Soft Vine Robot

Abdul-Malik Mustapha, Juan Battaglia, Kevin Abreu, Muhammad Gudaro

University of Central Florida, College of Computer and Electrical Engineering, Orlando, FL

Abstract— **'Soft' robotics can be defined as designing, controlling, and fabricating soft, flexible bodies, as opposed to 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 archaeology. The main goal of our project is to create a soft robot using flexible materials such as poly tubing that is programmed to navigate its way through an obstacle course of rough terrain, narrow spaces, and elevations.**

Keywords— Pneumatic controls, Artificial muscles, retraction, valves, autonomy, manual steering.

I. Introduction

Soft robotics is a growing area of research that deals with robotics designed with compliant materials, such as fabric, poly tubing, 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 archaeology. 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.

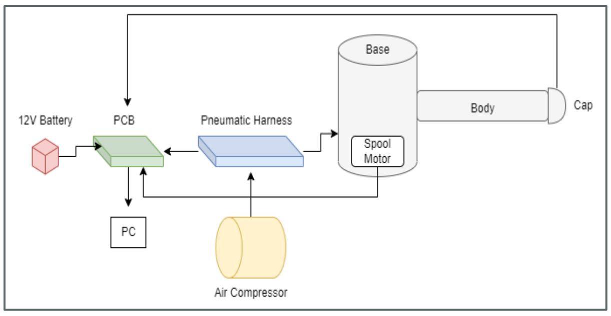


Fig 1: Design Overview

The goal of our project would be to design a soft vine robot, pictured above, using flexible materials that would be programmed to navigate the obstacle course. Using electronically controlled pneumatic valves, 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. We also plan to implement a graphical user interface where users can easily operate robot using a laptop. In addition, we will be using OpenCV, an open-source library for image processing platform. It will be used to identify and track any desired objects in the camera feed, which provides visual feedback on the vine robot's environment. To move the vine robot, we will be using three motor encoders and actuators to help guide the robot through the obstacle course. An air pump will be used to inflate the robot and use solenoid valves, ball valves, and relays to provide accurate control based on feedback to correct error signals or any degree of randomness when completing the obstacle course. In this paper, we would be discussing the design approach taken as a group as well as the integration of the subsystems involved in building the vine robot.

II. Mechanical Design

The mechanical design takes into account several components including an air compressor which provides the pneumatic pressure to the soft components, pneumatic controls (a system of solenoid valves, pressure regulator, poly tubing material, 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.

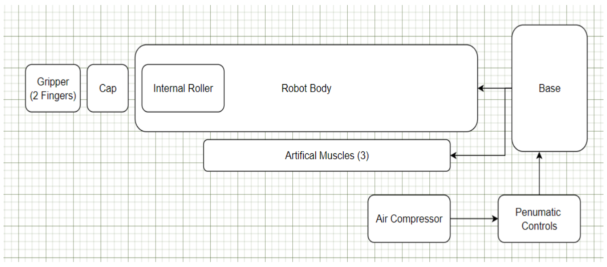


Fig 2: Mechanical Block Diagram

We will need a way to pressurize and store the air so that we can use it later. 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. 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.

A. Lengthening through Eversion

The moving parts of the vine robot will be made up of the main body tube and several auxiliary tubes (artificial muscles) 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. 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 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. 3**). 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. A benefit of eversion is that since 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.

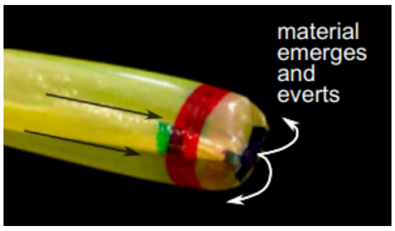


Fig 3: Polyethylene Tubing Eversion

B. *Reversible vs Non-Reversible Steering*

Vine robot steering can be divided up into two types: reversible steering and non-reversible steering. Reversible steering refers to the ability to change the steered direction of the robot without having to retract the robot. Non-reversible steering refers to steering mechanisms that do not have the ability to change the steered direction of the robot without first retracting the 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 non-reversible steering, there must be 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 3.**

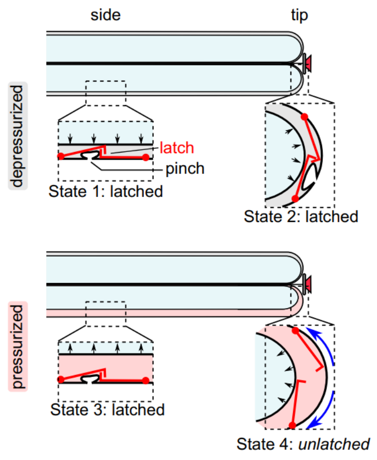


Fig 4: Demonstration of Latch Locking Mechanism

C. *Retraction Device*

The purpose of adding a retraction device on vine robots is to allow retraction without undesired bending or buckling of the robot body. These devices are 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. 4** 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.

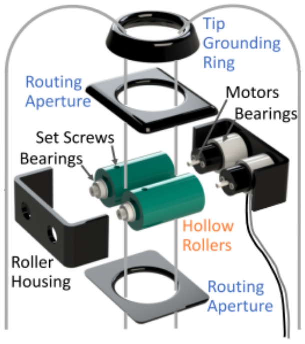


Fig 5: Internal Roller Assembly

D. *Pneumatic control and Gripper Design*

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. 4**. One of the most common materials used for vine robot pneumatic chambers is polyethylene, and **Fig.4**, 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.

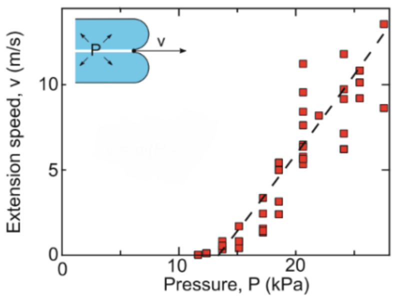


Fig 6: Robot lengthening vs air pressure

The gripper designed in **Fig. 5** 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.

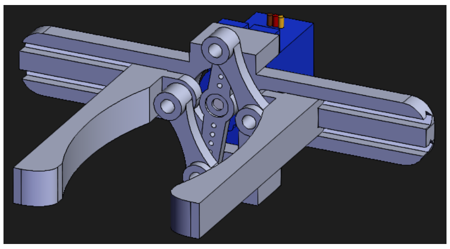


Fig 7: Gripper Assembly

III. ELECTRONICS DESIGN

The electronics design comprises the major electronic components that were used to build the soft vine robot, as seen in the diagram below. Our soft robot includes several pieces of hardware such as pneumatics, a temperature sensor, 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.

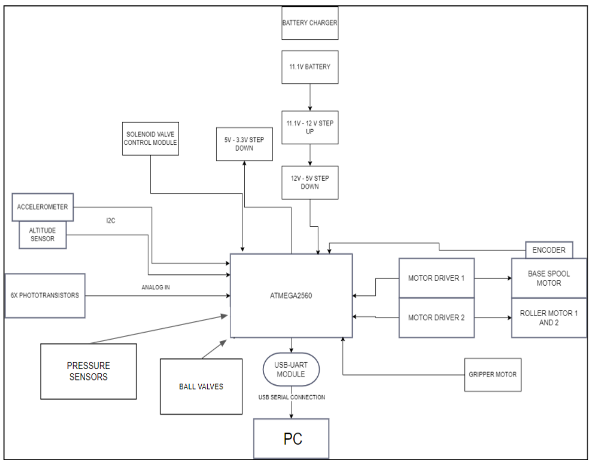


Fig 8: Hardware Block Diagram

A. *Microcontroller*

For this project, we selected an ATmega2560, as it has excellent processing power with multiple digital and analog pins. This microcontroller 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. **Fig. 6** 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 for power and UART, and USB serial header. The design also includes pins for ICSP programming. The PCB also includes headers for the digital and analog pins.

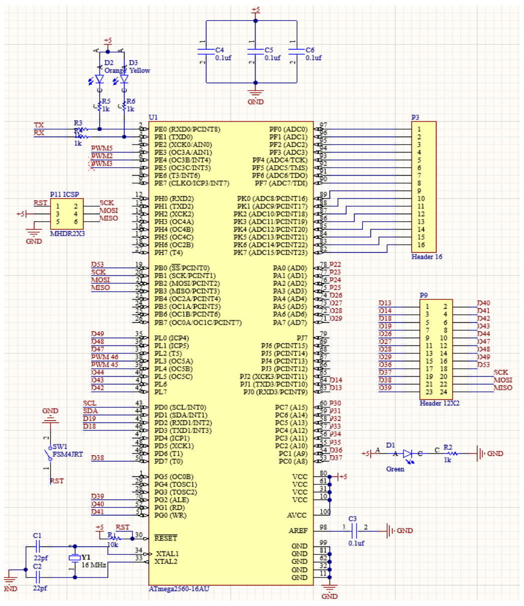


Fig 9: ATMega2560 Schematic

B. *Motors*

The vine robot will include a single 12V DC gearmotor 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. The robot will also include two roller motors at the cap of the robot, which will be used to pull the material back in at the tip to prevent buckling, as seen in the motor diagram below.

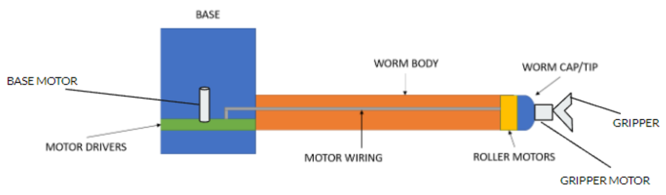


Fig 10: Motor Block Diagram

C. *Motor Drivers*

Furthermore, the motors will be driven by a motor driver that will be controlled by the GUI for growth and retraction. 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 figure below illustrates the microcontroller interaction with the Motors.

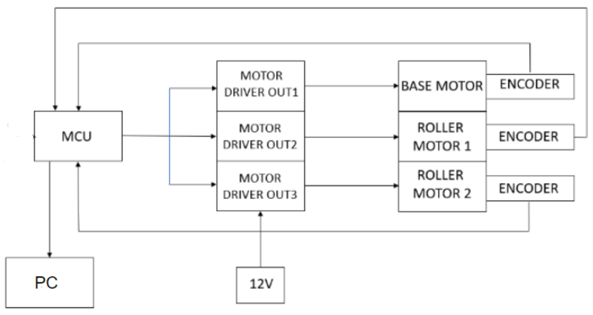


Fig 11: MCU-Motor Interaction

We selected an L293 motor driver, which 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. The spinning direction is controlled by changing the polarity of the input voltage, which can be accomplished by implementing an H-bridge circuit, which 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, it 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. The full motor circuitry for the robot can be seen in the figure below.

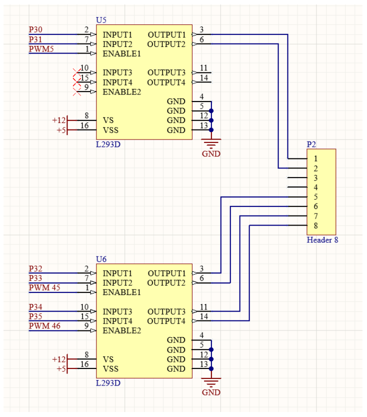


Fig 12: Motor Driver Design

D. Sensors

The vine robot includes a temperature sensor, pressure sensors, and an accelerometer. The purpose of the temperature sensor will be sent temperature data at the tip of the robot back to the user. The accelerometer will return the speed of the robot. The pressure sensors will be located at the pneumatic harness to send back pressure data in PSI to the user. This will allow the user to monitor PSI levels of the body and three steering tubes of the robot. The pressure sensors will be directly connected to the analog inputs of the Atmega2560 and powered with 5V. The temperature and accelerometer sensors will require 3.3V to 5V logical level conversion to read the I2C data. To accomplish this, a logic level conversion circuit was design using BSS138 N-channnel FETs, as seen in the schematic below.

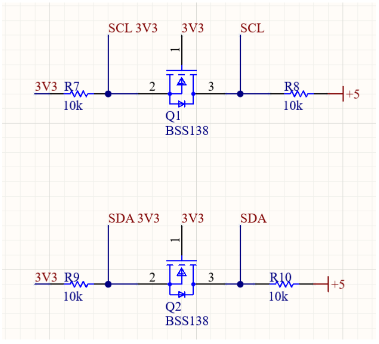


Fig 13: Logic-level Conversion

E. Phototransistors

The soft vine robot 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 vine robot will be a 161 phototransistor from 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 robot will include six phototransistors located around the cap of the robot. 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 figure below showcases the phototransistor alignment on the robot cap board.

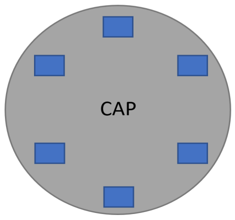


Fig 14: Robot Cap with Phototransistors depicted in blue

F. 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 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.

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 the semiconductor 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.

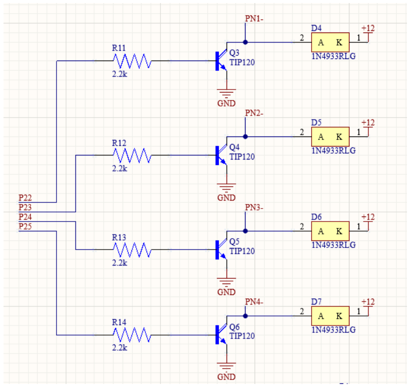


Fig 15: Solenoid Valve Control Circuit Schematic

To control the pressure coming from the air compressor, 4 motorized ball valves controlled by an 8-channel relay board will be used. The relay board will be connected to the 8 digital pins on the PCB. 2 relays control a single ball valve by switching the 12V between the open and close wires. Taking away 12V from both wires allows the ball valve to pause and control the flow of the air.

G. *Power Conversion*

Our electronic design also includes a power conversion circuit that can step down 12V to 5V to 3.3V, as seen in the schematic below. A PJ-002A power jack is used for the 12V power input. The 12V input is then stepped down to 5V with a R-78E5 switching regulator, which is then stepped down 3.3V using ADP160 linear regulator. The design includes bypass capacitors to reduce voltage noise. The 12V is used to power the pneumatics, motors, and ball valves. The 5V is used to power the MCU and pressure sensors, while the 3.3V is used to power the temperature and accelerometer sensors.

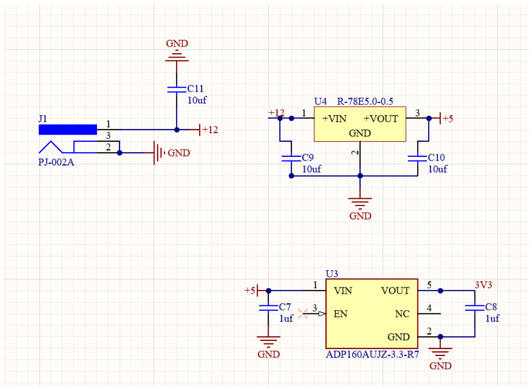


Fig 16: Power Conversion Design

H. *Microcontroller Firmware*

The Atmega2560 firmware, developed on the Arduino IDE, controls the motor direction, speed, and receives encoder data. The code also control the ball valves with active low switching and controls the pneumatics with active high switching. 10 analog pins are used as inputs to read phototransistor and pressure sensor data. The temperature and accelerometer data are also read over I2C communication. The robot components can be controlled by writing bytes over serial. The firmware expects certain bytes that are used to control the components. The firmware also returns the sensor values over USB serial at a 115200 baud rate. The GUI developed in Python is used to read and write data to the MCU, which allows for robot control.

IV: SOFTWARE DESIGN

A. *Overview*

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. The robot has its own User Interface (UI) developed in PyQT6 that allows the software to interface with the PCB via USB serial. The GUI allows COM port selection along with the ability of having the user choose which mode the robot will be set to. When in autonomous mode, computer vision and image processing will be used to interpret the environment the robot will be navigating in order to identify and track a light bulb. We will be using a USB camera connected back to the PC 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 robot's path. All the sensors will be connected to a PC that will act as a mediator for the pneumatics, ball valves, motors, and sensor data. The software GUI also includes a debug window, which allows each individual pneumatic, ball valve, and motor to be tested on its own. This is the main method of testing that will be used before integrating all the components. It allows us to test both the component and PCB, as well as their relation with the GUI.

B. *Manual vs. Autonomous*

The vine robot includes 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, we can see how the components interact in each different mode.

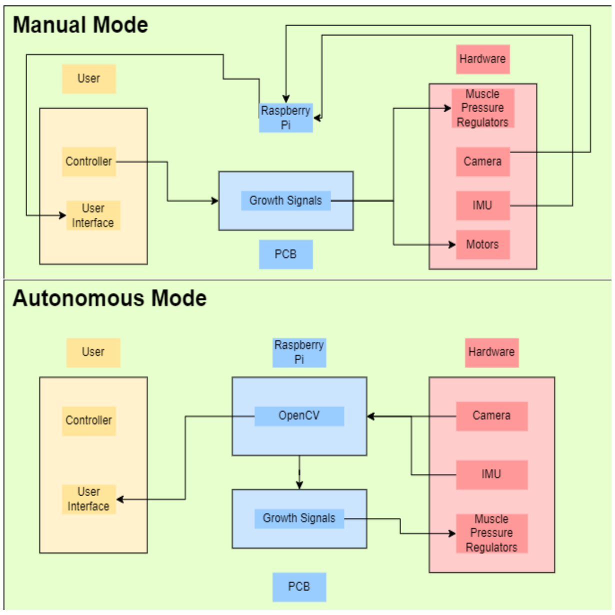


Fig 17: Manual vs. Autonomous Mode

C. *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 robot to map out its surroundings and give it a better understanding of its environment. The image processing is done using the OpenCV library which will allow us to identify a parametric object. Furthermore, 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. The figure below depicts the computer vision software diagram.

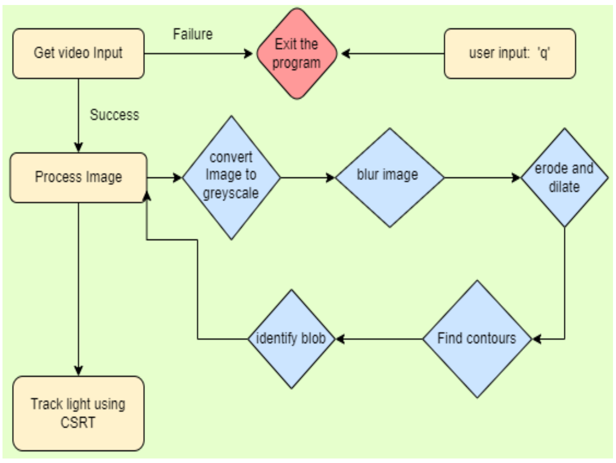
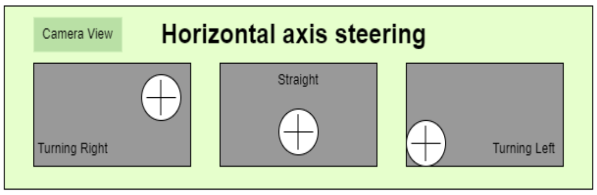


Fig 18: Tracking Algorithm Diagram

The vision software sets up the object tracker and then starts capturing the video input. Then, 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 has no problems reporting failures as well.

Furthermore, the Image processing module also interacts with the hardware. The OpenCV software maps out the video frames and calculates the angles in which the vine robot is 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. The figure below showcases how the steering works when tracking a light bulb.



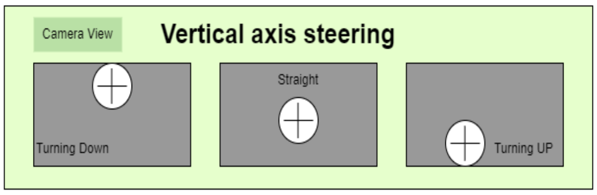


Fig 19: Steering Diagram

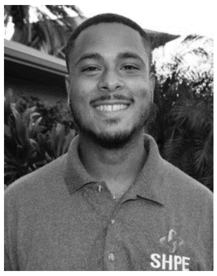
V: CONCLUSION

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. Our interest in soft robotics gave us the motivation to build a soft vine robot which is still currently cutting–edge technology that is slowly finding its place in industry applications. 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 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.

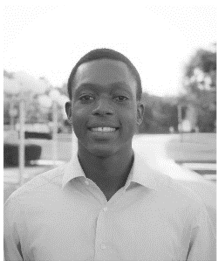
THE ENGINEERS

Juan Battaglia is an Electrical Engineering student who is passionate about control systems, robotics, and software development. He has completed internships at NASA, Lockheed Martin, and Renesas Electronics, and has also conducted undergraduate research at Siemens Digital Grid Lab. He has also served on the E-board of both IEEE and IEEE PES student chapters as the Vice President and founded IEEE UCF's Knight's Open Circuit Podcast.

Muhammad Gudaro is an Electrical Engineering student who is passionate about electronics design and firmware development. He has completed internships at Tesla, Lockheed Martin, and Collins Aerospace, and has also conducted undergraduate research at Florida Solar Energy Center. He has also served two consecutive years on the E-board of the IEEE student chapter as the Marketing Director.



Kevin Abreu-Aguila is a Computer Engineering student who is passionate about software development. He has completed internships at Apple, Lockheed Martin, and Autodesk, and has also conducted undergraduate research within UCF EXCEL. He has also served two consecutive years on the E-board of SHPE student chapter as the Event Coordinator.



Abdul-Malik Mustapha is an Electrical Engineering student who is passionate about power and renewable energy. He has completed an internship at Siemens and has also worked as an undergraduate teaching assistant within the college of ECE. He has also served as the Academic Excellence Chair and Professional Development Chair at both the NSBE and IEEE student chapters.

REFERENCES

[1] 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

[2] 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

[3] Joseph D. Greer, Tania K. Morimoto, Allison M. Okamura, and Elliot W. Hawkes "Series Pneumatic Artificial Muscles (sPAMs) and Application to a Soft Continuum Robot," 2017 IEEE International Conference on Robotics and Automation (ICRA) Singapore, May 29 - June 3, 2017

[4] Margaret M. Coad, Laura H. Blumenschein, Sadie Cutler, Javier A. Reyna Zepeda, Nicholas D. Naclerio, Haitham El-Hussieny, Usman Mehmood, Jee-Hwan Ryu, Elliot W. Hawkes, and Allison M. Okamura, "Design, Teleoperation, and Deployment for Navigation and Exploration," Digital Object Identifier 10.1109/MRA.2019.2947538 Date of current version: 28 November 2019

[5] Blumenschein, L.H., Okamura, A.M., Hawkes, E.W. (2017). Modelling of Bioinspired Apical Extension in a Soft Robot. In: Mangan, M., Cutkosky, M., Mura, A., Verschure, P., Prescott, T., Lepora, N. (eds) Biomimetic and Biohybrid Systems. Living Machines 2017. Lecture Notes in Computer Science(), vol 10384. Springer, Cham. https://doi.org/10.1007/978-3-319-63537-8_45