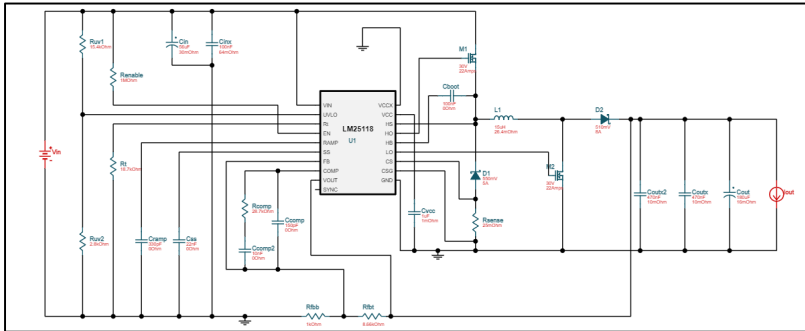
Figure 6.8.3.107- LM25118 Switching Voltage Regulator (11.1-12V Buck-Boost Converter)
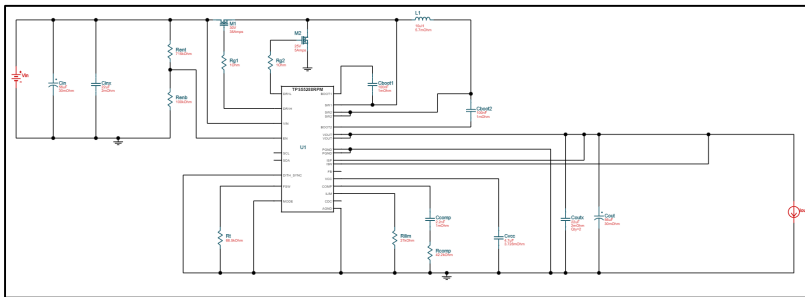
Figure 6.8.3.108 - TPS55288RPM Switching Voltage Regulator (11.1-12V Buck-Boost Converter)

The figure above is shown as a buck-boost converter stepping up voltage from 11.1V to 12V. This schematic was designed with the aid of the Webench tool on the texas instrument website. This provides us with an accurate representation of how we plan to implement this circuit on our PCB layout and this circuit aims to provide 5V to our microcontrollers (Raspberry Pi and Atmega2560). After looking at bth features closely, we decided to go with the LM25118 buck boost converter due to its higher efficiency.

Part	Manufacturer	Part Number	Quantity	Price ($)	Footprint (mm²)	Description

Rfbb	Yageo	RC0201 FR-0710KL	1	0.01	2.08	Resistance: 10 kΩ Tolerance: 1.0% Power: 50 mW
Rfbt	Vishay-Dale	CRCW040245K3FKED	1	0.01	3	Resistance: 45.3 kΩ Tolerance: 1.0% Power: 63 mW
Cinx	MuRata	GRM188R71H104KA93D	1	0.02	4.68	Cap: 100 nF Total Derated Cap: 87 nF VDC: 50 V ESR: 20 mΩ Package: 0603
U1	Texas Instruments	TPS56637RPAR	1	1.386	16	
Cin	TDK	C2012X5R1V156M125AC	1	0.21	6.75	Cap: 15 µF Total Derated Cap: 1.2 µF VDC: 35 V ESR: 1.67 mΩ Package: 0805
Cout	Kemet	C0805C106K8PACTU	3	0.03	6.75	Cap: 10 µF Total Derated Cap: 22 µF VDC: 10 V ESR: 3 mΩ Package: 0805
L1	TDK	VLP8040T-2R2N	1	0.22	113.42	L: 2.2 µH DCR: 15 mΩ IDC: 6.7 A
Rpg	Vishay-Dale	CRCW0402100KFKED	1	0.01	3	Resistance: 100 kΩ Tolerance: 1.0% Power: 63 mW
Cboot	MuRata	GRM155R71C104KA88D	1	0.01	3	Cap: 100 nF Total Derated Cap: 100 nF VDC: 16 V ESR: 1 mΩ Package: 0402
Cff	Yageo	CC0201JRNPO8BN101	1	0.01	2.08	Cap: 100 pF Total Derated Cap: 100 pF VDC: 5 V ESR: 0 Ω Package: 0201

Part	Manufacturer	Part Number	Quantity	Price ($)	Footprint (mm²)	Description
Rff	Vishay-Dale	CRCW040220K0FKED	1	0.01	3	Resistance: 20 kΩ Tolerance: 1.0% Power: 63 mW

Table 6.8.3.103 - BOM TPS56637 Switching Voltage Regulator (Buck Converter)

Part	Manufacturer	Part Number	Quantity	Price ($)	Footprint (mm²)	Description
Cout	Panasonic	35SVPF120M	1	0.73	151.29	Cap: 120 µF Total Derated Cap: 120 µF VDC: 35 V ESR: 18 mΩ Package: 10x12.6
Coutx2	MuRata	GRM188R61E474KA12D	1	0.04	4.68	Cap: 470 nF Total Derated Cap: 230 nF VDC: 25 V ESR: 1 mΩ Package: 0603
Cvcc	Taiyo Yuden	GMK212B7105KG-T	1	0.03	6.75	Cap: 1 µF Total Derated Cap: 1 µF VDC: 35 V ESR: 1 mΩ Package: 0805
D2	ON Semiconductor	MBRD835LT4G	1	0.22	101.71	Type: Schottky VRRM: 35 V Io: 8 A
L1	Coilcraft	MSS1210-223MEB	1	0.81	204.49	L: 22 µH DCR: 26 mΩ IDC: 6.7 A
Rfbb	Vishay-Dale	CRCW04021K00FKED	1	0.01	3	Resistance: 1 kΩ Tolerance: 1.0% Power: 63 mW
Rfbt	Yageo	RT0805BRD0710K4L	1	0.06	6.75	Resistance: 10.4 kΩ Tolerance: 0.1% Power: 125 mW
Css	TDK	CGA4J2C0G1H223J125AA	1	0.08	6.75	Cap: 22 nF Total Derated Cap: 22 nF VDC: 50 V ESR: 0 Ω Package: 0805

Ccomp2	TDK	CGA4J2 C0G1H3 33J125 AA	1	0.09	6.75	Cap: 33 nF Total Derated Cap: 33 nF VDC: 50 V ESR: 0 Ω Package: 0805
Ccomp	Samsung Electro- Mechan ics	CL21C4 71KBAN NNC	1	0.01	6.75	Cap: 470 pF Total Derated Cap: 470 pF VDC: 50 V ESR: 0 Ω Package: 0805
Cboot	AVX	08053C 104JAZ 2A	1	0.06	6.75	Cap: 100 nF Total Derated Cap: 100 nF VDC: 25 V ESR: 0 Ω Package: 0805
D1	SMC Diode Solutio ns	SBRD10 200TR	1	0.12	101.71	Type: Schottky VRRM: 200 V Io: 10 A
Rt	Yageo	RC0201 FR- 0718K7 L	1	0.01	2.08	Resistance: 18.7 kΩ Tolerance: 1.0% Power: 50 mW
Ruv2	Vishay- Dale	CRCW0 4023K8 3FKED	1	0.01	3	Resistance: 3.83 kΩ Tolerance: 1.0% Power: 63 mW
Coutx	MuRata	GRM18 8R61E4 74KA12 D	1	0.04	4.68	Cap: 470 nF Total Derated Cap: 230 nF VDC: 25 V ESR: 1 mΩ Package: 0603
M1	Texas Instrum ents	CSD175 71Q2	1	0.11	9	VdsMax: 30 V IdsMax: 22 Amps
Renable	Vishay- Dale	CRCW0 4021M 00FKED	1	0.01	3	Resistance: 1 MΩ Tolerance: 1.0% Power: 63 mW
Rsense	Susumu Co Ltd	PRL163 2-R020- F-T1	1	0.2	10.92	Resistance: 20 mΩ Tolerance: 1.0% Power: 1 W

U1	Texas Instruments	LM25118Q1MH/NOPB	1	2.672	71.4	
Rcomp	Yageo	RC0201FR-0715K4L	1	0.01	2.08	Resistance: 15.4 kΩ Tolerance: 1.0% Power: 50 mW
Cin	Panasonic	35SVPF39M	1	0.52	106.09	Cap: 39 µF Total Derated Cap: 39 µF VDC: 35 V ESR: 30 mΩ Package: 8x6.9
Cramp	MuRata	GRM1555C1H561JA01J	1	0.01	3	Cap: 560 pF Total Derated Cap: 560 pF VDC: 50 V ESR: 0 Ω Package: 0402
Ruv1	Yageo	RC0201FR-0723K7L	1	0.01	2.08	Resistance: 23.7 kΩ Tolerance: 1.0% Power: 50 mW
Cinx	Kemet	C0805C104M5RACTU	1	0.01	6.75	Cap: 100 nF Total Derated Cap: 100 nF VDC: 50 V ESR: 64 mΩ Package: 0805
Cvccx	Taiyo Yuden	GMK212B7105KG-T	1	0.03	6.75	Cap: 1 µF Total Derated Cap: 1 µF VDC: 35 V ESR: 1 mΩ Package: 0805
M2	Texas Instruments	CSD17571Q2	1	0.11	9	VdsMax: 30 V IdsMax: 22 Amps

Table 6.8.3.104 - BOM LM25118 Switching Voltage Regulator (Buck-Boost Converter)
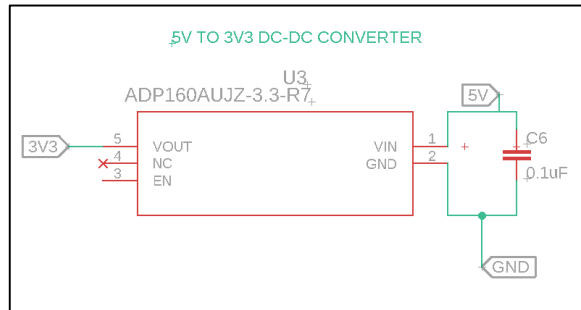
Figure 7.3.205 - ADP160AUJZ Voltage Regulator (5V-3.3V Buck Converter)

The figure above showcases a DC-DC converter, using the ADP160AUJZ voltage regulator, that steps down 5V from the Atmega2560 to 3.3V, which powers the altitude sensor and the accelerometer. This circuit will be included on the base control board, however, the other voltage regulators will be separate PCB's.

6.9 - Integrated Schematics

Our design will include a total of three printed circuit boards. The base control PCB will be utilized to interface with the various sensors on the soft vine worm and send data back to the Raspberry Pi via USB serial. The PCB will also include the motor driver and pneumatic control system. The cap sensor PCB will house the various sensors and connect to the base control board. The controller board will be a custom PCB that will be used to control the motor and movement of the robot through pneumatic control. The figure below showcases how the cap sensor board and base control board are mechanically connected within the worm body and base.
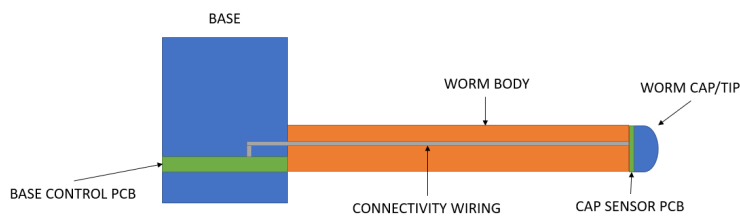
Figure 6.901 - Base Control and Cap Sensor PCB Mechanical connection

6.9.1 - Cap Sensor Board

The cap sensor printed circuit board will be connected to the worm cap and will be used to house the altitude sensor, accelerometer, and phototransistors, along with whatever passive components are needed. The figure below showcases the integrated schematic of this board designed on AutoDesk eagle. A pin header for both the sensors and the phototransistors are also used to connect to the base control board. We plan to use male pin headers and use female wires to connect the two boards within the robot body. The altitude sensor and the accelerometer will both connect to the I2C channel of the Atmega2560, while the phototransistors will connect to the analog inputs of the MCU.
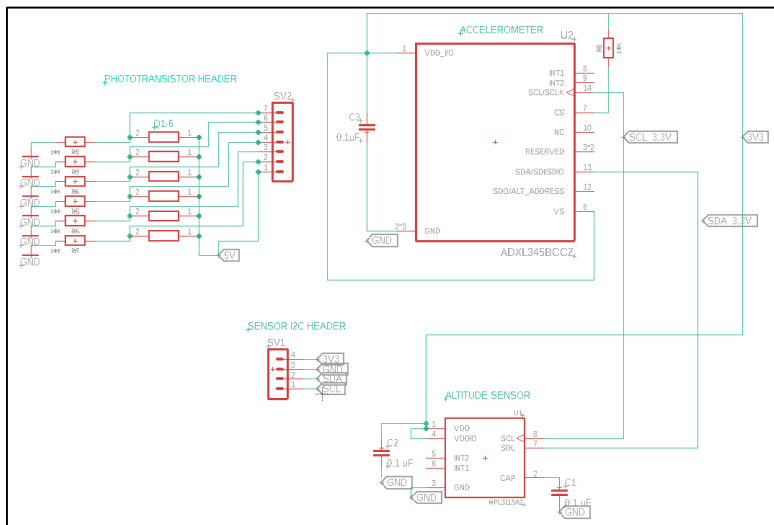
Figure 6.901 - Cap Sensor Board Schematic

6.9.1.2 - Cap Sensor Board PCB Layout Progress

The figure below showcases the current progress of the PCB layout for the cap sensor board.
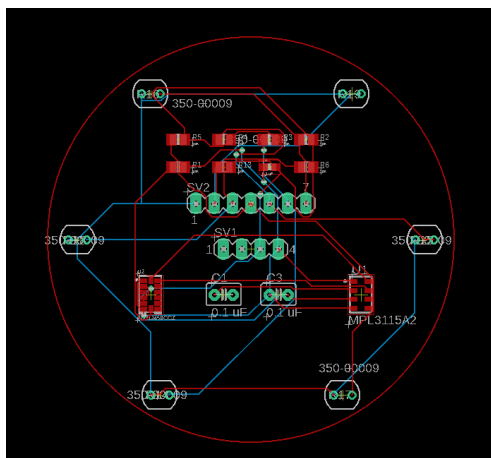
Figure 6.9.1.101 - Cap Sensor Board PCB Layout Progress

6.9.2.2 - Cap Sensor Board Part Summary

Name	Component	Value	Quantity	Vendor
R1, R2, R3, R4, R5, R6, R7	SMD-0805 RESISTOR	10k	7	Digikey
C1, C2, C3	SMD-0805 CAPACITOR	0.1uF	3	Digikey
Q1, Q2, Q3, Q4, Q5, Q6	PHOTOTRANSISTOR		6	Adafruit
U1	MPL3115A2 ALTITUDE SENSOR		4	Adafruit
U2	ADXL345 ACCELEROMETER		2	Amazon
SV1	4 PIN HEADER			Digikey
SV2	7 PIN HEADER			Digikey

Table 6.9.2.201 - Cap Sensor Board Part summary

6.9.2 - Base Control Board

The base control printed circuit board will be connected to the base of the worm and will be used to house the microcontroller, two motor drivers, pneumatic control circuit, a 5V-3.3V voltage regulator, header for the controller board, header for sensor board and phototransistors, header for USB-UART serial module, header for joystick and buttons, and a 12V power input, along with whatever passive components that are needed. The figure below showcases the integrated schematic of this board designed on AutoDesk EagleCAD.
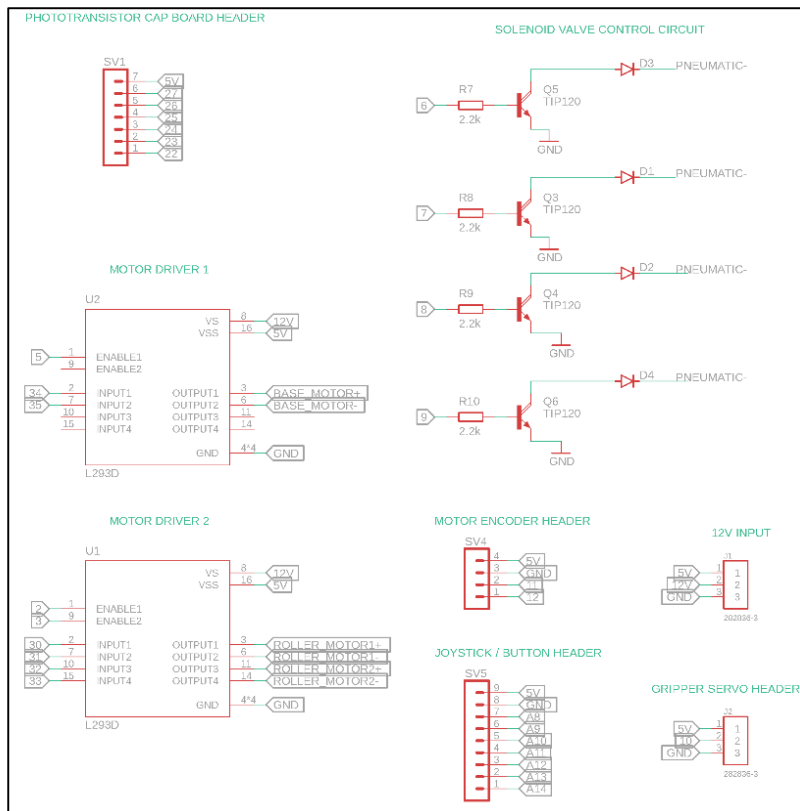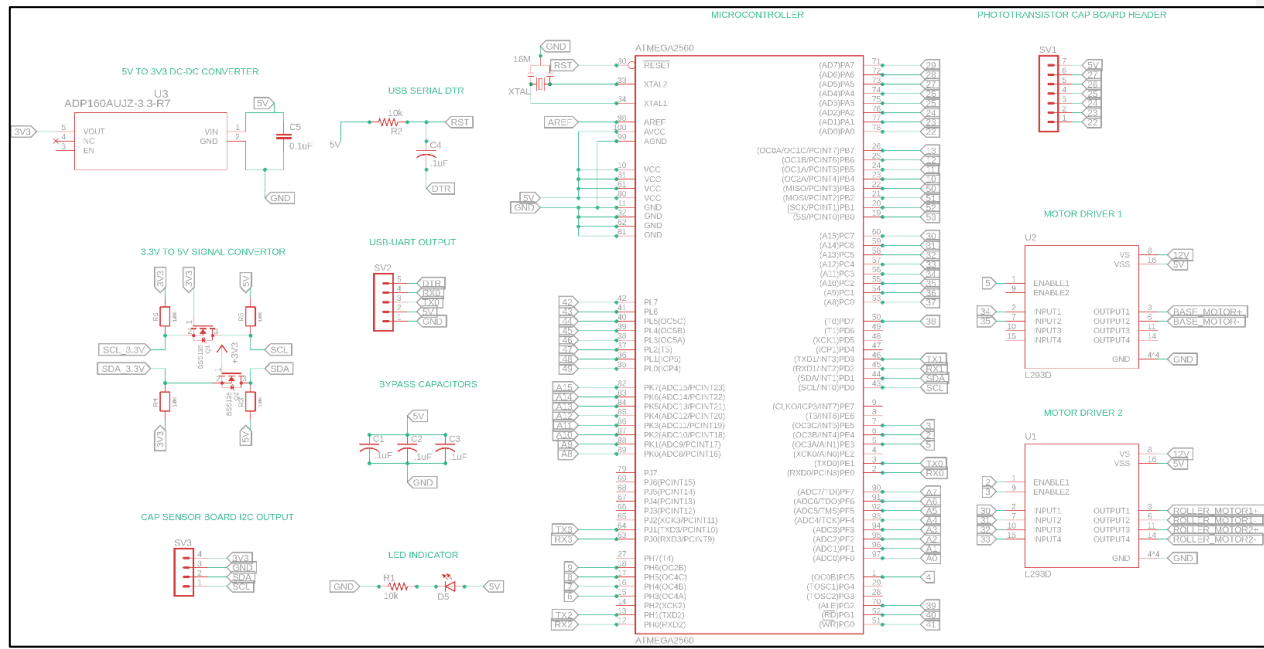
Figure 6.9.201 - Base Control Circuit Schematic

6.9.2.1 - Base Control Board PCB Layout Progress

The figure below showcases the current progress of the PCB layout for the base control board.
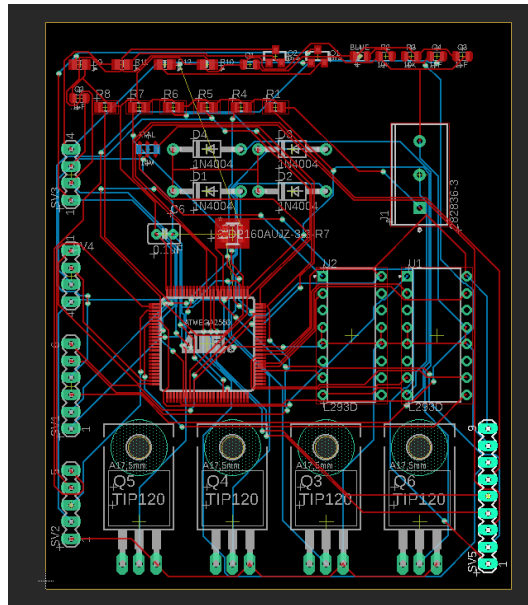
Figure 6.9.2.101 - Base Control Board PCB Layout Progress

6.9.2.2 - Base Control Board Part Summary

Name	Component	Value	Quantity	Vendor
R1, R2, R3, R4, R5, R6	SMD-0805 RESISTOR	10k	6	Digikey
R7, R8, R9, R10	SM-0805 RESISTOR	2.2k	4	Digikey
C1, C2, C3, C4, C5	SMD-0805 CAPACITOR	0.1uF	5	Digikey
Q1, Q2	BSS138 TRANSISTOR		2	Digikey
Q3, Q4, Q5, Q6	TIP120 TRANSISTOR		4	Adafruit

U1, U2	L293 MOTOR DRIVER		2	Amazon
U3	ADP160AUJZ VOLTAGE REGULATOR		1	Digikey
D1, D2, D3, D4	1N4933RLG DIODE		4	Digikey
D5	APT2012LVBC/D BLUE LED DIODE		1	Digikey
U4	ATMEGA2560-16AU		1	Amazon
J1, J2	3 PIN HEADER		2	Digikey
SV3, SV4	4 PIN HEADER		2	Digikey
SV2	5 PIN HEADER		1	Digikey
SV1	7 PIN HEADER		1	Digikey
SV5	9 PIN HEADER		1	Digikey
16M	16 MHZ RESONATOR		1	Digikey

Table 6.9.2.201 - Base Control Board Parts Summary

6.10 Electronics Bill of Materials

The following is a total bill of materials for the base control board and cap sensor board.

Component	Quantity	Vendor	Price
SMD-0805 RESISTOR 10k ohm	13	Digikey	$0.84
SMD-0805 RESISTOR 2.2k ohm	4	Digikey	$0.42
SMD-0805 CAPACITOR 0.1 uF	8	Digikey	$1.05
BSS138 TRANSISTOR	2	Mouser	$0.68
TIP120 TRANSISTOR	4	Adafruit	$15.70
L293 MOTOR DRIVER	2	Amazon	$8.99
ADP160AUJZ VOLTAGE REGULATOR	1	Digikey	$1.43
1N4933RLG DIODE	4	Digikey	$2.74
APT2012LVBC/D BLUE LED DIODE	1	Digikey	$0.58

Component	Quantity	Vendor	Price
SMD-0805 RESISTOR 10k ohm	13	Digikey	$0.84
SMD-0805 RESISTOR 2.2k ohm	4	Digikey	$0.42
SMD-0805 CAPACITOR 0.1 uF	8	Digikey	$1.05
BSS138 TRANSISTOR	2	Mouser	$0.68
ATMEGA2560-16AU	1	Amazon	$13.31
HEADER PINS	30	Amazon	$5.99
PHOTOTRANSISTOR	6	Adafruit	$5.70
MPL3115A2 ALTITUDE SENSOR	4	Adafruit	$16.50
ADXL345 ACCELEROMETER	2	Amazon	$6.71
COM-09032 Joystick	1	Digikey	$3.95
Buttons	5	Amazon	5.99
MAGNET ENCODER	1	Amazon	8.99
SG90 SERVO	1	Amazon	$6.00
METAL GEARMOTOR	2	Digikey	$59.90
SQUARE-FACE 12V DC MOTOR	1	McMaster-Carr	$62.74
16 MHZ RESONATOR	1	Digikey	$0.18
USB-UART CONVERTOR	1	Amazon	$6.99
Female to Female Wire Headers	1	Amazon	$6.98
RumCam Phoenix 2 Micro	1	Amazon	$29.99
LED Flashlight	1-2	DigiKey	$1
Motor Encoder	3	Amazon	$15.88
Accelerometer/Gyroscope	1	Amazon	$10
Motor Driver Controller	1	Amazon	$11.49
Pressure Sensor	1	SparkFun	$26.95
Servo motor	1	Amazon	$20.99
Battery pack (Lipo battery)	1	Banggood	$151.99
Battery pack (Lipo battery CHARGER)	1	Hobbymate	$119.90
Battery disconnect switch	1	Amazon	$12.99
Buck Converter (12V-5V)	1	DigiKey	$1.68

Component	Quantity	Vendor	Price
SMD-0805 RESISTOR 10k ohm	13	Digikey	$0.84
SMD-0805 RESISTOR 2.2k ohm	4	Digikey	$0.42
SMD-0805 CAPACITOR 0.1 uF	8	Digikey	$1.05
BSS138 TRANSISTOR	2	Mouser	$0.68
Buck boost converter (11.1V-12V)	1	DigiKey	$7.50
Buck Converter(5V-3.3V)	1	DigiKey	$1.43
AA alkaline batteries(12V) (8-PACK)	1	Amazon	$5.22
ATMEGA2560-PU	1	MicroChip	$2.46
Battery Disconnect Switch	1	Summit Racing	$14.99
Altitude/Pressure Sensor	1	Adafruit	$9.95
PCB Costs		PCBWAY	$20.00
Total			**$706.77**

Table 6.10.1.201 - Electronics Bill of Materials

7 Software Design

7.1 - Design Overview, Technology, and Architecture

The software components for this project allow for all the hardware components to communicate with one another, and more importantly, it allows the user to be in the loop. Below in **Figure 8.101,** we can see a diagram that closely follows this relation. Later, we explain in more detail each component and what purpose they serve in the overall scope of the project. These software programs are development tools that aid the worm to navigate both autonomously and by remote control. We will develop a User Interface (UI) that will allow the user to choose which mode the worm will be set to. When in autonomous mode, computer vision and image processing will be used to interpret the environment the worm will be navigating in order to identify and track a light bulb. We will be using the RunCam Phoenix 2 Micro Camera to give live video feed to the user, a video feed that will be processed using OpenCV. The UI will also be the host to all the different values that we will be receiving from the sensors as well as the live video of the worm's path. All the sensors will be connected to a Raspberry PI that will act as a mediator for the pneumatic controls and actuators. To make it easier for the user to navigate the worm, we add a variety of sensors for orientation purposes.
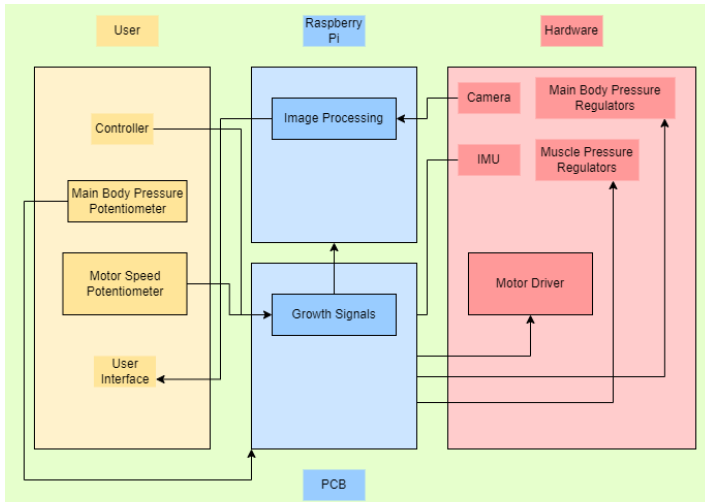
Figure 7.101 - Software Components Diagram

7.1.1 - Manual Mode vs Autonomous (stretch goal)

We are aiming to have the worm complete the obstacle course in two different modes which are autonomous and manual mode. The software design details the autonomous mode in great detail as opposed to the manual design which relies more on hardware components. However, some of these components need to interact with the MCU to convert those inputs into commands for the pneumatic controls. In manual mode, the MCU will receive the commands from the user, which will be sent over wire from a physical controller. When in autonomous mode, the robot will rely on the camera to recognize a light bulb and track it. In this mode, the camera and the sensors will report back data to the user on the UI but the user will not need to send any commands. In the diagram below, **Figure 8.1.101**, we can see how the components interact in each different mode.
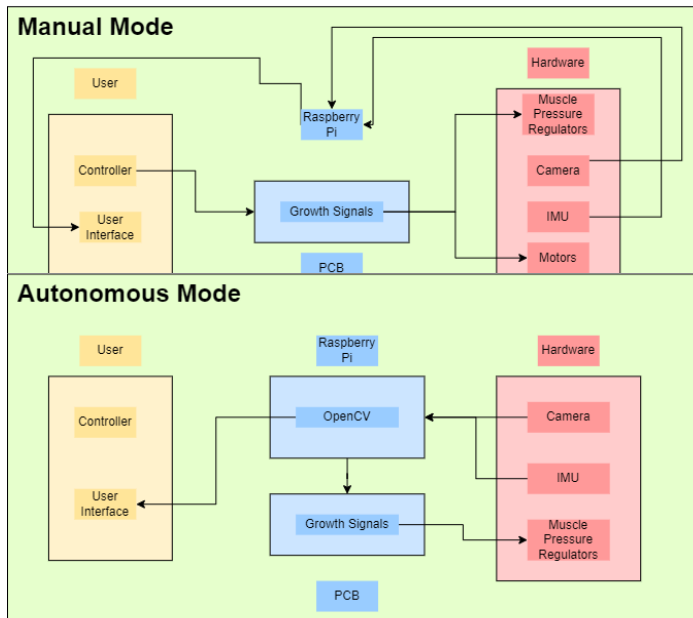
Figure 7.1.101 - Mode Specific Software Diagram

7.2 - Boot Up Sequence

The boot-up sequence details how the program will make sure that all components are turned on and ready to work. We plan to start this process from the Raspberry Pi. By running a command-line script we should be able to start the project, which will bring up the GUI and start running the main code. Once all the microcontrollers are on, the first step is to run a series of unit tests and integration tests to make sure that the functions are running as they are supposed to. After, if everything goes smoothly, we can just select the desired mode on the GUI and begin traversing. Below in **Figure 8.201**, we can see a sequence diagram demonstrating the order in which the classes and functions interact with each other.
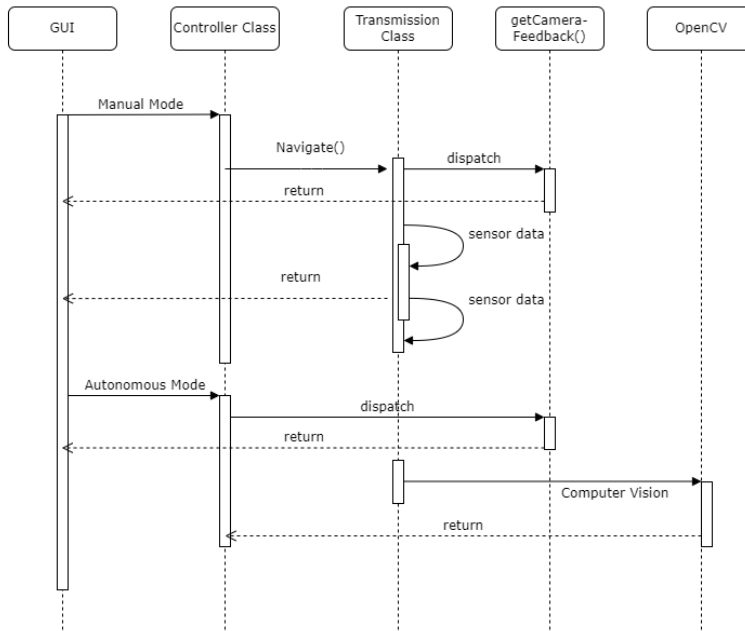
Figure 7.201 - Sequence Diagram

7.3 - UX Design

The UX design for this project, unlike most software products, does not focus on providing a friendly user interface but on functionality. Of course, we do want the user to be able to visualize the path of the worm as well as the feedback from the sensors. This will give each user a better understanding of the environment and how the robot is interacting with it. The UX will be developed in a GUI that is able to receive signals from different ports. Another important functionality of the UX is the ability to provide the user with a mode switching button (in case of autonomy implemented), which will direct whether the robot will be traversing autonomously or manually. This will interact with the MCU and allow the pneumatic muscles to be controlled by the user if in manual mode or it will create the reactive architecture needed for the worm to navigate by itself. Below in **Figure 8.301** is a demonstration of the layout of the Graphical User Interface.

Graphical User Interface

Video Feedback

Sensors	
Metric	**Value**
C/K	Temperature Sensor
Humidity	Humidity Sensor
Speed	Accelerometer
Pressure	Pressure Sensor
Orientation	Gyroscope

Object	
Identified	Y / N
Following	Y / N

Manual   Auto
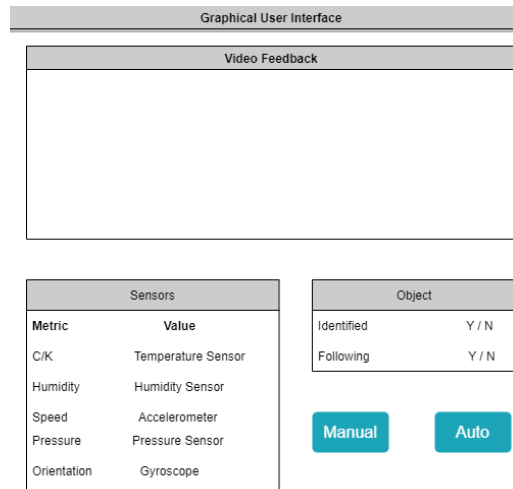
Figure 7.301 - UX Design Layout

7.4 - Computer Vision

Computer vision is one of the main components of this project that allows the robot to navigate autonomously and recognize the light bulb. Initially, we wanted to use a real-sense camera that allows the worm to map out its surroundings and give it a better understanding of its environment. Unfortunately, this feature was constrained by the size of the camera required to record and process the video feedback. Therefore, we will be using a RunCam Phoenix 2 Micro camera that allows for video processing without depth perception. The shortcomings of not having a real-sense camera are replaced by the other sensors that are explained in the next section. The image processing is done using the OpenCV library which will allow us to identify a parametric object.

7.4.1 - RunCam Phoenix 2 Micro

The RunCam is one of the favorite FPV (First Person View) cameras on the market right now. Mainly used for drone piloting and racing, this camera provides a high-quality image, even for objects moving at a high speed. The robot utilizes this camera as a guide when in Manual mode and for object recognition in Autonomous mode. This camera performs really well in dark spaces due to its f/2.0 Large aperture lens which is ideal for a robot that is meant to navigate tight spaces and even underground. The camera will be connected to the Raspberry Pi and transmit the video via analog and obtain power from it as well. Some specifications of the

camera are detailed below in **Table 7.4.101.** The video feedback will be presented to the user via the Graphical User Interface.

Model	RunCam Phoenix 2
Image Sensor	1/2" COMS Sensor
Horizontal Resolution	1000TVL
Lens	2.1mm (M12) FOV155° (4:3)
Screen Format	4:3 / 16:9 switchable
Mirror/Flip	Available
Signal System	PAL / NTSC Switchable
S/N Ratio	>50dB (AGC OFF)
Electronic Shutter Speed	Auto
Auto Gain Control (AGC)	Auto
Min. Illumination	0.001Lux@1.2F
WDR	Global WDR
Day/Night	Color/BW/EXT/Auto
Menu Control	Cable Control
Net Weight	9g
Dimension	19mm*19mm*20mm
Current	220mA@5V / 120mA@12V
Power	DC 5-36V

Table 7.4.101 - RunCam Feature Specs

The camera will mount to the cap of the robot at the tip and will be wired to the base of the robot by running both the communication wires and power wires through the inside of the body of the robot. To control the settings of the camera we must purchase a separate controller by RunCam which they simply call the RunCam Key Board. This is just a 4 button board that plugs into the camera and

allows you to navigate the settings menu while looking at the output display of the camera.

To transmit the video stream the camera will connect to a transmitter. We have chosen to go with the RUSH Tank II Ultimate which is a well-known FPV drone camera transmitter. The transmitter is very compact, has pins for a microphone expansion, and comes with a "RUSH Cherry" lollipop antenna. We will also be using the RUSH AGC dynamic gain microphone. This RUSH transmitter will transmit the camera feed at the standard 5.8 GHz that most FPV drones use to stream video, and we will also be able to hear the noise around the tip of our robot thanks to our added microphone. The audio will not be mission-critical to our robot, but it will be a nice addition for when we record our robot and present it later on. The RUSH transmitter will communicate with the Skydroid 5.8 GHz OTG receiver (another common piece of hardware amongst FPV drone enthusiasts), which will deliver the video and audio stream to the Raspberry Pi via USB.

7.4.2 - OpenCV

The video processing is mostly done using OpenCV which is a cross-platform open-source library that provides tools for any kind of image and video processing. This is a tool for Python developers even though most of its documentation is written in C++ and also supports other languages like Java and MATLAB. This software supports Windows, Linux, Android, and Mac OS. The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms.

Processing a video is just performing operations on frames which are images of a video at a certain point in time. OpenCV focuses on Image Processing, Video Capturing, and analysis including Face Detection and Object Detection. This is widely used in the Robotics field as a navigation and obstacle avoidance tool just as we plan to do for our robot. Some of the important modules that we will be using are .imgproc, .video, .Videoio and .imgcodecs. These modules will assist with the video analysis such as motion estimation, background subtraction, and object tracking as well reading in the video streams.

We will use this software to steer toward the light by sending the same commands that the controller would to the pneumatic controls. The location of the light bulb is computed and we can designate a radius from the center of the frame window, possibly of 100 pixels but we would have to test, then we can set a threshold of pixels so that if the light is that amount of pixels away we can send the respective command to move in that direction.

7.4.3 - Tracking Algorithms

Object tracking is one of the most difficult components that we want to implement in this project. Object tracking is where a detected object in a video, interpreted as a set of frames, and its trajectory is estimated. In this project we will only be

tracking one object, therefore making the algorithm we are using a Single Object Tracking (SOT) algorithm. In this section, we compare some of the algorithms that are available to us in the OpenCV library and we discuss which one is most beneficial for our case.

- **Boosting Tracker:** A tracker based on the AdaBoost ML algorithm. The classifier is trained at runtime and it takes time to learn the positive and negative examples of the object. Generally, really slow since it is over a decade old.

- **CSRT:** This tracker trains correlation filters with compressed features. It is slower than KCF algorithms but more accurate than counter-parts, and very robust to unpredictable motion. It can also recover from failures but does not include failures from total object occlusion.

- **KCF:** Tracker works by training a filter with patches containing the object as well as nearby patches. It is faster than CRST and adapts to scale and rotation but it does not recover well from failures or total object occlusion.

- **TLD:** This tracker trains a classifier that is then used to re-detect the object and correct any tracking errors. This tracker recovers well from full occlusion and adapts to scale and deformation but it also reports lots of false positives and does not report failures very well.

- **MedianFlow:** Tracks both forward and backward displacement of objects and measures the difference between the trajectories. This one does a good job at reporting failures but it can fail if there is a quick change in motion.

We are going to use the CSRT tracking algorithm since it seems more applicable to the project. Our object will not be moving at a fast speed if at all. The motion will come from the everting vine robot. One issue that we do have to account for is object occlusion since the light bulb might be located behind walls in the obstacle course, so we are choosing an algorithm that can account for that.

7.5 - MCU Configuration

Before we can establish a connection between the components that make up this project, we have to set up the control boards that we will be using. As previously mentioned, we have microcontrollers at the head of the robot, in the navigation controller, and the main control board as well. All these microcontrollers need to be set up so that they can interact with the motors, solenoid valves, Raspberry Pi, and with each other. To start the configurations we first include any of the necessary libraries that are to be used by the main functions.

The setup function is a small component in the code for the microcontrollers but they ensure that each has their protocols initialized as well as their input and output pins. In this setup function, we assign each pin a corresponding sensor, in the case

of the sensor control board, and for the main control board, we will need a set up of the pins for the relays and the 3 motor controllers. Of course, they will both need the I2C connection setup as well for data transmission.

The rest of the code within each microcontroller will reside inside of a loop that will run as long as the robot is active. There will be a variable that will be shared by all the components to indicate when we should exit out of the programs. The functions and commands inside the loops will vary from component to component but they all aim to accomplish the main tasks for each microcontroller, whether that is reading sensor data, sending motor commands, or doing the image processing, which is the case of the Raspberry Pi. Within the loops, we also include the I2C communication so that the components are actively trying to send data continuously.

7.6 - Sensors

There are many sensors that we will be using on this project. We initially wanted to use these sensors to create a map for the robot to be able to navigate autonomously. However, time being one of the biggest constraints, we had to limit the features that we wanted the robot to have. This does not mean that the sensors are rendered useless. These sensors are important for the user to have an understanding of the environment when they cannot experience it themselves. And so, the information acquired from these sensors will be displayed in the GUI for the user. This means that the software has to take care of the interface connecting the output from the sensors to the PCB and Raspberry Pi, which ultimately end in the user's computer screen.

7.6.1 - Sensor Signal Transmission

The signal received from the sensors have a long trajectory before they arrive at the display to be seen by the user. The data is transferred via I2C from the sensors to the MCU, which then sends it to the Raspberry Pi to be displayed on the GUI. The data incoming from the sensors will be passed as blocks of bytes but will be received at the Raspberry Pi as floats so it can be as detailed as possible. For the I2C connection, we will use the Python *smbus* library. This Python module allows access through the I2C/dev interface on Linux hosts. The main two functions from this module that are used are *write_byte_data* and *read_i2c_block_data.* This applies to all the sensors we are using which are the IMU, and the altitude sensor. This transmission is used for sensor-MCU connection; and for MCU-Raspberry Pi connection, we use the module *pyserial* to establish serial connection. The Raspberry Pi is used mostly for video processing purposes but it will also be in charge of creating the GUI so all the sensor information does ultimately have to go to it even if there is no processing of that data.

7.7 - Software Class Diagram

The software design includes a few classes that need to interact in order for the entire system to function appropriately. These classes contain the methods that will acquire the data and transmit it across the system. The class diagram below in **Figure 8.701** shows the interaction between classes and the functions within them.

The Transmission class is a collection that has the mission of assuring that the data flows as previously described. The sensors PCB will contain only two functions: i2c_rcv() and i2c_send(). These two serve to acquire the data from the sensors and then send them to the main control unit. The MCU will be doing most of the heavy lifting for control of the vine robot but within the Transmission class, it will also contain an i2c_rcv() function and a send_data() function that will interact with the Raspberry Pi.

The controller class is a bit more complex than the Transmission class. This is because there are more components that are actively interacting. This class's modules will be spread around the system since there are three different microcontrollers in play. The main code will be written in the Control Board PCB. Here we will have a global variable keeping track of the mode that the vine robot is set to run in. The main module in this class is called vine_control() which will send the directional commands using Python's Motor module.

The commands will come from either the Raspberry Pi if we are tracking an object or from the Controller PCB. The controller PCB itself will have a module, send_cmds(), which uses the *pygame* module to convert joystick input into receivable commands. The motor control is used for growth and retraction which will be one dimensional, having the grow() and retract() functions loop continuously until the vine body is fully extended or fully retracted. Pneumatic control will also come from this class from the Control Board and we plan to use Pulse Width Modulation technique for the solenoid valves.
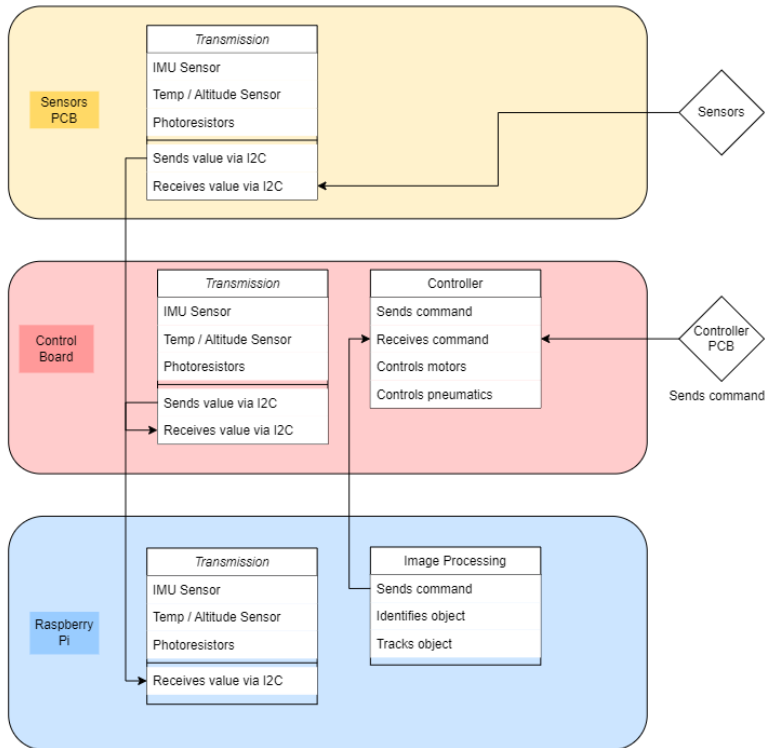
Figure 7.701 - Class Diagram

These classes take care of most of the firmware for the vine's hardware. However, we still have some major computing that needs to be done in the Raspberry Pi, and that is for the computer vision aspect of the project. To identify the light we use the cv2 library to modify the frames as we encounter them. We plan to process around 50 frames per second if needed but ideally, we want to process as low as 25 frames per second. This is because for tracking the light we want to use the CSRT tracker which is very accurate for lower fps. This algorithm works by training a correlation filter with compressed features. The filter is then used to search the area around the last known position of the objective in frames where it succeeded. This tracker is robust to unpredictable motion and can easily recover from failures when the object has not moved much. It does have some trouble recovering from when an object is fully hidden but due to the slow growth of our vine, this should not be a problem since it can quickly be retrained. Another great feature about this tracker is that it can adapt to scale, deformation, and rotation, which are some things we can encounter as we traverse the obstacle course.

7.8 - Final Coding Plan

During this section, we describe the main functions that we plan on including in the code for our system. There are two main classes that we will be designing which are a Transmission class and the Controller class. These two classes will divide the tasks of receiving the information from the sensors and sending navigation commands from the controller to the MCU and pneumatic controls. We will also describe the code that will create the Graphical User Interface for the user to see the video and sensor feedback. Should we have the time to implement the stretch goal of using computer vision to have the worm recognize an object, such as a light bulb, the GUI will also contain a button to activate said feature.

7.8.1 - Coding Plan for Controller

The *Controller()* class will consist of a series of functions and variables that will take the user input and transmit it to the vine robot. The controller we have chosen for this project is a custom-made controller that is wirelessly connected to the custom-made Control Board PCB. In order for us to be able to receive input from the controller, we need to install the *pygame* module to get the input from the controller's joystick as well as any buttons we need. There are two functions that will be separate from the main function and these are the *grow()* and *retract()* functions. This is mainly because these functions will only be called a couple of times for each run as opposed to the navigating input from the joystick. These functions will be bound to specific buttons which will set the robot in motion.

In contrast to these simpler functions, we have the main *navigate()* function which runs as long as the robot is active. This is because we need to control the vine robot as long as we need to achieve the task at hand. As opposed to the usual four-wheeled robot, this function will be a bit more complex than just mapping an axis angle of the joystick to a specific motor to complete a move to the right or left. Our robot has three directional axes making it a bit more complicated to measure the amount of pressure needed to distribute to each artificial muscle at a time. However, the basis still remains that this function will receive input from the controller and will pass said input to the pneumatic controls which act as our motors for navigation in this project. This is done using the *pyserial* module which lets us establish a serial connection to the Raspberry Pi and *smbus* for I2C connection with controller PCB.

7.8.2 - Coding Plan for Receiver

The *Transmission()* class (**Figure 8.8.201**) makes sure that we are receiving all the information from the sensors and using it correctly. The main function in this class is the *getSensorData()* function which is in charge of reading all the data from the accelerometer and the other sensors. Another function *getCameraFeedback()* will get the video feedback in real-time. All of this information will be stored into variables and directly sent to be displayed on GUI. The GUI itself will not have any

commanding functions besides changing the mode in which the vine will be operating. This is the *switchMode()* function. This function is responsible for setting the global variables for Manual or Autonomous mode.
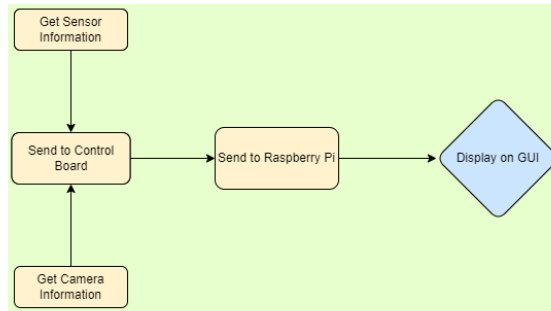
Figure 7.8.201 - Transmission class data flow

7.8.3 - Coding Plan for Image Processing

The image processing portion of the project is the most software-oriented portion even though it contains hardware integration as well. Not only do we need the video feedback from a physical camera, but we also need to communicate the pneumatic commands how to follow the light. We will use OpenCV which contains all the necessary libraries and methods for this feature.

First, our code will make sure to set up the tracker, which we will explain in the testing section, and then start capturing the video input. We import the library *cv2* for accomplishing all of the video capture and processing. We then proceed to create the bounding box that will surround the light blob as it identifies them. The purpose of the boxes is for the tracking itself; once the box is around a blob of light, the software will continue to track this blob. From there we will implement an infinite loop that will process every frame and use the tracking algorithm to focus on the light blob. This loop can be broken out by pressing a key which we will set to another button on the controller.However, the loop will not be broken when an object is not identified or gone into occlusion. The algorithm we use is able to recover from partial occlusion and should have no problems reporting failures as well. This process can be seen in **Figure 8.8.301** below.

111
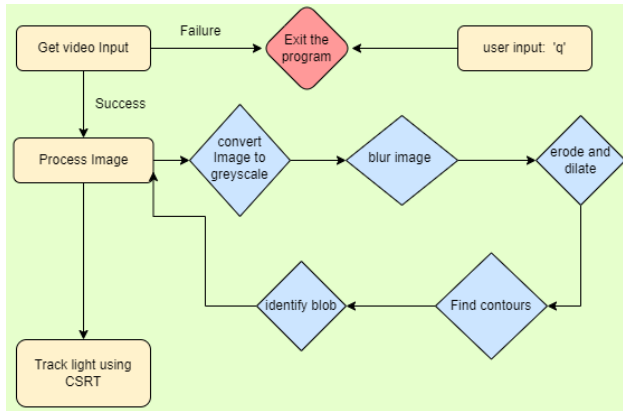
Figure 7.8.301 - Image Processing Coding Plan

The second portion of the Image processing module will be the one to interact with the hardware. Using OpenCV still, we can map out the video frames and calculate the angles in which the vine robot should be moving to keep the light blob centered. This is what actively lets the vine robot actually track objects. To accomplish this, we will calculate if the amount of the light blob is above the designated threshold of pixels that are outside of the center radius of the frame. These parameters are then passed to the Controller class' *navigate()* function. Here in this diagram, **Figure 8.8.302**, we can see how the steering works when tracking a light bulb
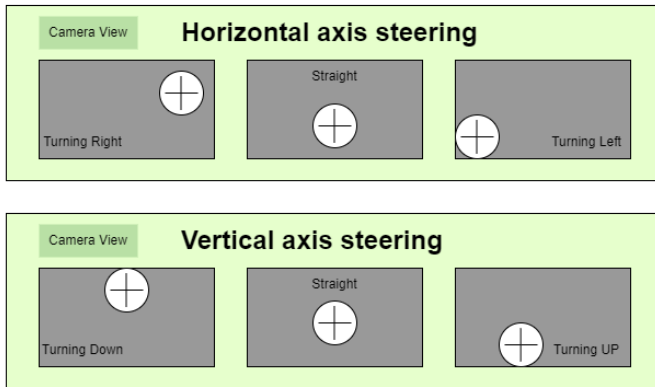
Figure 7.8.302 - Image Processing for Steering

7.9 - Summary of Software Design

This project does not heavily focus too many software components as opposed to the hardware components that make it up. Our software implementation will make sure that these hardware components can safely communicate with each other. The software in the MCU will receive information from components such as the sensors and report it back to the user via the User Interface. The MCU will also receive the input controls from the user who provides them via the controller. The input received will be the basis of the navigation of the vine robot. As a stretch goal for the software, we will use the camera mounted at the tip and image processing modules in order to have the robot recognize a light bulb and follow it.

8 Prototyping

For the prototyping section of our project, we will be implementing our PCB layouts based on the schematics we designed from the Autodesk Eagle software. We will also be comparing multiple PCB vendors and creating an outline to describe which vendor is more resourceful and dependable. We will be using 3 PCB boards which include one for the tip of the robot worm, the control board and the robot controllers.We will also be looking into the pneumatic and soft materials that we plan on using for our project and the process of going from the design phase to the implementation phase. In order to test the reliability of our project, we plan on using this summer semester to build a prototype so we can compare designs and come to a more reliable conclusion.

8.1 - PCB Vendor and Assembly

We took a look at two companies who have experience in the PCB industry: JLCPCB and Advanced Circuitry International. Before we went in-depth with the company, we ensured all information provided in this document was directly sourced from their website which is cited in the reference section.

There are several qualities we looked for while picking a company such as cost-effectiveness, high-quality reviews, and overall being trustworthy. Below is some information we gathered based on our specifications we made for our printed circuit boards.

Reliability: They currently have more than 200 customers per month who have given them high-quality reviews.

Content: They have a PCB design software with an extensive library

There are two ways for us to build our circuit boards: using breadboards or printing an actual circuit board. Although using a breadboard allows room for maneuvering and repositioning of components, it can be easy to loosen wires and make wrong connections which makes printed circuit boards the most probable choice. Printed circuit boards offer more precise and seamless connections. A PCB is a layered mask composed of a silkscreen, solder mask, copper, and a substrate. The types of substrates used are FR-4 which is more heat-resistant than the FR-2 which can be stacked onto multiple substrates. With these advantages, printed circuit boards allowing layers to be connected seamlessly makes it a better choice than breadboards. We plan to order our PCB from JLCPCB. The table below shows a detailed comparison between many PCB design software.

PCB Software	Cost	Features

Autodesk Eagle	> $780 *free for students	Extensive library Widely used by the online community Almost used by all manufacturers
Altium	> $7000	Extensively used by universities and leading schools Easy data migration Design environment with diagrams and specifications
KiCad	free	Can select the physical imprints of the components used on the diagram
DesignSpark	free	Diagram entry Routing and automatic component positioning Project interface to organize files

Table 8.1.101 - PCB Software Comparison
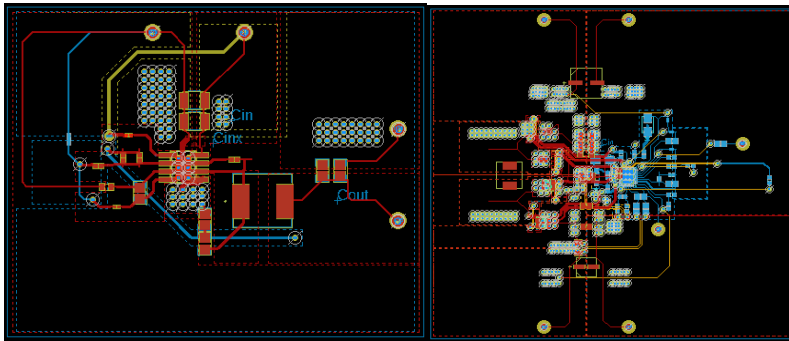
Figure 8.1.102 - Top view of PCB layout for buck converter (left and right)

For our PCB layout, we have decided to go with Autodesk eagle since it provides a free version for students. It also has a user-friendly interface which makes it easier to access tools to draw electrical components and wiring. We will be designing our circuit using the schematic and board layout. After the design process, we will be sending our final version to our preferred PCB manufacturer who will help us print out the boards so we can attach them to the control and circuitry component of our worm robot.

8.2 - Build Plan

The build plan details the steps our group will take to bring the design to life. Having done our research and selected the components, we now are faced with the hardest task which is to implement the design. We first describe the build for individual components for both the mechanical and the electrical aspects. Then we explain how the integration process will bring the vine robot to life. The integration will prove difficult as we have to test compatibility and realistic design flaws that we might have missed during our design process. However, this is to be expected and, hence, why we must build (a) prototype(s) to give us a better sense of the design. This way, we can avoid blocks further on the development process of the integrated system.

8.2.1 - Mechanical Build Plan

When we build the body and steering chambers the body must be twice as large in diameter as the steering tubes, and the steering tubes must be six inches longer than the body tubes. When using the LDPE material we can simply purchase tubing that is a certain diameter and another sized tubing that is half of the diameter of the first tubing. However, when using the TPU-Coated Nylon, we must cut the fabric and then heat seal it together to create the tubing shapes that we want. In order to do this, we must cut the fabric in the manner shown in **Fig. 5.3.101**. In our case, we would like our robot's body tube to be 5 inches in diameter and the steering tubes to each be 2.5 inches in diameter. We would also like the robot to be 10 feet long. Thus we will cut the W in the figure to be 15.71 inches, and the L in the figure to be 120 inches.
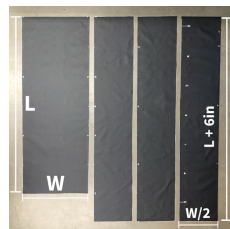
Figure 8.2.101 - TPU-Coated Nylon Cutting Guide
(reprinted with permission from vinrobots.org)

Once the fabric has been cut to size, we must trace around the end of the tube fitting, about 2 inches from the the end of the fabric while centered on one end of the width. We then cut a hole in the inside of the traced circle and twist the tube fitting through the hole, and finally, we also screw the opposite end tightly. We repeat this process for all the pouch motors we have. Once we have all the robot body and side pouches reade to be heat-sealed, we are going to evenly mark the textile at every C/4 inches, where C is the Circumference of the robot outlet,

starting from the end. Once we have the marks, we proceed to heat-seal the pouches at every mark and along the seam. If the heat-sealing proves successful, we should be able to run a pressure test with no air leaks. To finish preparing the body of the robot, we just need to measure about ⅓ of the circumference of the body, mark it, and partially invert the body so that the tail sticks out. Thereafter, we cut a piece of fishing line of around 2 feet longer than the length of the body and tie off the tail. We then can evert the robot's body and tape the pouches to the three sides previously marked, and finally, we connect the body to the outlet of the base using hot glue or tape.(**see Figure 8.2.102**)

Figure 8.2.102 - Vine Body Construction
(reprinted with permission from vinrobots.org)

To build the base of the vine robot, we need to 3D print some of the components and have a laser cutter readily available for cutting some plastic elements. To prep the body outlet where the main body is attached, we just cut a 3 inch plastic tube to a length of 6 inches. The base frame consists of making a plastic cylinder that we will measure to make some holes to let the air pumps in. We also need to make two holder plates that we need to glue to the inside wall of the cylinder base so that these holders can hold the plates containing the top and bottom closing plates, which we will glue to the cylinder. The next step would be to prep any wires we need to go into the base and make sure we air seal all of the holes we created for the wires and the air pumps. We connect the motor encoder to the proper wires and then place the spool inline with the motor encoder and resting on the acrylic holders inside the cylinder. Finally we tie in the fishing line to spool and attach the spool to the shaft components using hot glue. (**Figure 8.2.103**)

Commented [1]: double check section.

Figure 8.2.103 - Base Design
(reprinted with permission from vinrobots.org)

117

9 Testing Plan

9.1 - Prototype Test and Evaluation Plan

Testing the prototype will have multiple phases as we move through the components. We make sure to test the components in isolation before we can run an integration test. In the following sections, we detail the testing plan for each aspect of the project. We also describe the obstacle course since it is the final test that our project must pass. During each tests, we have points of evaluation that tell us if it is ok to move on to different tests, and this way, we can guarantee progress.

9.2 - Obstacle Course Design & Description

The obstacle course designs are meant to highlight the features that the team has implemented on the vine robot. The goal of this project is to have the robot successfully navigate these obstacle courses which are supposed to represent different real-life use cases. We previously mentioned that this type of robot has been used for archeological navigation but there are many other cases where the user does not have a predetermined path set for them. Many researchers want to implement similar products in search and rescue or search and deliver missions in cases of tragedy. Therefore, navigating around obstacles is one of the main goals of this project as well. Below there are three different designs for the obstacle courses with a brief description of them.

The material we will use for the obstacle is mainly plywood. There is no need for the material to be any harder or abrasive as we only want to demonstrate the navigation aspect of the vine robot.The ground base will consist of 3 of these plywood panels that are 4x4 feet in length and a ¼ inch thick so it is not too heavy for mobility.

We previously mentioned that the robot is capable of traversing different types of soil and environments so we want to introduce sand at the beginning of the first course right before it enters the arch. The obstacles and walls will be created using styrofoam rods and sheets. We can use a hot glue gun or styrofoam glue to create the shape of the obstacles and use the Velcro tape to make sure these are firmly secured to the ground as they will be run into and we do not want them to be displaced during trials. For the foam rollers, we will just use the beach pool tubes that are light and inexpensive. We will lightly tape them to the floor with Velcro or double-sided tape so they do not fall by themselves but only if knocked down. All obstacles should be easily removable for the sake accessibility to the different obstacles course options.

In **Figure 9.201,** we can see our first obstacle course which is a simple model representation. The course has a measurement of 12x4 feet. The walls depicted are just high enough to show that these are walls to get around and not over. On

118

the first wall, we can see that it contains a hole through which the vine robot will pass through. The radius of the arc will be just slightly bigger than the radius of the vine robot on the sides and just slightly longer than the diameter of the bot in height. This is meant so that the user has to be precise while controlling the robot. Shortly after we have a two-step ladder which will help demonstrate that the robot can navigate on a vertical axis. Note that the steps are not smooth so the robot cannot just rely on pure forward pressure to push itself up. Lastly, we have some foam rollers that will be loosely placed at the end of the course to once again demonstrate navigation and the tight buckling system we implemented, which is supposed to have the vine robot keep its figure as it turns instead of turning its whole body. To indicate success in this obstacle course, we want the vine to navigate easily through the obstacles, however, as retraction is also part of the implemented design, we also want to be able to safely retract it without causing any ruckus on the way back.
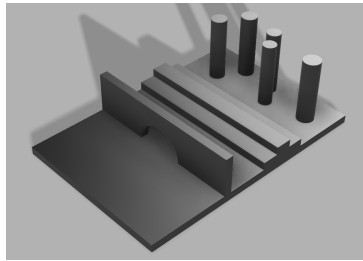
Figure 9.201 - Obstacle Course 1

Our second obstacle course will demonstrate the vine robot's ability to turn on a horizontal axis only. However, this course allows the bot to showcase how it can turn in the sharp corners as seen in **Figure 10.202** below. The course itself has a measurement of 12x4 feet as well. Furthermore, we also want to demonstrate how the robot can also be guided by its environment. This means that we have extended the walls so that the bot can just follow them after completing a turn. Just like the previous obstacle course, this course will consist of the same materials and the walls shall be just high enough so that the robot cannot skip over them, but not too high where the user cannot see into the obstacle course from different angles. We will have to test this height requirement but it is expected to be around 1 foot in height.
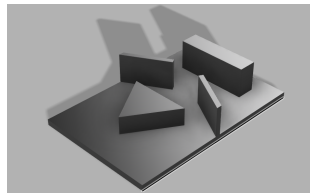
The third obstacle course is designed to showcase navigation at its peak. There will be no environment-guided navigation. The user will have to move the vine robot in an upward and sideways direction to make it fit through the tunnel provided. The course has a measurement of 12x4 feet and the tunnel will have a diameter of 7 inches which is just slightly bigger than the diameter of the head of the robot. This course shall also demonstrate the ability to navigate in tight spaces. The material used will also be plywood and the tube will be made of plastic. The tube may only be around 1 foot in length but we will need to test and make arrangements for the sake of demonstration. The height of the wall sustaining the tube may not be longer than 1 foot since, given the length of the obstacle course, it might prove to be a steep incline to get to the tube. This is something that we will also have to test as we go. Depending on the force and size of the actuators, the angle on which the vine robot turns will vary and so we will have to make adjustments.
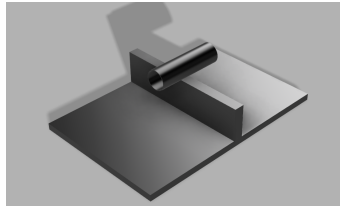
Figure 9.203 - Obstacle Course 3

To ensure mobility, we are going to design the obstacle course in a way that is portable. We previously showed all the obstacle courses as three different entities so it is visible that we will have three different courses. However, we are going to design it all in one base. We are going to buy 3 pieces of 4x4 feet plywood that will fold in accordion style. We will use hinges to accomplish this so whenever we need to carry the obstacle course, it looks like 3 4x4 feet pieces stacked. We will have the Velcro tape on the base in all the possible arrangements for the walls and obstacles. This way once we demonstrate or test our design, all we have to do is tape the desired obstacles to create one of the 3 above designs. We believe this is the most effective design in terms of both time and space constraints. Below in **Figure 10.204,** we demonstrate how the accordion style base should look.
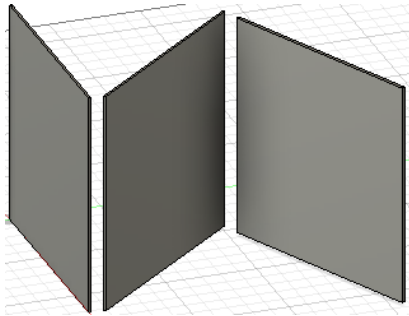
Figure 9.204 - Accordion-style Base

Below we detail the bills of materials as expected for the obstacle course. These materials we expect to buy ourselves but depending on the availability of some of the UCF facilities we might be able to save some money. (see **Table 9.201**)

Material	Quantity	Price	Link
Plywood	3	35.94	15/32-in 4-ft 4-ft Pine Sanded Plywood in the Plywood department at Lowes.com
Hinges	4	0.77	Gatehouse 1-in Zinc Mortise Door Hinge (2-Pack) in the Door Hinges department at Lowes.com
Velcro tape	3	7.47	Amazon.com: VELCRO Brand 6 Ft x 3/4 in	Sticky Back Tape Roll with Adhesive	Cut Strips to Length	Hook and Loop Fasteners	Perfect for Home, Office or Classroom, Black, 90975W : Industrial & Scientific
Styrofoam glue	2	10.99	Amazon.com: Beacon Hold The Foam Glue 2oz : Everything Else
Floating Pool Noodles Foam Tube - 3 pack	2	12.99	Floating Pool Noodles Foam Tube, Super Thick Swim Pool Foam Noodles, Bright Colorful Swimming Pool Foam Stick, Swimming Pool Accessories for Kids Adults : Toys & Games (amazon.com)

Commented [2]: shorten these names they seem excessive

Commented [3]: shorten these names they seem excessive

Commented [4]: shorten these names they seem excessive

Commented [5]: shorten these names they seem excessive

9.3 - Facilities and Equipment

Commented [6]: need to add to it. if you guys know of any more equipment and stuff pls add

This section details the environment in which the project will be developed, as well as some of the equipment. We are in the process of reaching out to professors to get a possible sponsorship or allowance to use their laboratory space. However, until we have concrete approval we cannot count on this type of assistance.

Therefore, we plan to use UCF's Texas Innovation Lab as the main facility for mechanical development. There we can use the equipment that is available to all students. We can build our PCBs there and do most of the cutting and soldering work there as well. For work that does not require mechanical design such as documentation and software development, we will work out of the Engineering Atrium or an office in the L3Harris Engineering Center. It is also noteworthy to mention that since our project continues in fall and we have the span of summer in between, we might all do some individual work at our respective preferred places. We also plan to have alternative sites where we can work by ourselves since the TI Lab is usually full of other teams that also need to use the equipment provided by the school.

In terms of equipment, we will have to rely on some of the school's laboratory multimeters for accurate measurement of the circuitry. The TI lab also is a great way to use free supplies that are necessary for soldering, hot gluing materials, and cutting materials as well. Since we are working with textile equipment, we are likely to need a sewing station which one of the team members can provide for us. On top of that, one of the biggest pieces of equipment that we will need is a 3D printer for some of the mechanical pieces including the cap design. The 3D printer will also be provided by a team member so we do not have to worry about finding one on the school campus.

Also, there is a building on campus known as the educational complex where they provide students with tools for laser-cutting and 3D printing. We plan on using that space very often because it is quite straightforward to implement and does not require a lot of technical skills to print out our parts. One of our team members has worked with the staff before which gives us more reliability as to how we intend on using that space effectively for our project.

9.4 - Mechanical Test Environment

To test the mechanical components several pieces of equipment and machinery will be needed. There are multiple 3D printed parts that need to prototyped and for this we will use Juan's 3D printer as well as mostly PLA filament. For the laser cutting of the acrylic we will use the laser cutter provided to us at the UCF TI Innovations Lab. However, this laser cutter can sometimes be unreliable. In the case that the laser cutter at the UCF TI Innovations lab is out of order or unavailable, the team will use the laser cutting services provided at the MakerFX Makerspace in Orlando.

The team will also need a space to test different pneumatic components. Pneumatic testing can often be loud and destructive. Because of this the team will need a nice open outdoor space near an electrical outlet. The picknick desk next to Lake Claire on the UCF campus are a great place for this. This area is often unoccupied and has plenty of outlets. There is plenty of space for the team to carry out pneumatic testing here.

Aside from the aforementioned facilities the team will need screwdrivers, cutting utensils, hammers, a sewing machine, and a heat sealer. Aside from the heat sealer, Juan has all these pieces of equipment plus a nice open desk to work on at his apartment. The team will need to purchase a heat sealer, but aside from that, all the construction of the robot base and body can be carried out at Juan's apartment.

9.5 - Mechanical Specific Testing

To test the pneumatic components of the mechanical system several steps need to be taken. In order to test the air compressor, solenoid valves, and pressure transducers a long line of poly tube will be connected to a pressure transducer and a solenoid valve. We will then calculate the volume of the poly tubing and set our pressure regulator to the rated air compressor PSI. From that we will be able to assume the rated CFM and thus we will know how long it will take for the poly tube to be completely filled with air at a certain PSI. We will then attempt to open and close the solenoid valve for that period of time (thus testing the solenoid valve). After the period of time is up we will check our pressure transducer readings to check the that the air compressor filled the tubing to the desired PSI in the desired amount of time (thus testing the CFM rating of the air compressor). We can also then fill the poly tubing til its bursting point to check if the pressure transducer reads the air pressure that the poly tubing is rated for (thus testing the pressure transducer).

We must also test that the series pouched motor steering chambers retract at the proper rate. To do this we must construct a series pouche motor chamber as in structured in section **8.2.1** and then check to see if it become 75% of its original length once full contracted. If it does not then a new chamber should be made.

The tip mount and retractor must also be tested. To do this we must calibrate teh robot to retract at the proper rate. We must check that the motors are able to retract the body without busting the poly tube or becoming engulfed in the robot or being ejected from the robot. Calibrations must be made until the robot retracts without problems. The tip mount must also be able to handle up to 5 pounds of free hanging weight in order for it to be considered secutly on the robot tip. While testing the retractor it would also be smart to test the base of the robot, most importantly the spool and retractor motor. Since the internal roller and base retractor motor should work in tandem in both expansion and retraction, it is important that they be tested at the same time.

9.6 - Electronics Test Environment

For the electronics portion of this project, we plan to see how the Raspberry Pi and other microcontrollers interact with each other through the serial peripheral interface. We plan to be testing the bot as a whole based on external conditions such as outdoor temperature or weather and relating it to how this can affect the

123

efficiency of the robot. The motor drivers for the actuators will be tested in similar settings where the PWM(Pulse Width Modulation) can be used to test how much voltage is being generated based on different types of duty cycles (i.e 25%, 75%, or 100%). Another factor we will be taken. Basic electronics test equipment such as multimeters, power supplies, function generators, oscilloscopes, along with breadboards and basic prototyping components will be needed for electronics testing.

9.7 - Electronics Specific Testing

We plan on testing the microcontroller, accelerometer, altitude sensor, phototransistors, darlington transistors, motors, motor driver, and motor encoder. To test these components we will prototype with an Arduino Mega, which contains the Atmega2560 microcontroller. The accelerometer and altitude sensor will be tested on the microcontroller's I2C channel and then we will try to send this information to a Raspberry Pi via USB-serial. To test the motors, we will design a prototype breadboard circuit with the L293 motor driver and connect it to the Arduino Mega, in order to test motor functionality, speed, PWM, and direction change. The phototransistors will also be connected to the analog inputs of the Arduino Mega and tested to see how reliable they are can be used with OpenCV when the data is sent back to the Raspberry Pi. Lastly, a basic prototype circuit with the TIP120 darlington transistor will be constructed and connected to a 12V load to see if the circuit can be switched on and off with a digital high and low.

9.8 - Power Test Environment

Our power circuit design is composed of the battery, disconnect switch, a 12V-5V buck converter, and an 11.1V - 12V buck-boost converter. The batteries will be tested in an enclosed environment while attached to the base of the worm. We tested our 2200mAh battery to ensure the right amount of voltage is produced using a voltmeter. We also plan on testing how much power can be delivered to the 3 motor controllers based on the power estimates we have made. The DC-DC converters will also be tested to ensure the input voltage is supplied to these components provides the desired output voltage. We will be measuring the voltages using a voltmeter. We will also use the battery charger to test how optimized the battery can be based on the specifications we need for the power consumption aspect of the robot. We also plan to incorporate a battery management system (BMS) that will help balance the load on each cell so as to avoid overcharging or short-circuiting. Fortunately, our battery comes with an installed BMS.

9.9 - Power Specific Testing

Testing in this section of the project will mainly be related to ensuring there is enough power going through the components of the robot. Since our robot will be

124

battery-powered, our main source of power will be the battery. Our battery specifications are 22Ah rated at 11.1V. To test the validity of this data, we will ensure the battery is fully charged. We will be using a multimeter to measure the rated voltage. After that, we will get a load with a known power consumption rate(i.e motor controller), so based on the time it takes for the battery to die, we will determine if the battery can provide the required amount of current within an hour. We also plan to apply this to all the electrical components, mainly the raspberry-pi because that is the main hub for all our electrical wiring.

9.10 - Software Test Environment

The software testing environment mainly consists of the interface that resides within the Raspberry Pi. Even though we also need to direct the hardware input to the PCB and then the R-Pi, for testing purposes we will first focus on making sure everything Computer Vision related is coded perfectly in the Python scripts. To do so, we will use the Python shell to run and test the unit tests. Some heavier computations involving image processing and videos can be run on a personal computer and run the tests on Visual Studio Code which also supports Python testing. VS Code also supports PyAutoGUI and integration testing, which makes it a viable candidate for overall development before taking it to the hardware.

9.10.1 - Unit Testing

Testing in this project is mainly related to the hardware integration and testing of the middleware connecting both hardware and software. However, it is good practice to test the software itself before it is taken to the hardware integration. We will be conducting unit tests which are ways to test specific methods, or units, in an isolated manner within the entire system. Unit test allows developers to test many scenarios that methods can be presented with. This way we can ensure that the entire system performs as expected. We will be working with the Python native framework PyUnit.

PyUnit is a framework that was inspired by the famous Java JUnit testing framework. This framework does not require any additional modules and since it comes with Python it will not cause any extra strain on the Raspberry Pi. It runs rapidly and generates reports upon failures of expected results. This is a feature that can be applied to both the software-specific code as well as middleware connecting the sensors and the MCU.

9.10.2 - Integration Testing

Integration testing is a bit more complicated than unit testing and more extensive as well. While unit tests check for a specific method or feature, integration tests make sure that all components in the system operate with each other. This means that after we see that everything works fine we gotta also confirm that the data and information are being relayed properly. What makes this complex is the fact that it

125

can navigate all the way from input to the camera to sending the navigation commands. It works like a cascade where a test passing or failing has a direct effect on the next test, and, thus, works like a cascade of tests. There is also a higher possibility of failing these tests and if we do not write the tests well enough, it is easy to get lost in error messages and not figure out what is failing or why.

9.11 - Software Specific Testing

The unit test that we will develop will be mostly local and will be run multiple times before we take the product into hardware integration. We will have unit tests for all of the classes and make sure that every method is accounted for. There will be different classes for the testing for each component of the software which we will detail below.

The manual testing, as opposed to the unit testing, will focus on having the software actually interact with the hardware and testing that we get all desired responses. This means making sure the pins are correctly mapped with the sensors and that data is being sent or received. We will Isolate each sensor to make sure that they work properly before combining the tests and then we will make sure that they can work together before we build the body of the robot.

9.11.1 - Image Processing

For image processing we are going to have unit tests for the methods that involve some kind of input or output and for those that perform a computation on the video frames. To furthermore test these methods, we will conduct training on a set of images and attest to their response accuracy. The second part of the image processing testing will require the presence of a camera, but not necessarily the one we are using for this project. This is so we can test the classes we need to create for object recognition and object tracking. We will have the software identify different objects just so we can attest that the algorithm is robust enough for different use cases. Finally, we will test the light blob identifying algorithm with a camera in real-time and a light bulb.

9.11.2 - Graphical User Interface

The GUI can also be tested using the PyUnit unit tests. This framework contains many methods related to asserting Python applications. We can test if the window is correctly displayed and if it is the right size. We can also write unit tests for making sure that specific fields are being filled up which we will need to assert that the sensor data received is being displayed. Finally, we can also test the button actions to confirm that the signals are being sent, that is, that we are correctly setting the robot to either manual control or autonomous mode. Furthermore, we can also use the library PyAutoGUI to test the UI automatically. This package is available for all operating systems and it provides the ability to simulate mouse cursor movement and clicks as well as keyboard key presses. This will allow us to

run the UI test each time we finish an integration test without having to manually do so and thus save us time.

9.11.3 - Sensor Testing

The sensors are components that are slightly more difficult to test than the code functions using unit tests. To make sure that the middleware is correctly implemented we all need to closely and carefully work together on the connection between the PCBs, the Raspberry Pi, and the sensors. For this to be done effectively we have to go through several trials in order to perfect it. Luckily, even though the sensors will be connected to the PCB, we can still use the Raspberry Pi for live debugging and testing. There we can test in the console if we are receiving the correct values from the sensors.

10 Administrative Content

10.1 - Budget and Financing

For our budget, we did not have a confirmed total cost of the parts needed to complete the project. We also had to compare what type of sensors we should use to help the robot navigate through the obstacle course and it was down to an ultrasonic sensor. They are also generally not affected by light and they measure 3D structures. In addition, using a gripper in Table 1 shows a cost estimate of our project, along with a range of what our total cost is projected to be with the current components.

Components	Cost	Quantity
BOM for Robot Body (Table - 5.4.1)	$195.37	1
BOM for Robot Base (Table - 5.4.2)	$673.53	1
BOM for Pneumatics (Table - 5.4.3)	$219.95	1
BOM for Obstacle Course (Tabel - 9.201)	$68.16	1
BOM for Electronics	$706.77	1
Total Estimated Cost	**$1863.78**	

Table 10.101 - Budget

10.2 - Milestone Chart

In this section, we will be talking about our milestones for the project and how we split major and minor tasks amongst each other. Before we began our first phase of the project, we spent an ample amount of time researching various projects that can be used as a reference for our main project. We initially decided on an autonomous robot lawn mower with solar panels attached as an alternative power source. After some brainstorming, we decided to structure our project around soft robotics. In order for all our ideas to develop fully, we split our brainstorming sessions over a few weeks to properly formulate a project that can be used to serve people in the real world.

In the first phase of the project, we discussed the engineering requirements needed to make this project successful such as how to navigate the robot and how the obstacle should be structured. We left our design requirements as open-ended as possible to allow complete creative license and optimize our project design while also making it feasible. After a lot of brainstorming, we decided to make our

robot's grippers a stretch goal in order to make our robot more efficient and feasible.

In the next phase of our project, we discussed the hardware and software designs which meant we had to do more detailed research on the devices and components required to make this robot function. For example, we needed to figure out how the pneumatic design will help the robot move through the obstacle course and what type of application software we plan on using to program the autonomous robot such as robot vision, and what type of micro-controller unit we intend on using to properly program the exact amount of pressure needed for the robot to move at specific locations in the obstacle course. As we move further in the next phases of our project, we plan on testing our individual components to ensure they meet the proper electric load needed to sustain our project.

As we begin to develop our project, we update our report with current research and findings that helps us to maintain order and a well-detailed timeline and help us run things smoothly in terms of implementing the project over the course of the year. We properly organized our documentation in various folders so we can track how far we have gone in the project and if any mistakes were made that can be resolved quickly since we made detailed documentation of the different milestones of our project. We also have a bibliography and resource file that helps us find similar research to the project we are currently working on and how other people have worked with soft robotics and the challenges they faced so we are less likely to face those encounters.
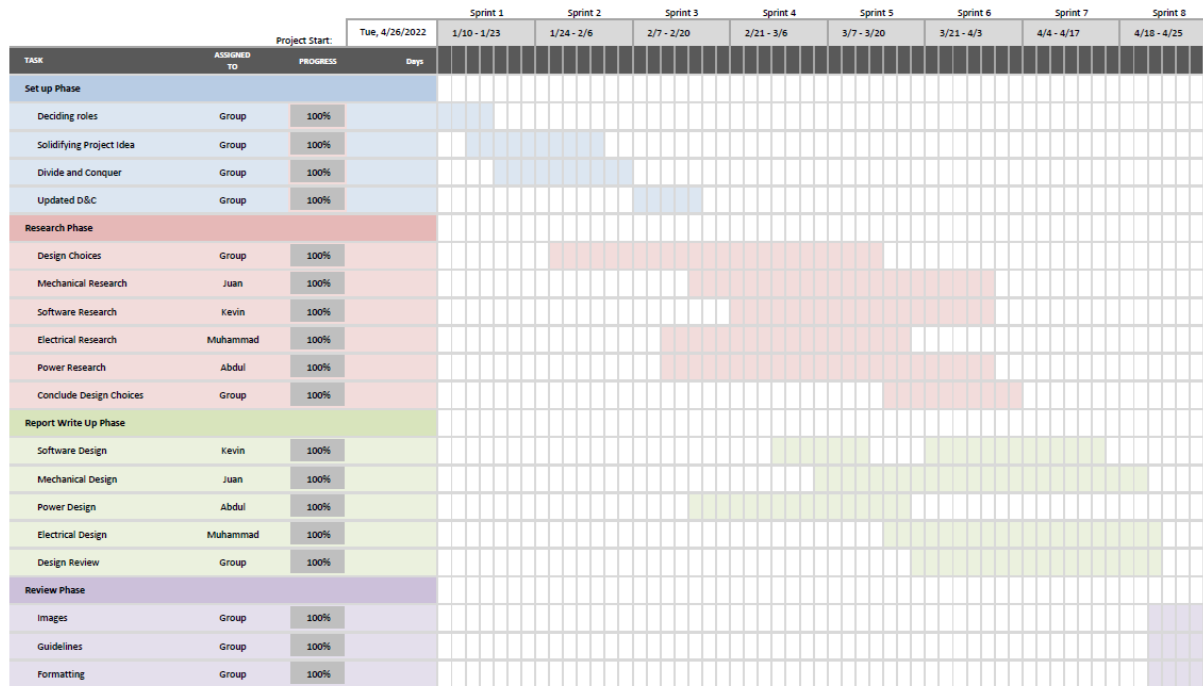
Once the design and brainstorming process was completed, we realized we needed to make some of our features stretch goals in order to make our project more realistic. For example, we agreed to make solar panels a stretch goal given the fact that we already have the lipo batteries as our main power source. We also removed lidar sensors and only focused on the pressure and accelerometer, which served us well in the long run.

Throughout the project timeline, we split up our tasks over several weeks in the project so all our members are aware of future deadlines that need to be met. We also implemented color coding which made our tasks clearer for us to achieve tasks that need to be done such as completing our page draft documentation. Each group member is assigned bi-weekly tasks to ensure our milestones are achieved before the end of the semester. In **Figure 10.201** we can see the timeline we followed for the design description and completing all the documentation needed. We stuck to the two-week sprints and only depicted some of the overarching goals for each of the phases. However, we do have a more detailed version where we make sure to account for every little bit of work we do so that everyone is held responsible. For Senior Design II (**Figure 10.202**) we only presume how things will proceed. However, having clear goals in mind helps narrow down those smaller tasks that we each need to complete. One thing that will be constant throughout the second semester is the constant updating of the documentation as we test the design. We all make sure to check these charts often and make any corrections to

129

it as needed, but doing so indicates that progress is slowing down in some areas. So the boards also serve as a judge of difficulty for tasks that allow us to know when a teammate needs help in an area.
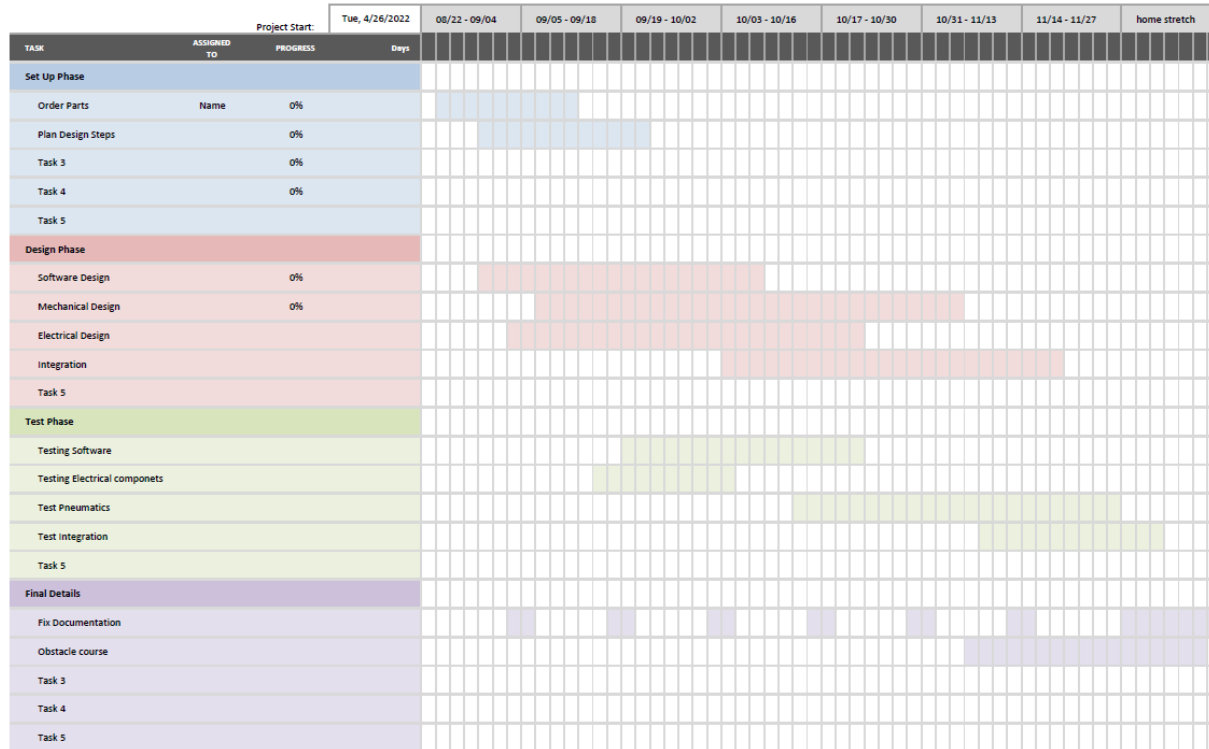
PROJECT TITLE

Vine Robot Project

TASK	ASSIGNED TO	PROGRESS	Days	Project Start: Tue, 4/26/2022	Sprint 1 1/10 - 1/23	Sprint 2 1/24 - 2/6	Sprint 3 2/7 - 2/20	Sprint 4 2/21 - 3/6	Sprint 5 3/7 - 3/20	Sprint 6 3/21 - 4/3	Sprint 7 4/4 - 4/17	Sprint 8 4/18 - 4/25
Set up Phase												
Deciding roles	Group	100%										
Solidifying Project Idea	Group	100%										
Divide and Conquer	Group	100%										
Updated D&C	Group	100%										
Research Phase												
Design Choices	Group	100%										
Mechanical Research	Juan	100%										
Software Research	Kevin	100%										
Electrical Research	Muhammad	100%										
Power Research	Abdul	100%										
Conclude Design Choices	Group	100%										
Report Write Up Phase												
Software Design	Kevin	100%										
Mechanical Design	Juan	100%										
Power Design	Abdul	100%										
Electrical Design	Muhammad	100%										
Design Review	Group	100%										
Review Phase												
Images	Group	100%										
Guidelines	Group	100%										
Formatting	Group	100%										

Figure 10.201 - Senior Design I Milestone Chart

131

PROJECT TITLE

Vine Robot Project
Note: Projected Milestones and Goals

TASK	ASSIGNED TO	PROGRESS	Days	08/22 - 09/04	09/05 - 09/18	09/19 - 10/02	10/03 - 10/16	10/17 - 10/30	10/31 - 11/13	11/14 - 11/27	home stretch
Set Up Phase											
Order Parts	Name	0%									
Plan Design Steps		0%									
Task 3		0%									
Task 4		0%									
Task 5											
Design Phase											
Software Design		0%									
Mechanical Design		0%									
Electrical Design											
Integration											
Task 5											
Test Phase											
Testing Software											
Testing Electrical componets											
Test Pneumatics											
Test Integration											
Task 5											
Final Details											
Fix Documentation											
Obstacle course											
Task 3											
Task 4											
Task 5											

Project Start: Tue, 4/26/2022

Figure 10.202 - Senior Design II Milestone Chart

These figures are an overview of the milestones that we have as a team. However, we keep track of our individual milestones in a document containing a Kanban Board with more detailed tasks and a timeline that requires progress information for every day. This board will also be actively updated as we progress through our Senior Design II class. We will make adjustments and plan ahead in order to have a more detailed milestone. It is also worthy to note that some of the tests will be done on dates prior to the ones mentioned. These dates only reflect the work that will be done on the integrated design with all components on it.

In order to effectively communicate between ourselves and keep each other accountable for our tasks, we used this milestone chart as a template and used discord very often to communicate and schedule our weekly meetings on Tuesdays and Sundays. We have been able to successfully use google drive as our main hub of documentation. We have folders addressed to the main sections of our project such as meeting notes, prototypes, reports, research, and software.

10.3 - Team Organization

For us to meet these milestones we have to have a certain method of organizing the team, the information we find, as well as sharing our progress. This organization was discussed early in the semester so we can proceed in a timely manner. We use a mix of software methodologies such as Agile and Scrum meetings which have proven effective and surely it will as well during Senior Design II. We also have delegated tasks according to our roles and using a Kanban Board we have nailed down the tasks that we each need to accomplish to ensure success.

10.3.1 - Project Roles

During Senior Design I, we have divided our roles based on our strengths and weaknesses. We wanted to have each member work on something they are good at while also interacting and learning about other areas. The software component is really the only component that is the most isolated from the rest of the project but even so, the software is actively interacting with the MCUs to receive and send data and commands. The project makes sure that all components are integrated into a single unit, and thus, gives us all an opportunity to work on different areas. Regardless, we still planned and came up with roles that were fitting for each. Therefore we spent the time in Senior Design I researching and writing in different sections that corresponded to our roles. We still met up frequently to make sure that the information we were adding was cohesive. We divided the main components into software, mechanical, electrical, and power designs, and in the table below (**Table 10.3.101**) we can see how the roles were distributed.

Name	Primary role	Secondary roles
Juan	Mechanical design of the robot. Designing the pneumatic system of the worm	Recommending power converters to use for the electrical components.
Muhammad	Responsible for designing the electronics schematics and block diagrams	Creating PCB layouts and making wiring connections to the raspberry pi
Abdul	Responsible for designing the power supply and choosing the battery specifications and researching types of power converters to use.	Budget and financing
Kevin	Responsible for the software design and obstacle course design	Microcontroller and Sensor design

Table 10.3.101 - Project Roles

10.3.2 - Team Communication

The most important aspect of any group contribution to a project is always to have open means of communication. Being full-time students who also work and have other responsibilities, we all have to keep in contact with updates regarding our responsibilities. During Senior Design I, we were not all present on campus at the same time and it was hard to coincide but having various means of communication helped us stay in touch. We use Discord mainly since it is a platform that allows for group chat with several channels to divide the subjects of conversation. It also allows for voice calls, video calls, and screen sharing. We also make sure to have each other's information in case we ever need rapid contact. In Senior Design II we will continue to use this app but we will also be meeting in person more regularly. We hold meetings every Tuesday and Thursday and sometimes on Sundays as well, where we hold our scrum meetings. Since our group will be off for the summer, staying in contact will be huge for the project to progress steadily and we all come back ready.

In terms of sharing information, we have Google Drive where we hold all of our project documentation. This includes our reports, brainstorming sessions, bibliography, and any design diagrams that we deem useful. This way we can all access all available resources, and data and be able to rate our performances and progress. We are also able to leave comments on specific parts of the documents which have proved to be super helpful. Using the agile methodology we are able to see what progress has been made in every sprint which we have designated to be a 2-week period. To share our source code, we will use GitHub which is open source and will allow us to all have access.

11 Project Summary and Conclusions

11.1 - Consultants, Sponsors, Subcontractors, and Suppliers

At the beginning of the project timeline, we realized how costly this project was going to be based on the previous research we have done on soft robotics. Our initial estimate amounted to about $1200 which meant we needed to start looking for potential sponsors and suppliers. Our initial plan was to reach out to faculty advisors with a strong background in robotics and research. This will help us not only receive adequate funding, but also provide us with the necessary knowledge we need to understand the scope of this project. A potential sponsor/advisor was a professor who is partaking in control system research which plays a huge role in how we want to successfully complete this project as a team. In addition, we hope to reach out to companies who have an interest in the development of our project since we are one of the very few teams whom have undertaken this type of project.

11.2 - Concluding Remarks

To conclude, this project has been a great challenge for us as a team. We all had to develop a growth mindset by trying to learn previous research and relevant technology based on our robot worm. Our interest in soft robotics gave us the motivation to build a soft robot worm which is still currently cutting–edge technology that has not been applied for a long time in the industry. Our biggest challenge was trying to make this project feasible and realistic. We initially wanted to have multiple features such as incorporating solar panels as an alternative power source to batteries so we can promote reducing pollution in the environment and also give our robot the ability to work both day and night even when the battery has been depleted. We also planned on making the robot autonomous, then we realized how software-heavy the task will be because we only had one computer engineer on our team. In addition, our budget was quite expensive which meant we had to reach out to sponsors and faculty advisors to help provide funds for our project.

This project also gave us the opportunity to apply most of the engineering concepts we have learned during our undergraduate career as electrical engineering students. From designing our electrical schematic diagrams to developing our own printed circuit board layout using Autodesk Eagle. We were also able to touch on the mechanical aspects of engineering such as designing the internal and external surfaces of the robot and how we want it to function using motors, relays, and solenoid valves to control the amount of pressure needed to regulate the movement of the robot throughout the obstacle course. Not only did we grow in our technical prowess, we were also provided the opportunity to build our soft skills which will serve us well in our professional careers. We had to decide our main

mode of communication which was discord. Before we began senior design, we had brainstorming meetings where we created a server and channels under the server for different purposes(i.e general chat, tasks, and resources). We also had a voice channel where we held our weekly meetings on Tuesdays and Thursdays at 9 pm. Our first order of business was to talk about our current updates which kept us accountable for each task we were assigned to. Secondly, we drew up a project timeline to ensure we were all on track to finishing the project documentation by the end of the semester. Finally, we color-coded our overall block diagram to delegate the main responsibilities of each section of the project to each team member so we can be held accountable for each major part of the development of the project.

12 Appendices

12.1 - Bibliography of Related Work

[1] Science Journals — AAAS (escholarship.org)

[2] DF Robot LIDAR Sensors - Getting Started with LIDAR | DroneBot Workshop

[3] Raspberry Pi 4 LIDAR Data Visualization | Python, Plotly, Websockets, D3.js - YouTube

[4] https://softroboticstoolkit.com/book/control-board

[5] https://softroboticstoolkit.com/book/controlboard-using-board

[6] https://softroboticstoolkit.com/book/controlboard-advanced-control (edited)

[7] https://www.vinerobots.org/

[8] https://www.vinerobots.org/build-one/the-simple-vine-robot/

[9] https://www.vinerobots.org/build-one/pre-formed-vine-robot/

[10] https://www.vinerobots.org/publications/

[11] https://www.hawkeslab.com/projects

[12] http://charm.stanford.edu/Main/Resources

[13] https://youtu.be/qevIIQHrJZg

[14] 2021-controlling-combined.pdf (gatech.edu)

[15] https://ieeexplore.ieee.org/document/8977330

[16] https://www.youtube.com/watch?v=GgJt6vIbiso

[17] https://www.softroboticsinc.com

[18] https://pcbtrace.com/landing/flex-pcb-assembly-and-manufacturing.php?gclid=Cj0KCQiA0eOPBhCGARIsAFIwTs63bNWXNZvEvf17UhAwG4lbFcAjasbFNRfPcLk8Li8JHqPXg4ssQ-caAkWJEALw_wcB

[19] (1) Tracking Objects | OpenCV Python Tutorials for Beginners 2020 - YouTube

[20] A Soft, Steerable Continuum Robot That Grows via Tip Extension. (escholarship.org)

[21] Getting Started with OpenCV | LearnOpenCV

[22] M. M. Coad et al., "Vine Robots: Design, Teleoperation, and Deployment for Navigation and Exploration," in IEEE Robotics & Automation Magazine, vol. 27, no. 3, pp. 120-132, Sept. 2020, doi: 10.1109/MRA.2019.2947538

[23] M. Selvaggio, L. A. Ramirez, N. D. Naclerio, B. Siciliano and E. W. Hawkes, "An obstacle-interaction planning method for navigation of actuated vine robots," 2020 IEEE International Conference on Robotics and Automation (ICRA), 2020, pp. 3227-3233, doi: 10.1109/ICRA40945.2020.9196587.

[24] 1912.08297.pdf (arxiv.org)

[25](1) AIM2020 Workshop - Bio-inspired Vine Robots and A Promising New Application - Elliot W. Hawkes - YouTube

[26] Self-Driving Car using Raspberry Pi - Computer Vision Zone

[27] Object Tracking with OpenCV (livecodestream.dev)

[28] Getting Started With Testing in Python – Real Python

[29] Raspberry Pi - MPL3115A2 Precision Altimeter Sensor Python Tutorial : 4 Steps - Instructables

[30] (1) Sending Data via I2C (RaspberryPi to Arduino) How to Wire & Code - YouTube

[31] Cheat Sheet — PyAutoGUI documentation

[32] DIY Arduino RC Transmitter - How To Mechatronics

[33] https://softroboticstoolkit.com/stretchsense

[34] http://www.wseas.us/e-library/conferences/2008/algarve/EEESD/036-588-384.pdf

[35] https://www.doitpoms.ac.uk/tlplib/batteries/battery_characteristics.php

[36] https://www.aurorasolar.com/blog/solar-panel-types-guide/

[37] https://geothermhvac.com/mono-vs-poly-better/

[38] https://ases.org/thin-film-solar-panels/

[39] https://www.analog.com/en/technical-articles/energy-harvesting-with-low-power-solar-panels.html

[30] https://softroboticstoolkit.com/low-cost-ep-circuit/background

[31]Design, Modeling, Control, and Application of Everting Vine Robots - PMC (nih.gov)

[32] https://arxiv.org/pdf/2003.09113.pdf

[33] https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8917931

[34] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7805729/pdf/frobt-07-548266.pdf

[35] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7805729/pdf/frobt-07-548266.pdf

[36] https://arxiv.org/pdf/1910.11863.pdf

[37] https://link.springer.com/chapter/10.1007/978-3-319-63537-8_45#citeas

[38] https://arxiv.org/pdf/1912.08297.pdf

[39] https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7989648

[40]
https://www.hawkeslab.com/_files/ugd/00c802_ebe50f767a7a45aabe97978d93d72a34.pdf

[41] https://onlinelibrary.wiley.com/doi/epdf/10.1002/advs.201800541

i.

12.2 - Copyright Permissions

Vine robot

Source: https://www.vinerobots.org/

PERMISSION: GRANTED

Using picture of worm robot for final research paper

Abdul-Malik Mustapha
Tue 4/5/2022 7:59 PM
To: aokamura@stanford.edu

Good evening professor Okamura,
My name is Abdul-Malik Mustapha and I am currently a senior majoring in Electrical Engineering at the University of Central Florida. My team and I are planning to build a soft vine robot for our senior design and we were inspired by your research on soft robotics at Stanford. I would like to ask if we could use some of your pictures taken as figures for our research paper?

Re: Using picture of worm robot for final research paper

Some content in this message has been blocked because the sender isn't in your Safe senders list.
I trust content from aokamura@stanford.edu. | Show blocked content

Allison Mariko Okamura <aokamura@stanford.edu>
Tue 4/5/2022 9:34 PM
To: Abdul-Malik Mustapha

Dear Abdul-Malik,

Thanks for your email. Yes, you are welcome to use our images, please just attribute to the relevant paper or website. If you have not seen it already, you may find this website useful: https://www.vinerobots.org.

Best,
Allison

Tip Mount Design

<u>Source:</u> S. Jeong*, M. M. Coad*, L. H. Blumenschein, M. Luo, U. Mehmood, J. Kim, A. M. Okamura, and J. Ryu, "A Tip Mount for Transporting Sensors and Tools using Soft Growing Robots," IEEE/RSJ International Conference on Intelligent Robots and Systems, 2020, pp. 8781-8788. *Equal contribution

<u>PERMISSION:</u> GRANTED

MC Margaret Coad <mcoad@nd.edu>
Wed 4/20/2022 12:20 AM
To: Juan Battaglia
Cc: Sang-Goo Jeong <jsg1215z@koreatech.ac.kr>

Hi Juan,

Thank you for your email. I'm glad you are interested in building a vine robot. You are welcome to use our tip mount design.

Sang-Goo, would you be able to share the CAD files for the design?

You can cite the work using the following citation: S. Jeong*, M. M. Coad*, L. H. Blumenschein, M. Luo, U. Mehmood, J. Kim, A. M. Okamura, and J. Ryu, "A Tip Mount for Transporting Sensors and Tools using Soft Growing Robots," *IEEE/RSJ International Conference on Intelligent Robots and Systems,* 2020, pp. 8781-8788. *Equal contribution

Also, you may be interested in our other tip mount design, presented in our other paper: W. E. Heap, N. D. Naclerio, M. M. Coad, S. Jeong, and E. W. Hawkes, "Soft Retraction Device and Internal Camera Mount for Everting Vine Robots," *IEEE International Conference on Intelligent Robots and Systems,* 2021, pp. 4959-4965.

This second design can allow a vine robot to carry a camera without having the cap slide along the environment. We did not test its ability to carry objects, but we expect that it will do well at that too.

Best,
Margaret

—
Margaret M. Coad, Ph.D.
Assistant Professor
Aerospace and Mechanical Engineering
University of Notre Dame

300A Cushing Hall of Engineering
Notre Dame, Indiana 46556
(574) 631-5688
https://ame.nd.edu/faculty/margaret-coad/

Battery

3.2 You agree to use the Site or Services solely for your own private and internal purposes. You agree that (a) you will not copy, reproduce, download, re-publish, sell, distribute or resell any Services or any information, text, images, graphics, video clips, sound, directories, files, databases or listings, etc available on or through the Site (the “Site Content”), and (b) you will not copy, reproduce, download, compile or otherwise use any Site Content for the purposes of operating a business that competes with Banggood.com, or otherwise commercially exploiting the Site Content.

Battery Disconnect Switch

Terms of Use

Site Access

By accessing this website, you agree to be bound by the terms and conditions of use appearing below, such use being non-commercial in nature and conditioned on your acceptance. The materials on this website are Copyright © 2020 Autosales, Incorporated. All rights reserved. You are hereby authorized to view, copy, print, and distribute these materials subject to the following conditions:

1. The materials may be used for internal informational purposes only;
2. Any copy of these materials or any portion thereof must include the above copyright notice; and
3. Autosales, Incorporated may revoke or modify any of the foregoing rights at any time.

You also expressly agree that you will not use any robot, spider, or other automatic or manual device or process to interfere or attempt to interfere with the proper working of our website, nor act as a conduit for others to affect the same result.

Battery charger

Source: https://usa.banggood.com/HOTA-D6-Pro-AC-200W-DC-650W-15A+2-Dual-Channel-Lipo-Charger-With-Wireless-Charging-for-NiZn-or-Nicd-or-NiMH-Battery-p-1480536.html?utm_source=googleshopping&utm_medium=cpc_organic&gmcCountry=US&utm_content=minha&utm_campaign=minha-usg-pc&currency=USD&cur_warehouse=CN&createTmp=1&ID=477586175626&utm_source=googleshopping&utm_medium=cpc_us&utm_content=nolan&utm_campaign=aceng-pla-usa-all3-20220402&ad_id=513001116163&gclid=Cj0KCQjw3v6SBhCsARIsACyrRAliJuRHnZ8rPrdk_m3ljuoKgIYWDMnldFfG_-2ZV1kWDIX4pQJhFwIaArCOEALw_wcB

PERMISSION: GRANTED

12.3 - Datasheets

https://www.ti.com/document-viewer/TPS565208/datasheet

https://www.ti.com/document-viewer/LM25118-Q1/datasheet

https://www.ti.com/lit/gpn/tps55288

ADP160/ADP161/ADP162/ADP163 (Rev. H)
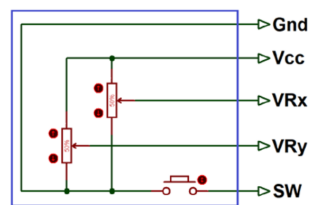
12.4 - Pin Layouts

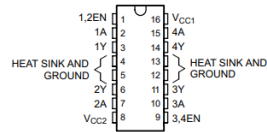**Figure 12.401**: Joystick Pin Layout
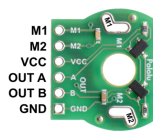
Figure 12.402: Motor Driver Pin Layout

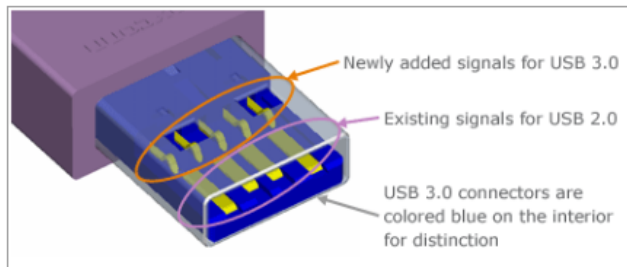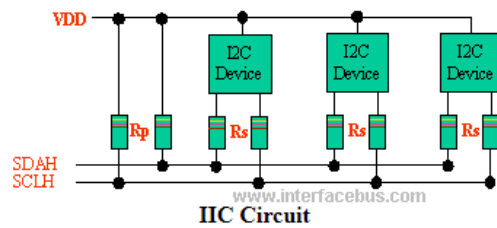**Figure 12.403**: Motor Encoder Pin Layout

Figure 12.404 - USB 3.0 Pin Layout

Figure 12.405 - I2C IIC Circuit

v.