UCF Senior Design

ArticuLight

Department of Electrical Engineering and Computer Science

University of Central Florida

Dr. Lei Wei

<u>Group 9</u>

Benjamin Simms	Computer Engineering
Michael Taub	Electrical Engineering
Keng Chu	Electrical Engineering
Scott Bell	Electrical Engineering

Table of Contents

1. <u>Executive Summary</u>

The objective of the ArticuLight device is to implement a gesture controlled articulating light fixture through the use of computer vision. A camera connected to an embedded computer analyzes the gestures of the user and performs the relative action associated with the gesture. The gesture implementations are resetting the arm position, moving the light to a desired location through the articulating arm, and an activation/deactivation gesture for the arm to follow the hand.

ArticuLight's technical specifications are as follows: the system is powered by 120V AC at 60Hz. This powers everything in the system including the embedded computer and light bulb. The arm articulates with a lighting accuracy of less than or equal to one cm in distance from the gestured location that is instructed to the system. The system contains a software to hardware interface of less than three seconds per gesture and the computation time of issuing an instruction from a gesture is less than two seconds. The entire system weighs in at less than seven pounds and contains an arm that is twenty-four inches in length. Motors connected to a motor controller are the driving force that implements the articulation of the arm, and the arm articulates with up to 90 degrees of movement in both the yaw and pitch orientations. The physical footprint of the system is designed so that it can be oriented both horizontally and vertically.

NVIDIA's Jetson Nano architecture is the embedded computer responsible for computing the gesture recognition tasks. The Jetson Nano is useful for robotics in an environment that employs the use of machine learning and is therefore a good fit for this project. A motor controller for arm articulation connects to the pins present on the ATmega328 microcontroller and offers many more pins for expandable functionalities such as a light controller. A Raspberry Pi Camera 2 is connected to the Jetson Nano in order to capture images for gesture recognition.

The software that is implemented to perform computer visions and issuing the respective signals to perform actions such as activating the motors is written in Python. Python is a very popular scripting language for building computer vision algorithms and deep neural network models for image classification. Because the Jetson Nano contains a Python library for using its pins for signal generation, not much code outside of Python is necessary. The images from the camera are processed through OpenCV and captured through Jetcam. The image classification task is performed by an 18 layer deep convolutional neural network machine learning model supplied by NVIDIA. The outputs of this model are fed into a Support Vector Machine to classify the points on the hand as a gesture. These gestures then control the articulating arm by sending information to the ATmega328 microcontroller present on the developed PCB.

2. Design Constraints

When discussing realistic design constraints, the goal of this project is to minimize the possibility and probability of any negative implications. Delineating each area, we should be able to achieve this objective.

2.1 Social, Political, or Ethical Design Constraints

A social constraint of this project is the fact that it is mainly operable for one individual only. In the case that multiple people want to make use of the project (to light a car engine that requires multiple people to be working on it at the same time for example), the project may act erroneously with multiple hands in the frame. To prevent this, only one hand is parsed by the computing device at a time, and the camera should only have one hand in the frame at a time to take complete control of the system.

As this project is a lighting fixture meant to be used on a personal basis, the team who built this project did not identify any political or ethical constraints.

2.2 Environmental Constraints

From the aspect of an environmental constraint, we have opted to use materials that are robust enough to last for years given proper maintenance and care. Furthermore, plastic components selected are either recyclable or biodegradable to minimize environmental impact in the event of replacing parts and throwing them away. The metal components can also be readily recycled or scrapped. There are no current plans to use batteries, unless designed as a stretch goal, we will use rechargeable batteries to extend the usability of those as a power source which in turn will reduce the waste generated over the lifetime of the project. Furthermore, the only other minor environmental concern is the epoxy that was used to secure all of the segments of the arm wire guides in place. As the amount used is very small, less than 5 oz, there are no regulatory requirements for this project.

2.3 Economic Constraints

The only economic constraint that may develop from this project would be if it becomes a possible device we choose to sell. However, as it stands, there are no plans to make this a commercial enterprise but will be considered if the opportunity arises.

The manufacturability of this project does rely heavily on the ability to get the machine learning processing components which have been difficult. The limited availability has made these components either unavailable or priced four to five times higher than what they have historically been priced. As such, if this project were to be considered for manufacturing at a larger scale a better solution would need to be found for this aspect of the design.

The overall design components are commercially obtained and do not require specialized circuits or manufactured parts. With the exception of the 3D printed frame which will be made available so sustainability may require minor research to replace parts as needed but will not be otherwise unavailable.

2.4 Health and/or Safety Constraints

A minor health and/or safety concern is a viable constraint. There are moving components that require some maintenance, be it lubricant oil if made of metal. The current plan is to have everything enclosed to avoid potential injury from the motors, fan, and electrical components.

The motors selected for this project have been chosen so that the arm does not supply a force great enough to cause injury. Since the camera is at the end of the arm, as long as no body parts are present in the trajectory of the arm movements, the arm should not collide with the individual. However, since this is always a possibility, motors that do not supply a large force were selected.

During the build and testing of this system, some of the mounts have reasonably sharp edges and the components can become hot while in use. As such this adds some minor safety concerns if opening the case while in operation. To prevent components from becoming too hot that touching them will cause injury, we implemented fans on the case and the computing device, as well as used material to mount the servos that would act as a heatsink.

There are also safety considerations to consider when the system is articulating. Since the reset and move to functions have a very fast, instant movement with a metal and plastic arm that weighs just under 1 lb, this presents a minor bludgeoning hazard. The short, quick movements aren't enough to cause any real damage to a person, but can hit objects and the high torque of motors can have the arm push things off of a work surface. So a safe distance practice is needed when this system is in use. Smoothing the arm movement algorithms will alleviate most of this constraint when implemented in the future.

2.5 Mechanical Design Constraints

Mechanical design constraints that we came across while building the Articulight includes the case material originally chosen, wire selection, servo motor selection, heating issues with the motors, and system mounting position/distance.

2.5.1 Case Material

The case material that was originally chosen for our project was a ¼ inch tinted plexiglass. This was going to be assembled with 3D printed corner connectors that would allow for assembly without mechanical fasteners if we chose. During testing, it was observed that the servo motors flexed during actuation, skewing our results. To account for this, the servo mounts and stands were made more robust, however, our testing results still had a major degree of variation. After further investigation it was determined that the case was flexing. As a result, the case needed to be built from a much more rigid material. We used wood, however, a thicker plexiglass may work as well as metal, PVC, rigid plastics, etc.

2.5.2 Wire Selection

The wire selected needed to be strong enough to sustain continuous movement for long periods of time without stretching or deforming. String, rubber tubing, vinyl rope were all considered but all of these had a tendency to either break or stretch. Uncoated steel wire was also considered and tested, but this had a tendency to damage the wire guides and was difficult to attach. After researching our options, we decided on using a deep sea fishing line which is a very thin steel cable coated in plastic.

2.5.3 Servo Motor Selection

Our original motors were a 15 kg/cm servo which, with a 5 V power supply, was able to articulate the arm during testing, without the camera and light attached. Once the weight of the light was added it was observed that the accuracy and overall performance of our system degraded due to the extra length and weight at the end of the arm. We determined a more powerful servo motor would be needed to accommodate the length and weight of the arm. Since one of our specifications was also to make this system water/dust proof, the servo motors needed to be both strong enough for actuation and waterproof.

2.5.4 Motors Overheating During Actuation

During actuation we discovered that the motors had a tendency to become very hot. In order to prevent the motors from becoming damaged by heat, the stands for the servo and servo mounts were made of steel and copper to act as heat sinks. A 12 V fan was also added to the top of the case to assist in cooling the motors.

2.5.5 System Mounting Position/Distance

With the current iteration of design, the Articulight’s only fully functional position is to be mounted vertically. This is due to limitations with the ATMega28P code not being complete to account for a horizontal position. Furthermore, the system is limited on how close an individual’s hand can be to the light source due to the limited camera frame size. These are both things that will be expanded upon by using a wider frame camera in the future but for now constitutes a constraint.

2.6 Time Constraints

This project was built during the Spring - Summer 2022 semesters at UCF. As such, we had a considerably low amount of time to complete this project when compared to finishing a project during a Fall to Spring semester. To ensure that our project was finished in time given these time constraints, a minimum of weekly group meetings and progress checks were made and help was given to team members if a task was not completed. We also communicated through Discord to allow quick communication through multiple devices rather than a communication method such as text, which is only available on mobile devices.

The constraint of time on finishing this project as it must be completed in the semester had a large influence on our design decisions. Some of the design decisions that were made such as materials were not necessarily the most cost effective or best options available, but instead were chosen because it would save us time in either assembly or receiving the component. A specific example of the time constraint on this project is the choice to use a pre-trained convolutional neural network supplied by NVIDIA for hand detection - if time permitted, we would have trained our own from the ground up.

2.7 Hand Constraints

As this project uses machine learning and computer vision to analyze gestures of the hands, using this project requires the user to have at least one hand. In the event of a user who does not have hands and has a prosthetic that is complex enough to allow at least three unique gestures, the system can be retrained through the available python scripts to learn the gestures that the individual is able to make. However, in the case of no hands and no prosthetics, the system will not be able to work with the user.

2.8 Power Input Constraint

This project was designed for use in the United States with 120V 60Hz AC electricity commonly found here in the United States in households. This is a constraint, because if someone is from a country that does not use the 120V 60Hz AC electricity found in the United States, they must redesign the power electronics to handle the electricity that is native to where they are from.

2.8 Electrical Constraints

Our PCB design required the use of a four layer board in order to keep the traces to a size applicable to our small form factor design. The traces need to be able to accommodate up to 2.5A for each motor and 4A for the Jetson Nano, for a total of 9A of current from the PNP transistor and 5V regulator. A two layer board of the size that would fit in our case would have traces that would overlap and short circuit between the motor power, ground, and pwm lines.

3. Related Standards

3.1 Power Supply Standards

One of the standards that is applicable to this project is the standard of power. The system will need to have a power source of 120V at 60Hz of alternating current because this is the standard residential power output in the USA. The power electronics that are designed to power the system as a whole will need to be designed around this source of alternating current voltage. The majority of light bulbs sold in stores also fit this standard.

3.2 Case Standards

The screwing mechanism in common lightbulbs is referred to as the Edison Screw. The sizes of these Edison Screws have been standardized differently in North America and Europe. In the US, these socket sizes have been standardized in many different sizes. The standardized sizes we consider later on are the E-12, E-19, E-26, and E-39 socket sizes.

3.3 Digital Communication Standards

Communication standards are also relevant to this project. The computer vision computational device (embedded computer) employs some communication protocol in order to communicate to the motor controllers and any other controllers that are applicable. Some of these standardized communication protocols include Universal asynchronous receiver-transmitter (UART), Inter-integrated Circuit (I^2C), and Serial Peripheral Interface (SPI). These protocols are outlined later in the document.

3.4 Electrical Connection Standards

Since this project implements the use of an embedded computer and PCB, connection types are another standard discussed here. Pin headers are one of the standards that are used in modern embedded solutions. Since we are using an embedded computer in this project, pin headers are one of the standards that will be relevant in this project. Pin headers are metal pins that allow connection via wires or jumpers between one device to another. It is used in this project for communication between the microcontroller and embedded computing device. Within this standard is the standard at which voltage the digital pins will produce for on state vs off state (usually 0V). The microcontroller will output the supplied voltage between 1.6 - 5V and the embedded computing device standardizes the output pin voltage to 3.3V. Because these standards are instantiated and documented in their respective datasheets, precautions were taken so as not to let the embedded computation device receive 5V on its pins, potentially burning the pin.

Another standard connection type that is relevant is the Camera Serial Interface (CSI). This standardized connection type specifically for cameras defines the way in which a camera sends data to the processor on a computer. It is commonly found on embedded computers as a camera solution and is more efficient in transmitting the data from the camera sensor than USB, which is also a connection type that is one of the most well known standards today, as seen in the next paragraph.

Another electrical connection standard that is used in this project is the Universal Serial Bus (USB). USB is widely used because it is the most popular connection type for peripheral devices to a main device as it was standardized in 1996. Since its standardization, multiple forms of the USB connection and protocol have been developed. These include but are not limited to USB, USB 2, USB 3, micro-USB, USB-C, USB-A and more. USB 3 is used extensively in this project to connect peripheral devices to the computer vision computational device such as a keyboard for text input and a mouse for navigating the cursor.

3.5 Software Standards

Finally, software standards are also relevant. In developing machine learning algorithms to train and use models, Python has standardized the type of data that are fed into these models in order to train and test them. The standardized version of this data type is called a tensor. It is an abstract data type that allows a seamless experience in computing the mathematics (such as partial derivatives) required to perform modern machine learning modeling algorithms. NumPy's ndarrays, one of the largest used data preparation libraries in Python, is considered to be a tensor as well. The standardized tensor fits well in the name of one of the most popular deep learning libraries to date, TensorFlow.

Python also standardizes the way that external snippets of code called packages are obtained. One of these standardized methods is by using Pip. Pip, Python Package Installer, allows a programmer to obtain necessary packages by issuing the command pip install <package>. If the code that a programmer wishes to obtain is not obtainable through pip, the programmer may check the release for the package. It is a common standard that in supplying the source of the package on a version control site such as github, an installation python file will be provided. Running the python installation file will allow the programmer to install the necessary package.

Along with Python, C++ is utilized in this project. C++ was developed on top of the C programming language and was released in 1983 from Bell Labs. Since then, it has been updated and worked on and is still a popular language today for writing applications and embedded technology due to its efficiency. Because C++ has been updated for almost 40 years as of writing this, it has been standardized in multiple facets. One of these facets is the syntax in which programs are written.

Commenting is a standard that is employed in the majority of programming languages that are currently in use. Comments allow a programmer to explain functionalities of code without needing a separate document to follow along the code with. However, documentation is an important part of the software design process but is generally used to explain use cases of code files and other high level concepts of the code. Comments allow quick and precise information that pertain to specific lines of code that may be confusing to someone who is seeing it for the first time, and is a common standard in writing and reading programming language code.

3.6 PCB Standards

PCB standards are relevant for almost any electronic device if used. As a designer when we create our design, we have to ensure we have a robust design, we need to understand how to design for manufacturability (DFM), how to design for the environment (DFE), how to design for reliability (DFR), design for test (DFT), etc. But, why are these so important to know? If standards aren't followed, material for the PCB board can suffer from poor reliability, cost of PCB can be higher than needed, and PCBs of the same design can be inconsistent. A solution to testing these standards is by testing the performance and quality of the end product by using cross-sectional testing.

A specific set of standards that are employed in today's engineering are the IPC standards. IPC, formerly called Institute of Printed Circuits, has created many standards in the world of PCB design. By using our PCB manufacturer JLCPCB and complying with the design rules they have set to successfully print our PCB, we have indirectly complied with the relevant IPC standards. From JLCPCB's website, they state that they have complied with all of IPC's standards.

3.7 RoHS Standards

RoHS, Restriction of Hazardous Substances Directive 2002/95/EC is a directive established in the European Union in the 2000's that has employed standards for the use of substances in electronics. Some substances can be cheaply used in electronics but have negative consequences biologically if they come into contact with a human. As such, RoHS standards are commonly used in any electronic manufacturing to ensure the safety of individuals who come into contact with manufactured electronic devices.

Since our project contains significant PCB design and deals with many purchased electronics, we decided to be RoHS compliant. To keep our project in line with RoHS standards, we deliberately chose components that do not contain the harmful materials listed in the RoHS directive at a larger than described amount. These materials are: Cadmium (Cd, < 100 ppm), Lead (Pb), Mercury (Hg), Hexavalent Chromium (CrVi), Polybrominated Biphenyls (PBB), Polybrominated Diphenyl Ethers (PBDE), Bis(2-Ethylhexyl) phthalate (DEHP), Benzyl butyl phthalate (BBP), Dibutyl phthalate (DBP), Diisobutyl phthalate (DIBP), all at < 1000 ppm.

4. <u>Administrative Abstract</u>

In this section, you will see an overview of what you will be reading throughout the report. This will cover the research problem and objectives, our methods, key results or arguments, and lastly our conclusion. The problem that we had to address in our ArticuLight project was to find the best way to connect our components to where they wouldn’t overload on voltage and burn out. Our objective of this project is to build a working model of a motion controlled lighting system that has advanced features such as adjustment to brightness, movement controlled via hand gestures that are recognized through a camera, and allows two different power options for more location choices. Our main method to accomplish each objective and to overcome the problem of design was through prototyping. We were able to prototype and test most of the parts, except some due to back orders from websites we ordered the parts from. Another method we used was discussing these goals and problems within our group and using our knowledge learned from our classes, we were able to put together multiple solution plans to overcome constraints. Some of the key results that we learned from our methods was how each component goes hand in hand with one another. Not only that, but we also learned how hardware and software affect one another which led to our ArticuLight to be a success. Overall, we hope that after reading this report it will give you a solid idea of what it takes to build your own device, similar to our ArticuLight system. Furthermore, we hope this gives you more in depth knowledge about the components used and how they work and why it is important to choose a specific component based on principles and/or factors of efficiency, cost, voltage and amperage input/output ratings, and how they affect the design/system as a whole.

5. Technical Abstract

In this section, we will cover similar items to that in the administrative abstract, but will be going more in depth about the components and how they function. Our ArticuLight design runs off of both hardware and software aspects. Without one of these, our design would not function properly. For example, one of our hardware components we chose, which you will read about further on is the Jetson Nano. This particular component will be acting as the master or “the brain” of our system. Not only is it a hardware component, it is also used for the software aspect of our design as it will be programmed to send messages to the rest of the system. This will basically tell other components such as the motors, relays, and the camera.

The camera is an important component needed for the design that pairs well with the Jetson nano, as it would be used to recognize the hand gestures made to control the movement of the light. As you read the report you will see how the camera is used for “seeing” the hand gestures made and sending a message over to the Jetson Nano, which will then send the message to the motors, which in turn, activates the gears, thus adjusting the light to the desired position that the user commands.

6. Project Narrative

The introduction of electric power has led to many advancements in the modern home. One of these advancements is the use of light bulbs to illuminate an area while it is being used. While lighting is widely used for general illumination, it is also imperative to have a good point source light for certain activities such as mechanical work, wood working, electronics soldering, etc. As such, some activities may also require reorienting a point source light to highly illuminate a specific area that requires detail. The purpose of this project is to facilitate this need by designing a fully functional articulating light system with the ability to reorientate to a gesture-controlled location. The light system will also be expandable, allowing more lights to form an array that work together to illuminate an area.

The lighting system will perform gesture-controlled tasks using computer vision. The software will intake an image of the user and identify the hands of the user. When a gesture is performed, the software will recognize the gesture and issue commands for the gesture such as reorienting the position, turning the light off, and resetting the position of the light to the default setting. The timing between a user's gesture and the action that is mapped to the gesture will be reasonable, within a few seconds. There will also be physical buttons/switches that perform tasks that don't require a parameter such as a position to shine (turning on/off, resetting position of light, etc).

Each light will have its pitch and yaw controlled by a pair of servo motors connected to an articulating arm and gear system which will allow for a wide range of motion. The available range of motion should allow the system to light up anywhere in range of the camera. The light arm will have many joints that form a spine and a flexible wire loom that will protect the connecting wires and add rigidity to the system so that the light can angle itself without moving the whole system. Materials chosen for the light should be light enough to be considered portable but heavy enough to be sturdy. The light will feature a modular mount that allows wall, ceiling, and stand mounting with appropriate equipment.

Humidity plays a crucial role in electronics components. Humidity can affect circuit boards by causing corrosion. Humidity in electronic components could cause enough water to enter the system and cause a short circuit in any of the electronic components. Since water is highly conductive, electricity will surge in the path of least resistance causing significant damage in components. Another issue with humidity and moisture in the air will cause solder components to oxidize making the solder joints weak to eventually fracture and fail over time.

The lighting system should be robust enough that it can withstand the humidity and heat of a Central Floridian garage. There are a couple of ways to make the electronic circuit more robust. For circuit boards that could be damaged exposed to humidity, we could use a urethane conformal coating to seal up all the sensitive electronics or if the electronic components are sealed in an enclosure we could use suction fans to reduce the humidity level. If the electronic components produce enough heat, it will act as a heating element which in itself will lower the relative humidity.

According to a google search on Florida's humidity, the yearly average humidity of a Florida garage can range from as low as 54 to as high as 91, as shown in the table below. Furthermore, the average humidity throughout a year is 74.3. The daily heading in the table will show the average humidity in that specific part of Florida. Moreover, the morning heading in the table will show the humidity percentage that was taken at 7:00

AM in the specific part of Florida, and lastly, the afternoon heading in the table will show the humidity percentage that was taken at 4:00 PM in the specific part of Florida. The only section in the table that is different is the time humidity was taken in Pensacola. The time humidity data was taken in Pensacola was at 6:00 AM and 3:00 PM. It is also important to note that these times were gathered and recorded in local standard time (Eastern Standard Time - EST). These were findings from 1961 – 1990.

Place	**Morning**	**Afternoon**	**Daily**
Daytona Beach	86	62	75
Fort Myers	90	58	75
Jacksonville	89	58	76
Key West	79	67	74
Miami	84	62	73
Orlando	89	56	74
Pensacola	85	60	74
Tallahassee	91	54	75
Tampa	88	57	74
West Palm Beach	83	63	73

Table 1: Florida humidity in percentages

From our findings, we can conclude that we will need to make sure our ArticuLight system can handle the humidity and hot temperature that Florida brings year round. Furthermore, due to ozone layer depletion, Florida is only going to get more humid and warmer temperature year round. From this, we are able to plan our design to have a good heat resistance to keep it cool year round, so Florida weather won't be too much of a constraint.

6.1 Specifications

- Weight
    - No more than 7 lbs
- Power Input
    - 120V AC, 60 Hz
- Power Output
    - 120V AC, 60 Hz LED light socket
    - 5V USB Camera
- Footprint (base and length)
    - 10x10x10 inch gear box
    - 2 feet length
- Pitch/Yaw Angle/Degrees of Freedom
    - Up to +/- 90 degrees of movement left and right (yaw)
    - Up to +/- 90 degrees of movement up and down (pitch)
- Price
    - under $350
- Computer Vision
    - Will use a camera to identify hand gestures to control the movement of the actuators
- Stand
    - The mount will be modular
        - Mountable
        - scalable by adding additional lights
- Environmental robustness
    - Able to handle temperature and humidity variation
        - 20 - 100 degree temperature range
        - waterproof
        - dust proof
- Response Time?
    - Software to hardware interface will have a response time less than 3 sec
- Manual reset/zeroize
    - Device will have a way to manually stop and reset the system

6.2 Key Engineering Specs

Engineering Specification	**Value**
1) Accuracy	**<=1 cm**
2) Computation Time	**<= 2 seconds**
3) ActuationTime	**between 3 - 5 sec**
4) Power Input	120V AC
5) Weight	<=7 lbs
6) Durability	> 1000 uses
7) Arm Length	24 inches
8) Light Voltage	?? V

Table 2: Key Engineering Specifications

The computation time (#2) is the time it will take the system to read in a gesture from the camera and send a stimulus to perform some kind of actuation, depending on the gesture. For example, if a gesture is performed to turn the light off, the computation time is the time it takes from performing the gesture to the instruction being sent to the microcontroller to turn the light off.

Actuation Time (#3) is the time it will take when the computation is finished to complete the movement. This product isn't meant to be used in such a way that the arm can touch the user, but to enable a margin of safety, a minimum time will be given to ensure the arm doesn't move so fast as to strike the user with substantial force to cause damage.

Accuracy (#1) is determined by how close the device will move the light source to focus on an indicated location. Measurements will be taken from the center of the light focal point for consistency. Testing was performed by using a laser pointer mounted in place of the light in order to verify the center of the light would map correctly to the landmark used, the thumb in this case, and measurements were taken with a ruler based on where the laser light lands with regard to the tip of the thumb.

6.2.1 Stretch Goals

Within the time frame allotted for building this project, the design specifications will be a challenge to meet, however, if time does permit, a couple of stretch goals are being considered to allow for a more user friendly and adaptable functionality.

6.2.1.1 Solar and Battery Power

The power supplied for this project has focused primarily on using a wall outlet to supply AC voltage to an AC - DC converter in order to power all of the components. Adding a rechargeable battery option would allow greater portability for the light. Because we already have a reasonably light weight design, the extra weight added from adding a rechargeable battery circuit housed within the same base as the voltage regulator would not add much to the design. The batteries could be recharged either from a solar cell or the outlet.

6.2.1.2 Adding a Third Degree of Freedom

Adding an additional degree of freedom is an aspect that was highly considered from the onset of design, however we decided to take a more conservative approach for specifications. This would allow the ArticuLight to expand and contract allowing the user to focus the light on a tighter spot or diffuse it as needed. The gesture recognition would act similarly to the dimming feature. However, adding this feature would require at least one more motor added to the design and to alter the frame so that it can compress. This would alter the movement to be a lot smoother and flexible, essentially creating a continuum robotic arm.

6.2.1.3 Interconnectability

This concept has a few implementation possibilities, but in general being able to expand the design to incorporate multiple lights would be an amazing feature to impart. This comes with a few complications that would require major changes to the design. The ArticuLights could either act independently, each covering a specified area. Another option would be connecting multiple lights together with one acting as master to the others. Doing so we would just have to determine if each light would have its own camera or if all of them shared one camera. Either way, the additional processing power needed for all the cameras would need to be considered and determined to set a limit on how much interconnection is possible. The other aspect for this is the power requirement for the extra motor. For power supplied through an outlet, this will not likely be a large concern, but for battery operation this could reduce the sustained use time by quite a bit. The overall cost of having individual systems connecting together would just multiply the cost of components as well. If design for this aspect is attempted, we will likely just try to incorporate a new arm, with motors, into the system that is currently in place. That would minimize the additional resources needed.

6.2.1.4 Incorporate FPGA Design

A final stretch goal we are considering is to add a Field Programmable Gate Array, FPGA, design element to the project. FPGA is a highly adaptable circuit board that allows programming at the logic level to develop virtually any functionality, even in parallel operation. This could be used in place of a motor controller since the FPGA could be programmed to produce PWM signals at any output pin based on the serial data provided. The largest advantage to this would be the capability to control all of the motors in parallel instead of sequentially as a microcontroller would. The adaptability of this comes at a cost however. Most FPGA boards do not have a processor embedded on the board and are generally limited in memory. So if power is lost to the system or disconnected, the FPGA could use its programming. There are newer boards that do include more non-volatile memory, but they are expensive. The logic level programming for FPGA is made easier by using high level synthesis, HLS, done through programs such as Verilog HLS or Vitas HLS. These programs can translate C++ code to hardware description language, HDL, such as Verilog, SystemC, or Verilog HDL. From this program, it would also be possible to specify design constraints such as size, power, and speed so when converted the hardware is optimized to those specifications. HLS also supports machine learning and has libraries for OpenCV. This will be considered, but an FPGA board large and fast enough to handle this would be outside of our budget and as such will not likely be attempted. If we do implement an FPGA for motor control, it will enable us to generate PWM signals and even PID control within one board (explained later in this report) as well as reducing the maximum time required for a full range movement in half.

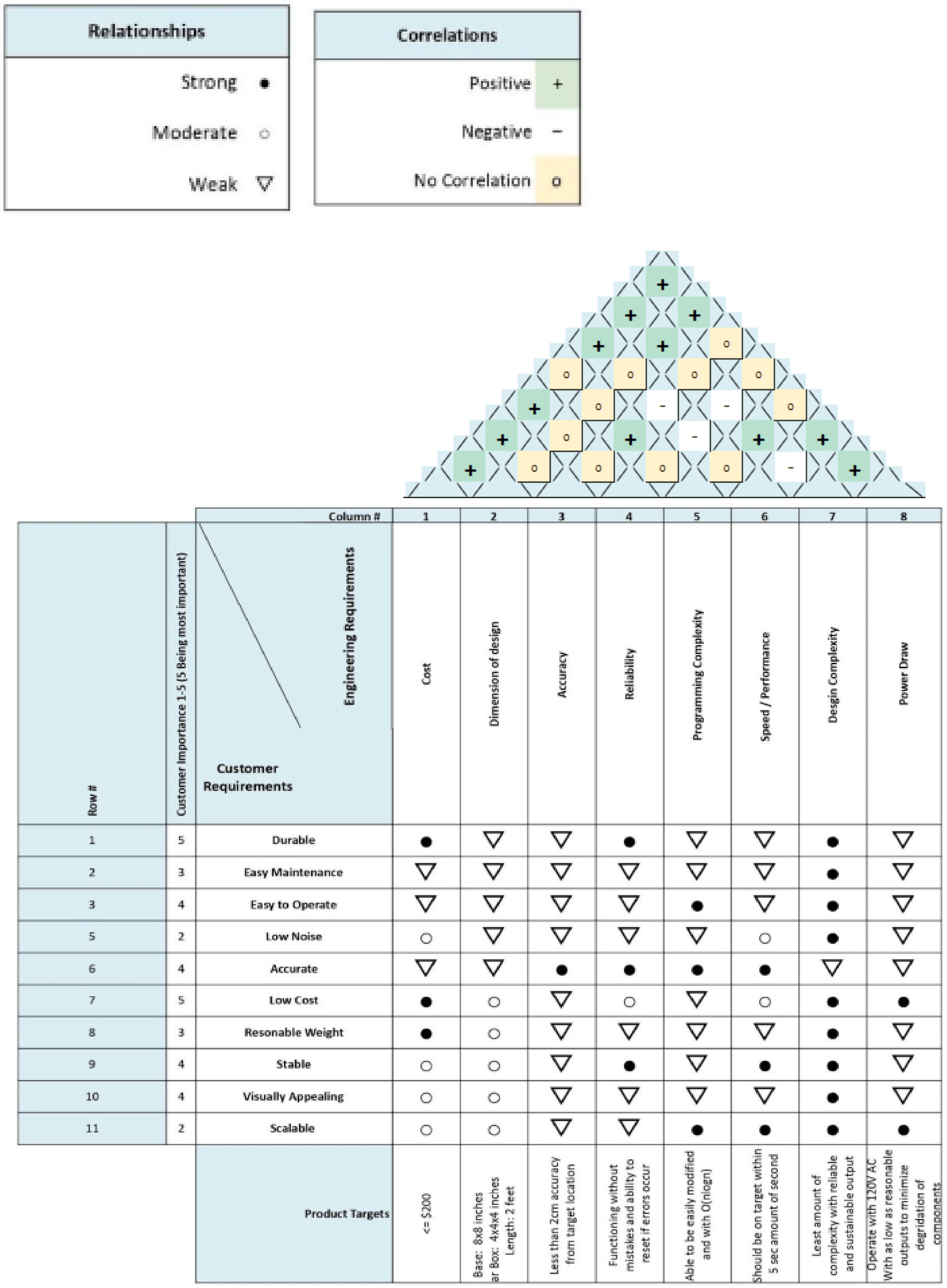
6.2.2 Related Work

Inspiration for this design came from a video created by the youtube channel, NY CNC, who created an octopus chandelier, a link to the video can be seen in the Citations. The design is effectively one leg of the chandelier with the stretch goal of being able to in essence duplicate something similar to the video. To elaborate on this piece, we decided to add machine learning, gesture recognition to make a highly functional fixture that can be used in most work environments.

6.3 House of Quality

The house of quality figure (figure 1) is a shorthand way of outlining the relationships and correlations that the engineering and customer requirements have in common. The "roof" of the house of quality shows the correlations between the engineering specifications and the body of the house shows the relationships between customer requirements.

The legend at the top shows the symbols for each relationship type. In the roof, the relationships correspond as follows: For each column (cost, dimensions, accuracy, etc) the boxes that align diagonally correspond to this specification.



Relationships	
Strong	●
Moderate	○
Weak	▽

Correlations	
Positive	+
Negative	−
No Correlation	o

Row #	Customer Importance 1-5 (5 Being most important)	Customer Requirements \ Engineering Requirements	Cost	Dimension of design	Accuracy	Reliability	Programming Complexity	Speed / Performance	Desgin Complexity	Power Draw
		Column #	1	2	3	4	5	6	7	8
1	5	Durable	●	▽	▽	●	▽	▽	●	▽
2	3	Easy Maintenance	▽	▽	▽	▽	▽	▽	●	▽
3	4	Easy to Operate	▽	▽	▽	▽	●	▽	●	▽
5	2	Low Noise	○	▽	▽	▽	▽	○	●	▽
6	4	Accurate	▽	▽	●	●	●	●	▽	▽
7	5	Low Cost	●	○	▽	○	▽	○	●	●
8	3	Resonable Weight	●	○	▽	▽	▽	▽	●	▽
9	4	Stable	○	○	▽	●	▽	●	●	▽
10	4	Visually Appealing	○	○	▽	▽	▽	▽	●	▽
11	2	Scalable	○	○	▽	▽	●	●	●	●
		Product Targets	<= $200	Base: 8x8 inches ar Box: 4x4x4 inches Length: 2 feet	Less than 2cm accuracy from target location	Functioning without mistakes and ability to reset if errors occur	Able to be easily modified and with O(nlogn)	Should be on target within 5 sec amount of second	Least amount of complexity with reliable and sustainable output	Operate with 120V AC With as low as reasonable outputs to minimize degridation of components

Figure 1: House of Quality

6.4 Hardware Block Diagram

The hardware diagram below (Figure 2) shows the components that we will be implementing into our design. As shown we will be using a 120V AC power source wall outlet plug-in. The power from the wall outlet will then flow through a voltage regulator which in turns connects to our microcontroller (in our case the Jetson Nano) to provide a voltage that won't overload and cause any damage to our design. After that connection is created it then breaks into 3 different branches. First we look at the top of the flowchart. From the microcontroller, it will then flow voltage through the relay which will increase the voltage flow to 12V to allow the motor controller to function. depending on the component we choose the relay may or may not be needed for the motor controller. Next the connection then goes into the servo motors. The servo motor maximum input voltage we will choose will be anywhere in the range of 5V up to 12V but no more than 12V since we won't be needing anything more powerful than that for our design. The servo motor then will control the gears which will be used for the movement of the arm. For the middle branch of the flowchart (i.e. microcontroller to camera connection) the microcontroller we will be using for this design will be the Jetson Nano which will be used to program the camera to recognize hand gestures. These specific hand gestures will allow us to make the motors and gears shift into the position for each specific gesture we specified. For the last branch, the microcontroller will send a signal to the second relay to turn on the LED light or lights depending on if we will incorporate more lights into the system. The relay is needed to control the higher voltage supplying power to the LED light.

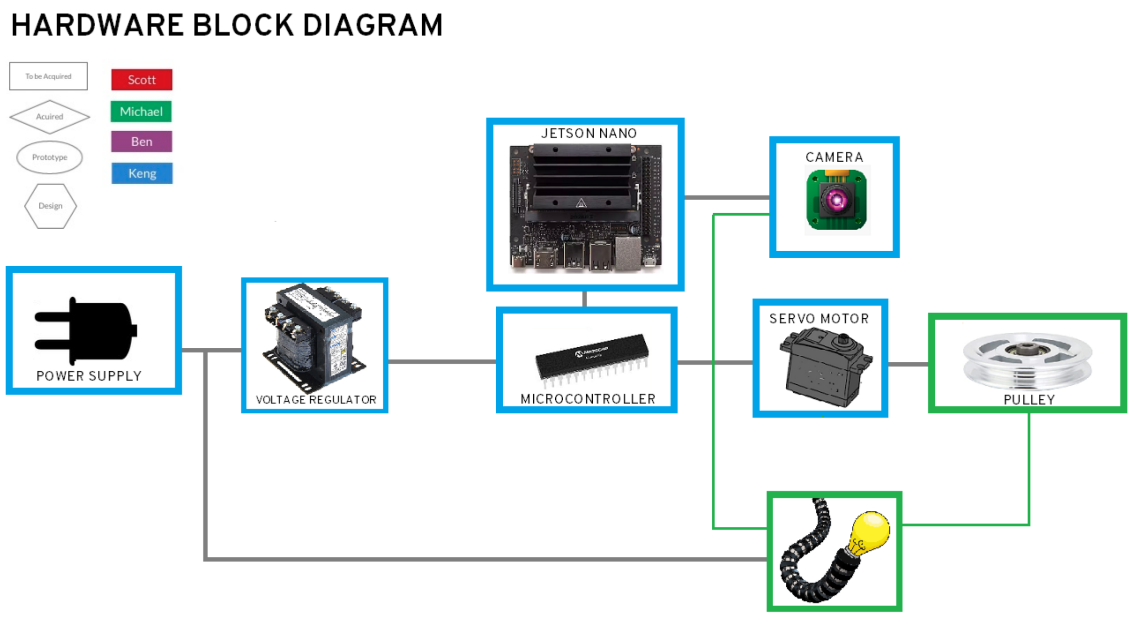


Figure 2: Hardware Block Diagram

6.5 Software Block Diagrams

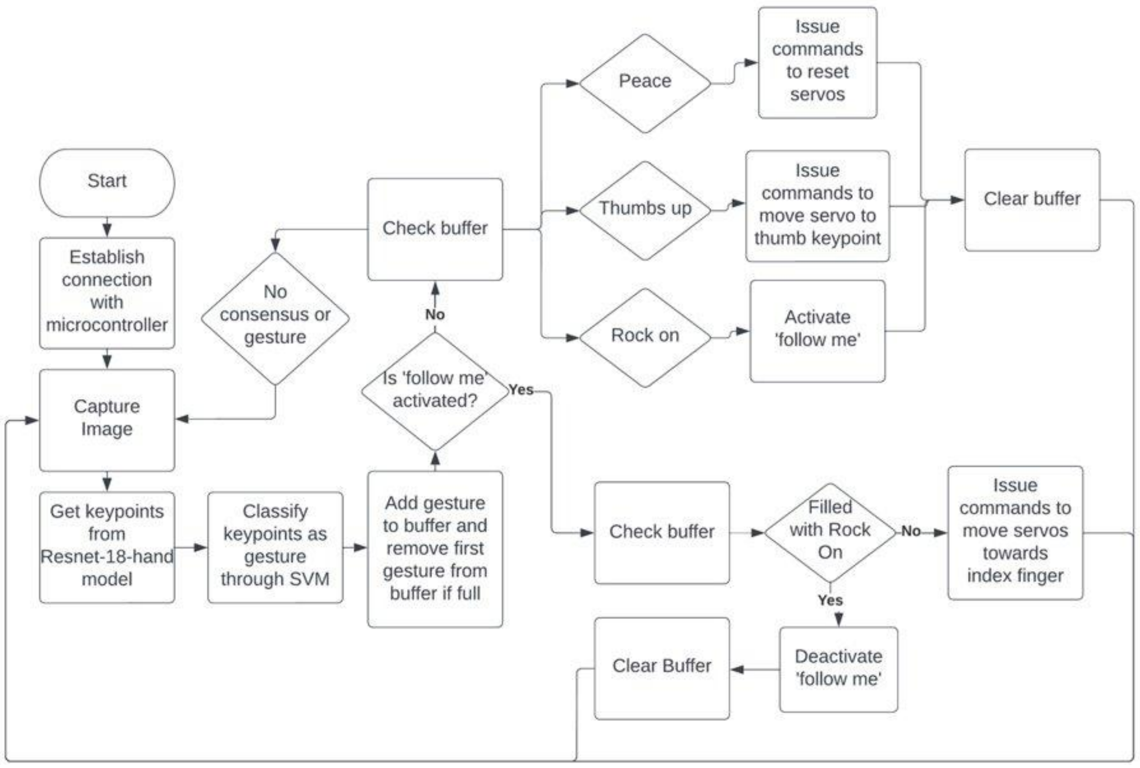
Outlined in this section are two software block diagrams: the software on the Jetson Nano (Figure 3) and the software on the ATmega328 microcontroller (Figure 4). The outputs of the Jetson Nano software are the inputs for the ATmega328 microcontroller, and the communication between the two devices is set up so that communication is one way only.

The machine learning and computer vision tasks run on the Jetson Nano, and send signals through serial communication after performing decisions based on the computer vision and machine learning outputs. The Jetson Nano upon compilation of the main script sets up communication with the microcontroller and enters an infinite loop where an image is captured, classified, and then added to a buffer of gestures to ensure the gesture has been held to avoid action on a single erroneous classification.

The main body of the loop has two states: follow me activated and follow me deactivated. When follow me is activated, it checks the buffer for the deactivation gesture. If the gesture is not persistent in the gesture buffer, it sends commands to the MCU to allow the servo motors to center the index finger to the frame of the camera. Notice that the index finger does not need to be part of a gesture for the script to send the necessary information to the microcontroller. If follow me is deactivated and the gesture buffer is full of a defined gesture that has behavior, the action is taken that is defined for that gesture. The gestures and their corresponding actions are listed in Table 3. After an action has been executed other than issuing the follow me commands to the microcontroller, the gesture buffer is cleared to prevent an action happening multiple times when only one action was necessary.

Figure 4 has the software block diagram that is instated on the ATmega328 microcontroller. The ATmega328 instantiates the servo connections and serial communication with the Jetson Nano and starts a state loop. It checks for data on the serial buffer (data sent from Jetson Nano) and checks the first byte of the data. ‘w’, ‘x’, and ‘r’ are acceptable bytes. The format of the information sent on the serial bus from the Jetson Nano is byte, value, byte, value.

The first byte dictates the type of operation and the second byte is there to separate the values. In the case of the first byte being ‘w’, it maps the values from pixels sent from the Jetson Nano to servo angles and issues commands to the servos to move to the mapped value. This corresponds to the thumbs up gesture in the Jetson Nano block diagram. In the case of ‘x’, it steps through the corresponding angle until the Jetson Nano stops issuing the commands. This corresponds to the ‘follow me’ functionality on the Jetson Nano. Lastly, when the first byte is ‘r’, it resets the positions of the servos to the default state, which is 90 degrees.

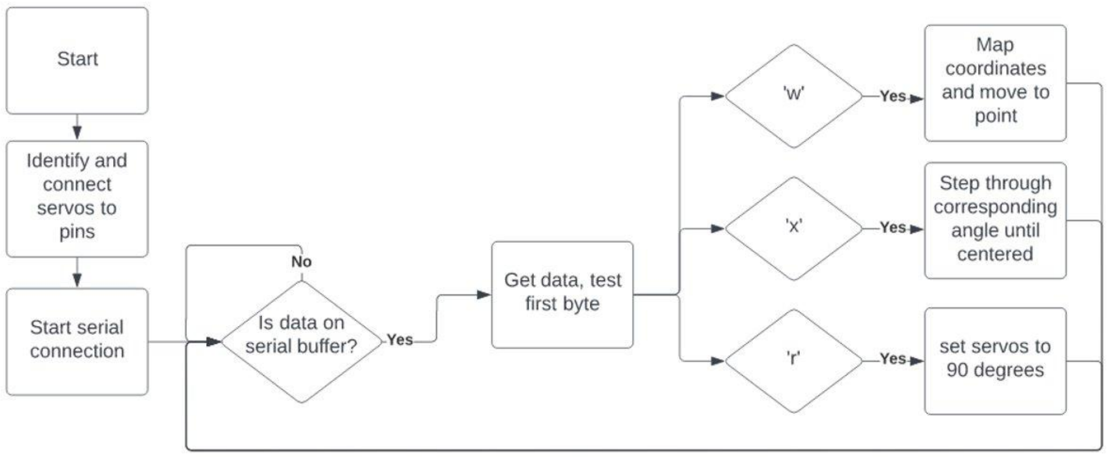


Start
Establish connection with microcontroller
Capture Image
Get keypoints from Resnet-18-hand model
Classify keypoints as gesture through SVM
Add gesture to buffer and remove first gesture from buffer if full
Is 'follow me' activated?
No
Yes
Check buffer
No consensus or gesture
Peace
Thumbs up
Rock on
Issue commands to reset servos
Issue commands to move servo to thumb keypoint
Activate 'follow me'
Clear buffer
Check buffer
Filled with Rock On
No
Yes
Issue commands to move servos towards index finger
Deactivate 'follow me'
Clear Buffer

Figure 3: Jetson Nano Software Block Diagram

Peace	Reset motor states
Thumbs up	Issue command to MCU to center thumb
Rock On	Activate/Deactivate follow me

Table 3: Gesture to System Behaviors



Start
Identify and connect servos to pins
Start serial connection
Is data on serial buffer?
No
Yes
Get data, test first byte
'w'
'x'
'r'
Yes
Yes
Yes
Map coordinates and move to point
Step through corresponding angle until centered
set servos to 90 degrees

Figure 4: ATmega328 Software Block Diagram

7. Research and Part Selection

The first step in designing the Motion Controlled Lighting System is to research the components that will be used in the design. This includes hardware components such as microcontrollers, motors, motor controllers, light socket/bulbs and so on. Furthermore, it also explores software aspects such as computer vision for gesture control and microcontroller software. Each component and their purpose will be discussed so that the best component will be chosen for use in the project. As you will see further on in the report, there are various components and models of these specific chosen components that will be compared in different tables and figures.

7.1 Microcontroller Boards

The goal of this project is to have a light fixture that is self movable and controlled based on hand-gesture commands. As such, some electrical stimulus will be required to activate the motors. A microcontroller chip is a small computing device that can send stimulus to the motors by outputting Pulse Width Modulation (PWM) signals to perform tasks such as moving the fixture. As a separate computer vision module is required, the module will have to communicate with the microcontroller to employ needed functions. A chosen microcontroller board must meet the following specifications for this project:

- Minimum of four pins for motors (two pins each motor), two of which are PWM or analog output for control (assuming no motor controller)
- Device communication (I^2C, UART, etc) for communication with the computer vision device

In the case that multiple chips meet these needs, the devices should be compared based on other qualities such as online or community support, development environments, running temperatures, storage, space, price, and availability.

7.1.1 Specification of Microcontrollers

Table 4 below shows the summary of comparable specifications between the three microcontroller chip technologies discussed in this section. The table outlines the measurable specifications that are of importance in the design of the articulating arm. Because the ATmega 328p meets the requirements necessary for the scope of this project and is convenient to obtain, it is chosen for this project. More information about each of the microcontrollers is supplied below the table.

MCU	ATmega328p	MSP430G2553	STM32G031F8P6	MSP430FR6989
Input Voltage (V-DC)	1.8 - 5.5	1.8 - 3.6	1.7 - 3.6	1.8 - 3.6
Digital I/O Pins	23	24	44	40
PWM Channels	6	3	6	5
Flash Memory (KB)	32	16	64	128
Length (mm)	35.5	24.33	6.6	16
Width (mm)	8.6	9.4	6.6	16
Retail Price	$3.26	$3.13	$3.54	$6.88

Table 4: Specifications of Microcontrollers

7.1.2 Microcontroller Board Options

The following embedded microcontroller chips are covered in this component research section: the ATmega328P, MSP430G2553, STM32G031F8P6, and MSP430FR6989. These microcontrollers all would be capable devices for completing the required tasks that a microcontroller must do in this project. However, since the ATmega328 microcontroller is currently obtainable with no lead time, can be flashed with the code necessary to perform motor control, and has a lot of community support in part to its use in the Arduino platforms, it is chosen for this project.

7.1.3 ATmega328P

The ATmega328P microcontroller is the microcontroller commonly used on the Arduino family of boards. One of the benefits of using the ATmega328 is that the Arduino bootloader can be flashed onto them, allowing the programmer to make use of the Arduino libraries, and in our case, the motor control libraries for servo motors. This is a very convenient way for us to implement motor control without the need for manual generation of PWM signals. This is one of the deciding factors on why we chose to use this chip in our embedded design.

The ATmega328 features an input voltage of 1.8 - 5.5V, 23 digital input/output pins, 6 PWM channels, 32KB of flash memory, and is 35.5mm x 8.6 mm. It is priced typically for its functions, at approximately $3.26 per chip.

7.1.4 MSP430G2553

The MSP430G2553 is the microcontroller that is present on the MSP-EXP430G2ET launchpad development kit that is used in the engineering analysis course here at UCF. It features a 1.8 - 3.6 DC Voltage range, 24 digital IO pins, 3 PWM channels for motor control, and 16KB of flash memory.

Although this microcontroller contains the specifications that are required for this project, motor control with this chip will be considerably harder than motor control with the ATmega328 microcontroller. This is because as of researching this microcontroller, it would be necessary to output PWM signals manually through the code, which would require much more time to implement. As this project was built during a summer semester, it is imperative to make design choices that save time in implementation.

7.1.5 STM32G03IF8P6

The STM32G03IF8P6 microcontroller is a very good choice of microcontroller for motor control specifically. It features 6 PWM channels for PWM servo control and is very robust in GPIO pins. It features 44 GPIO pins, being the most communicative microcontroller to be discussed in this paper. This microcontroller has 64KB of flash memory, which is more than enough for the tasks a microcontroller will need to complete in this project, which is primarily motor control. It also has a small footprint, only 6.6mm x 6.66mm. It

7.1.6 MSP430FR6989

The MSP430FR6989 microcontroller is the microcontroller that is used in the MSP430FR6989 LaunchPad Development Kit that is used in the embedded systems course here at UCF. Using this microcontroller would be an advantage in the sense that the students who have built this project are comfortable with the software development principles on this chip due to the embedded systems course. However, as the main function of the microcontroller in this project is to send PWM signals to servo motors, it suffers the same issue as the MSP430G2553 microcontroller, in that servo signal generation will be much more time consuming than using the ATmega328 microcontroller.

7.2 Computer Vision Computational Device

The computer vision computational device is a computing device that can perform computer vision identification tasks by streaming images from a camera, analyzing them, and sending signals to the microcontroller to perform gesture controlled tasks. Because the task of training a computer vision model manually is out of the scope of this project, the device should be able to run software that streamlines this process. As such, the device should have some sort of GUI based operating system that allows the installation and compilation of custom software. The device will need to be able to run a trained computer vision model at a rate such that a gesture will not be missed or be inaccurate. To achieve this, multiple images from the camera must be analyzed every second, as a gesture may rely on movement such as setting the dim level of the light.

7.2.1 Computer Vision Computational Device Specifications

Table 5 shows the specifications of the considered computer vision computational device options; The Raspberry Pi 4, OpenMV microcontroller, and the NVIDIA Jetson nano Developer Kit. From the specifications below, the Nano Developer Kit is the device that is chosen for this project as it performs the best in deep learning scenarios such as computer vision.

Device	**Raspberry Pi 4 B**	**OpenMV**	**NVIDIA Jetson Nano Developer Kit**
GPIO Pins	28	10	40
Has GUI Based OS	Yes	No	Yes
RAM	1 GB - 8 GB	1 GB	4 GB
Processor	BCM2711 up to 1.5 GHz	STM32H743VI 480 MHz	Quad-core ARM A57 1.43 GHz
Operating Voltage	5 V	3.3 V	5 V
Length	85.6 mm	31.5 mm	69 mm
Width	56.5 mm	35.56 mm	45 mm
Weight	46 g	16 g	249.5 g
Retail Price	$55 for 4 GB RAM, $75 for 8 GB RAM	$65	$99

Table 5: Computer Vision Computational Device Specifications

7.2.2 Computer Vision Computation Options

Since much computation is needed in any deep learning application and computer vision is certainly a deep learning application, the device that will be used for processing images and running identification algorithms will need to be able to perform general purpose computing. Since the goal of this project requires the computing to be embedded, the computer vision device will need to be some small form factor computer.

The Raspberry Pi devices are extremely popular for embedded computing and show some promise in being able to handle basic computer vision tasks. OpenMV, an embedded computer vision device, is marketed as such and is a relatively rare type of device as it is a full blown embedded computer vision processor. Although it is incapable of hosting a GUI based operating system, it is capable of using the OpenCV computer vision library as an embedded system, and contains a camera embedded into the system.

Lastly, the NVIDIA Jetson Nano Developer Kit is a general computing device with a GUI OS (Linux) that is designed for the use of deep learning algorithms, such as autonomy and computer vision.

7.2.2.1 Raspberry Pi 4 Model B

The Raspberry Pi series of microcomputers have been around since 2012. Since then, there has been development on optimizing and increasing its computing capability. Often used as a general computer, it is also a good choice for embedded computing due to its size and low power consumption. The Raspberry Pi runs on a unique Unix based operating system specifically designed for the hardware on the Raspberry Pi. Because the Raspberry Pi OS is Unix-based, the amount of available software libraries and support are abundant. It is also a very popular component for at home projects such as use for an entertainment system, a small and lightweight portable game emulator, smart home applications, and much more.

Preliminary research in projects that use computer vision show that it can perform object identification at a relatively low rate of frames per second. This is not optimal for the objective of this project and may pose a problem for movement based gestures, such as setting the dim of the light on an interval with the use of distance between hands. A low frame sampling rate of a movement based gesture may result in stuttering or unpleasant light behavior. The camera options available are also quite limited, as the operating system the Pi runs on (Raspberry Pi OS) is only officially compatible with cameras made for the Pi. The cameras available for the Raspberry Pi 4 are outlined later in the camera options section.

The MSRP price of the Raspberry Pi 4 is dependent on the amount of ram contained in the computer. The available options for RAM size are 1 GB, 2 GB, 4 GB, and 8 GB. These versions retail at $35, $45, $55, $75 respectively. The 4 GB and 8 GB versions would be the only ones considered for this project as that amount of RAM is needed for the application of deep learning.

7.2.2.2 OpenMV Cam H7 R2

The OpenMV Cam H7 R2 embedded computer vision device is not a general purpose computing device, but is a microcontroller itself. It runs on a software called MicroPython, which is a python based language that is used on some microcontrollers. MicroPython can be beneficial for use in machine learning applications, as Python is one of the most common languages to develop machine learning models in. Because it is a microcontroller and not a computer, there is no visual operating system that a user can interact with. It must have a separate computer connected to the device in order to change its function, unlike the Raspberry Pi and NVIDIA Jetson Nano Developer Kit.

Since many computer vision models are trained using libraries in the Python language, this allows more abstraction between the models that are trained and the microcontroller tasks that will need to be performed. The performance of the OpenMV device is not the best, similar to a Raspberry Pi through looking at example projects on the internet. It does contain an embedded camera which is convenient, but only has an image resolution of 640x480, which could pose problems in recognizing gestures when the user is further away.

Priced at $65, the OpenMV board is a great option for computer vision projects with a deep learning model that will compute at a low amount of frames per second. It is specifically a good choice for computer vision computation as well as a microcontroller. Because the OpenMV system only houses 3 servo pins, it would not bode well for this project and would be used solely as a computer vision computational device.

7.2.2.3 NVIDIA Jetson Nano Developer Kit

The Jetson Nano Developer Kit, developed by NVIDIA, is a computer that runs on a Linux based operating system and the Jetson AI architecture and is marketed for deep learning use. A Linux based operating system in this context is very useful as the amount of software libraries available for use are going to be plenty. Since computer vision is a deep learning application, it is a natural choice for a project that contains computer vision tasks. It is particularly useful in deep learning because it contains a GPU architecture that allows neural network model training and algorithms to occur in parallel. Parallel computation of machine learning tasks increases the efficiency of the model processing speed and is greatly beneficial for any system that requires deep learning computation in real time. It also contains a multitude of camera options, and has over 100 cameras officially supported by NVIDIA. Some projects that use this device for computer vision applications show a performance of frames per second of >20, which is enough for accurate detection of movement based gestures.

The NVIDIA Jetson Nano Developer Kit also contains many GPIO pins that could potentially replace the functionality of controlling motors done by the microcontroller board, although the amount of pins that are PWM capable may be limited in our use-case. The pins are also useful in the context of using a dedicated microcontroller for controlling the other peripheral devices such as the light and motors as the pins on the Nano may be used to transmit information to the microcontroller without the need for USB for communication.

The Jetson Nano Developer Kits come in a 2GB and 4 GB version, with an MSRP value of \$60 and \$100 respectively. Although these are more pricey than the other options outlined previously, the performance boost is worth it. Computer vision is the heart of this project and as such spending more on the performance of the deep learning model is a wise decision. The Jetson Nano is the best option out of the options that were outlined.

7.3 Cameras

The camera that the computer vision module will use is very important because the image that is fed into the computer vision algorithm needs to have high enough resolution to effectively recognize objects from a distance. Additionally, the camera should prioritize delay since this project is a real time system. Reducing the delay from capturing light in the lens to sending the data to the computer vision algorithm will decrease the overall computation time and lead to a faster running articulating light. The camera should also have an interface that is compatible with the device that is running the computer vision algorithm. Note that the OpenMV board contains an on-board camera, so no options will be outlined here.

Cameras are capable of connecting digitally through the use of many different data transfer and power protocols. Two of the most popular interfaces for connecting cameras to embedded devices are the CSI (camera serial interface) and USB interfaces. USB is one of the most common interfaces for general data transmission in the computing world. Most computers will have multiple USB 2.0 and 3.0 ports in order to connect multiple peripheral devices that can perform a wide array of functions. However, the MIPI Serial Camera Interface is one of the faster interfaces from lens to processor. In order to reduce the time it takes for the computer vision algorithm to receive information from the camera the MIPI serial camera interface will be focused.

Two common camera sensor types are rolling shutter and global shutter. A shutter is the method in which the camera sensor observes light and converts it into a digital signal. In the rolling shutter, pixels in the camera sensor are updated row by row, “rolling” down the sensor. In the global shutter scheme, every single pixel is updated at the same time. Rolling shutter may result in artifacting and transformation of the image in objects that are moving. For example, in a rolling shutter camera, the blades on a helicopter will show as curved because they are rotating at a fast rate. In a global shutter camera the blades would be seen as straight. Though the gestures that will need to be captured in this project may be movement based gestures, a camera with rolling shutter will suffice as the gestures will not be high speed movements. Rolling shutter cameras tend to be cheaper, so they will be considered for use in the project.

Although the Jetson Nano is built for AI tasks such as computer vision, it still isn't a very powerful device when compared to the hardware typically used for such an application. As such, the python script running on the Jetson Nano ends up parsing frames at a rate of eight per second. So, as long as the camera can handle eight frames per second, it is a high enough frame rate.

7.3.1 Camera Specifications

Table 6 shows the specifications of the camera modules that are outlined below. The cameras that are the best quality with budget in mind are reviewed. The Raspberry Pi Camera Module 2 is one of the most commonly used cameras with the Jetson Nano. Although the Arducam IMX219 Visible Light Fixed Focus Camera contains the same sensor as the Raspberry Pi Camera Module 2 and is cheaper, it is not chosen for this project as it is not as supported as the Raspberry Pi cam V2. The full specifications for each camera is shown below.

Camera	**Raspberry Pi Camera Module 2**	**Raspberry Pi High Quality Camera**	**Arducam IMX219 Visible Light Fixed Focus Camera**	**Arducam IMX219-AF Programmable/Auto Focus Camera**	**Arducam IMX219 Low Distortion M12 Mount Camera**
Image sensor	IMX219	IMX477	IMX219	IMX219	IMX219
Maximum resolution	8 Megapixels	12 Megapixels	8 Megapixels	8 Megapixels	8 Megapixels
Horizontal FOV (degrees)	62.2	Depends on lens sold separately	62.2	65	75
Changeable Lens	No	Yes (required)	No	No	Yes (required)
720p frame rate	60 fps	60 fps	120 fps	120 fps	120 fps
Price	$25	$50	$19	$30	$32

Table 6: Camera Specifications

7.3.2 Camera Options

Since the OpenMV device contains a camera out of box, the Raspberry Pi and NVIDIA Jetson Nano compatible cameras will be explored. The Raspberry Pi camera options are limited, as only a few cameras are officially supported by the device. The Jetson Nano however has many more options available including the Raspberry Pi cameras. Cameras specifically for the Jetson Nano that are outlined here include the IMX219, IMX219-AF, and the IMX219 M12 cameras. All of these cameras use the IMX219 photosensor, which is the sensor also used on the Raspberry Pi Camera Module 2. There are hundreds of cameras compatible with the Jetson Nano Developer Kit, but only the low profile and lower budget cameras that are interfaced via the MIPI serial camera interface were considered for use.

7.3.2.1 Raspberry Pi Camera Module 2

The Raspberry Pi Camera Module 2 contains the IMX219 sensor and is able to capture fixed focus images at 8 megapixels and can capture video at up to 1080p. It boasts a high frame rate at lower resolution, as will be used in the project. Considering the scope of this project, these specifications are definitely adequate to capture high quality images at a high frame rate for the computer vision model, even at a decent distance.

Because the Camera Module V2 is connected through CSI, it will have less latency than a camera that is connected through USB. This is very beneficial because it will decrease the latency between capturing images and inputting them through the computer vision model to analyze the images and make decisions.

The camera module 2 is priced at $29.95 MSRP and is connected to the Pi device through CSI (camera serial interface). The NVIDIA Jetson Nano has a CSI connector on the device, so this camera is an option for use with both the Raspberry Pi and Jetson devices. This camera is also notably cheap as many of the cameras officially recommended for the Jetson are priced similarly to the budget of this entire project.

7.3.2.2 Raspberry Pi High Quality Camera

The Raspberry Pi high quality camera is priced at $50 and does not contain a lens, which would need to be purchased separately starting at $25. It is capable of capturing images from 12.3 megapixels. Because the price is much steeper than the camera module 2 and extremely high quality imaging is not a necessity for gesture recognition, the Camera Module 2 should be chosen over this camera in the context of the Raspberry Pi computing devices.

7.3.2.3 NVIDIA Jetson Nano Developer Kit Cameras

Because the NVIDIA Jetson Nano is a Linux machine, cameras that connect via USB2/3 and are compatible with Linux are able to be connected and used as a source of images.

The Jetson also contains the ability to connect cameras in many different connections via CSI, MIPI, ethernet, FPD-Link III, GigE, GMSL, PoE GigE and V-by-One HS connection methods. While NVIDIA has explicitly mentioned specific camera hardware that is officially supported for the Jetson Architecture, many more computer cameras will be able to work with the device. Although the Jetson Nano is capable of using cameras with many interfaces, only CSI will be considered since it is efficient in data transmission and is popular for embedded vision.

7.3.2.4 Arducam IMX219 Visible Light Fixed Focus Camera

The Arducam IMX219 fixed focus camera uses the same sensor that the Raspberry Pi camera module 2 uses. It is also capable of high frame rates at lower resolutions (30 fps at 1080p, 120 fps at 720p) and is a rolling shutter camera. It is also a fixed focus camera, so objects that are close to the camera (<200 mm away) will be out of focus and the computer vision model will have trouble recognizing hands and gestures.

The Arducam IMX219 Visible Light Fixed Focus Camera is comparable to the Raspberry Pi camera module 2 in almost every way. Because it uses the same sensor and has the same communication interface, this can be expected. It was designed specifically for the NVIDIA Jetson Nano series and is not compatible with standalone Raspberry Pis but is compatible for the Raspberry Pi Compute models. One notable difference between this camera and the Raspberry Pi camera module 2 is price: $23. This is 7 dollars cheaper than the camera module 2 with comparable stats. Therefore this camera should be chosen over the Raspberry Pi camera for use with the Jetson Nano.

7.3.2.5 Arducam IMX219-AF Programmable/Auto Focus Camera

The Arducam IMX219-AF Programmable/Auto Focus Camera is comparable to the similar fixed focus camera but is programmable to adjust the focusing. This would allow the use of reading gestures from hands that are very close to the camera. This is a desired function, but it results in a higher price at $39.99. Since the price is much more and it is not likely that gestures will be performed closer than 200 mm to the camera, the fixed focus version is a more budget conscious decision.

7.3.2.6 Arducam IMX219 Low Distortion M12 Mount Camera

The Arducam IMX219 Low Distortion M12 Mount Camera has the same camera sensor as the previous two cameras discussed and behaves similarly, but contains a lens that allows for very minimal distortion. It does lose approximately 2 degrees from the field of vision (FOV) from the lens however. It is priced at $31.99, about $10 more than the fixed focus camera and $7.99 cheaper than the programmable variable focus camera. If distortion is an issue with gesture detection, this camera should be considered.

7.4 Microcontroller Programming Software

Since the ATmega328p device was chosen as the microcontroller for this project, the arduino bootloader was burned onto the device. The Arduino series of devices uses a modified version of C++ for microcontroller behavior and a dedicated development environment called Arduino IDE. The IDE allows the programmer to implement libraries and functions available for programming. Along with built-in libraries and functions, it is also possible to upload external or self-made code to the ATmega328 through the Arduino IDE. Such is the case in this project. By including and flashing the Servo.h library to the ATmega328, sending PWM signals to the servos to move them is very straightforward. When this library is appended to the ATmega328, the microcontroller waits for serial messages from the Jetson Nano to control the motors based on the gestures implemented.

7.5 Primary Programming Language

Machine learning models in general can be trained using many different programming languages and libraries available on all kinds of development environments. These libraries exist for many different programming languages such as Python, C/C++, Java, MATLAB, R, JavaScript, and more. Additionally, some of the different environments you can find machine learning libraries for are desktop operating systems such as Windows, MacOS, and different Linux distributions. More environments available are mobile phones including iOS and Android.

Python is an excellent candidate for programming a computer vision model for this project as the course algorithms for machine learning here at UCF employs the use of Python for training a multitude of model types using many different libraries. Additionally, Python is a very popular language for learning data science and machine learning.

7.6 Computer Vision Software

The software that is used to allow streamlining of the images from the camera into a python script are outlined here. A few options with their pros and cons are discussed. Any computer vision software must perform image processing functions which the mentioned libraries are capable of. The goal of this software does not encapsulate the whole functionality of the computer vision tasks. Another set of software must also be used in conjunction in order to reach our goal of gesture recognition, which is outlined in the next section.

The main goals of the computer vision software that must be met are allowing images from the camera to be accessible in a python script, perform image segmentation for object detection, and be fast enough to perform object detection in real time. A mixture of

OpenCV (for image manipulation) and Jetcam (for image gathering) is used in this project.

7.6.1 OpenCV

OpenCV is an open source computer vision library that is aimed at the use of computer vision in real time. Since our project plans to implement real-time gesture control, this software is a solid candidate. It includes thousands of machine learning and computer vision functions that allow the training and use of a model to perform tasks such as object detection.

OpenCV is extremely popular within the computer vision community for image functions for a variety of computer vision tasks. According to OpenCV's about page on their official website, they advertise over 47,000 community members, over 18 million downloads and the fact that the library is used in many different commercial, research, and government applications.

OpenCV was originally developed in the C++ programming language and has interfaces for programming in Python, Java, and MATLAB available for the Windows, Mac OS and Linux operating systems. Because Linux is the operating system that the computer vision hardware will be running, OpenCV is available for use.

OpenCV contains over 500 algorithms to train and assist the model with even more functions that support these algorithms in testing and using computer vision models. OpenCV offers official courses in using the library to perform computer vision tasks, but because OpenCV is so popular in developing computer vision models there exists many free user made tutorials for using the library. Additionally, the official website for OpenCV includes documentation for the library that outlines the behavior of available algorithms and functions for use.

7.6.2 SimpleCV

SimpleCV is a computer vision library that prides itself on being used for "computer vision made easy". It is essentially a compilation of multiple libraries made for training computer vision models in the Python programming language. Since Python is a likely language to be used in developing the computer vision model, SimpleCV is a valid choice. Because the topic of training deep learning models in general includes a lot of concepts such as matrix and linear algebra, fundamentals of machine learning algorithms, and more that require a decent amount of studying, SimpleCV is an option for individuals interested in hitting the ground running with computer vision.

SimpleCV's website sports installation guides and tutorials in using the library to perform computer vision tasks. However, since SimpleCV is not exactly one of the most popular libraries in computer vision, dealing with issues and performing specific tasks may prove to be harder than it would be in other libraries. This is due to the limited amount of community made tutorials and documentation, given that it is not as widely used as the traditional computer vision libraries.

7.6.3 Scikit-image

Scikit-image is a Python library that contains functionalities for image processing. It contains hundreds of functions that allow a large amount of operations to be done on images. One downside of using this library is that it is less geared for real-time applications and more for general image processing. This will lead to a larger amount of research and work required to get the system up and running. It can however be used in conjunction with some other deep neural network library to achieve image detection. It may also prove to be a useful tool in preparing the images to be fed into the computer vision model.

7.6.4 Jetcam

Jetcam is a library for accessing the camera inside of python scripts as an object. It features support for both CSI and USB cameras, and is compatible with the functions used in OpenCV. As such, OpenCV is used in this project for image manipulation and the capture of images for the machine learning software is performed by Jetcam.

Jetcam allows a python programmer to get started with the camera connected to the Jetson Nano in as little as 3 python lines of code. Importing the library, defining the camera, and capturing an image.

7.7 Machine Learning Software

Along with the software required for modifying and feeding images from the camera into a python script, we must separately use some machine learning to train and run a model in order to effectively reach gesture recognition. The goal of the machine learning software is to feed in the images from the computer vision software as a Python variable in order to first train the model then predict one of the set gestures as a classification of a specific gesture. TensorFlow, Keras, and Pytorch will be the considered Python libraries that will complete this task.

The chosen library that will be implemented in the Articulight project is Pytorch. It is chosen because it allows the advertised easiest and streamlined programming and training of a Deep Neural Network, in the scope of this project, for classifying images of hand gestures. It is used in the project trt_pose_hand, which is developed by NVIDIA for the Jetson Nano platform expanded and modified for this project. At the time of writing the research section, the plan was to develop a model for hand recognition from the ground up. However, as this project was built in a summer semester and we only had one person assigned for the computer vision and machine learning portions of this project, the choice of using a pre-trained model developed by NVIDIA for this task was made. Trt_pose_hand is outlined in its own subsection below.

Classification is the heart of the problem that we face in developing the machine learning model to recognize hand gestures. One of the most common models for developing this type of model is a Convolutional Neural Network (CNN). CNNs run on the same principle as barebone neural networks (such as the one shown in Figure 4) but implement more layers and data manipulation in order to get increasingly complex feature extraction from an image (such as the features on a hand). Explaining exactly how CNNs work is outside the scope of this document, but some of the concepts in CNNs will be explained here.

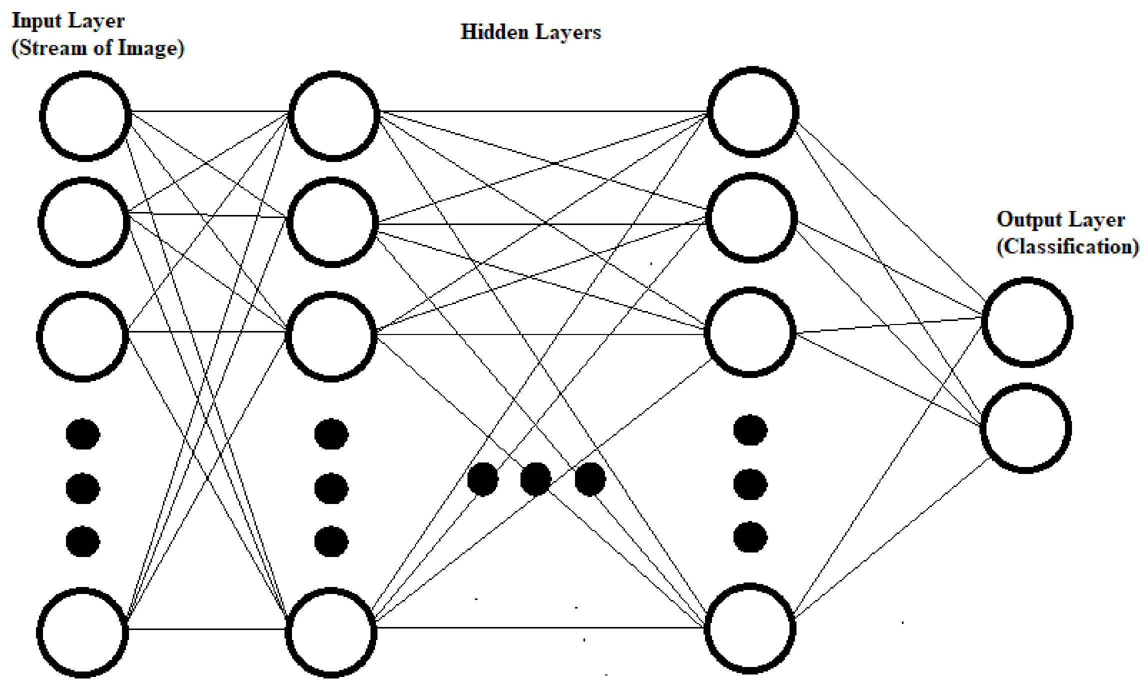


Figure 4: An Arbitrary Neural Network with Two Classification Neurons

Convolution is one of the aspects of a CNN, hence the convolution in the neural network. The purpose of convolution in this regard is to extract the high level features of an image such as the edges of an object in the image. Implementation of multiple convolution layers results in more complex edges and features in the image to be learned in the network. With more and more of these layers it is possible to detect objects with unique features such as the hand gestures we wish to recognize in this project.

Another technique implemented in CNNs is the pooling layers. Pooling layers take the result of the convolution layers and reduce the amount of data via dimensionality reduction in order to reduce the computation needed to run the network.

Lastly, the fully connected layer is implemented. The fully connected layer can be thought of as the simple neural network also known as a feedforward network. This layer takes the outputs of the convolution and pooling layers and applies the classification from the features that were extracted.

The following outlined software libraries all contain the ability to implement CNNs and are relevant to the goal of this project when it comes to training a model.

Another type of model that was implemented is a support vector machine (SVM). Support vector machines are machine learning models that employ supervised learning to perform classification. In a nutshell, it finds the hyperplane between the training data points so that the data points are as far away from the hyperplane as possible. By using NVIDIA's model for hand recognition, we can feed the outputs of the model into an SVM to effectively achieve gesture recognition.

7.7.1 TensorFlow

TensorFlow is one of the two machine learning libraries (the other being PyTorch) that has official releases specifically for the Jetson Nano. Because there is an official release, the installation and potentially the performance of the modeling, testing, and fitting algorithms may be higher than an unsupported version.

TensorFlow is a machine learning platform originally built by Google Brain. Since then, it has expanded into an open source project that has APIs for many programming languages and has many different builds including TensorFlow Lite, TensorFlow.js, and TensorFlow Extended. Although TensorFlow is traditionally run through Python code, it is very efficient in the computation required to train and run machine learning models because it computes using C++ and CUDA. Because NVIDIA developed both CUDA and the Jetson Nano device we intend to use, this allows the computation from TensorFlow to be streamlined and efficient.

TensorFlow is a library that focuses on the creation and implementation of Deep Neural Networks (DNNs). DNNs are one of the most common solutions to the implementation of image classification, a problem that is well relevant to the computer vision tasks we wish to perform. The image from the camera will be captured and formatted appropriately for use by OpenCV and the data after the appropriate transformations will be fed into TensorFlow to develop and use the DNN for image classification.

TensorFlow allows the programmer to have access to both high and low level APIs. High level APIs are to be considered more 'user friendly' than low level APIs in this context. Low level APIs tend to have a lot more intricacy in the settings available as parameters and require the programmer to be more involved in using them. High level APIs are simplified to allow the programmer to write less code in order to accomplish what they want to.

Because TensorFlow gives the programmer access to the aforementioned low level APIs, the efficiency of the built model will have the ability to be considerably high when compared to a model that is trained using only high level APIs. There is the tradeoff of using low level APIs that makes the programming process considerably more convoluted when compared to using a library with high level APIs such as Keras. Keras is outlined in the next section and is actually a part of the TensorFlow library one level down from TensorFlow itself.

7.7.2 Keras

Keras is a python API that is included in TensorFlow. It is considered a lot easier to pick up and start developing as it was developed with the goal of fast experimentation in TensorFlow DNNs. Although Keras is considered to be simple, the Keras website states that it is not a simplistic API and will allow the development of efficient DNNs. According to the Keras website, Keras is powerful enough that it is at the industry level standard and companies such as NASA and YouTube use it. Keras is available in the TensorFlow Python library as TensorFlow.keras.

Keras allows the programmer to develop differing complexity of deep learning tools. The most simple of these tools is the Sequential class that creates a linear stack of layers. The programmer can also utilize the Keras Functional API to build graphs of any amount of layers or use subclassing to to write DNNs from the ground up. It also has many more functionalities implemented to make the machine learning process easier.

7.7.3 PyTorch

PyTorch is similar to TensorFlow in the regard that it utilizes CPU and GPU architecture to perform mathematical operations using tensors. A tensor is a data structure that is similar to NumPy, which allows precise mathematics to be computed. It is also similar to TensorFlow in the fact that it is an open source project and contains a large community of developers and users. These tensors are then able to form deep learning structures through the various APIs that are available in the PyTorch library.

PyTorch contains the following components: torch, torch.autograd, torch.jit, torch.nn, torch,multiprocessing, and torch.utils. torch is a tensor library that is the basis of the library as a whole. It allows the instantiation of tensors that use GPU processing to achieve greater efficiency. torch.autograd is a library that has functionalities that perform differentiation. torch.jit is a compilation stack that allows the creation of models from PyTorch. torch.nn covers the neural networks that use tensors and is used in conjunction with autograd. torch.multiprocessing allows Python multiprocessing that is fine tuned to raise the efficiency of PyTorch code. Lasty, torch.utils covers the utility functions that may be used with the other libraries that were mentioned.

PyTorch is becoming very popular in the machine learning research community as it is, according to their github repository, very fast and contains minimal framework overhead. This allows the researcher to have a streamlined process in developing machine learning models that fit the needs for their problem or research.

7.7.4 Resnet-18-hand

Resnet-18 is an 18 layer deep convolutional neural network. It was developed by a group of individuals through PyTorch, and is commonly used in the NVIDIA AI fundamentals on the Jetson Nano for computer vision projects. Resnet-18-hand is a retrained version of resnet-18 that was retrained via transfer learning for outputting key points of the hand (parts of the fingers, palm, etc) on a given frame. This model is used in conjunction with pytorch, and allows us to spend less time building a hand recognition model from scratch, and focus on getting the system to work properly. The input of this model is a 224x224 image and it outputs a list of keypoints with their respective location in the frame. Gathering the outputs of this model when a gesture is held allows us to train a support vector machine that analyzes the key point patterns in these gestures for gesture classification. More information about SVMs in the subsection below this section.

7.7.4.1 SVMs & Trt_pose_hand Project

Trt_pose_hand is a project that was developed by an NVIDIA employee for using the resnet-18-hand model to perform gesture recognition. It consists of a set of Jupyter notebook files that use a USB webcam for grabbing the images. Because this project is supposed to be a standalone embedded project, utilizing Jupyter Notebooks would not be a good embedded solution. So, in this project, it was required to translate the code from NVIDIA dealing with the resnet-18-hand model into python scripts.

Trt_pose_hand also makes use of a support vector machine to perform gesture classification on a set of gestures that was defined for the purpose of gesture controlled interaction with a computer and a mini-paint program to draw with your hands. Using the predefined gestures in this project was an option, but it was more substantial and effective to train our own SVM and develop the software required to gather training images.

To create and train an SVM, scikit-learn was used. Scikit-learn is a python package that allows simple machine learning models to be instantiated, trained, and used programmatically. Using the pickle package, the models created in one file can be loaded in another for use by saving and loading the model after it is initially trained. The outputs of the hand model are compiled into a list and fed into the SVM by using the fit function. Then, supplying a new output to the SVM will output a number mapped to a gesture.

7.8 Light Socket

For any type of light fixture, whether that is a lamp, ceiling fan, chandelier, etc,. They all require a light socket. A light socket is a key aspect for our design as the specification of the light socket plays a role in terms of the bulb we will select. Not only that, but the light socket that is chosen has specifications on how much wattage and voltage it can handle to flow to the bulb. It is also important to note that light sockets can be extremely dangerous if not handled correctly. Sometimes, a light socket can transmit more current than it was designed to handle, it may generate enough heat to melt the components inside the socket. This can lead to issues in the design if this happens.

7.8.1 Specification of Light Sockets

Specification/ Light	**E-12**	**E-19**	**E-26**	**E-39**
Power (W)	75	60	60	660
Voltage (V)	125	125	250	240
Luminosity (Lumens)	400	840	800	36000
Size (mm- Diameter)	12	19	26	39
Weight (oz)	6.4	2.04	0.564	8.2
Price	$9.44 (4 pack)	$13.95	$7.99 (4 pack)	$15.53

Table 7: Specifications of Light Sockets

7.8.2 Light Socket Options

There are multiple light sockets in the market today. The Motion Controlled Light System team's first choice was an E-26 (Medium or Standard socket) type light socket due to the fact that this particular light socket is used for most lamps (Medium bulbs 26mm), which is our design. The other options came from other common household light sockets such as E-12 (Candelabra socket), which uses the smallest type of bulbs (12mm), E-19 (Intermediate socket), which uses the next size of bulbs (19mm), and E-39 (Mogul socket), which uses the largest type bulbs (39mm). Research was done between all four types of sockets and the different specifications each one has such as wattage, voltage, size of bulb it can hold, price, and weight.

7.8.2.1 E-12 Light Socket

This model socket was chosen and researched as it is one of the most common types of sockets seen in everyday households, mainly chandeliers. This particular socket comes in a pack of 4 for a total price of $9.44.

The E-12 socket has a power wattage of 75 W with a voltage of 125 V. Using an E-12 bulb with this socket will allow us to have a maximum luminosity of 400. Despite being the smallest size socket in the market it weighs the most out of all four sockets with a weight of 6.4 ounces.

7.8.2.2 E-19 Light Socket

This model socket was researched as it is one of the types used in lamps. The cost of this socket is $13.95 per unit. It is one of the more expensive type sockets as you will see further on. Despite the cost of the E-19 socket, it has better specifications than the E-12 socket. The E-19 socket has a power wattage of 60 W and a voltage of 125 V. Using an E-19 bulb with this socket will allow us to have a maximum luminosity of 840. It has the 2nd smallest weight of 2.04 ounces.

7.8.2.3 E-26 Light Socket

The E-26 light socket is the most common household bulb used for all electronic appliances, hence why it was chosen as a team's first choice. The cost of this socket is $7.99 for a pack of 4. This is the cheapest socket out of the four types that we researched.

The E-26 light socket is widely used everywhere because of its top of the line specifications and because of its cost efficiency. It has a wattage of 60 W and a voltage of 250 V. It has a maximum luminosity of 800. It is the lightest type of socket available with a weight of 0.564 ounces.

7.8.2.4 E-39 Light Socket

This particular light socket model is mainly used to illuminate large facilities such as warehouses, factories, and arenas. This is the most expensive type of socket at a price of $15.53 per unit. The reason we didn't choose this particular socket was because of how bright of a light it produces. The team felt we didn't need that bright of a light as it may blind the camera that will be used for the computer vision.

Furthermore, this socket has a wattage of 660 W and a voltage of 240 V. The maximum luminosity of this socket with an E-39 bulb is 36000. It is the heaviest socket with a weight of 8.2 ounces.

As shown in table 7, the E-26 light socket is the best in terms of weight, cost, and the amount of lumens we are able to receive. It is about 8 ounces less than the E-39 socket, about 6 ounces less than the E-12 socket, and about 1.5 ounces less than the E-19 socket making it the lightest of all the types of sockets. Between the E-19 and E-26 socket, the justification why the team chose the E-26 was because it was more cost effective, despite the lumens only having a difference of 40.

7.9 Light (Bulb)

In today's world, there are many different types of bulbs used in society. The bulbs that we chose to focus on for our team's design were incandescent, halogen, fluorescent, and

LED. Each type of bulb has its own characteristic specs in terms of life, cost,and energy consumption. We have chosen to use LED bulbs for the project. We chose LED bulbs because they consume less energy while producing the same amount of light output as a traditional incandescent bulb. The bulbs are significantly safer to use because the overall material is just made out of plastic and when dropped they won't shatter like an incandescent bulb and won't spill mercury like a compact fluorescent bulb when broken. When it comes to heat, the LED bulb has a heatsink to dissipate the heat usually around the base of the bulb but wont get as nearly as hot as an incandescent bulb which will burn if you touch it anywhere. All these factors make the LED the right choice for our project. From **Table 8** we will be comparing the pros and cons of the most common type of bulbs in the market.

Types of bulbs	**Pros**	**Cons**
Incandescent	● Cost efficient ● Produces high light output ● Can be dimmed using rheostats	● Lifespan of 1000 hours ● 5 to 20 Lumen per watt
Halogen	● Uses halogen gas ● Compact size ● Can be set to turn on/off via timer	● Can get extremely hot after long period of usage ● Possibility of bulb exploding due to high increase of temperature
Fluorescent	● Energy efficient ● Cost savings ● Long light life	● Contains mercury ● Higher initial cost ● Limitations such as noise, flickering light, and not all bulbs are dimmable
LED: Light Emitting Diode	● Energy efficient ● Bulb generally stays cool ● Lifespan of 50000 hours	● Expensive cost ● Consequences of blue light

Table 8: Pro and Con Table of Different Bulbs

7.9.1 Light (Bulb) Options

For our design, the team decided to look at four different types of bulbs to compare their benefits as well as their drawbacks. As listed above, they consist of incandescent, halogen, fluorescent and LED. Research was done between the four types of bulbs and we will be looking at the specifications of each type of bulb.

7.9.1.1 Incandescent Bulb

An incandescent bulb is an electric light with a wire filament heated until it glows. The great thing about this bulb is that it is cost effective and produces a high light output. These particular bulbs can also be dimmed by using rheostats.

It produces light by heating a filament inside a vacuum bulb filled with an inert gas preventing the filament from burning out due to oxygen. It's basically a fire burning inside the bulb. With fire comes heat and an incandescent bulb produces a lot of heat.

Incandescent bulbs have a short lifespan somewhere between 800 to 1000 hours compared to LED bulbs which last about 50,000 hours. Incandescent bulbs are being phased out because there is no real advantage in using these types of bulbs. They are not energy efficient because they waste a lot of energy. It wastes about 90% of energy in the form of heat.

7.9.1.2 Halogen Bulb

Halogen bulbs are similar to that of an incandescent bulb. In order to increase the life and light output of the bulb Halogen gas is used. One of the best things about this type of bulb is that it is inexpensive (only $4-10 per bulb), produces bright light, has compact size, and can be set to turn on/off through a timer. A halogen bulb can produce a brighter lighter because unlike incandescent light,a halogen light burns off the tungsten filament particle but the particle then gets redeposited back into the filament allowing the particle to be reused or burned up again thus producing a brighter light and longer life. Life expectancy of a halogen bulb is about 2500 hours.

Some advantages to using halogen bulbs are that they have better quality of light output than incandescent lights. They are compact in size and have the ability to be dimmable. They have become relatively inexpensive, but with new and modern bulbs on the market there is no real advantage in using these types of bulbs anymore.

The disadvantage about this bulb is the safety hazards with it. The justification why it is a safety hazard is because these bulbs tend to get extremely hot when operated for long periods of times. Furthermore, due to the high increase in temperature of the bulb, there is a possibility the bulb can explode since the glass in the bulb is also in such pressure that an explosion can send glass outward. Life expectancy can be shortened just by touching the bulb with your hand. The oil in your skin can react to the bulb giving it a hotter spot in the area touched thus reducing the life expectancy. Unless you're looking for a heat lamp the amount of heat generated by a halogen bulb can make someone uncomfortable being around or near the bulb.

7.9.1.3 Fluorescent Bulb

Fluorescent bulbs are another type of light that was created to be more energy efficient with less heat than an incandescent bulb. How it works is that you have a glass tube and at each end you have an electrode called cathodes. The electrode has filament inside the electrode that will send a current through the gas inside the lamp. The lamp is filled with an inert gas commonly argon and a little bit of mercury. Basically when current is sent through the lamp it creates an arch sending UV light down the tube. The glass tube is coated with phosphor and when the UV light hits the phosphor coating which makes the light glow.

Fluorescent bulb uses about 75% less energy than a traditional incandescent bulb. A 20 watt Fluorescent bulb would equal a 75 watt incandescent bulb. Fluorescent lighting does not produce a lot of heat which keeps the room cooler. They typically last about 10,000 all the way to 50,000 hours

Disadvantage to using Fluorescent lighting is it contains mercury. Even though the bulbs contain a very small amount of mercury, about 4 milligram, it can still pose a health hazard to people. Another drawback is that in colder climates, Fluorescent lighting doesn't perform as well in colder weather. You run the risk of the light not turning on or might come on but wont be as bright or even take a very long time for the light to warm up before full brightness can occur.

Furthermore, another disadvantage of fluorescent light is the limitations it has. One of these limitations is that not all fluorescent bulbs are dimmable. Another limitation is a buzzing sound that occurs when the light is on and active. However, the buzzing noise constraint can be overcomed by using an electronic ballast. The last, but not least limitation of the fluorescent bulb is flickering light. This constraint or limitation is still in the process of being solved using advanced technology that is available in today’s society as well as better ballasts.

7.9.1.4 LED Bulb

The Light-Emitting Diode (LED) is the most energy efficient bulb in the market today. it's a semiconductor device, when current passes through the semiconductor it emits a light. How a LED works is that electrons and holes are in an energy band. The bandgap determines the light particles that are emitted by the LED. Different materials such as Indium gallium nitride (InGaN) can produce a blue, green or UV light and Aluminum gallium arsenide (AlGaAs) can produce colors such as red and infrared lights.

It is able to produce light 90% more efficiently than incandescent bulbs which decreases the power cost. Some benefits of this bulb is that it has a low safety risk, so if someone touches the bulb while it's on it won’t burn as the light stays generally cooler. It has a lifespan of 50000 hours, and although it is one of the most expensive types of bulbs, it pays for itself long term as it doesn’t waste heat energy. Another advantage of using LED bulbs is the wide variety of colors to choose from. There are literally thousands of color options that you can pick. LEDs are quick when you flip the switch you get instant lighting. you don't need to wait until it warms up for the correct temperature of the lighting to be correct. Temperature has no effect on LED bulbs and it's instant and quick.

As well as having multiple color options to choose from, there are five main types of LED lights. These include miniature, dimmable, high power, lighting, and flashing LEDs. We will cover each type of LED light in the table below, after discussing the disadvantages of LED bulbs.

Moreover, some of the downfall of using LED light is the consequence of blue light, which can affect vision problems later in the future. However, in today’s society, there are many different devices that help protect our eyes from the cause of blue light. compared to incandescent bulbs, LED bulbs have a higher upfront cost. Usually LED bulbs emit light in one direction and not a 360 degree all around or spherical direction, which could pose a problem when trying to light up a whole room but leaves dark areas where the light could not emit the glow.

Types of LED bulb	**Specification of bulb**
Miniature	Extremely small, and come in a single color or shape; Usually used in remote controls, calculators, and mobile phones.
Dimmable	Create a dimming effect; Used for creating unique settings such as a romantic bedroom setting.
High Power	Deliver higher levels of brightness.
Lighting	LED bulbs that are normally found in households and often take the shape of the traditional Edison bulb.
Flashing	These are normally used to attract attention; Used for storefront signage, displays at exhibition, and automobile indicators.

Table 9: Types of LED Bulbs

7.10 Motors

Motors come in many different sizes and shapes and power ratings. There are three common types of motors: DC motors, stepper motors and servo motors. Each motor is unique to their own application, and each has their advantage and limitation, but all convert electrical energy into mechanical movements. Table 10 shows the pros and cons to each type of motors. We will be looking at more of these specifications in depth in the table, as well as the headings that follow.

Motor Types	**Pros**	**Cons**
DC Motor Brushed	● Simple to control ● Excellent torque at low RPM ● Inexpensive	● Brush can wear out over time ● Brush can arc and generate EMF noise
DC Motor Brushless	● Reliable ● High speed ● Efficient	● Requires low starting load ● Typically requires specialized gearbox to drive application
Stepper Motor	● Excellent position accuracy ● High holding torque ● High reliability ● Very simple to program usually plug and play ● Have higher torque at low speed	● Limited in speed ● Possibility to miss a step during load ● Constantly drawing current
Servo Motor	● Highly precise ● Can be precisely controlled ● Gives position feedback to location ● Has a higher torque at high speed	● Limited in movements usually 0 – 180 degrees ● Requires tuning for feedback loop ● Most complex than the other motors ● Could easily be damage by overloading mechanically

Table 10 : Types of Motors

7.10.1 DC Motor

There are two types of DC motor, brushed DC motor and a brushless DC motor. The working principles between the two are essentially the same. DC motors or direct current motors are very basic motors, easy to use. A voltage is applied to the terminals thus spinning the motor.

Depending on the polarity it can either go forward or reverse. Controlling the speed of the DC motor is as simple as varying the input voltage, the higher the voltage the faster it will spin the lower the voltage the slower the speed. Torque and load vary depending on the motor size, but the DC motor has a higher torque at low speed.

7.10.2 Brushed DC Motor

Brushed DC motor has three key components, the stator, the armature, and the commutator.

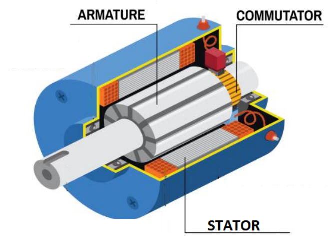


Figure 5: DC Brushed Motor

The stator is the part inside the motor frame that doesn't move (magnets) and the armature is the rotating part of the motor. The permanent magnet has a north and south pole and when voltage is induced the armature will spin until opposite poles line up. For the motor to keep spinning the voltage needs to switch polarity so the magnets can switch poles and keep rotating. To switch the polarity, it uses a commutator.

A commutator is a ring that has air gaps, the air gaps cause the armature to switch polarity of the armature causing the magnet to switch poles and in turn keeps the motor spinning in one constant direction either forward or reverse. These motors are relatively inexpensive and very basic. These motors are used for continuous rotation application like a toy car or power tool.

DC brushed motors operate by sending current through the brushes to the motor windings which turns the motor. The advantage of a brushed motor is that it is simple to use, does not require any controller to run the motor and relatively inexpensive. The disadvantage of using a brushed motor is the arcs that it generates in the motor creates electromagnetic noise in the circuit. Life expectancy is short due to the contact wearing out and the speed range is not as high as a brushless motor due to the brushes on the motor.

7.10.3 Brushless DC Motor

DC brushless motors have no brushes just like the name implies. Unlike a brushed motor which delivers current through the commutator into the coil and then into the rotor causing the motor to move, a brushless motor uses permanent magnets to rotate the rotor. Unlike a brushed motor where the winding was on the rotating rotor, the winding on a brushless motor is stationary and does not move. Brushless motor operates by controlling the magnetic field in the stationary windings moving the rotors causing the rotation.

Brushless motors in comparison to brushed motors are more complicated. You cannot simply connect the positive and negative leads and hope for it to turn. A controller is needed to control the timing of the current to the coil to rotate the stator. The controller energizes one of the permanent windings to start the rotation. The rotation is maintained by constantly chasing the rotating magnet.

The advantage of brushless DC motors is that it's more efficient, has a longer life span and higher torque. It has a higher RPM speed since it does not have brushes. It generates low electrical noise compared to brushed motors.
Disadvantage of a brushless DC motor is that it’s a little pricey and more complex to control.

7.10.4 Stepper Motor

There are generally three common types of stepper motor, A Permanent Magnet type, a Variable Reluctance Stepper motor and Hybrid stepper motor but how they work are generally the same. The characteristic of a stepper motor is that the rotation is not continuous like a DC motor, it rotates in steps.

A stepper motor is attached to a motor driver. A motor driver is nothing more than an electronic switch turning on and off at high-speed. As electricity gets sent in pulses to the stepper motor, each pulse that is applied causes the motor to rotate one step. The number of steps will determine the angle at which the stepper motor will make with the normal plane when considering the project. This would be a simple matter of mapping steps to reference angles in order to allow movement with accurate precision. However, the precision is strictly dependant on the number of steps.

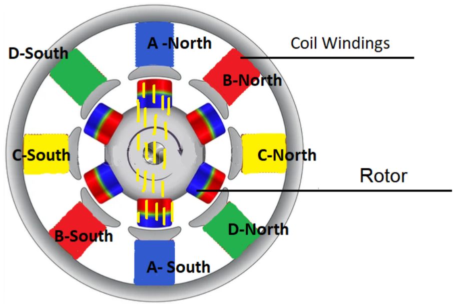


Figure 6: Variable Reluctance Stepper Motor

In **Figure 6**, it shows how the Variable Reluctant (VR) stepper motor movement occurs. Depending on the type and size of the motor will depend on how many stator poles and how many rotor teeth will depend on the traveling angle of the stepper motor. The less it has the more the motor has to travel for one step and the more it has the less the motor travels and this is important when it comes to accuracy. A step occurs when one pair of stators is energized. Only one stator is energized at a time, so when A is energized it creates a magnetic flux line through the rotor and the stator lining up the two pairs and only one pair of rotor teeth would line up while the other are not. When A is de-energized B will become energized and the same sequence occurs and so on.

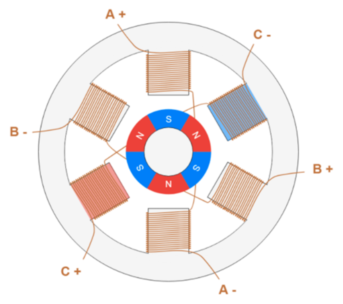


Figure 7: Permanent Magnet Stepper Motor

In **Figure 7** is an example of the inner workings of a permanent magnet stepper motor.

There is a slight variation on the construction between a variable reluctance stepper motor and the permanent stepper motor but how they work is about the same.

The only difference between the variable reluctance stepper motor and the permanent magnet stepper motor is that the variable reluctance stepper motor has no permanent magnets either on the stator or the rotor while the permanent magnet stepper motor has the magnet on the rotor. The rotor shaft itself is a smooth permanent magnet. Same principle as the VR motor, if coil A is energized then the rotor will align opposite poles, when it is de-energized and B is energized the rotor attracts the opposite poles and movement occurs.

The last stepper motor is a hybrid stepper motor because it’s between a permanent magnet stepper motor and variable reluctance stepper motor.

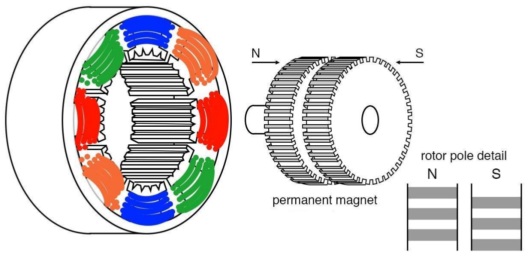


Figure 8: Hybrid Stepper Motor

In **Figure 8** you see that the rotors have two rotor caps. The teeth on the north and south rotor caps do not line up with each other but are alternating with each other. when a pair of coils is energized the tooth on the rotor cap will align with the opposite poles on the stator. Due to the number of teeth on the rotor, the hybrid stepper motor can move in very small increment steps thus resulting in high resolution, speed, and torque.

Just like DC motors, rotation can either be forward or reverse but with the advantage that it can stop and hold a certain position. The advantage of a stepper motor is that it offers excellent speed control while providing precise positioning; it has a constant holding torque without the need for the motor to be powered on. It has a greater torque than that of the same size servo motor and costs less than a servo motor. Stepper motors do not require position feedback to the controller therefore makes them easier to program.

Disadvantage of a servo motor is that its torque declines as speed increases. They draw a substantial amount of power which in turns into heat. Compared to a servo motor, a stepper motor will have a lower accuracy than that of a servo motor and are slower. Another drawback to not having positional feedback is that if the stepper motor misses a step there is no way the controller knows what it missed.

7.10.5 Servo Motor

A servo motor is fast and highly precise. They are a little more complicated than the two motors we mentioned before.

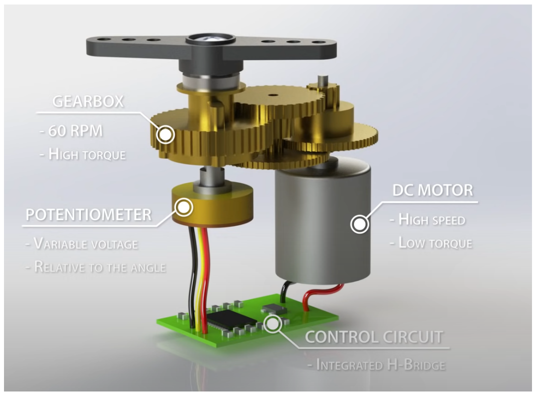


Figure 9: Servo Motor

Figure 9 shows the general breakdown of a servo motor. A servo motor consists of a DC motor, gear assembly, controller, and position sensor.

The DC motors are usually high-speed motors and to reduce the speed and increase the torque of the motor a set of gears connected to the dc motor. The position sensor is attached to the output shaft so when the motor rotates so does the positioning sensor as well creating a voltage of the absolute angle of the output shaft basically telling you what position you are currently.

Servo motors provide higher accuracy and precision in an angular or linear position. The sensor inside the servo provides position feedback. Has a constant torque across its speed range due to the gears inside the servo motor and are quicker than a stepper motor.

Most servo motors have a limited rotation anywhere between 180 degrees to 270 degrees. Programming a servo motor is more complex than that of a stepper motor. It requires tuning and positional feedback and has a higher cost than that of the other motors. When it comes to repeatability, accuracy and speed, servo motors have the advantage and obvious choice for this project.

7.11 Pulse Width Modulation Servo Control Driver

We decided that since we are using a servo motor for our design, that a pulse width modulation (PWM) signal type controller would be the best, as it will be able to send a signal when the user wants the light on or off. We decided to go with the Adafruit 16 - channel 12 - bit PWM/Servo driver - I2C interface - PCA9685. When we found this particular chip online we realized what an excellent add-on this would be for our design. Using only two pins, we can control 16 free - running PWM outputs. We can go as far as even chaining up to 62 breakouts to control up to 992 PWM outputs which would be insane, however glorious to see if applied. During our project we found we could use the Jetson Nano to control the servo motors. From testing, and learning how the Jetson can be implemented, we were able to save money on our design by not needing a PWM servo controller.

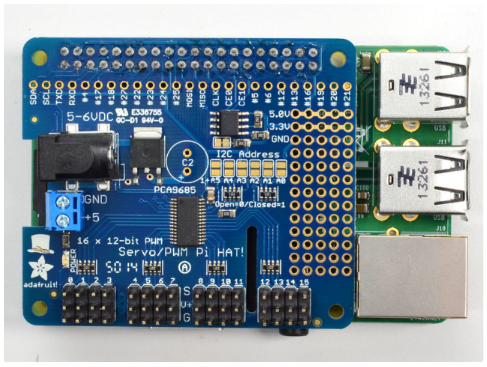


Figure 10: Adafruit 16-channel 12-bit PWM Servo Driver

This servo control driver has many benefits. A major benefit of the Adafruit PWM servo driver is that it can be used to make “chainable” designs. This means we can hook up multiple motors and have them all running at the same time. This is especially beneficial for our design since we will be using 2 to 4 servo motors to power the gears that will be used for the rotation of our device. Moreover, this PWM servo driver has 3 - pin connectors that are in groups of 4, which will give us the option to plug in up to 16 servos based on the size of the servo motors. Another beneficial reason we chose this specific servo driver is because it is 5V compliant. This means that if we used a 3.3V microcontroller, with a 3.3V logic pull-ups we can control this PWM servo driver through our Jetson Nano and still safely sink up to an output voltage of up to 5.5V. One of the last benefits why we decided to choose this servo controller is because it is an I2C - controlled PWM driver with a built-in clock. This clock can be useful for our design as if the clock is programmed correctly, that way we can set the servo drivers to activate at certain times of day when scheduled. These benefits and others will be shown in the table below:

Description of Adafruit 16-channel 12-bit PWM servo driver
It's an I2C-controlled PWM driver with a built-in clock
It is 5V compliant, which means you can control it from a 3.3V microcontroller, with 3.3V logic pull-ups and still safely sink up to 5.5V outputs
6 address select pins, so you can wire up to 62 of these on single I2C bus
Adjustable frequency PWM up to about 1.6 KHz
12-bit resolution for each output - for servos, that means 4us resolution at 60Hz
Configurable push-pull or open-drain output
Output enable pin to quickly disable all the outputs
Terminal block for power input (or can just use the 0.1" breakouts on the side)
Reverse polarity protection on the terminal block input
Green power-good LED
3 pin connectors in groups of 4, - you can plug in up to 16 servos at once (base on size)
"Chain-able" design
A spot to place a big capacitor on the V+ line
220 ohm series resistors on all the output lines to protect them
Solder jumpers for the 6 address select pins

Table 11: Description of PWM servo driver

7.11.1 Other Features of the Adafruit PWM Servo Driver

Another main reason why we decided to go with this specific PWM servo driver is because it comes fully tested and assembled breakout as well as 4 pieces of 3x4 male straight header (for servo/LED plugs), a 2 - pin terminal block (for power), and a piece of 6 - pin 0.1" header (to plug into a breadboard). Moreover, it only requires a small amount of soldering to assemble and customize the board by attaching the desired headers. This particular task takes about 15 minutes and even a beginner can do it. It even came with technical details, which include the dimension of the Adafruit PWM servo driver, the weight (with no headers or terminal blocks), the weight (with headers and terminal blocks), and the type of addresses used as shown in table 12 below:

Technical details of Adafruit PWM Servo Driver
Dimensions (no headers or terminal block) 2.5” x 1” x 0.1”
Weight (no headers or terminal block): 5.5 grams
Weight (with 3 x 4 headers & terminal block): 9 grams
This board/chip uses I2C 7-bit address between 0x40 - 0x7F, selectable with jumpers

Table 12: Technical specifications of PWM servo driver

7.12 Specifications for Microcontroller Relay

Specification/Relay	**Electromechanical Relay**	**Solid State Relay**
Operating Current	70mA	12.5mA
Output voltage DC	30VDC	Cannot drive DC OUTPUT
Output current	0.1A to 10A MAX	0.1A to 2A MAX
Single Channel Relay Price	$7 to $12	$9 to $15

Table 13: Specifications for Microcontroller Relay

Relays are nothing more than a device that sends a small electrical signal from a controller to open or close a circuit controlling something of a larger current. It’s basically a switch turning on and off. There are two common types of relays, electromechanical and solid-state relay. When testing our design we found that we didn’t need relays for our design. Instead, we used a full wave rectifier to help control the current in our project.

7.12.1 Electromechanical Relay

Electromechanical Relay is just a mechanical relay with moving parts inside. They come in many different sizes, shapes, and power ratings. The relay is either normally open or normally closed. **Figure 11** shows when low current energizes the coil on the low current side, it moves the armature closing the air gap thus completing the circuit.

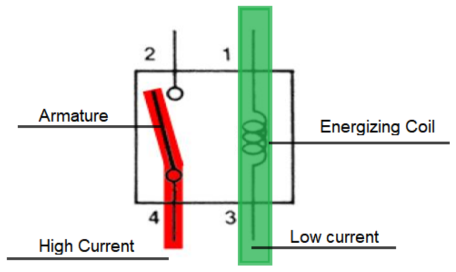


Figure 11: Mechanical Relay

The one advantage of a mechanical relay is that it can carry a large amount of current. The disadvantage to a mechanical relay is when you have moving parts making contact, Parts tend to wear out much quicker thus the life span gets shortened. They tend to be noisier when it's switching on and off.

7.12.2 Solid State Relay

Unlike mechanical relay which uses moving parts to do the switching a solid-state relay (SSR) uses semiconductor such as diode, transistor, thyristor, or triacs to do the switching, which means there are no moving parts to wear out. No moving parts means no contact and no sparks are generated and quiet. Without any moving parts the relay would last longer, can withstand shocks, and drops which is the obvious choice for our project.

The drawback to a solid-state relay was it has a higher cost, had a higher output resistance and will generate a large amount of heat so a fan may be required to dissipate the heat it generates.

For our project we did not require the use of relays. The original design was going to have our light activation controlled by a gesture to turn it on and off and to activate communication between the Jetson and ATMega328P. Neither of these were required for the Articulight system. We used a 120 Vac light that was tied to the outlet and the ATMega328P could poll the serial communication buffer for data which prevented any syncing issues due to the delay of turning on the relay.

7.13 Voltage Regulator

We had multiple discussions about whether a voltage regulator should be added as a design component. The primary job of a voltage regulator is to reduce the voltage to a specified range. It works as a shield for protective devices from damage. It can regulate both DC or AC voltages, depending on the implementation of the design. There are two types of voltage regulators. The first type is linear voltage regulators and the second type is switching voltage regulators. In **Figure 12** and table below, we would be looking at the ideal placement of the voltage regulator, as well as the comparisons of both the linear and switching voltage regulators. The comparison table will consist of the pros and cons of each type of regulator and how each of them act in a circuit and their design.

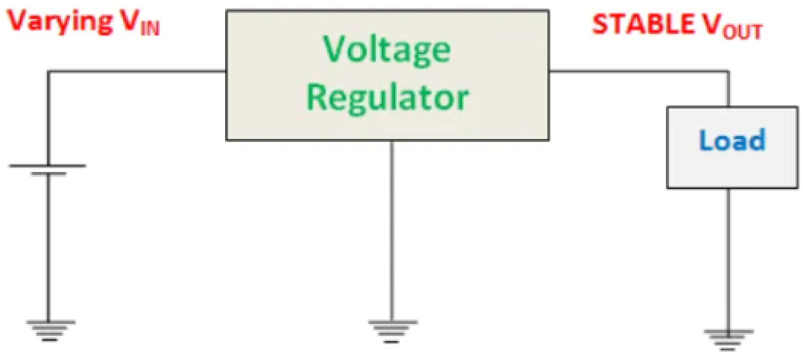


Figure 12: Ideal placement for Voltage Regulator

Types of regulators	**Pros**	**Cons**
Linear Voltage Regulator	● Gives a low output ripple voltage ● Fast response time to load or line changes ● Low electromagnetic interference (EMI) and less noise	● Efficiency is very low ● Requires large space – heatsink is needed ● Voltage is above the input cannot be increased
Switching Voltage Regulator	● More complex design, which allows for handling higher power efficiency ● Able to provide output, which can be greater than or less than or that inverts the input voltage	● Higher output ripple voltage ● Slower transient recovery time ● EMI produces very noisy output ● Very expensive

Table 14 : Types of Voltage Regulators

7.13.1 Linear Voltage Regulator – Series

There are two types of linear voltage regulators. They consist of Series and Shunt. A series voltage regulator uses a variable element that is placed in series with the load. When the resistance of that series element is adjusted, the voltage dropped across can be changed, while voltage across the load remains constant. These types of voltage regulators are mainly used with lower load currents because at higher current loads, it does not give good voltage regulation. There are two types of series regulators we will look at. They are Discrete and Zener Diode series voltage regulators.

7.13.2 Discrete Transistor Series Voltage Regulator

The discrete transistor series voltage regulator controls the input voltage magnitude and gives it to the output. The output is then fed-back through the circuit. Moreover, it is then sampled by the sampling circuit and is given to the comparator. Following that, it is then compared by the reference voltage and given back to the output. The block diagram below shows how a discrete transistor voltage regulator works.

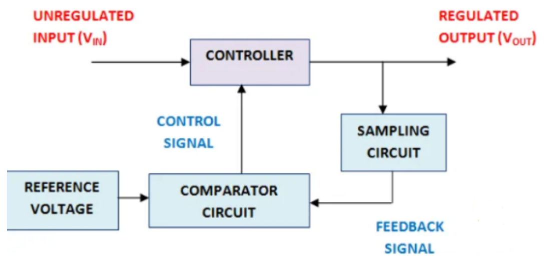


Figure 13: Block Diagram of Discrete Transistor Regulator

7.13.3 Zener Diode

A zener diode is a silicon P/N junction device that allows current to flow in both the forward and reverse direction. A zener diode allows current to flow in the reverse direction, if the voltage is greater than the breakdown voltage (also known as the zener voltage). Due to these aspects of the zener diode it makes it act as a voltage regulator device and is referred to as a zener controlled transistor series voltage regulator. As shown in **Figure 14** below, it shows how the transistor is used as an emitter. The emitter and the collector terminals of the series pass transistors used here that are in series with the load. In this particular built circuit design, the variable element is a transistor and the zener diode supplies the reference voltage. The output voltage is determined by the formula $V_{Out} = V_{Zener} + V_{BE}$ where, V_{Zener} is voltage of zener diode and V_{BE} is voltage that falls between the base and emitter of a bipolar junction transistor. V_{BE} is approximately 0.7 V, since a zener diode is a silicon transistor.

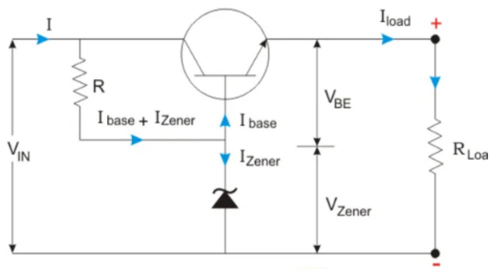


Figure 14: Circuit Diagram of Zener Diode

7.13.4 Linear Voltage Regulator – Shunt

A shunt voltage regulator functions by creating a path from the supply voltage to the ground through a variable resistance. This type of regulator is less efficient than a series regulator because the current in the shunt regulator is diverted away from the load and flows uselessly to the ground. Moreover, it is also connected in parallel with the load. However, shunt regulators are much simpler as they consist of just a voltage reference diode. Furthermore, it allows good voltage regulation at high load currents. In the table 15 below we compare these two types of voltage regulators more in depth.

Comparison number	**Series Voltage Regulator**	**Shunt Voltage Regulator**
1	Regulator is connected in series with the load	Regulator is connected in parallel with the load
2	At high load currents, it does not give good voltage regulation	It gives good voltage regulation at high load currents
3	DC output voltages are not constant	Output voltage is constant
4	Has good efficiency for higher load currents	Comparatively has less efficiency
5	The regulator is appropriate for heavy loads	Appropriate for light loads

Table 15 : Comparison Between Series and Shunt Voltage Regulators

7.13.5 Switching Voltage Regulator

Types of switching regulators	**Pros**	**Cons**
Dielectric isolation	● Safety compliance ● Breaking of ground loops ● Floating outputs and level shifting	● Expensive ● low efficiency ● larger size
Non-isolation	● Cost efficient ● Smaller size, more compact ● Efficiency	● Higher output ripple voltage ● Slower transient recovery time ● EMI produces noisy output

Table 16: Topologies of Switching Voltage Regulators

A switching voltage regulator switches a series device on and off rapidly. The duty cycle of the switch sets the amount of charge transferred to the load. It is controlled through a feedback mechanism similar to that of a linear voltage regulator. Switching regulators are efficient because the series element is either fully conducting or switched off it dissipates nearly no power. Switching regulators has two types of topology. They are Dielectric isolation and Non-isolation. After reading about the two types of topology for a switching voltage regulator, you will see table 16 that consists of the pros and cons of each.

7.13.6 Dielectric Isolation

Isolated switching voltage regulators isolate the input from the output by electrically and physically separating the circuit into two sections preventing direct current flow between input and output, which is typically achieved by using a transformer. A result of isolation is that each of the isolated circuits has its own return or ground reference. There are two types of isolated voltage regulators. They are flyback and fly-forward regulators.

7.13.7 Non-isolation

Non-isolated switching voltage regulators have a single circuit in which current can flow between both the input and output. Non-isolation regulators are beneficial in terms of cost, size and performance. There are two classified types of non-isolated voltage regulators. They are step-down voltage regulators and step-up voltage regulators. Step-down voltage regulators take an input voltage that may be high and lower the voltage. Step up voltage regulators take a low input voltage and increase it to provide a higher output voltage.

Type of voltage regulator	Pros	Cons
Linear voltage regulator	● Design is very simple ● Less output ripple ● Response time is fast ● Less noise	● Low efficiency ● Space requirement is large ● Voltage cannot be increased ● Heat sink is sometimes required
Switching voltage regulator	● Efficiency is high ● Size and weight are very low ● Boost or buck or inverting or buck/boost converters are possible	● Complex design ● Expensive ● Noise is high ● Transient recovery time is time consuming

Table 17: Pros and Cons of Linear and Switching Voltage Regulators

7.13.8 Other Voltage Regulators – Alternator, Electronic, And Transistor

First, we will look at the alternator voltage regulator. Alternators produce the current that is required to meet a vehicle's electrical demands when the engine is running. It also replenishes the energy used after startup. Furthermore, an alternator has the ability to produce more current at the lower speeds than the DC generators that were onced used by most vehicles. The alternator has two parts. These parts are the stator and rotor/armature. The stator is a stationary component which contains a set of electrical conductors wound in coils over an iron core. The rotor/armature is the moving component that produces a rotating magnetic field by any one of the listed three ways. These are (1) induction, (2) permanent magnets, and (3) using an exciter.

Figure 15: Alternator Voltage Regulator

Next we will look at the electronic voltage regulator. This type of voltage regulator can be made simply from a resistor in series with a diode or series of diodes. Due to the logarithmic shape of the diode V-I curves, voltage across the diode changes only slightly due to changes in current or changes in input. There is a flaw to this design however. If your design needs precise voltage control and efficiency, this design is not the way to go, however if precise voltage control and efficiency are not important or needed for our design then this voltage regulator would work fine.

Figure 16: Electronic Voltage Regulator

Lastly, we will be looking at the transistor voltage regulator. These voltage regulators have an abstable voltage reference source that is provided by the zener diode, which is also known as the reverse breakdown voltage operating diode. Moreover, the transistor voltage regulator maintains a constant DC output voltage, while the AC ripple voltage is blocked. Although, while the AC ripple voltage is blocked, the filter cannot be blocked. Furthermore, this regulator also has an extra circuit for short circuit protection, current limiting circuit, over-voltage protection, and thermal shutdown.

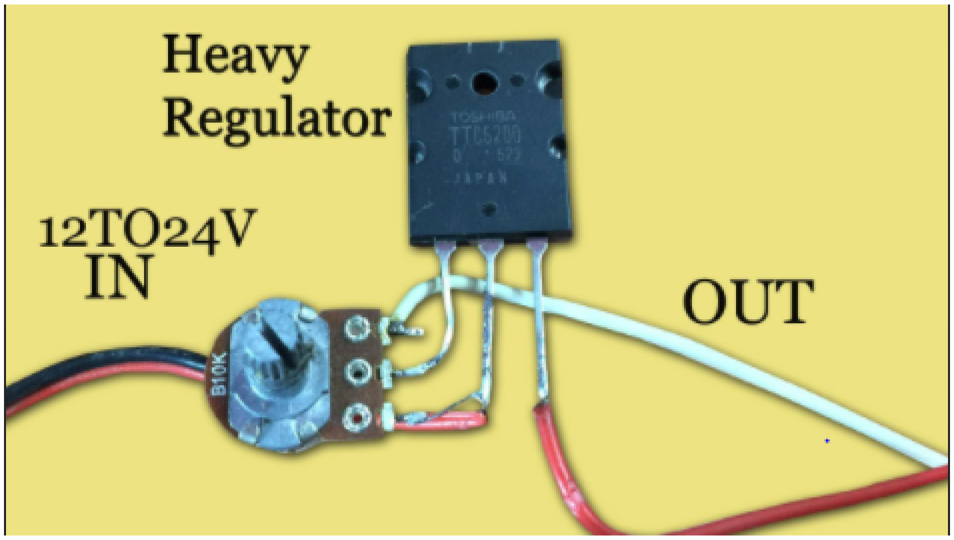


Figure 17: Transistor Voltage Regulator

7.13.9 DC - DC Converter

Since our design would use DC voltage, a DC to DC converter is an important design component as it would allow us to increase or decrease the voltage to other hardware components such as connecting our Jetson Nano. DC to DC converters act as a voltage regulator that consists of components such as switches, inductors, and capacitors. The main justification why we use DC to DC converters is because it simplifies power supply systems, isolates primary and secondary circuits, and matches the loads to the power supply. Even if the input or output voltage varies based on whether we use a plug-in power supply or batteries, DC to DC converters ensure a constant output in terms of the voltage thanks to closed feedback loops, thus allowing it to maintain efficiency as a power conversion circuit.

We decided to choose a DC to DC converter over a step-up or step-down transformer because they proved to be highly inefficient and wasted enormous amounts of energy in the form of excess heat. We will look at the pros and cons of DC to DC converters in table 18. There are four main types of DC to DC converters. These are buck converter, boost converter, buck boost converter, and SEPIC converter. We will look at more specifics of each type below.

Pros	**Cons**
● Provides technique to extend potential from partly reduced cell potential ● Available as whole hybrid circuit element and requires very few additional components ● DC - DC choppers are used to regulate the voltage ● It is constructed to make best usage of the energy yield for photo-voltaic systems ● Isolated converters has more energy transformer on condition of the barrier ● Output of isolated converter is organized as positive or negative	● Prone to noise ● Expensive ● Choppers are inadequate due to unsteady voltage and current supply ● Fly-back type: More EMI due to gap, more ripple current, more input/output capacitance, higher losses, etc.

Table 18: DC - DC Converter Pros and Cons

7.13.10 Buck Converter

A buck converter is a DC to DC converter that acts as a step-down transformer which produces a voltage that has been reduced from the input voltage, meaning the output

voltage is less than the input voltage. It regulates the unregulated DC voltage to stabilize the lower output voltage.
When the switch is closed the voltage flows through the inductor, the voltage flowing through the inductor will be limited by the inductor capacity. It will then travel into the capacitor where the energy is stored. when the switch is turned off the stored energy flows through the diode maintaining the reduced output current. The cycle repeats thousands of times a second helping maintain the regulated output voltage.

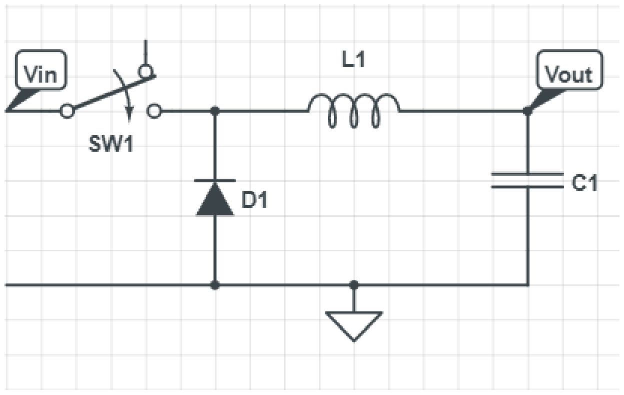


Figure 18: Buck Converter

7.13.11 Boost Converter

A boost converter is a DC to DC converter that acts as a step-up transformer which steps the voltage up and thus produces a higher output voltage as opposed to lower input voltage. This type of converter is generally used in the likes of LED lights, such as the one in our design.

In **Figure 19** the circuit works by passing current through the inductor (L1) which creates a magnetic field. the inductor magnetic field will collapse and thus creates a voltage spike as the magnetic field in the inductor collapses. Since the voltage is going to be higher than the input voltage the current is going to be much lower than the input voltage in the end.

The switch (SW1) turns on and off thousands of times a second. When the inductor creates a high voltage spike the high voltage gets passed through a diode (D1) and stored into the capacitor (C1). The D1 diode is there to stop the current from traveling backwards from the capacitor once it's stored. everytime the inductor induces a voltage spike the capacitor stores these higher voltages which steps up the voltage.

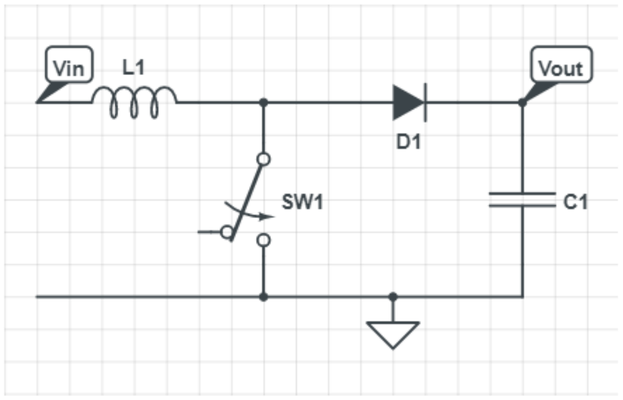


Figure 19: Boost converter

7.13.12 Buck Boost Converter

A buck boost converter acts as a dual DC to DC converter. This means it can act as both a step-up or step-down transformer, meaning it can produce an output voltage that is either higher or lower than that of the input voltage. Furthermore, this type of converter can be used to produce both negative or positive voltages, and thus can be used in various places throughout the circuit.

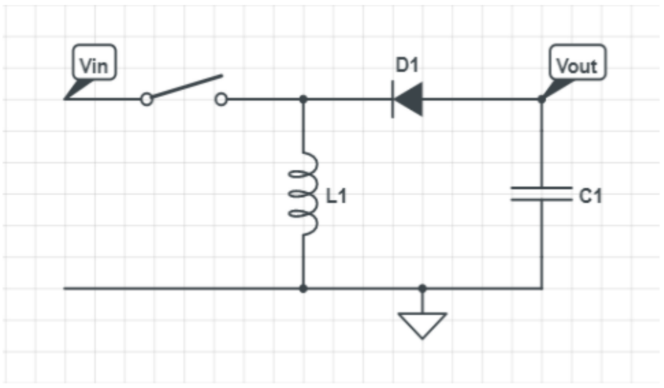


Figure 20: Buck Boost Converter

7.13.13 SEPIC Converter

The single-ended primary-inductor converter (SEPIC for short) is a type of converter that allows the output voltage to be greater than, less than, or equal to that of its input. A SEPIC is essentially a boost converter followed by an inverted buck-boost converter, therefore it is similar to a traditional buck-boost converter, but has advantages of having non-inverted output as shown in the Figure 21 below. As shown below, the SEPIC exchanges energy between the capacitors and inductors in order to convert from one voltage to another. The output of the SEPIC is controlled by the duty cycle of the control switch (S1), which acts as a transistor such as a MOSFET.

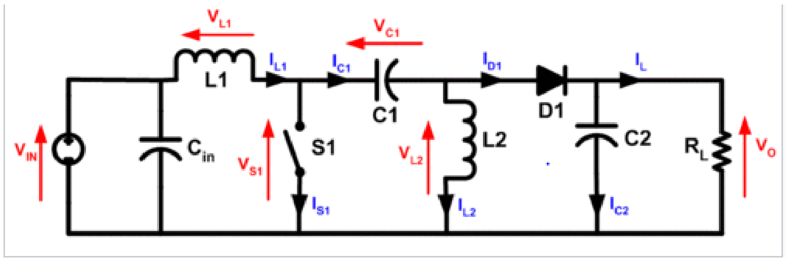


Figure 21: SEPIC Schematic

7.14 Motor Controllers

Motor controllers are used with both direct current (DC) and alternating current (AC) motors. The selection of the motor controller is an important deciding factor as it allows for us to control the electrical power supply flow. This is crucial for our project because it will allow us to overcome any constraints in the form of overloading current and/or overloading of voltage. The group came to a selection of three specific controllers that we thought would work best.

These choices are the HiLetgo 16 channel servo controller, MakerFocus 16 channel servo controller, and Robotdigg 3 channel servo controller. In Table 19 below, we can see the specifications of each of these motors. These specifications include the type of control signal each controller has, type of communication module, input/output voltage, the number of channels, and lastly the cost differentiation between each type of motor.

7.14.1 Specifications for Motor Controller

Specification/ Controller	**HiLetgo16 Channel Servo Controller**	**MakerFocus 16 Channel Servo Controller**	**Robotdigg 3 Channel Servo Controller**
Control Signal	PWM	PWM	PWM/Analog
Communication Module	IIC	IIC/USB	Analog 0 - 3.3 V
Input Voltage (V)	5 - 10	5	4.7 - 7.4
Output Voltage (V)	5 - 10	6 - 12	3.3
Number of Channels	16	16	3
Price ($)	9.09	13.59	8.00
	HiLetgo 16 Channel Servo Controller	MakerFocus 16 channel Servo Controller	Robotdigg 3 channel Servo Controller

Table 19: Specifications for Motor Controller

7.15 Cooling Fans

A cooling fan is a relatively cheap but important component in the context of this project. The Jetson Nano embedded computer we intend to develop our computer vision tasks on will be performing computation in parallel using the GPU/CUDA cores on the board. Running these parallel artificial intelligence operations on the Nano will make the GPU and CPU very hot. So in order not to sacrifice efficiency (the system will stall or slow down when it gets too hot) it is important to install a cooling fan on the heatsink of the device. There are two types of fans we will be looking at. These two fans consist of Direct Current (DC) and Pulse Width Modulation (PWM) fans.

First we will look at DC fans. DC fans are powered with a potential of fixed value such as the voltage of a battery. Typical voltage values for DC fans are 5V, 12V, 24V and 48V. The higher the voltage of the fan, the quicker the fan speed, and the greater the cooling. This allows you to lower the fan speed by reducing the voltage, although most fans will stall below a certain speed. However, controlling this voltage change can be found to be very difficult. DC fans are traditional computer fans. They have a 3-pin connector which consists of power supply, ground, and signal pin. The signal pin collects information about how fast the fan is spinning (tachometer output) and alerts if the fan stops working.

Now let's look at PWM fans. PWM fans and/or pumps are normally found in some CPU coolers and GPU (a.k.a graphics card) coolers. Furthermore, these types of fans use an integrated circuit to control the speed of a fan or pump and, therefore, how much cooling it is providing to the CPU or GPU. PWM fans are very similar to DC fans, but differ in a major key aspect. This key aspect is that they have an extra pin for pulse width modulation. This particular pin takes input from the motherboard to directly control the fan's speed. Moreover, voltage control to determine what speed you want the fan at is very simplistic, thanks to the PWM pin that takes input directly from the motherboard. Not only that, but PWM fans have a 0% chance of the motor stalling, thus making the fan function 100% of the time. In table 20 below, we will be comparing these two types of fans and explaining our justification why we chose which fan.

We decided to choose a PWM fan because we thought it would pair best with the Jetson Nano as you will read later on in our experimental design heading. The PWM fan model we decided to choose was the Noctua NF-A4x20. In table 21 below we will look at more in depth specifics of this fan. Note that L.N.A stands for Low-Noise Adaptor, U.L.N.A stands for Ultra Low-Noise Adaptor, and AAO stands for Advanced Acoustic Optimisation. This specifically chosen PWM fan has a manufacturer's warranty of up to 72 months, which is equivalent to 6 years, because Noctua fans are renowned for their immaculate quality and longevity. We were also able to find the cheapest company to order this fan from by doing a google search of the EAN barcode, which is 9010018100112. Using this barcode, we were able to find the company selling the NF-A4x20 for a fair cheap cost of $14.95.

This fan also has a MTBF (mean time between failures) of 150,000 hours, thus having a good life-span of use and would not need to be replaced often. Another benefit of this PWM fan is the extensive cabling options that it offers. The fan's 20cm primary cable minimizies cable clutter in typical applications, while the supplied 30cm extension cord provides reach when needed. Both cables that come with this fan are fully sleeved and a 4 - pin y - cable allows the connection of a second fan to the same PWM fan header for automatic control.

DC fan	**PWM fan**
● 3 - pin fan	● 4 - pin fan
● Voltage control	● PWM control
● The supply voltage is varied for speed control	● The supply voltage is constant throughout the operation, the PWM signal duty cycle controls fan speed
● Precise speed control is difficult	● Seamless speed control
● Limited in reducing speed below that which corresponds to the minimum threshold voltage	● The minimum speed achieved can be below DC fans
● Speed can be lowered up to 40% of the rated speed	● The lowest speed can be less than 20% of the rated speed
● Possibility of motor stalling below the minimum threshold voltage	● No chance of motor stalling
● Commonly used as chassis fans with low power consumption	● Commonly used as a CPU cooler with higher power consumption

Table 20: Comparison Between DC and PWM Fans

This fan operates with a voltage of 12V which is easily given from our chosen DC - DC regulator. The fan has a dimensional size of 40x20mm. Moreover, when the fan is active and in-use, it is very quiet. This is due to the flow acceleration channels that this fan offers. These channels allow the fan to increase the speed of airflow at the crucial outer blade regions. Due to this factor, it reduces suction side flow separation allowing better efficiency and lower vortex noise. Furthermore, the stepped inlet design of the fan adds turbulence to the influx, in order to facilitate the transition from laminar flow to turbulent flow, therefore reducing the tonal intake noise, as well as, improving flow attachment and increasing suction capacity, especially in space - constrained environments. This fan also includes inner surface microstructures which allow the tips of the fan blades, plow through the boundary layer which is created by the inner surface microstructures. This creates flow separation from the suction side of the blades, which significantly suppresses them, and results in reduced blade passing noise and improves both airflow and pressure efficiency.

Another reason why we decided to go with the NF-A4x20 instead of the NF-A4x10 is because the NF-A4x20 has twice the thickness which allows for increased static pressure performance and thus creates the NF-A4x20 fan ideal for demanding applications with high flow resistance.

The reason our design benefits greatly from this fan is because it stops at 0% PWM. This means if the motherboard or PWM controller were to set the duty cycle of the PWM to 0% it will stop. This works with our design because it allows us to create setups with semi-passive cooling that will automatically turn the fan off, and thus run completely silent at an idle speed if thermals allow. Another key factor that influenced us to go with the NF-A4x20 is that it comes with the Omnijoin^{TM} adaptor set. This set will allow us to connect our fan to the Jetson Nano. To do this, we will cut the original wires that the fan comes with and connect the adapter using the supplied cable connectors. We can plug the NF-A4x20 into the proprietary fan headers straight to the Jetson Nano, thus powering our fan, as well as keeping the Jetson Nano cool and keeping it from overheating, as this is the reason for the fan in the first place.

Specifications	NF-A4x20
Dimensions	40 x 40 x 20 mm
Screw hole spacings	32 mm
Connector	3-pin
Blade Geometry	A-series with flow acceleration channels
Rotational Speed (+/- 10%)	5000 RPM
Rotational Speed with L.N.A. (+/- 10%)	4400 RPM
Rotational Speed with U.L.N.A (+/- 10%)	3700 RPM
Airflow	5.5 CFM
Airflow with L.N.A	4.9 CFM
Airflow with U.L.N.A	4.1 CFM
Acoustical Noise	14.9 dB(A)
Acoustical Noise with L.N.A	12.2 dB(A)
Acoustical Noise with U.L.N.A	8.5 dB(A)
Max Input Power	0.6 W
Max Input Current	0.05 A
Voltage	12 V

Table 21: NF-A4x20 PWM Fan Specifications

7.16 Powering Methods

As the ArticuLight will implement lighting features and other electronics, assessing the power electronics is integral to the success of this project. Wall power (120V at 60Hz), battery power (DC) and a hybrid solution will be considered in this section of the report.

7.16.1 Wall Plug-in Power Supply

Wall plug-in power supply is beneficial because it gives us unlimited power as long as it is plugged in. However, having unlimited power does come with constraints. A major constraint is the length of the cord the power supply has. It is useless to use a wall plug-in if it can't reach the outlet. However, to overcome this constraint, we can use an extension cord which we can plug the wall power supply into.

Another constraint is that different wall outlets could provide different output voltages to our plug-in power supply thus supplying our master component with a different input voltage which may overload the board as we are not sure how much power is being supplied from the wall plug-in device.

Another issue with wall plug in power is the hazard of the cord being in the way of tripping. This could have the potential for the light to fall and be damaged. To overcome this constraint we would hope to somehow implement a command strip hook to place on the device or workbench to help hold any excess wire that is not needed. Furthermore, another constraint with the wall plug-in cord is that it could be easily accessible to small children and/or pets. For example small children may pull on the wire causing the light to fall and be damaged or pets can chew through the wire, which may ruin the system.

7.16.2 Battery Pack Power Supply

Battery pack power supply is great because it is very portable and comes in various form factors. These factors include different voltages, currents, and power capabilities. Battery power supply also provides very stable and clean power, although we have to keep in mind the discharge of the battery overtime. We can overcome this constraint by using a battery level indicator. This will allow us to make sure that our batteries never discharge completely, thus keeping them alive longer.

Going battery powered does have some advantages. You eliminated the tripping hazard of a power cord being in the way. The unit is portable and can be placed anywhere without needing a wall outlet present or using an extension cord. With improved battery technology battery life is greatly increased.

Some drawbacks to using a battery: depending on the battery size you might not have a long operating life before you need to recharge the battery. We would need a way to incorporate a two battery pack system while one is being used, and the other one is charging and on standby ready to get swapped out when the one being used is depleted.

As the battery gets charged and recharged the battery begins to degrade and overtime wont hold a charge anymore. This could eventually get costly as specific types of batteries can be more expensive than others, due to their output voltage, durability and whether or not the battery is rechargeable or just thrown away. When we tried prototyping a battery pack power supply, we saw it lasted only 5 minutes with AA batteries. As a group, we all agreed that no work bench project could be completed within 5 minutes time, so we decided to go with the wall outlet power supply.

7.16.3 Hybrid Version (AC and DC Power Sources)

A hybrid version of the system would utilize both batteries and a 120V from the wall outlet. When the device is close to a reachable wall outlet the device would use the outlet wall thus having an endless amount of electricity powering the system and when no wall outlet is present the unit would use a battery powered system.

Using this type of system would eliminate any kind of constraint. If you don't have one power source available to you, you always have the other power source. This will eliminate any wasted time on a jobsite. Furthermore, it would save cost on our overall electrical bill since we have the choice to use a wall plug-in power source or to just turn on a battery pack power source.

Battery size	**Chemical system**	**Nominal voltage**	**Type**
AA, AAA, C, D	● Zinc - Manganese Dioxide ● Nickel - Metal Hydride	1.5 V 1.2V	Primary (Non - Rechargeable) Secondary (Rechargable)
9V	● Zinc - Manganese Dioxide; Lithium - Manganese Dioxide ● Nickel - Metal Hydride	9V 8.4V	Primary (Non - Rechargeable) Secondary (Rechargable)
Coin cell	● Lithium - Manganese Dioxide	3V	Primary (Non - Rechargeable)
Flat pack	● Lithium - Ion (Li-ion); Lithium Polymer (LiPo)	3.7V	Secondary (Rechargable)
Car battery	● Lead - Acid	12V	Secondary (Rechargeable)

Table 22: List of Battery Sizes, Chemical Systems, Voltages, and Types

Power supply	Pros	Cons
Wall plug-in	- Stable voltage - Can plug into extension cords to give it more length - No discharge	- Not portable - Different wall outlets may have different voltages - Plug may be too short to reach outlet
Battery pack	- Stable and clean power - Battery level indicator - Different types of batteries - Portable	- Voltage decreases as batteries discharge - Batteries can erode causing erosion in the battery pack

Table 23: Pros and cons of power supplies

7.17 Wire Loom

A wire harness, or loom, is a piece of hardware used to run wires between components of multi-stage devices. It is to protect the bundle of wires, securing it and keeping things organized. It also helps protect it from vibration or abrasion and enhance the aesthetic. There are many styles of looms for electronics, but this project requires something that will be durable, weather resistant and able to withstand a large range of ambient temperatures for the garage or outdoor environment it will be expected to function in. The following are various wire looms that were considered.

7.17.1 Plastic Loom

These looms are a cheap option that still maintains most of the critical functionality we are considering. They are highly flexible which will reduce strain on the motors, fairly weather resistant while in good condition, but will likely suffer from cracks from frequent use and when exposed to temperature extremes. These looms are very lightweight which will reduce the strain on the motors and pulleys during actuation.

7.17.2 Tinned Copper Braided Sleeve

This is a durable material typically made of a nylon material and coated with tinned copper. This material is very lightweight and flexible much like the plastic loom, however, the braided sleeve can fray over prolonged use. This style of wire loom also provides very little weatherproofing aside from some insulation, but does provide some minor electromagnetic shielding.

7.17.3 Metal Loom

A metal loom, typically referred to as "gooseneck" which is a flexible hose that has some rigidity which allows the loom to maintain the shape it is placed in. This rigidity has the drawback of requiring more force to move the arm, putting more strain on the motors, but will also act as a strain relief on the wires. The metal loom is the most durable and provides excellent weather protection as well as electromagnetic shielding.

7.17.4 Specifications for Wire Loom

Specification/Material	Plastic	Braided Sleeve	Gooseneck Pipe
Operating Temp (^{0}C)	-70 to 125	-50 to 150	N/A
Weight (lbs)	0.203	0.101	N/A
Price ($)	10.99 (25 feet)	10.98 (10 feet)	19.95 (18 inches)

Table 24: Specifications of Wire Looms

Though there are cheaper options, the most robust and reliable will be the metal loom. A stainless steel loom will provide weather protection that can withstand a large range of temperature fluctuation. The rigidness of the loom can also be used as a measure to hold the arm in the desired position while not adding strain to the wires inside, as long as the loom itself is connected to the light socket housing. The project will run on 110V AC, so supplying power to the motors should not be an issue.

7.18 Pulley Wire Guide

The pulley system will be the main mechanical method of controlling the movement of the device with a servo motor driving each pulley, for precise angular accuracy. The pulley will reduce the amount of torque the servo requires to output for the length of the arm. A metal wire will be attached to opposite sides of each pulley and will be run down the sides of the arm, four in total, two for each pulley. When a pulley is turned, the corresponding side will be wound about the pulley, causing that side to flex. The size of the pulley will need to correlate to the degrees of movement from the servo such that it allows for a full 90 degree range of motion. The pulley's themselves need to be durable enough to handle the friction and abrasion from the wires while being wound and the strain from holding the arm in a set position for extended periods of time. Following are the materials that we have decided to consider.

7.18.1 Plastic Pulleys

These pulleys are very cheap and durable, but will degrade faster than a metal pulley from abrasion and environmental conditions. one advantage of plastic is having the ability to create or fabricate new parts from a mold injected process or 3D printed from a design sketched on the computer. plastic has the ability to be durable and lightweight verus the same part that is made out of metal. Some plastics have the ability to resist chemical reactions such as oxidation or corrosion. Parts made from plastic are usually easier to machine as well.

As with anything made out of plastic there are some disadvantages to using plastic parts. Plastic is not as durable as something made out of metal. Prolonged exposure to the sun will cause the part to become brittle and crack or break over time. When plastic parts are exposed to friction and create heat, they tend to deform from their original shape causing issues down the road.

7.18.2 Metal Pulleys

These pulleys are very strong and durable for all environmental conditions, but will require preventive maintenance over time from rust or corrosion which would cause actuation problems.

The size of the pulley will be linked to the range of motion of the light, so having the appropriate size is important in completing the project. Testing will be done with 3D printed plastic pulleys which will allow for rapid prototyping and determining the size needed. Another option would be to have smaller pulleys with gears to accommodate the differential in size with an appropriate gear ratio. These gears could be 3D printed as well to determine an optimal ratio. Once the appropriate size is found, a metal pulley of comparable size will be purchased.

7.19 Comparison of Filament Types

Filament	**Pros**	**Cons**
PLA	Easy to print with lower print temperature	Degrades over time when exposed to outdoor conditions
PETG	High degree of flexibility and resistant to warping	Higher print temperature Broken down by ultraviolet light, not suitable for long term outdoor use
ABS	Tough, impact-resistant material. Water resistant	Higher print temperature Broken down by ultraviolet light, not suitable for long term outdoor use Ventilation required when printing due to fumes produced
HIPS	Impact and water resistant. Very rigid material with high impact strength.	Higher print temperature Ventilation required when printing
Metal Filled PLA	Gives a metallic finish. Increases weight and is much more resistant to degradation than standard PLA. Acts as a thermal conductor.	Much harder on the printer itself Very expensive, around $30-40 per kg roll.

Table 25: Comparison of Filament Types

7.19.1 Polylactic Acid (PLA)

This is a polymer plastic that is made from biodegradable materials, typically cornstarch or sugarcane. It has a melting point between 180 to 200 degrees C, but will start warping around 50-60 degrees C. The plastic is tough and fairly resilient to abrasion. Due to the material's biodegradable nature, it is not water or chemical resistant and will be slowly consumed by various common bacteria when exposed to humid environments, so would have to be kept in a temperature controlled, dry environment. With proper care, this is a viable option for a workshop or vented garage. PLA is low cost and prints fast at a lower temperature, this makes it ideal for rapid prototyping and for indoor use primarily with enough toughness to handle occasional outside use, but ultimately will not be ideal in the environmental extremes, high humidity, heat, or cold, that should be accounted for.

7.19.2 Polyethylene Terephthalate Glycol (PETG)

PETG is an extremely high strength filament that can achieve very sturdy and strong prints. When printing it has very low shrinkage, so is less likely to warp while printing. The glycol in the filament keeps the plastic from crystallizing, making it softer than other materials, this has the benefit of allowing flexibility in the material and is also less likely to warp after printing is completed. PETG melts between 230 to 260 degrees C and will not warp until temperatures exceed 80 degrees C. Due to the chemical composition of PETG, it will break down under ultraviolet light (UV). This constraint can be overcome by using a flexible, waterproof paint or opaque sealant to block sunlight from reaching the material. PETG is also more chemically resistant and won't break down from solvents such as Acetone or Methyl Ethyl Ketone, common cleaning materials in workshops for electronics or jobs that use oils. Because of its robustness this filament is commonly used in mechanical applications and robotics especially when painted or with additives to make it more UV resistant which makes this a strong candidate for our robotic arm frame.

7.19.3 Acrylonitrile Butadiene Styrene (ABS)

ABS is the same plastic that is used to make Legos. It has similar properties to PETG, strong, durable plastic that is easy to shape and tough to break. It melts around 220 degrees C and will start warping in temperatures in excess of 80 degrees C and can handle temperatures as low as -40 degrees C before becoming brittle. ABS produces unpleasant fumes when heated so ventilation is needed when printing this material. ABS is water and chemical resistant but will break down over time from UV radiation, slower than PETG still a consideration if left outside for long periods of time. This can be mitigated by using a waterproof paint or sealant to prevent direct UV exposure. This is one of few materials researched that gave a low temperature for consideration. Though ABS is chemically resistant, unlike PETG, ABS will dissolve in solvents that contain acetone and MEK. This reduces the applications where this can be used, it is still possible if care is taken to limit the probability and duration of contact with those chemicals. There are also chemical resistant polyurethane coatings that can be used to mitigate this constraint. ABS is a good consideration for filament for this project due to its robustness, but may need extra considerations for the environment it will be used in.

7.19.4 High Impact Polystyrene (HIPS)

HIPS is a very tough and rigid plastic that has high impact strength which can be punched, routered, or sawn like wood or aluminum. Generally used for toys, packaging, signs, kicking plates seen on the bottom of doors, and displays. The method of printing HIPS is similar to ABS but is generally less likely to warp. Similar to ABS, HIPS will start warping in temperatures in excess of 80 degrees C and melts at 225 degrees C. It is water resistant and won't readily break down in UV radiation and can be dissolved with L-Limonene, a typical solvent adhesive remover that contains the peels of citrus fruits. These solvents can be fairly common in a workshop that handles oils and adhesives, but the chance of direct contact for long enough to dissolve the frame is likely to be negligible but will be considered. Another constraint is that this material would require ventilation while printing. The robust nature of HIPS makes it a viable candidate for the final product, offering the most durability and requiring the least additional treatment to handle environmental constraints.

7.19.5 Metal Filled PLA

There are three options being considered for a Metal Filled PLA, also known as composite filament, these are copper, bronze, and stainless steel. These filaments have metal powder mixed in the PLA filament to give the prints a metallic aesthetic but also adds structure to the model depending on the percentage of metal mixed in. The overall specifications of the composite filaments is dependent on the type of metal being printed. Bronze and copper can corrode and oxidize over time, creating the patina discoloration that is typically seen on bronze and copper structures. Stainless steel filaments, on the other hand, are immune from rust and corrosion, however this is the only magnetic 3D printing material that is being considered. These composite filaments add extra hardness and weight to the design of the arm, but can be very abrasive to the printer and in some cases have been found to be somewhat fragile. Due to the presence of metal in the PLA plastic, it removes most of the constraints PLA has regarding environmental conditions
The largest considerations for this material will be if the overall design will make the arm too heavy for the servo controllers and due to the conductive properties of these metals, the wiring will need to be insulated to ensure safety for the user and the electronics.

From the data collected, the best candidates for 3D printing are PETG, HIPS, and Metal Filled Filaments. All three of these materials offer the strength and durability that would be required for sustained use, if this is the ultimate route taken for building the frame. Each material has the durability to withstand environmental factors such as UV radiation, rain, dust, heat, and cold.

For rapid prototyping PLA was used as a cheap and easy to work with alternative for designing the components. From testing, we have discovered that PLA has been robust enough to handle constant, sustained movement from the arm while using a coated wire as well as springs mounted within the wire guide holes, protecting the plastic from being damaged by the wire.

8. Mechanical Design and Prototyping

8.1 Pulley and Arm System

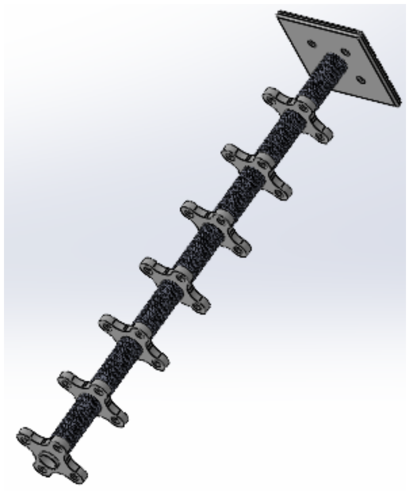


Figure 22: ArticuLight 3D Model

8.2 Frame Design

The frame for this project needed to be robust enough to handle various workspace environments. While researching the various types of wire looms available, Loc Line was determined to be a viable candidate for the frame structure of this project. Loc Line is a series of interlocking segments connected together to form a tube. Each joined segment has a 360 degree freedom of rotation and can flex in all directions. Commercial versions of Loc Line are made of plastic and are generally used as flexible hoses for water applications, i.e. aquariums, coolant lines, etc. which can be used indoor or outdoor, handling temperatures up to 170 degrees fahrenheit, that holds a shape when arranged, and is lightweight, 2oz per 20cm. It also has the advantage of allowing for a variable length by adding or taking away segments. However, commercial versions of this system have 3cm for each segment and .6cm internal diameter and are expensive for short segments, generally around and would require a method to attach a wire guide to the tube without altering the functionality of it.

Figure 23: Commercial LocLine

Using the loc line design allows for rapid prototyping of the frame but can easily be adapted for any of the wire looms researched, a wire guide will need to be attached to the loom.
Further research found a scaled up version of the Loc Line segments on Thingiverse.com for .stl files to 3D print the segments which have the dimensions of 6cm length and 1.6cm internal diameter. These increased parameters allow for fewer segments to make up the overall length of the project and ample space for the wiring that will be needed for the light source from the control module. Furthermore, the materials that can be 3D printed provide a myriad of options to suit the design robustness aspect of the project, outlined below.

8.2.1 Frame Prototype 1

A set of the new loc line segments were printed using PLA filament as a very quick means to test the initial design. In order to connect each segment, a heat gun was used to warm and soften the socket of each segment so they can be connected together. This first iteration of the loc line segments connected together but a few of the segments were heated for too long which made the socket very loose to the point where the ball and socket wouldn't hold the position it was moved into. An attempt to correct this mistake was taken by both submerging the loose segments into an ice bath, and by placing the segments in a freezer for a few hours. This was done in an attempt to allow the cold to shrink the plastic and salvage what would be a wasted print. Unfortunately, this attempt did not produce any change in the functionality of those segments. The segments were also brittle enough that they could not be pulled apart without either breaking or using the heating method to loosen the socket again. Another attempt to re-tighten the socket was to apply heat and compress the socket around the ball joint, but this was also not a viable strategy. The conclusion that was derived is that once the loc line segments were assembled they would have to remain assembled and can not be altered post-production.
A new set was printed and assembled, carefully controlling the heat applied while assembling the segments. During this assembly a few of the segments broke, revealing that these segments are thin and will break if too much pressure is applied.

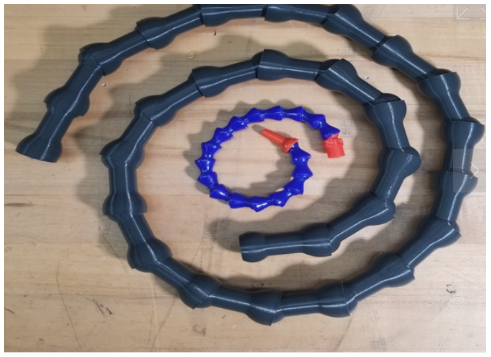


Figure 24: Thingiverse LocLine

8.2.2 Frame Prototype 2 - Fixing the LocLine

In order to address the issues with the thickness of the segments, the loc line system was recreated in Solid Works, using the same parameters from the model found on Thingiverse and doubled the overall thickness. Doing this reduced the internal diameter but gave the wires more protection from the outside environment and increased the overall strength of the frame. The next set of loc line segments printed were able to connect together and function correctly, holding the position it is moved into, and when fully assembled has an overall length of 66cm with the ability to add or remove segments to fit the overall dimensions needed for this project.

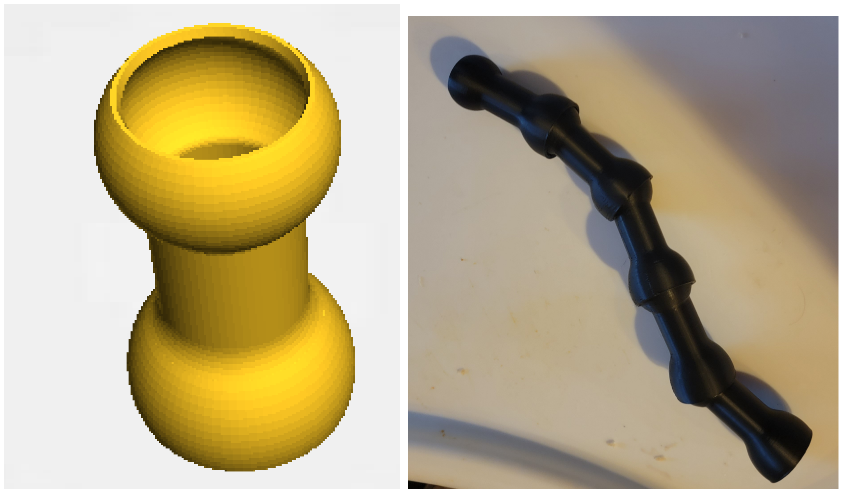


Figure 25: Modified 3D Model of LocLine and Printed Prototype 1

8.2.3 Frame Prototype 2 - Designing the Vertebrae

The next step in developing the frame was to determine how the motors would control the movement of the system. An organic movement approach was taken when approaching this subject. The arm does not need to bend more than 90 degrees in a single direction for this design specification. The Loc Line segment system allows for a complete freedom of movement of the arm itself, but not for the light mount. Because of this limitation, where the light will be pointing will need to be controlled by the position of the arm in order to keep the system as simple as possible.

A vertebrae design was chosen to replicate the free movement of a spine-like structure. This form is simple enough to design and could be incorporated into the loc line segments by expanding on the design that was already there, creating a base on the segment with through-holes that guide wires can be run through. Since the design expanded on the original loc line segment, these new vertebrae segments could be attached to each other or to the original segments. When developing the base, a basic square was used, then shaped for a more natural and smoother dimensions, making sure to keep the edge distance large enough so that the guide wires are not going to break the frame during repeated movements. The holes were aligned in such a way that they are symmetrical with the others, on the corners. This gives the maximum amount of space to place the holes while affording the aforementioned edge distance and due to the symmetry, will allow two gears to control all directions of movement, with opposite corners acting as the counter movement to one another. Another advantage to the vertebrae segments, is that each guide wire passthrough hole will act as a pulley for the next segment which has the added effect of reducing the torque from the overall lever length of the entire system to that experienced from just one segment, and should reduce the size of the motors needed to operate the system. The final advantage of the vertebrae design is that it reduces the overall footprint of the arm itself. By keeping the wires closer to the frame, there will be less chance for them to get twisted or tangled with other objects in the area or on itself as it's moving.

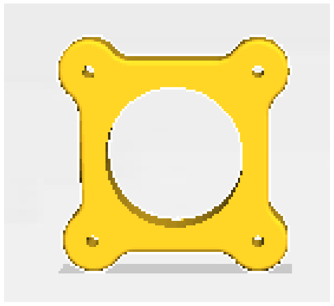


Figure 26: Vertebrae Wire Guide 3D Model

Since the design of the wire guides does not allow for the base to move freely, the only possible twisting that may occur is from the segment connection points themselves. This may need to be addressed later with further testing, however, if the tension on the control wires is high enough, this may not be a design constraint.

8.2.4 Frame Prototype 3 - Incorporating the Vertebrae

Once the wire guide was designed a method to attach these to the loc line segments needed to be addressed for it to be fully realized. An option to attach these pieces that was considered was to use set screws along one or more of the faces between the wire guide holes. This has the advantage of simplicity and these vertebrae could be placed anywhere along the length of the arm. Since plastic material is not difficult to drill through, the implementation would be very quick and easy. The disadvantage to this option is that drilling holes in the structure ultimately weakens the integrity of the the structure but more importantly, this would add between 1 to 4 screws holes per each wire guide that would need to be tapped adding 1 to 4 places per wire guide what would need some sort of preventive maintenance and/or replacement to ensure these components don't come loose and slip over time. This is still an option, but is more viable for a metal frame.

The design that was ultimately decided on while using the 3D printing strategy is to incorporate the vertebrae into the loc line design. This not only creates a segment structure that includes the wire guide holes, thus not requiring much maintenance for the final product, and giving the wire guides more stability in their location since each piece is effectively identical. This design has the added benefit of reinforcing the socket of the loc line by adding the entire wire guide structure around the weakest part of the segment. This makes the segments harder to put together, but the results are very strong and secure.

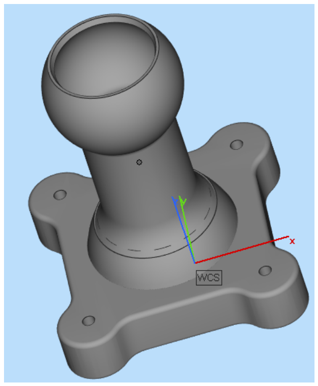


Figure 27: Combined Vertebrae and LocLine 3D Model

Connecting the loc line and these hybrid segments together allows for a partially realized design that could have guide wires and a pair of pulleys connected to the system. Doing so enabled us to fully visualize the movement pattern of the system in action.

8.2.5 Frame Prototype 3 - Complications with the Design

From operating the pulleys and wires manually a couple of design flaws become apparent almost immediately. Initially the arm articulated as expected while each pulley operated opposite sides of one degree of freedom, i.e. one pulley controlled pitch and the other yaw, the problem becomes evident when a new direction is applied to the system while it's already holding a position. Instead of unwinding to a neutral state, then moving into the new position, the arm tends to contract or crimp on itself due to the rigidity of this design.

The next flaw is due to the mobility of the ball and socket design which allows each segment a large angle of pitch and yaw, more than the 90 degrees needed for this project, the problem comes from the ability to rotate 360 degrees at each segment. When repeated movements are applied to the system, the overall structure has the potential to rotate at any given point along the length, which will tangle the control wires around the structure itself. The next flaw comes from the head of the devices, where the light socket will be placed. This design does not allow the head to be positioned except by the movement of the arm itself, which may lead to the device moving to the right position but the light not pointing at the right spot.

Another constraint that developed during testing is a matter of using the control wires with plastic guide holes. While checking the freedom of movement from flexing the arm repeatedly, it was observed that the metal wire used was slowly abrazing the plastic of the wire guide holes. The process does not appear to be fast, but would over time break through the plastic, causing a structural failure which would result in the need to replace segments of the loc line.

8.2.5.1 Frame Prototype 3 - Addressing the Complications

The largest complication of the design and thus the first to be addressed is the crimping of the arm itself. The underlying problem is that the device needs to have a zero initial state that can be used as a reference for movement. Ways to address this flaw include, having the device undo the previous movement from the actuation. This would place the device back to its initial state before moving to the new location, but would effectively double the movement time which could result in not meeting this critical specification. Another option is to add motors to the head of the arm. This would effectively double the power and component requirements, four motors and relays instead of two, but because the action of undoing the previous actuation could be performed in parallel with the new movement, this has a high probability of maintaining the movement time specification but would increase the cost and complexity of the device itself.

The next option being considered is to completely loosen the ball and socket joints so that the arm is not rigid. This has the advantage of allowing the arm to be completely flexible and allow gravity to return the device to an initial state, but would require the motors to hold the actuation position without aid from the frame itself. This would put more strain on the motors, but the weight of the material being tested so far has proven to be extremely light compared to the expected weight from the initial design, 4oz not including the light versus the specification of 80oz. The final option being considered is to add a tube down the length of the loom/loc line. This has the benefit of giving the wires inside more environmental protection, though it may still need to be replaced periodically, but if the tube is stiff enough, it would act as a counter force, pulling the arm back to an initial state. This has the same cost of requiring the motors to compensate and hold the position, but has a high chance of fixing the crimping issue completely.

The rotating flaw has a couple of options being considered so far. First, a key and keyway feature can be added to the design of both the vertebrae and/or the loc line. This would be a quick and easy fix that places a mechanical stop on the segments, not allowing the segment to rotate more than the designed amount or not at all. These would be placed on varying sides of the segments to clock the pieces, which will make sure there isn't an alignment issue that would minimize the change of the locks to slip or break. Another option is to add a spring or flexible joint between each vertebrae segment. This helps to pull each section back into the initial state, however it would give more force for the motors to compensate for when holding a position. The final option that is still being considered is to use a loom which has a much smaller chance to rotate on itself, although the wire guides would need to be attached in a different manner and most looms would still have the potential for crimping.

There is still a possibility that once the wires are fully secured and pulled tight, the tension force from the wires will prevent the structure from rotating on itself, negating the need to address this flaw. The final design complication will need to be addressed more once the first two are completed. A rigid light fixture isn't likely to have any problem once smooth, fluid actuation is developed but is still considered in case further design is needed to address this later.

To address the wire guide constraint, grommets will be added into the guide holes to give the components a metal sleeve to run the wires through. This will decrease the likelihood of the wire breaking through the plastic loc line, however adjustments to the model will need to be made to account for the extra space needed to fit a grommet in the guide wire holes.

8.2.6 Frame Prototype 4 - Final Prototype Design

The design complication of free rotation posed a challenge to correct. A simple key-lock design would prevent the system from rotating. For convenience, we will identify pitch and yaw as movement in the x and y-axis, with rotation about the z-axis. A simple notch with a raised connector would prevent movement about the z-axis but would also limit at least one of the other axis of movement. To fix this new constraint, the key was changed from a square to a dome shape, this reduces the number of sharp edges that would impede movement, and the lock segment was changed to a channel that would allow the key to slide freely. This was added at the four corners where the wire guides were designed. This section afforded the extra material that would be cut out of the segment so the overall structure would not be compromised. Even after designing the channel, when the position of the connected pieces were at a full extension on either the x or y-axis, this exposed the key and would reduce how smoothly the key fit back into the channel when the position of the link was not in a straight 90 degree along one axis. This could still work in function, but would add undue friction on connecting pieces as well as adding a potential point of failure at those locations. A chamfer of 45 degrees was added at the entrance of the channels which widened the opening and added a guide for the key to fit back into the channel. Doing so allows the same overall design and freedom of movement on the x and y-axis from before and keeps the system from getting stuck when at a full position on those axis, while simultaneously preventing the system from rotating on the z-axis.

The constraints with the wire guide were resolved by widening the wire guide hole in order to add a bushing that could hold a 14 gauge wire. This value was picked as an arbitrary, small, wire size and can be adjusted as needed within the sketch of the model. The holes were designed to be 0.1mm smaller than the actual bushing size in order to tap them in, allowing the external pressure from the frame to hold them in place as opposed to using a mechanical fastener. These bushings can also be used to secure the wire in place to prevent them rubbing against the material. A base and a mount for the lights still need to be incorporated, but the overall arm design has been completed unless a better solution presents itself. However, the plan will be to use a flexible arm design since it has the potential to be the most versatile and will require less mechanical design elements compared to anything more complicated.

Below, **Figure 28** shows the new model for the segments, the design iterations that were printed for testing, and finally how the assembly will look once they are all completed.

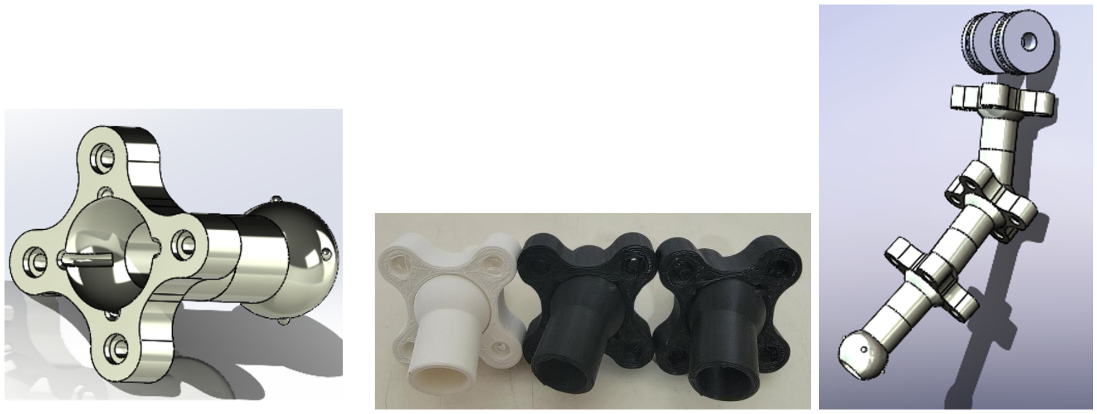


Figure 28: Prototype 4 3D Model, Printed Prototypes 2,3, and 4, and Assembled 3D Model

8.3 Continuum Robotic Arm Redesign

Upon completing the prototype of this design and testing the model with pulleys and wires, we discovered that though the design moved as intended, because it did not try to return to a neutral steady state the arm had a tendency to crimp and try to fold on itself. The only way to alleviate this mechanical design flaw would be to have the servo motors return to center before moving to a new position. This, in turn, would increase our overall actuation time. After further research, we decided to design the arm as a continuum robotic arm. This utilizes a spring or some other interlocking segments that try to return to a steady state when not being controlled by actuators. The overall design of the system can be broken down into five major sections; The segments/wire guides, camera mount, servo mount and stands, and the case/housing.

8.3.1 Segments/Wire Guides

The center spring has multiple segments with the wires fastened at the top, by the camera. This shifts the pivot point from the base of the arm near the servo motors, to the top. Each segment is essentially the base of the designed interlocking connectors which have four holes near the edges to act as the wire guides, with a small plastic spring added to prevent the wire from damaging the plastic by allowing the wire to flex around the hole better and a center hole the diameter of the spring which provides a very tight fit. The base plate and bottom two segments have copper tubing run through the wire holes to give the system a very stiff base. This was done to reduce the strain on the motors and pulleys when the system is in the horizontal position and prevents a sharp bend in the arm from the base plate. A segment was added to one side of the center hole to act as a further guide for the spring and as a mounting point for adding compression springs in the future to add a third degree of freedom.

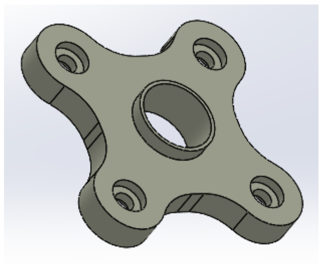


Figure 29: Segment/Wire Guide with Compression Spring Support

8.3.2 Camera Mount

Two versions of the camera mount were designed. One for an Onn usb webcam which has a slot in the top that allows the camera to be mounted with a bolt. The second is designed to accommodate the legs of a raspberry pi camera stand.

A segment model was used as the template with two legs extended from the top wire guide hole and a mount option for each camera. The legs for each mount were specifically tailored to accommodate the respective camera to allow the bottom of the frame to be just over the light bulb edge. The center hole was also widened to accommodate the light socket threads. Copper tubes were also used to connect the camera mount and the last segment so they move as one unit.

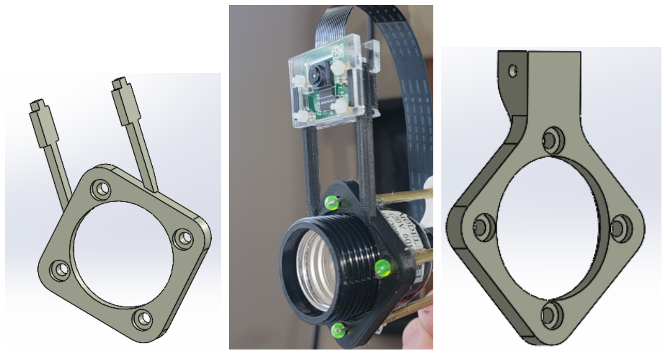


Figure 30: Camera Mounts

8.3.3 Servo Mounts and Stands

The servo mounts were designed as an L bracket with bolt holes that lined up with the servo mounting holes. The original design was not strong enough to account for the amount of torque placed on them by the actuation of the arm, so we made the mounts nearly 3 times thicker.
The stands are made from L brackets used for corner joints, with the Y-axis stand utilizing two brackets to prevent flexing in both directions, and sheet copper. The brackets provide stability for the servos during actuation, not allowing them to flex and the copper extends behind the motor to both give them more rigidity and act as a heat syncs.

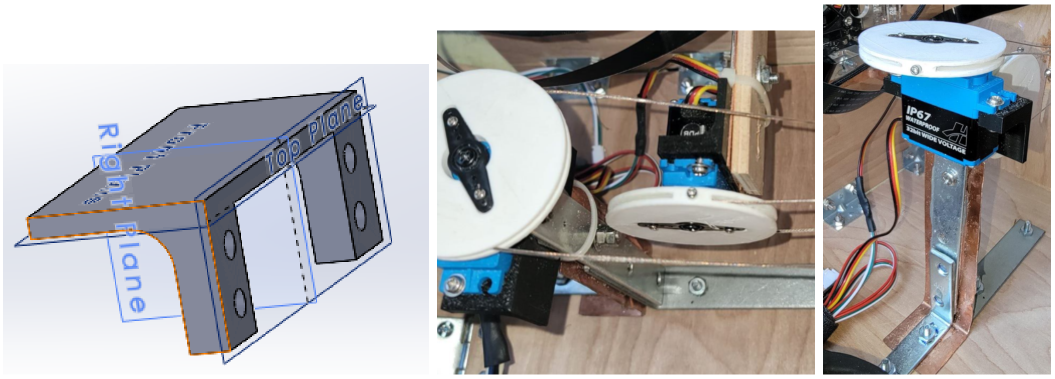


Figure 31: Servo Mounts and Stands

8.3.4 Pulleys and Wire Selection

We were unable to find a pulley system that would work for our system, so a new design was created that meets the precise specifications of our system. The pulleys are designed to allow the motor center gear to connect to an inset actuation control mount. Along with the mounting holes for bolts to secure it in place, the inset mount distributes the force across the pulley instead of the mounting bolts. A set screw holes was added to fix the wire to the pulley so we do not have to rely solely on the tension between opposite sides of the wire guides. The channels for the pulleys are deep so that the wires are not likely to slip out during actuation.

Deep sea fishing line was selected for the guide wire because it is a coated steel wire that is very strong and very flexible. The coating increases the longevity of the wire guides by reducing the abrasion caused by the metal wire rubbing against the plastic. The wires are connected at the camera mount using a fishing wire tube and beads which are pulled tight to give the most rigidity that we could manage.

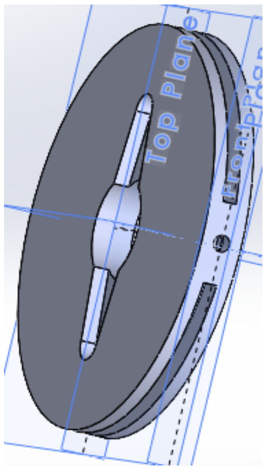


Figure 32: Pulley 3D Design

8.3.5 Case/Housing

The case was originally built out of plexiglass with 3D printed corner connectors which enabled us to assemble and dismantle the case quickly without the use of hardware such as bolts and screws. Upon testing, we noticed that the plexiglass would flex when the arm was moving. This was skewing our testing results since the amount of flexing was not constant and would likely cause the case to deform or break over time. The case was redesigned to be sturdier, using wood. This added the rigidity and strength needed to prevent excess movement from the servos.

The mounting hole for the fan was made from a piece of the plexi glass leftover as an homage to the original design as well as a way to view the inside of the case.

The case is designed with a double hinge with the PCB mounted on one of the walls. This allows us to easily access the servo motors and the ATMEGA328P chip on the PCB since with all of the components installed and mounted, which allowed us to minimize the size of the case. The hinged walls have strong workshop magnets mounted such that when the case is closed, the magnets hold the hinged walls to the rest of the container. The magnets were positioned as far away from electrical components as possible to minimize the effect of EMF on the circuit. During testing we have found that this design has not impacted the functionality of the system or PCB.

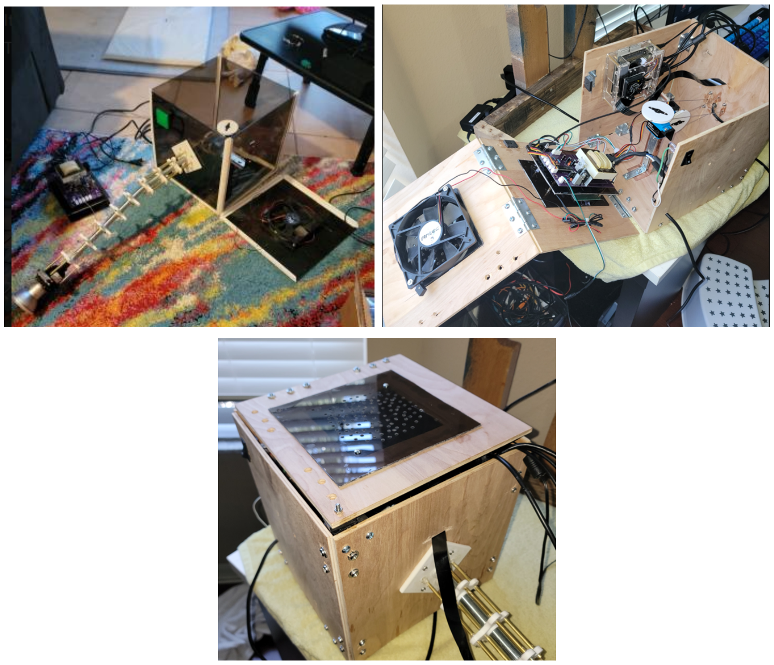


Figure 33: Original Case, Final Case (Opened), Final Case (Closed)

When fully assembled, the Articulight meets the footprint and weight specifications within a reasonable margin of error, weighing 6.3 pounds with an arm length of 23.4 inches.

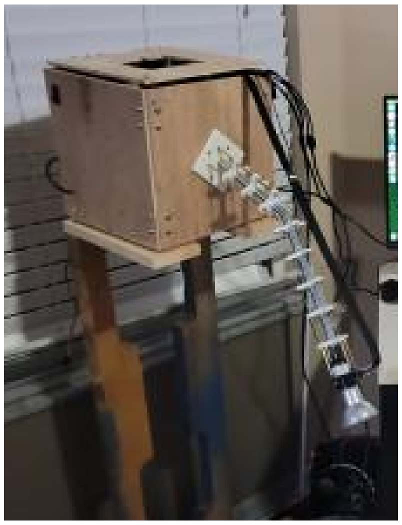


Figure 34: Complete Articulight System

8.4 Printed Circuit Board (PCB) Design

When creating our design, we implemented two different softwares to show the schematics of the connections with each of the components. First, we decided to create the schematic through a software called easyEDA.

We decided to format the components in a scheme similar to that of our hardware block diagram. As shown below in the easyEDA schematic we had our voltage source connected to both of our DC – DC voltage regulators. You can also see both regulators connected to their desired components, such as one regulator connected to the ATMEGA microcontroller and one regulator connected to the Jetson Nano.

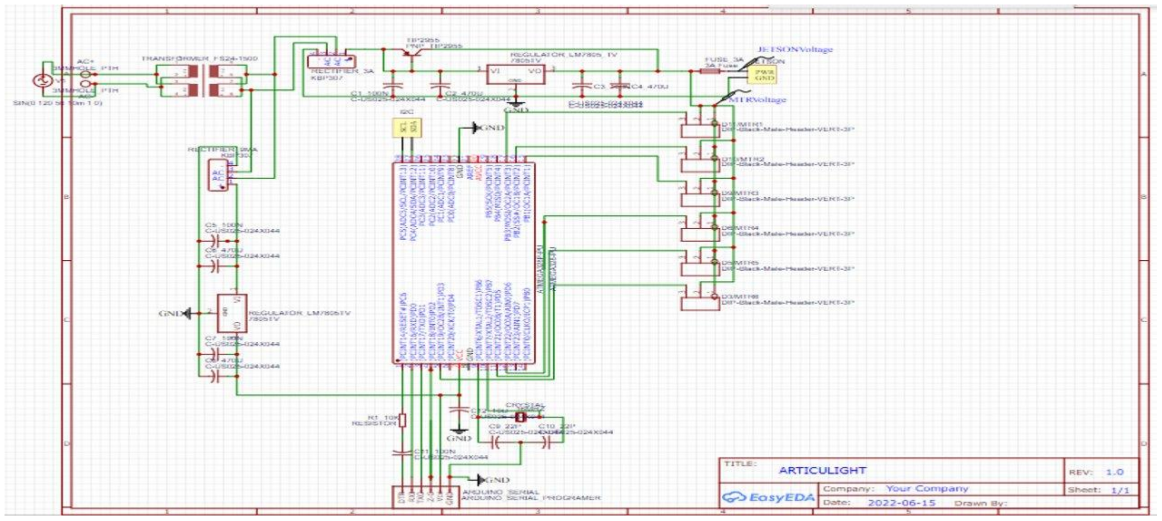


Figure 31: EasyEDA Schematic of Hardware Components

There were multiple constraints that we had to account for to be able to design all the connections for the components. For example, the Jetson Nano has a max of 5V and 4A. This means if the Jetson Nano exceeds a power input of more than 20W, then we risk burning out the board. As you will see below in further schematics, the Jetson Nano can act as the brain or master of the design which will help with cost reduction, since we won't need to incorporate a microcontroller or a motor control. Due to the constraints of the certain components it was crucial for us to implement a voltage regulator and even a DC - DC converter. But, how do we calculate how much total power (in Watts) do we need? There are two ways and it depends on how we build and connect our components. These consist of designing our circuit components in series or in parallel connection.

8.4.1 Power of Components

Knowing the power of our components is a crucial constraint to our design. One miscalculation and our design can overload and burn out our components. This is why the implementation of voltage regulators and DC – DC converters are important to overcome this crucial constraint. Below we will be finding the max power (in Watts) for each component which will help us determine the placement of our components. Power (Watts) = Voltage (V) * Amps (A).

- Wall plug-in: If 2 prong: 120V * 15A = 1800 Watts; If 3 prong: 120V * 20A = 2400 Watts
- Jetson Nano: 5V * 4A = 20 Watts
- Servo motors: Use about 1-2A when being operated - we need 2 so 2(1-2)A = 2-4A
- Motor controller: Max current is 18A for a 6-MOSFET controller, 25A for a 9-MOSFET controller, 35A for a 12-MOSFET controller, 40A for a 15-MOSFET controller, 50A for a 18-MOSFET controller
- Microcontroller Unit: 2V * 25mA = 0.05 Watts
- Relays: Assuming we use a standard 12V relay: 4 Watts
- LED light: 220 Lumen bulb = 4 Watts
- Voltage regulator / DC to DC converter: Must have at least 1300 Watts of 12 V output power
- Total consumption of power in Watts: 2400 + 20 + 0.05 + 4 + 4 + 1300 = 3728.05 Watts

8.4.2 Experimental Setup

First, we were dealing with 120V, so safety precautions needed to be taken when we were designing. To protect the circuit to the transformer, we are going to be installing a 15A fuse. This fuse would be necessary to have as it will help with safety. This should protect the circuit and sensitive equipment from damage if something were to go wrong, such as overloading. We needed a way to control the light with our microcontroller. Since the microcontroller was not capable of dealing with such high voltage, we decided to use a relay to receive the input signal from the microcontroller to turn the switch of the relay on and off, thus controlling the lights to turn on and off.

To utilize the 120V to power our NVIDIA Jetson Nano, we needed a transformer to take the 120V AC and convert it to 5V DC. The Jetson Nano has an operating input voltage recommendation of 4.75V, but recommends a power delivery of 5V. This would ensure that the microcontroller is not over voltage. The transformer also has a 5V – 12V output side that we are going to utilize for the two – servo motors that will be controlling the arms to the light.

Figure 32: Servo Motor

We are currently going with an HPS-3527SG 35kg High Torque Coreless Digital Servo motor. The servo motor has a working voltage between 5V – 7.4V, with the maximum torque at 7.4V. The output of the transformer should be more than enough to handle the power required to operate two servo motors. In the table below, we will be comparing the various motors below. We decided to choose the HPS-3527SG motor as it has a competitively fair price, can handle the most weight, has great operating voltage needed for our design, and lastly has the greatest rotational degree angle of 270.

Motor	Operating Voltage	Rotation Angle	Stall Torque	Price
HPS-3527SG	5-7.4V	270 degrees	35kg	$33.99
DS3218MG	4.8-6.8V	270 degrees	20kg	$15.99
DS3225	4.8-6.8V	180 degrees	25kg	$20.99
SB-2270SG	4.8-7.4V	100 degrees	32kg	$116.99
HST-35H	9-12.6V	240 degrees	35kg	$23.99

Table 26: Specifications of Motors

The motor has a stall torque of 35kg-cm, which means the motor will stop rotating when trying to move anything that weighs 35kg or more which is approximately equivalent to 2-lbs.

During the research phase, we realized that the microcontroller would not be able to run two servo motors. There wasn't enough power from the board to run both the two motors. For us to be able to control and have enough power to drive the two – servo motors, we would need a motor controller with an external power source to have enough power to operate the servo motors.

Servo motors in general do not need motor controllers, since they already have their own controller built in them. Specific types of servo motors just operate on power and grow an output signal that generates a pulse width modulation (PWM) signal to the controller.

The Jetson Nano acts as the brains for our design. Every second the system handles millions of calculations and with all the power consumption during the calculation, it generates lots of heat. To prevent the controller from overheating and lowering the clock speed to prevent the system from shutting down or shutting down all together. We also decided to install a fan into our design as a precaution to assist in cooling down the Jetson Nano.

It is recommended that the Jetson Nano uses a 5V PWM fan. We looked at different types of fans and decided on using the Noctua A4x20 5V PWM fan. This fan we chose has 4 pins. A 5V power supply, a ground, a tachometer and PWM. It will be mounted on top of the heat sink on the controller for our design. The Jetson Nano can supply the power and ground to the fan. The tachometer from the fan can relay to the Jetson Nano what the fan speed is depending on the current temperature of the board to cool it down.

A feature we would like to implement and test is the ability to dim the lights. For this to be successful, we will need an AC LED Bulb Light Dimmer Module as shown in the figure below.

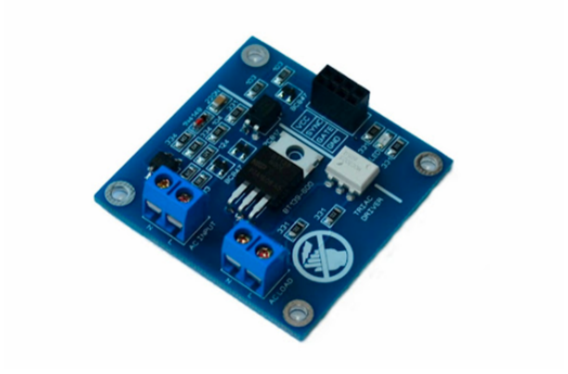


Figure 33: AC LED Bulb Light Dimmer Module

In **Figure 34** is a module that uses a TRIAC to control the dimming of the lights. They are used for resistive load lights. Incandescent lights would not work well with certain LED light bulbs, thus choosing our particular bulb was crucial for our design. The

TRIAC works by controlling the high voltage letting it pass through a regulated current letting a certain percentage of the current to flow. Wiring is simple, power is connected to the AC – INPUT side while the AC – LOAD is connected to the light bulb that you want to control.

During our research and testing/implementation phase of the LED bulb, we ran into an issue. We saw trying to dim the LED bulb that is not dimmable causes the LED bulb to flicker, as soon as the light starts to dim. The justification why incandescent bulbs do not flicker is because the filament inside the bulb is glowing red hot. When you start to dim the incandescent bulb, the filament inside starts to cool down. While during the process, the AC voltage is a sin wave. It is always changing from on to off, but you don't notice it with a traditional incandescent bulb.

LED bulbs don't have a filament inside the bulb to glow red hot. During the dimming process, the LED bulb is basically turning on and off multiple times causing the flickering you see when you try to dim non dimmable LED lights. For us to overcome this problem, we will be using dimmable LED lights.

There are two ways LED bulbs are dimmed. These two ways consist of either using pulse width modulation (PWM) or constant current reduction (CCR).

Bulbs that use PWM, it basically splits the on-time cycle by splitting it in half. The bulb still turns the LED on and off, but at a very high speed that the human eye doesn't notice the change. The benefits of the PWM are less time that the light is on, prolonging the life of the bulb and lowering the temperature of the bulb because it is not constantly on. The drawback to PWM bulbs is that it can generate noise in the circuit.
Constant current regulator (CCR), current is flowing continuously. The light output is proportional to the current output and when current is reduced, the brightness of the LED lights are also reduced.

Either way PWM or CCR can be used to dim the LED lights.

Dimming the Light was a stretch goal for our project. We unfortunately were unable to accomplish this goal during the given allocated time for our project. As much as we really wanted to add the dimmer, it caused issues with the rest of the PCB in the desired functions.

This was a stretch goal for our system which we were unable to implement, however for future iterations of our system, this will be added.

The camera is connected and powered by the JETSON Nano connected by an CSI ribbon cable which supplies power to the camera as well as allows data transfer.

8.5 Servo Control

To control the servo motors for this application we will require a pulse width modulation (PWM) signal which will be used to determine the output position of the motor. This can be done either from our main PC board, such as the Jeston Nano or Raspberry Pi, from a microcontroller, Arduino Uno, or from a servo controller. Research indicates that using the Arduino or Raspberry Pi's PWM pins, though they can perform the function desired, do not produce a clean enough signal for the accuracy desired for this project, so if these devices will be used then a servo controller will be implemented in conjunction to ensure a functional product. For digital logic this is described as a glitch which signifies the non-ideal arrival of either a rising edge or falling edge of the signal which leads to a shift in the signal timing. These shifts, usually in the microsecond range won't result in a large drift from the intended position. However, when paired with continuous operation and movement these miniscule drifts from the target will eventually start having an impact on the macro scale, leading to larger and larger inaccuracies, be them timing or position. This is why having a clean signal is vital to the operation of this device.

The Jetson Nano by itself has the capability to produce clean PWM signals needed to accurately control the servo motors for speed and accuracy. The position of the servo motors is determined by the PWM signal, also known as the RC Servo Pulse, sent through the control wire. The parameters of the pulses that are sent to the motor defines the position of the rotation, which will vary depending on the constraints of the motor itself. For this project, a 180 degree rotation will be selected because it will give a full 90 degree flex, i.e. parallel to the ground for the hanging position, on both x and y-axis of rotation. A typical servo motor uses a 2.0 msec, or 50 Hz, pulse width as the center which can be defined as the 0 degree position on its particular axis. A smaller pulse width will cause the servo to rotate counterclockwise to the -90 degree position whereas a larger pulse width will cause the servo to rotate clockwise to the 90 degree position. These constraints will change based on the servo selected to control the actuation, but the overall logic should remain the same in controlling the movement of the light.

The first portion of the algorithm will be to map a fixed area for the camera, based on pixel density, to the movement of the servo motor. This is done so that we can have the program automatically adjust the angle of movement based on how far away the gesture is. The data received from the detection algorithm will be read as serial data to allow for a higher throughput of data, which will then be converted to an integer angle value. Once the angle value is determined, the relay to the motors will be activated to supply power to the motors. This design decision to use these relays is to prevent constant power to the motors which will reduce the overall power draw from the system. Once activated, the angle value will be put into a function that will determine the PWM signal to move the light to that location.

After researching how a servo motor is controlled works with PWM, we experimented. Using an oscilloscope and connecting a servo motor through a PWM signal, we saw it create a square wave with a specific period of duty cycle. Shown below in **Figure 35**, it shows the duty cycle percentage of each PWM signal. The percentage of each duty cycle represents the period that the signal is on. We tested the servo motor at a 25 percent duty cycle and a 50 percent duty cycle. When the PWM is at a 25 percent duty cycle over a 1 millisecond period, it is on for 225 microseconds before turning off. When the PWM is at a 50 percent duty cycle over a 1 millisecond period, it is on for 500 microseconds and so forth.

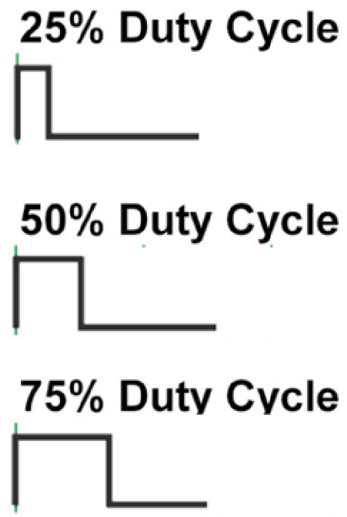


Figure 34: Duty Cycles of PWM

Moreover, we are going to be experimenting with the PWM on a servo motor by varying the duty cycle, thus changing the amount of time that the signal is on the servo motor. The specifications of the servo motor we are currently testing has three wires, power, ground, and signal and/or data wire. The signal or controller wire is going to be hooked up to an Arduino nano digital pin number three.

8.5.1 Stability Control for Servo Motor

Stability control for the servo motor is very crucial. Stability is important because servo motors operate with continuous current to reach a specified position, velocity, or torque. The precise amount of current to be delivered to the motor is determined by the servo controller used, which is based on information supplied by an encoder regarding the system's actual condition such as position, speed, and/or torque.

The servo controller sends these specific current commands to the servo drive. The servo drive is also referred to as an amplifier. This provides the servo motor with precise current that is needed to adjust and vary for any differences between the command value (given value of position, speed, or torque) and the actual value. The current command is crucial because it relies on accuracy. This is important because if the servo motor doesn't receive the necessary torque (either too little or too much) to drive the system to the intended position it is set to.

8.5.2 Information about Controllers

Further stability control of the servo motors may be warranted because the automation aspect of this project will require a feedback system where the actuation system will generate data for the position of the light which will be reprocessed by the machine vision algorithm. This feedback system allows us to use the closed loop system and design a controller that can compensate for the imprecise movements or imperfect PWM signal. Due to the nature of this system which only has motors at one end that will be used to move the length, for simplicity, the system will be modeled as a robotic arm with a lever. This simplifies the system for a macroscopic method of viewing the system kinematics and taking measurements to determine the best type of controller for this device. The types of controllers include, Proportional, Integral, Derivative controller algorithm, which are generally designed as P, PD, PI, and PID controllers, a Lead controller/compensator, a Lag controller/compensator, and if needed, a feed forward controller. Each component affects the output of the system's rise time, settling time, overshoot, and steady state error, experienced by the servo motor position when the PWM signal is applied and the actuation has started.

8.5.2.1 Proportional Control

This component allows for the correction value in proportion, P, to an expected error. This can be an amplification or attenuation. This controller reduces steady state error but increases overshoot and can produce instability at high proportional gains.

8.5.2.2 Integral Control

An integral action, I, has the greatest effect on reducing steady state error and even eliminating it all together. This can be thought of as a running sum of previous errors, generally errors left over from a proportional control. This type of control is generally essential in eliminating steady state error which will directly impact the accuracy and speed of our actuation; it can deteriorate the dynamic response of the system by increasing any overshoot and increasing the settling time. This is important to our system because the motors require a PWM signal, these are far shorter than a typical step input which means the system will need to respond within the delay given for movement.

8.5.2.3 Derivative Control

The derivative control, D or can also be described as a velocity controller, V, makes a system better damped and more stable. This is done by finding the difference between any current error and any previous error and will cause the output to adjust accordingly. This control allows for a quicker settling time and decreases the maximum overshoot while applying small increases to rise time and steady state error.
How the PID controllers respond are relative to the frequency in which they will be operating. Derivative control is most effective at higher frequencies, integral control is most effective at low frequencies, and proportional at moderate frequencies. The

frequency that will be used for our design will likely be in the moderate range of 20 kilohertz which translates to a PWM signal having a period of 20 milliseconds. This is a reasonable response time frame for a step input.

8.5.2.4 Lead Compensator

This controller increases the speed and/or stability of the system. When viewing the system transfer function from the root locus plot, a lead compensator shifts the root locus to the left which is what increases the speed of the response and increases the region of stability for the system. This controller can also increase the gain of the system at higher frequencies which will contribute to the increased response speed of the system, decreasing rise time and settling time. The cost of using this control method is that the gain that is experienced by the system can also amplify the noise experienced by the system causing phase shifts in timing which can accumulate over time, even though the system will ultimately readjust based on the feedback from the camera, this can potentially add significant time to the overall actuation time specification.

8.5.2.5 Lag Compensator

The lag compensator in essence works opposite to the lead compensator. When viewed as a root locus plot, this device will shift the root locus to the left. The effect this has on the system is that it will increase the rise time, increase the settling time, and shift the system so that it reduces the region of stability and can in fact push a system to be unstable. The advantage to using a lag compensator is that it can change the steady-state response of the system. This has a unity gain at moderate to high frequencies and will have gain at the lower frequencies. This causes all of the position, velocity, or acceleration constants, generally identified as Kp, Kv, or Ka, to be adjusted at the same time. This has the effect of causing the steady-state error of the closed-loop system to be decreased while not causing much change in the transient response of the system.

Each controller has advantages and disadvantages to how they affect the system and should be considered for the best combination that will offer the best arm movement at the lowest component and power cost.

8.5.2.6 Feed-forward Gains for Highly Dynamic Applications

The PID controller gains as described above are reactive corrections to the system’s behavior over time. If we wanted to make a highly dynamic application, or a system with better responsiveness, another type of gain – referred to as feed-forward gain – works outside the feedback loop and provides a proactive approach to error correction by predicting the commands needed to achieve zero error. These types of gains are either classified as velocity feed-forward or acceleration feed-forward. In **Figure 37**,it shows a model of the feed-forward gain controller.

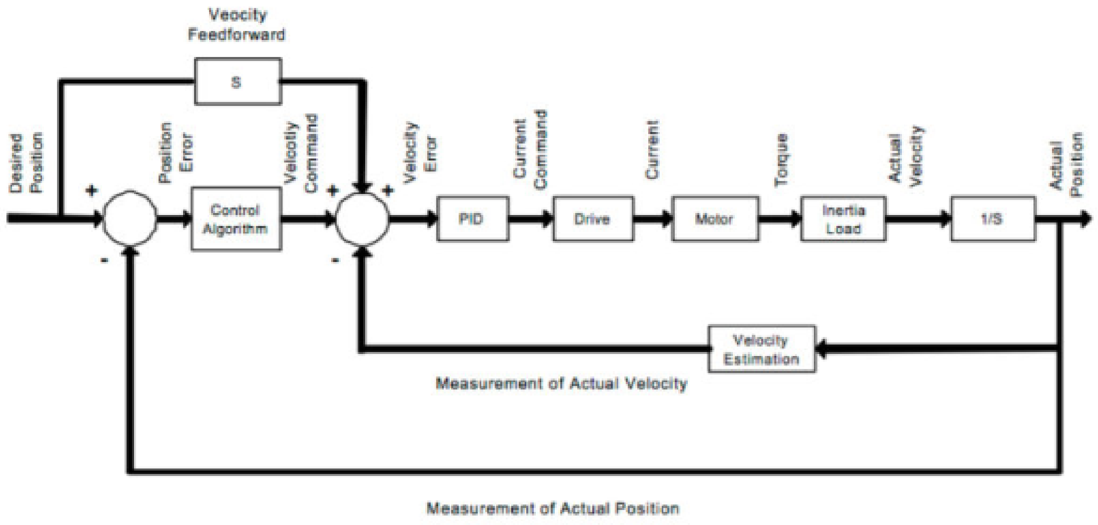


Figure 35: Feed-forward Gain Controller

Velocity feed-forward works during the constant velocity portion of the move profile. This minimizes error and helps improve response time by multiplying the derivative of the position command by the velocity feed-forward gain.

Overshoot caused by the velocity feed-forward can be eliminated by acceleration-feed forwarding. Acceleration feed-forwarding works during both the acceleration and deceleration portions of the movement. Furthermore, acceleration feed-forward multiplies the second derivative of the position command by the acceleration feed-forward gain value.

8.5.6 Servo Tuning via PID Loops

Servo tuning via PID loops involves a system that involves adjusting the gains in the motion controller to minimize the servo system's response time, settling time, and overshoot. The goal of servo tuning using a PID controller is to minimize the error between the commanded position, speed, or torque and actual value achieved. 'P' refers to proportional gain, 'I' refers to integral gain, and 'D' refers to derivative gain. A gain is a ratio of output to input or V_o / V_{in}. In a servo control loop, the "gains" determine how and to what extent the controller tries to correct the errors detected by the feedback device.

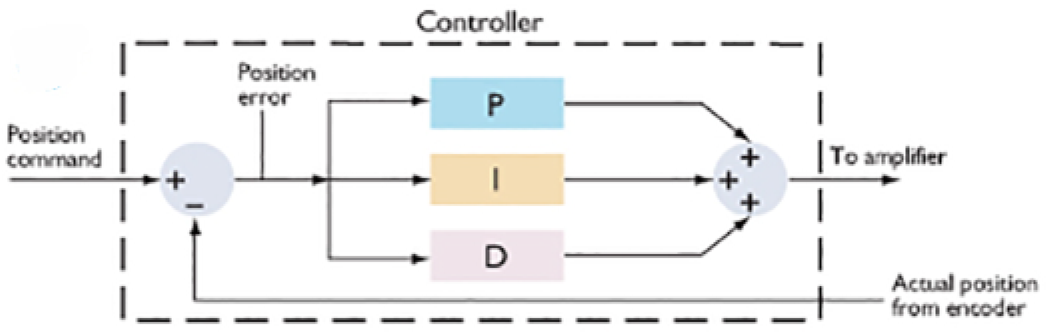


Figure 36: PID Controller Layout and How it Works

The amount of proportional gain in the control loop allows us to determine how much restoring force is applied to overcome the error between the command value and actual value. This proportional gain is multiplied by the error and generates the contribution to the output for the next time period. We use the term “proportional” because the amount of restoring force is directly proportional to the amount of error at any instant in time.

We use the integral gain to “push” the system to zero error at the end of the move. The integral value increases with time, however, due to the integral gain increases at the end of the move, it can cause the system to oscillate or overshoot. If the integral gain is too low, the system will have a slow response time. This “gain” is mainly used when the system is subjected to static torque loads.

Derivative gain is proportional to the rate of change of the error. It is primarily and often used in conjunction with proportional gain to reduce overshoot and provide a damping factor. However, a derivative gain that is too high can reduce system response time and cause oscillations.

8.6 Modeling the System

The ArticuLight actuation arm will be attached to gears, controlled by servo motors, generally called the load gear of the system. The flexibility and freedom of movement afforded by the LocLine style segments is what allows us to represent the system as a flexible link. This system is an electromechanical system with the servo motors controlled by a DC voltage, Vm. The mechanical aspects of the arm requires us to consider the torque produced by the motor combined with the damping experienced when the system experiences oscillations after movement. Using a model that best represents the components we plan on using, values for the stiffness, modeled as a linear spring, of the arm can be extrapolated and used to create a control system to ensure our servo motors operate within design specifications that we determine to be necessary for the quick and precise operation of our project.

8.6.1 Calculating Torque

A general model for this system was created to work out the kinematics of the system as well as calculate the torque that the motors will need to generate to compensate for the load applied from the arm, 24 inches, with a weight of around 3 pounds.

There are three different methods in calculating torque. The first method we will be looking at is how to find torque through perpendicular forces. First, we need to find the length of the moment arm. The distance from the axis or rotational point to the point where force is applied is called the moment arm. Next, we have to work out the force being applied perpendicular to the moment arm. This is done by using Newton's Second Law (F = m x a). Once we know the force, we can then calculate the torque by multiplying the force with the distance (or radius → **τ** = F x r). Lastly, the direction of the force can show whether our object is rotating positively or negativity.

Method 2 is calculating torque for angled forces. To start this method, we need to start with the distance of the radial vector. The radial vector is the line that extends from the axis or point of rotation. Next, we will then work out the amount of force being applied. The amount of force applied is measured in Newtons and is applied at a specific angle to give us our radial vector. Following that, we then want to calculate the angle made by the force and radial vector. When calculating the angle θ, we want to find the sine of that angle. Lastly, to calculate the torque we will multiply the force, distance, and sineθ (**τ** = r*F*sineθ).

The last method we will be looking at is determining torque with moment of inertia and angular acceleration (Method 3). First we have to determine the moment of inertia. This can be done using the formula 1/2(MR^2). Following we have to determine the angular acceleration α, which is the amount of velocity that is changing as an object rotates. Lastly, to find torque using this method we multiply the moment of inertia (I) by the angular acceleration (α) → **τ**=Iα

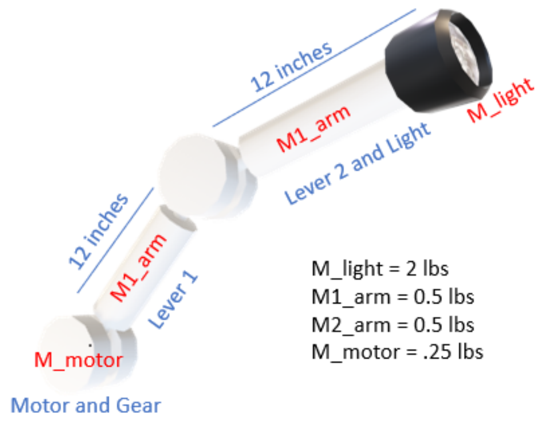


Figure 37: ArticuLight Model for Calculating Torque

The robotic arm we are built has a similar structure except with multiple joints that are controlled via the control wires. This model is a simplified version of that system which is a worst case calculation for the torque due to the motor compensating for the entire length and weight with only one joint to consider. This represents a full, straight arm that is moved. Each one of the values given and the calculation performed is not precise but will be used as a starting point for the actual implementation. The parameters used to calculate the torque are the length of the arm, signified as Length Lever 1 and Length Lever 2, in this case it will be considered as the full 24 inches, the weight of the system, including the M_light, M1_arm, M2_arm, and M_motor, and finally the joint rotational acceleration.

In order to allow our arm to move a full 180 degrees at worst case scenario within 3 seconds requires a rotational acceleration of approximately 45 degrees per second which should allow for enough time to complete the movement. The calculated torque comes out to be 339 lbs per inch or 60.5 kg per cm which are the torque values typically seen on motor datasheets. These torque values mean that the motor can move 339 pounds at one inch from the axis of rotation. This value is cut in half every inch which is why a 3.25 pound system needed a motor that can handle the relatively small weight across the length. This can further be adjusted by adding a gear system to the motor which increases the load that the motor can handle.

8.6.2 Develop the State Space Representation

The state space representation takes a model of a system or plant and allows us to see how that system reacts to inputs which can be used to determine the stability and controllability of the system. Our design has two servo motors connected to a base which then connects a load gear to the arm in order to control the pitch and yaw of the arm. Because of this setup, the system is modeled as a rotary flexible link with a high stiffness, so the parameters can be seen in **Figure 38**:

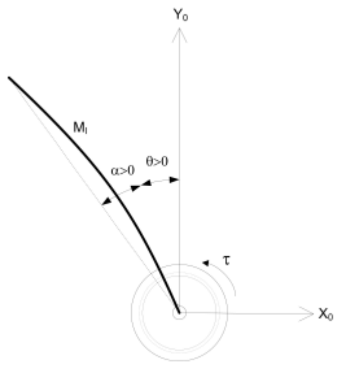


Figure 38: Flexible Link Rotation and Torque Model

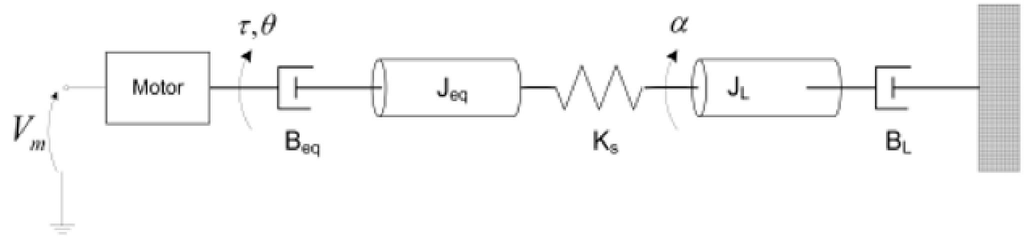


Figure 39: Flexible Link Free-body Diagram

These images most accurately capture the system we are trying to replicate and allow for an accurate representation to develop the state-space model which will be used to determine the control methodology for this device. This will be done by using the kinematics of the model and freebody diagram to derive the Legrangian equation to relate the potential energies of the system with the kinetic energies. The first step is to define the variables present in these models, noting that not all values will be able to be used or determined until the selected parts are identified and connected to a tachometer to establish the overall stiffness of the system. From there the state-space representation can be derived and gains for controllers can be calculated and implemented if it seems necessary to keep the articulation of the system smooth and precise. The above Figures were discovered and derivation was enabled from the *Rotary Flexible Link Workbook, Student Version by Quanser Inc. 2011*. The data from this workbook was adapted to derive the values required by the arm we are developing.

Parameter	**Symbol**	**Unit**
Flexible Link Mass	M_l	kg
Servo Angle	θ	deg
Deflection Angle	α	deg
Load Gear Torque	τ	N*m
Control Voltage	V_m	V
Viscous Friction of Servo	B_{eq}	N*m/(rad/s)
Viscous Damping Coefficient of Flexible Link	B_L	N*m/(rad/s)
Moment of Inertia	J_{eq}	kg*m^2
Center of Mass for Flexible Link	J_L	kg*m^2
Total Length of Flexible Link	L_T	m
Flexible Link Stiffness	K_S	N*m/rad

Table 27: Parameters Specified by Symbols and Units

For the parameters listed above, some of the values will have to be assumed while others will need to be tested for.

The flexible link mass, M_l, will be estimated at 3.25 lbs, or approximately 1.5kg to account for the weight of the arm, wires, and light that will be used this value can vary but once components are purchased and testing is done, this parameter is vital in calculating the torque of the system for the servos.

The servo angle, θ, is the angular position we want the servo to move to, is one degree at a time. That value was mapped out in the software, however this parameter is variable based on the PWM signal that is communicated to the servo, and is not needed for the calculations but will be used to verify the output angle is accurately executed.

The deflection angle, **α**, is the amount the link flexes away from our center position once a movement has been completed. These values are extremely low degrees due to the rigidity of the material and structure of the arm, but are accounted for similarly to the servo angle.

The load gear torque, **τ**, is the force experienced by the servo motor to move the system. From estimations calculated, the torque that is used in simulation is 60.5kg*cm, or approximately 6N*m. This torque is generated from the servo motor voltage, V_m.
The viscous friction of the servo, B_{eq}, which is the friction experienced by the servo motor when there is no load, and the viscous damping coefficient, B_L, which is the friction experienced by the servo motor when under load. Oppose the torque and movement generated from the servo motor. B_{eq} was determined from the datasheet from the servo that is selected, though research values for this are generally less than 0.05 and due to the highly flexible nature that the arm ended up being, a conservative number for B_{eq} will be selected for simulation and B_L will be set to 0.
The moment of inertia, J_{eq}, also known as the angular mass or rotational inertia, is the quantity that determines the torque needed for a desired angular acceleration about a rotational axis or the amount of force needed to achieve a desired acceleration. This is another parameter that assumes no load and will be found in the datasheet of the servo motor. The moment of inertia, J_L, of the link will take the link as a rod in determining this value compared to that or J_{eq}.

The total length, L_T, of the link is approximately 2ft, or about 0.6m. This represents the entire arm including the light and any components from the base where the motor and gear will be mounted.

Finally, the flexible link stiffness, K_s, is the value that determines how rigid the link is. This value is set based on the spring material used for the arm. This can be determined by attaching the arm to the servo motors and applying a step input and observing the response with a tachometer. Taking the natural frequency, this can be used to determine the stiffness of the link. We were unable to accomplish this and as such could not completely account for the stiffness of the arm, however because a spring was used instead of a wire loom, this value was estimated to be very low.

This system can be viewed as a linear system and as such can be represented with the state space equation:

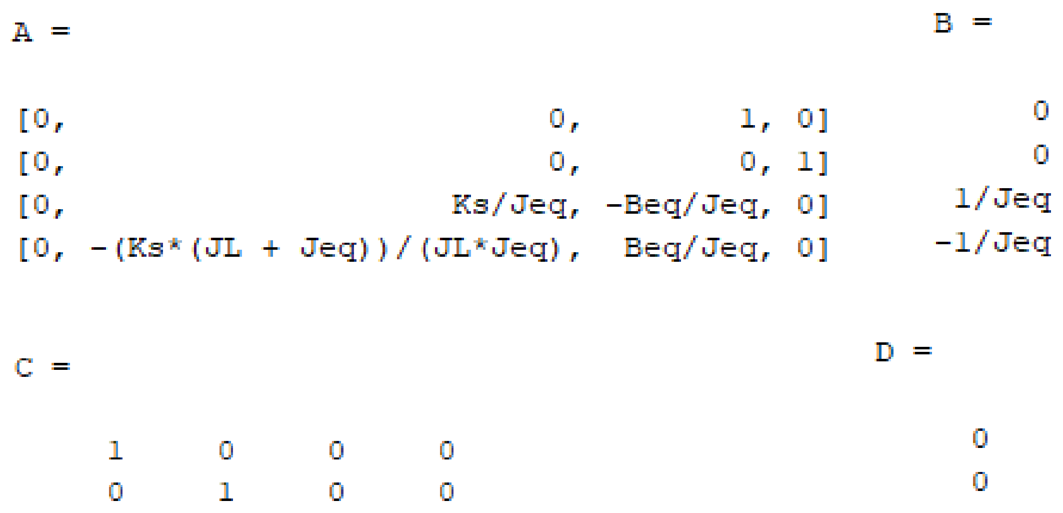
$$\dot{\mathbf{x}} = \mathbf{Ax} + \mathbf{Bu}$$
$$\mathbf{Y} = \mathbf{Cx} + \mathbf{Du}$$

These equations are a set of matrices that can be used to model the system, where **A** is the system, or state, matrix. This vector describes the system or plant. **B** is the input matrix, **C** is the output matrix, and **D** is the feedthrough or feedforward matrix. These matrices are used to define the state-space vector matrix, $\dot{\mathbf{x}}$, and the output vector, **Y**. The advantage of this method is that the state-space representation allows us to establish a model of the system and operate on it within the time domain, but also offers a quick way to move to the laplace and frequency domains readily to determine stability of the system.

Using the above parameters we can deduce the transfer function of the system:

$$G(s) = \frac{-J_{eq}J_L s - B_{eq}J_L - (J_{eq}+J_L)K_s}{J_{eq}^2 J_L s^3 - B_{eq}^2 J_L s + (J_{eq}+J_L)K_s^2}$$

Converting this to the state-space representation, using MatLab, we were able to derive the state-space representation:



```
A =                                                          B =

[0,                          0,          1, 0]                   0
[0,                          0,          0, 1]                   0
[0,                    Ks/Jeq,   -Beq/Jeq, 0]               1/Jeq
[0, -(Ks*(JL + Jeq))/(JL*Jeq),    Beq/Jeq, 0]              -1/Jeq


C =                                                   D =

     1     0     0     0                                  0
     0     1     0     0                                  0
```

In order to create a representation of the system, we decided to go with some conservative values for the servo controller torque and mass. These values will be updated as the design changes and new data becomes available. For the purpose of development, estimated values for J_{eq} from the selected servo motor, wll be 0.0035, Ks of 1, and B_{eq} of 0.005 for virtually no damping. This shows us a state-space representation that can be plotted in order to determine an estimated natural frequency to generate a closer representation of the actual system. The initial values of the state-space representation are:

```
A =

     0         0    1.0000         0
     0         0         0    1.0000
     0  285.7143   -1.4286         0
     0 -287.5661    1.4286         0
```

```
B =

         0
         0
  285.7143
 -285.7143
```

```
C =

     1     0     0     0
     0     1     0     0
```

```
D =

     0
     0
```

From this, the characteristic polynomial is determined to be:

$$G(s) = \frac{-0.00189s - 0.5462}{6.615\text{x}10^{-6}s^3 - 1.35\text{x}10^{-5}s + 0.5435}$$

The eigenvalues, or closed loop poles, can be determined from matrix A:

```
Poles =

   0.0000 + 0.0000i
  -0.7097 +16.9425i
  -0.7097 -16.9425i
  -0.0092 + 0.0000i
```

```
Zeros =

 -288.9947
```

All of the poles are on the right half plane of the real and imaginary axis which means that the system is already stable. This is an advantage because all that is needed is to fine tune the response such that it stays stable but has the steady state response we want. The second design specification for the controller would be a very fast settling time, ts, which would decrease the step time between movements for our system allowing us to meet the 3-5sec total movement time specification for our system. A 5% percent overshoot, PO, was selected just as a measure to ensure the server system won't be damaged when these gains are applied. Finally, the torque of the system was determined from using a 12V DC input because this has been a common value found for high torque servo motors to generate enough power to move the weight expected for the system.

The design specifications selected to determine how the system will be controlled are arbitrary due to the lack of information for specifics of the hardware that will be used.

However, the values selected will be conservative to ensure a good output for the system. The most important will be the steady-state error, ess, of zero. We want a very precise movement of the servo position so having no error is a key parameter to achieve this.

Applying a step response to the system, as would be from the PWM signal, a plot of the system dynamics can be estimated.

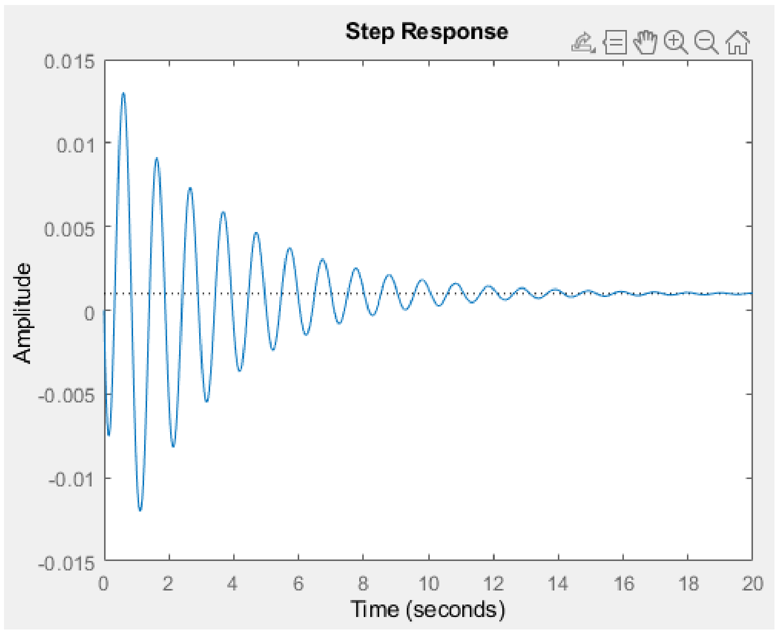


Figure 40: ArticuLight Plant Step Response

The natural frequency can be determined to be approximately 13.8rad/sec. Which will translate to a K_s of 98.27. When added to the system dynamics, the matrices change to:

```
A =

   1.0e+04 *                                            B =

         0         0    0.0001         0                      0
         0         0         0    0.0001                      0
         0    2.8077   -0.0001         0               285.7143
         0   -2.8259    0.0001         0              -285.7143
```

Which changes the characteristic polynomial equation to be:

$$G(s) = \frac{-0.00189s - 53.41}{6.615\text{x}10^{-6}s^{3} - 1.35\text{x}10^{-5}s + 5249}$$

with poles and zeros:

```
Poles =

   1.0e+02 *

   0.0000 + 0.0000i          Zeros =
  -0.0071 + 1.6810i
  -0.0071 - 1.6810i
  -0.0001 + 0.0000i            -2.8261e+04
```

From this data, it is possible to design controller gains for the system using a linear quadratic regulator, LQR, which enables us to design a dynamic controller at minimal cost. This means that the performance index is minimized, tends towards zero, at the same time, the required input for the system is minimized. This is achieved from the Riccati equation:

$$\mathbf{A^tP + PA - PBR^{-1}B^tP+Q} = \mathbf{[0]}$$

Using a diagonal Q matrix:

```
Q =

     1     0     0     0
     0     1     0     0
     0     0     1     0
     0     0     0     1
```

where **P** is the unique positive definite solution to the discrete time algebraic Riccati equation. Once solved, we can use it to establish the gains required to meet the desired performance index.

```
P =

    1.4482    0.0292    0.5486    0.5451
    0.0292  140.2763   -0.5028   -0.4996
    0.5486   -0.5028    0.7919    0.7868
    0.5451   -0.4996    0.7868    0.7867
```

The gains needed to meet the specifications outlined are

```
K =

    1.0000   -0.9259    1.4432    0.0292
```

The P and K matrices are what determine the values of an optimized Full-State Feedback controller. This creates a static gain, keeping the order of the closed-loop system the same but enabling us to meet the specifications for our system. Once this baseline is implemented, we can fine tune these values as well as the input **R** to generate the optimal response for the system. Furthermore, the addition of an LQR controller would only require a circuit to be added in series between the motor controller and the servo motors themselves with power routed to them from an analog pin from either the Jetson or microcontroller if the power demand is low enough which the results above show that the gains needed should be within the voltage that can be supplied via analog pin.

The overall servo control method that was used is in essence a P controller, for both stepping through the angles during the follow gesture and having a direct mapping movement of the move to and reset gestures. During the latter of the two movements, this control adds oscillations to the system, but still manages to have a zero steady state error. Future efforts will implement a Derivative controller at least to slow and smooth the movement of the arm to reduce the oscillations but maintain the zero error.

8.7 Communication Between Devices

Communication between the computer vision computational device (Jetson Nano) and any other devices are critical in order to have a functioning light system. This section outlines some of the more popular communication protocols that are used today.

The NVIDIA Jetson Nano Developer Kit contains the capability to use the UART, I2C, and SPI communication protocols. The protocol that we chose to use for this project in communication between the Jetson Nano and the ATmega328 microcontroller is UART through the Serial packages in Python and C++ respectively. We chose UART because it was the simplest method to get running and is fast enough to achieve responsive communication in the application of our project.

8.7.1 UART Communication Protocol

The Universal asynchronous receiver-transmitter (UART) communication protocol is the oldest communication protocol outlined in this document. It can also be considered the simplest of three communication protocols outlined as well. In UART communication, two devices are connected in an integrated circuit where each device contains a transmission line and a receiving line. One device's transmission line is the other device's receiving line, shown in **Figure 41**.

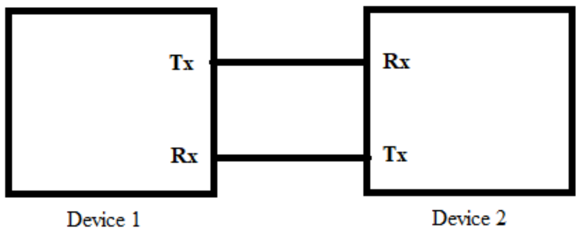


Figure 41: Simple UART Communication Scheme

Since this communication protocol is asynchronous, the transmission lines do not contain a dictated clock signal, and so the two devices must agree on the frequency of sent information called the baud rate. If the baud rate is different between the two devices, the information that is broadcasted through the lines will not be received correctly and garbage data will be read. A baud rate of 1000 indicates 1000 bits of information sent or received each second. In order to effectively communicate, the UART protocol begins with a single start bit and ends with a single end bit. Additionally, there is the parity bit that is broadcasted to ensure no erroneous transmission. As you will see in the pseudocode for using UART with these devices, these caveats happen under the hood and mustn't be managed by the programmer. However, the baud rate is indeed selected by the programmer.

For the purpose of communication with the NVIDIA Jetson Nano and Arduino Uno, the UART protocol will be used. Although SPI is the fastest communication protocol outlined here and I2C is faster than UART, UART is considerably easier to use than any of the other protocols. Since the Jetson Nano will be administering commands to the Arduino Uno that do not require speedy and constant communication (issuing motor and light commands), UART is a satisfactory choice for communication.

8.7.2 I2C Communication Protocol

Inter-integrated circuit (I2C) is a communication protocol that was developed in 1982 by Philips Semiconductors.

As such, pseudocode is supplied below for each device in how these communication protocols may be used. Because the NVIDIA Jetson Nano Developer Kit will be controlling the behavior of the Arduino Rev3 device, the Nano is considered a master device in their communication.

Unlike the serial peripheral interface the Inter-integrated circuit can use more than one master device in the circuit. communication only requires two lines instead of four like serial peripheral interface which minimizes connection so this means fewer pins or wires in the circuit cutting down the cost. It is less complex which does not need any chip select lines to add extra devices like the SPI which makes adding extra devices to the system easy to implement. When it comes to debugging the hardware it's easier to diagnose because it's easier to trace the problem. hardware can be added or removed without affecting the circuit on the bus system.

A disadvantage to the Inter-integrated circuit system is increases the complexity when you start adding master/slave device into the circuit. it's not a full duplex system but rather it's a half duplex system meaning the signal won't happen simultaneously which means communication can move in both transfer and receiving just not at the same time like SPI system.

8.7.3 SPI Communication Protocol

The serial peripheral interface (SPI) is a communication protocol that was developed by the Motorola company in the 1980s. It is extremely common among communication between embedded devices. It is one of the fastest industry standards in communication lines between devices.

We use a serial peripheral interface to transfer data from one electronic device to another electronic device since it is the most popular and standard communications protocol. The serial peripheral interface communicates between microcontroller and sensor shift resistor and so on. one reason we use a serial peripheral interface is because it is fast, around 8 megabits per second or more and supports higher clock frequency as well.

The advantage of using a serial peripheral interface is that it's a simple protocol, supporting full duplex communication. In full duplex communication data can be sent over and received over one channel or 4 lines without waiting for data to be received before sending the data over. Serial peripheral interface also consumes less power.

Some drawbacks are there is no acknowledgement or handshakes for its protocol. adding multiple slaves makes it more complex and there is no in herent arbitration.

8.7.4 UART Communication With Jetson Nano

The Jetson Nano has two pins capable of UART communication on the board and has the capability to establish serial COM ports through USB and through the use of 2 different headers on the pins. J41 is a free pin, but the other UART pin (44) is assigned as the Serial Debug Console. In order to perform simple UART communication on the Jetson Nano, the serial debug console service must be stopped, as it will prevent the protocol from executing with another device that uses the protocol in a bare bones manner.

For our system we were able to connect the ATMega328P TX and RX pins to the Jetson Nano using a USB adapter. This enabled us to use serial communication over the /dev/TTYUSB0 port on the Jetson Nano. Upon startup, the ATMega328P establishes a connection with the Jetson Nano with a baud rate of 9600 then polls the Serial Buffer until there is data on the BUS. From there, we have the ATMega328P parse the data byte by byte and extract the X and Y coordinates and perform actuation based on what the gesture requires.

9 Software Design

This section outlines the software design for both the NVIDIA Jetson Nano using Python and the ATmega328 microcontroller using C++. The Jetson Nano made use of three python scripts on top of the code that was forked from NVIDIA to perform the gesture recognition: image_collection.py, svm.py, and articulight.py. Each of these scripts and their functions are outlined in the below sections.

9.1 Software Design for Jetson Nano

The software for this project running on the NVIDIA Jetson Nano is built on the project trt_pose_hand from NVIDIA. Although we would have preferred to build a hand detection model from scratch, there simply was not enough time. The software on the Jetson Nano is composed of three python scripts: image_collection.py, svm.py, and articulight.py. Each of these files are outlined in the sections below.

9.1.1 image_collection.py

Written for gathering a dataset for the program svm.py, it loads a GUI using the PySimpleGUI library to navigate saving images to the Jetson Nano and which gesture the image corresponds to. When the script starts, it prompts the user for the name of the dataset and tells the user the path that it is going to save the dataset to. Once the name of the dataset is inputted, the number of gestures is then prompted. It loops through the number of gestures and gathers the names of the gestures from the user. Then, the GUI is opened.

Shown in Figure 42, the GUI has buttons for the following: Save Image, Save 600 Images, Save 100 Images, Next Gesture, Previous Gesture, Toggle Train/Test, and Exit. When any of the save image buttons are pressed, image(s) are saved to the dataset directory in their own directories depending on the gesture it represents. The next gesture and previous gesture buttons are for selecting the gestures the user wishes to save images to. The toggle train/test button allows the user to save images to a training and testing dataset. Once the exit button is pressed, the program cleans the directories up, renames them to iterative names (1, 2, 3, etc) and creates a json file for the labels of the respective gesture for each image. Then, this directory is used in svm.py to train a support vector machine

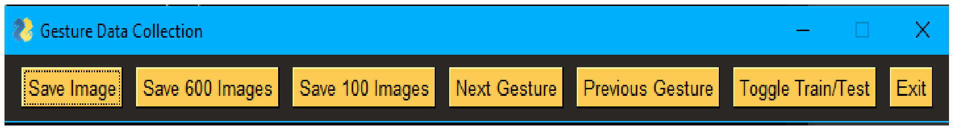


Figure 42: image_collection.py GUI

9.1.2 svm.py

A support vector machine is used in this project to classify the data that the hand model returns when a hand is passed into it. Svm.py takes the dataset images and trains the returned values of the hand model supplied by NVIDIA. After the svm model is trained, a json file is created with the respective labels of the hand gestures, as an svm outputs a number representing the classification. The svm model is saved to the project directory to be loaded in another script later through pickle, a file loading python library.

Once the svm model is trained, it is loaded and the camera is booted up to test the performance of the model. An issue that we faced in this software is that the svm model seemed to randomly assign gesture classifications and the programming of the json file would not result in expected behavior. To overcome this, we named the gestures that the json file mapped to the svm outputs as ‘one’, ‘two’, ‘three’, etc and mapped them manually through trial and error. If more time was available, this would be investigated further and fixed if possible.

9.1.3 articulight.py

ArticuLight.py uses the trained SVM model from svm.py to achieve real-time gesture detection. It passes each frame into the hand model and the outputs of the hand model into the SVM model, and performs actions based on these gestures. The gestures and their actions are outlined in the gesture section. An issue that we came across developing ArticuLight.py was that the SVM model would identify a gesture that was not intentionally supplied by the user, and act on a single erroneous gesture. To overcome this, we added a gesture buffer that keeps track of the previous gestures. For a gesture’s behavior to process, it must be held for one second. Because the program processes frames at a constant frequency, in this case 8 frames per second, we can know that if the buffer has the same gesture 8 times in a row, the gesture was held for one second in real time. Therefore, instead of taking the gesture and performing an action based on the gesture, we check if every gesture in the buffer is the same and compare that gesture with the expected behavior.

This script establishes a serial communication with the ATmega328 and issues commands to the microcontroller to move the servo motors depending on where the gesture is in the frame. If it is in the center of the frame with a 50 pixel error, it does not send these commands. To see the exact behavior of this program, see the Jetson Nano software block diagram section.

9.2 Software Design for ATMega328P Microcontroller

The ATMega328P code established serial connection with the Jetson Nano upon power on for the microcontroller. This is established once power is applied to the PCB. Due to the minimal power requirement of the ATMega328P, we determined that it wasn’t necessary to have an interrupt function to turn on the microcontroller.

9.2.1 Arduino_Articulight.ino

This script was programmed in the Arduino IDE. The program initializes the servo motor pins used for our system and connects them. Then a serial connection is created between the Jetson Nano and the ATMega328P with a baud rate of 9600 bits per second. From here, the ATMega218P polls the serial buffer until any amount of data is in the memory. Then, the first byte is read.

If that byte is the char ‘x’, this means that the gesture recognized was for the follow me function. The ATMega328P script will then parse the serial buffer for the next integer, that is, it will capture the next integer and save it to a variable. Once the x coordinate is established, the program reads the ‘y’ character and saves the next integer to a y coordinate variable. These x and y values are used as a center point for our gesture following algorithm. The servos step through angles based on the location of this center point until the index finger landmark is within +/- 20 pixels from that center point. When stepping through the x and y angles, if the values that the servo would move is outside of the 0 to 180 degree range of the servo motors, the angle variables are set at a minimum or 0 or the maximum of 180 ot prevent issues with the PWM signal generated for the angular movement.

If the first byte of the serial buffer is ‘w’, this means that the move to, or move here function was activated. Similar to the follow me function, this portion of the program captures the x and y center point, but instead of stepping through the angles, we mapped the size of the frame, 223x223, to the angle range of the servo motors. These values have to be changed based on the distance the camera is from the hand for recognition. for our test stand, the most accurate angular range for this movement was from 70 to 105 degrees. This means that at the minimum value for the frame x = 0, the servo motor will move to the 70 degree position which would center the thumb on the x coordinate.

The final option if the first byte reads the character ‘r’, the system resets to a pre-programmed position. For testing, this was at the center, (90, 90) where the arm is pointing straight down. For an actual applicable position, the articulight’s home position was set at a 45 degree angle up on the y axis, so that the light is pointing at an angle towards where our hands would likely be working.

Once the serial buffer and motor movements have completed, the program runs a serial.flush() function which has the ATMega328P read the remaining bytes from the serial buffer in order to clear the buffer before polling for the next available data to be passed into the program.

At this stage, our project works as intended, but only in the vertical mounting. Future endeavors will have the arm able to operate horizontally once the ATMega328P script is refined. We would also like to implement a PID control to smooth the movement when performing the move to and reset functions.

10. Planning

10.1 Build Plan

Before, we started building an outline for a design, it took us about a week to decide what we wanted to do. After some time passed we all came up with different ideas. We decided to go with the ArticuLight.

As a group, we felt the ArticuLight is the best thing we could currently build to help provide current engineering companies and individual engineers with an at-home workbench or benches with a light that can be controlled by a camera via hand gestures. Not only does this control the light to be bright or dim, but these gestures will also be used to position the light up to 180 degree rotations.

We are hoping that our design will provide innovative individuals to not have to worry about bad light positioning or brightness, as this design will successfully implement a hands free light system that can be controlled and positioned to one's free will to be the most beneficial setup.

A big part of the build plan will be the computer vision software necessary to recognize gestures. As outlined in other sections, the software will be largely based on a deep neural network that is trained to recognize gestures and OpenCV, a computer vision software library. To build the computer vision software, a prototype will be built that will stream images from the camera, analyze the image and run it through a deep neural network, and perform the action that a gesture implies if detected.

The first step to building this software is ensuring the environment has the required libraries (Jetson GPIO, OpenCV, TensorFlow, Keras, etc). After the environment is set up, controlling of the motors and light will be implemented through use of the GPIO pins on the Jetson Nano to control these components. The exact behavior of the pins and subsequent control will not be developed at this stage, but a basic activation of pins will be planned. Then the object detection and image analysis will be developed through TensorFlow/Keras and OpenCV respectively. To see how this is planned, check the Machine Learning Software section. When these operations are developed and are performing accurate gesture detection, the behavior of the pins and therefore the system will be fine tuned to ensure the best performance and accuracy of the Articulight system. Some examples of fine tuning available are the amount of movement the motors generate and the sensitivity of the dimming gesture (how quickly the light dims and brightens from movement of the appropriate gesture).

While the machine learning algorithm is being developed, the hardware control will be developed. Once the platform is purchased, the code will be developed to allow an interface between the gesture recognition and the microcontroller. This will allow development in a parallel fashion, setting up the functions to allow a serial input and output to the motor controller or relays.

The hardware will have a power regulator that will either be purchased or designed to take an 120 volts AC input from an outlet and output 12 volts DC. Because the Jetson Nano uses a 5 volt DC input, a node will be made to have 12 volts going to the motor controller relay and another to a DC to DC converter to step down the voltage for the Jetson. Fuses will be used at various stages to prevent over voltage in order to protect the circuits from power surges, short circuits, or accidental overdraw of power to sensitive components. The Jetson will connect to a motor controller, microcontroller, and relays to send an enable signal to the component as needed. A signal to the motor controller will include the PWM that will activate the motors, gear, and pulleys to move the light to the appropriate position. A signal to the light system will be connected to a potentiometer circuit to regulate the amount of AC voltage going to the light, enabling us to dim or brighten the light as desired.

10.2 Prototype Plan

Prototyping our design was a struggle. It was a struggle because specific components we needed weren't available to order as many items were out of stock or on back order. However, we made due with components that we had from past classes and from what was in stock.

A main part of our design we prototyped was the frame and arm that will be controlling how the light moves. We used a 3D printer and filament material to create this product. We used metal wires that ran through the arm and/or frame that would be connected to gears when the project would be fully designed and tested. The segments were designed in SolidWorks, then assembled within the program to verify dimensions, then each segment prototype was cut in half such that the top portion and bottom could be connected. This enabled a print in half the time for a full piece as well as saving filament while allowing the interconnections and movement could be accurately tested.

For the time being, we tested these wires by pulling them in certain directions to see how the arm of the light would bend and react in different circumstances. From prototyping this functionality of the design we noticed there was a lot of tension if the arm of the light was not returned to a zero - steady - state position (i.e. starting position). From this prototype, we decided to create and prototype a reset button or gesture for the camera that will return the light to starting position so tension doesn't build up in the wires.

10.3 Vision Software Prototyping

To build a prototype of the main object detection vision software, a python script using OpenCV-contrib-python that can do basic analysis on images will be developed. This prototype will be expanded on in developing the software that will be used in the final Articulight project. A pre-trained deep learning network will be used to facilitate the object detection tasks, as the process of developing a deep network from scratch is too involved for simple prototyping.

The functions of the computer vision software prototype are as follows:

- Read images from the connected camera frame by frame
- Store each frame as a temporary variable and perform any image analysis required for object detection
- Perform arbitrary object detection through Keras
- Power an LED on a breadboard with the Jetson Nano pins when an object is detected

Having these functions in a prototype on the Jetson will allow a more seamless experience in getting started with developing the software for the main computer vision tasks. Since gesture recognition is an extension of object detection, object detection is a good place to start. And although the light we intend to power is more complex and requires much more power than a simple LED, learning to use the pins on the Jetson Nano will benefit for later communication from the Jetson Nano to any controllers such as a motor controller.

10.4 Test Plan

In our ArticuLight design, we had to test a few main components. These components included the wall plug-in power supply, the DC – DC voltage regulator, the Jetson Nano, the servo motors, camera, and the LED light.

We tested the wall plug-in power supply to make sure it connected to the DC – DC voltage regulator as we had to open the power supply case and solder the wire to our converter. The reason these components are important to test is because this voltage from the wall power supply to the DC – DC converter is what will be flowing into our master (brains - Jetson Nano) component which has a strict max voltage of 5V and max amperage of 4A. Without any type of voltage regulator placed between the power supply or Jetson Nano would cause a voltage overload and destroy any other components connected.

We tested a code that will control both of our servo motors. This specific code works for a PWM signal that will program our servo motors to position the arm of the light that is given to us based on the hand gesture made in the camera's view. We also tested multiple codes for the camera to recognize hand gestures that are made to position the light, as well as determining how bright or dim the user wants the LED light to be.

In order to test the functions of the NVIDIA Jetson Nano, the Jetson Jetpack development environment had to be set up. The Jetpack is an Ubuntu installation with software curated by NVIDIA to utilize the CUDA processing and physical features such as the GPIO pins and connectors. To do this, the Jetpack image must be mounted on a microSD card and inserted into the microSD slot on the Jetson Nano. The Jetson Nano performs setup initialization and a modified Ubuntu installation boots. At this point, the Jetson Nano acts as a general purpose linux machine with the added software features that allow the hardware that the Jetson Nano has to function properly.

Pictured in **Figure 42** is the Jetson Nano in the development stage. It acts as a general purpose computer running Ubuntu Linux, and requires a mouse, keyboard, HDMI output to a monitor, and ethernet connection to connect to the internet. These connections are shown in the figure with their respective connection types. The micro USB connection is solely for power.

After the Jetson Nano was set up, it was tested by running common commands in the Ubuntu environment. Additionally, the output pins were tested by powering an LED using the Jetson.GPIO python library.

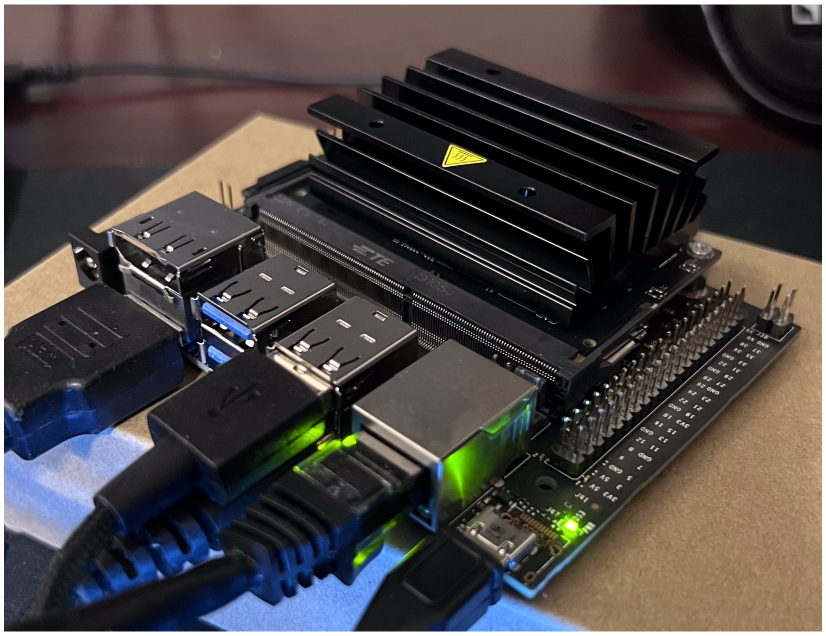


Figure 42: Jetson Nano In Development Stage for Component Testing

Lastly, the last major component that we needed to test was the LED bulb. Although it was one of the easiest components for us to test (i.e make sure it turns on), we also had to make sure that the particular gesture given would be able to brighten or dim the LED light.

10.5 Evaluation Plan

After testing our many components that we will be using in our ArticuLight design and prototyping it, we are then able to evaluate it. From our prototype we were able to see that the code used to control and program both of the servo motors complied and ran efficiently as proposed. Furthermore, when testing our design we saw how important it was to include a DC – DC voltage regulator. From basic knowledge of circuity, we know when input voltage exceeds that of a specific component, it causes voltage overloading and puts that specific component and/or other components to burn/fry.

Looking at our design as a whole, we noticed how each component chosen goes hand-in-hand with one another. We carefully selected each type of component based on research of components we needed. This consisted of many various types of microcontrollers, power supplies such as wall outlet plug-ins or battery packs, relays, voltage regulators, motors, LED bulbs, sockets, fans, and cameras. Each aspect of the system will need to be evaluated independently to verify they are working. Once the gesture recognition algorithm is developed and the signal is passed to the motor controller, an oscilloscope can be used to verify the appropriate PWM signal is being sent as well as any noise that may need to be accounted for. From this data, we can make adjustments to the circuit to reduce the error or add another control element to ensure the signals being generated are as clean as necessary to produce the desired output. As a last case, we can make the adjustments in code but that is less ideal, assuming all calculations done to produce the PWM signal are correct. A runaway error coming from hardware could stem from a larger problem with the components, as such should not be compensated by just code.

As a group, we evaluated each and every various type of component that could possibly be needed for our design and chose the best one that could handle our key specifications that were described in the beginning of our project outline.

11 Project Operation

The Jetson Nano that is powering the gesture control and issuing commands to the motors and light controllers will need to run a script on startup that opens the computer vision and related softwares. This is to avoid the need for the end user to have any interaction with the Jetson Nano Jetpack operating system environment and go straight to using the product.

The following sections will inform the user of the intended operations and familiarize them with the available functions and tutorials on how to perform them. Turning the system on or off and the available hand gestures for functionality will be listed here.

11.1 Turning the System On and Off

The ArticuLight system will contain a power button. This power button will allow the user to turn the system on from a completely off state. At the point of the off state, the Jetson Nano will not be running and the light will be off. Upon pressing the power button, the Jetson Nano will boot and run the script responsible for the computer vision and control mechanics.

Currently the system is fully controlled by the power switch which turns on the PCB, sending power to the ATMega328P, Jetson Nano, and Light. Future plans include designing the system to have the light gesture activated and to include an auto-on feature which will allow the system to activate upon receiving power and completely modular to use without the need of a mouse, keyboard, and monitor.

11.2 Gesture Controls

This section outlines the available functionalities and the gestures that control them. The user must be in the vicinity of the camera so that the user's hands will be visible so the computer vision algorithms will successfully perform the function that the gesture is aimed at completing.

Each gesture is outlined through their respective function. The functions that will be outlined in the following sections are moving the arm the 'follow me' mechanic, moving the arm to a desired location, and resetting the position of the arm to a neutral state after it had been moved.

11.2.1 Follow Me

This gesture activates the 'follow me' function of the ArticuLight system. Known by us as 'rock on', it is a hand gesture where the pinky and index finger are raised and the other fingers are tucked, as seen in Figure 45. Once this gesture is held, the system toggles the follow me flag and centers the camera on the index finger of the detected hand until the flag is disabled. To disable the follow me functionality, the same 'rock on' gesture must be held and it will be disabled, resulting in a neutral system until the system recognizes another gesture and acts on it.

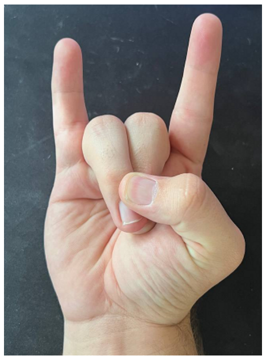


Figure 43: Follow Me Activation/Deactivation Gesture

11.2.2 Moving the Arm to a Desired Position

Moving the arm to a desired position will be implemented through the use of a single action gesture in which the system will move the arm so that the thumb is centered in the frame. This gesture is implemented through the gesture commonly known as 'Thumbs up'. This gesture is visualized in Figure 46.

When this gesture is held, the system will issue commands to the servo motors through a mapping function that detects how far away the thumb is from the center of the frame in pixels. When passed to the mapping function, the motors move so that the thumb is centered.

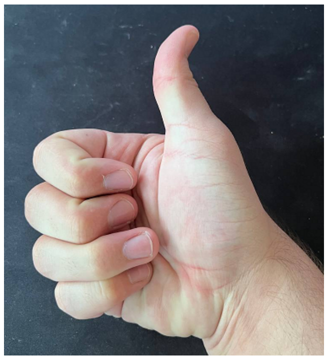


Figure 44: Move Arm to Position Gesture

11.2.3 Resetting the Position of the Arm

After the arm had been moved through the use of the other gestures, the user may implement the reset gesture so that the arm will result in a neutral state. This gesture is performed by extending the index, middle and ring fingers and tucking the thumb and pinky. This gesture is shown in Figure 47.

When this gesture is held, the system issues commands to the servo motors that results in the arm being in a neutral state. When the system is mounted vertically, this results in a completely vertical arm. Likewise, if the system is mounted horizontally, it results in a completely horizontal arm.

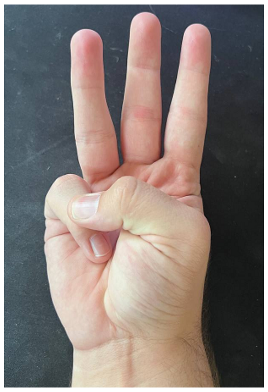


Figure 45: Reset Arm Gesture

11.3: Retraining Gesture Recognition

This section details how the software interacts with the user to retrain the support vector machine that is responsible for gesture detection. Two scripts must be edited slightly and run, and the final script needs to change the SVM model that it relies on.

11.3.1: Gather Dataset: image_collection.py

The first step to retrain the gesture detection in this project is to gather a new dataset. Upon running the image_collection.py script, it will prompt the user with the name of the requested dataset. The user then would enter the name of the respective dataset. Then, the script will prompt for the amount of gestures that the user plans on training. The user should enter the number of gestures they wish to train and an additional 'gesture' for a lack of hand so that the model does not have to choose between gestures when there is no hand present. Also, it is a good idea to have another 'gesture' for a no gesture sample where the hand is in a neutral state.

The script will then loop through the number of gestures that the user wishes to train with a prompt for a name for each one. The user should remember the order in which they enter these names if they wish to run a testing dataset on the SVM so that they can test its accuracy. Then, the GUI outlined previously in this document will start and the user can start collecting images for the gesture training. It is recommended to supply at least 600 images per gesture for training, as this will supply an adequate amount of training data for the SVM to learn the gestures. For the testing dataset, the user should capture at least 100 images. It is important to translate and rotate the hand during the image collection so that the SVM will be able to identify the hand in different orientations.

After the images are collected, the user should then click the exit button. The images and their respective directories will be removed and added to the training/testing directory in the root directory of the dataset. The images will be renamed numerically starting from 0. A JSON file is created for assisting with the SVM training and should be moved to the directory of the dataset.

11.3.2: Retrain SVM: svm.py

After the image dataset is collected, a new support vector machine can be created for gesture detection. To do this, svm.py must be updated to use the directory of the dataset that the user wishes to train. This is available through the path variables that are instantiated throughout the code. The boolean flag train_svm must be updated to true if it is not already. Then, below the flag, the name of the .sav file that will save should be updated to a name that the user believes is representative of the dataset.

After all necessary variables are updated to use the dataset that the user just created, the script can be ran. After the SVM is trained and a .sav file is created, the SVM is loaded and will show gesture detection on the screen. One issue that is present in this script is that the SVM file will attempt to read gestures from a JSON file in preprocess/gesture.json. If the file is edited to the gestures from the order in which they were captured in image_collection.py, the model will erroneously show which gesture is which on the camera feed. The simplest way to overcome this is to edit gesture.json so that the gestures are ‘one, two, three’ etc then seeing which gesture maps to which order and manually update gesture.json with the user’s findings. If more time was present in this project, this would be investigated further and at least somewhat automated.

11.3.3: Update articulight.py

Now that a new SVM model is created, articulight.py must be updated. The SVM that the script references should be updated to the SVM model that the user wishes to use. Then, the gesture outputs that map to actions in the main body of the loop in this file should also be updated. By default, these gestures are “Thumbs up”, “Three”, and “Rock on”. If the user wishes to use different gestures such as in this case, they should be manually updated in the scenarios of code that result in the various actions that the system executes. After these steps are complete, the new gestures that the user specified will be checked when the system is ran.

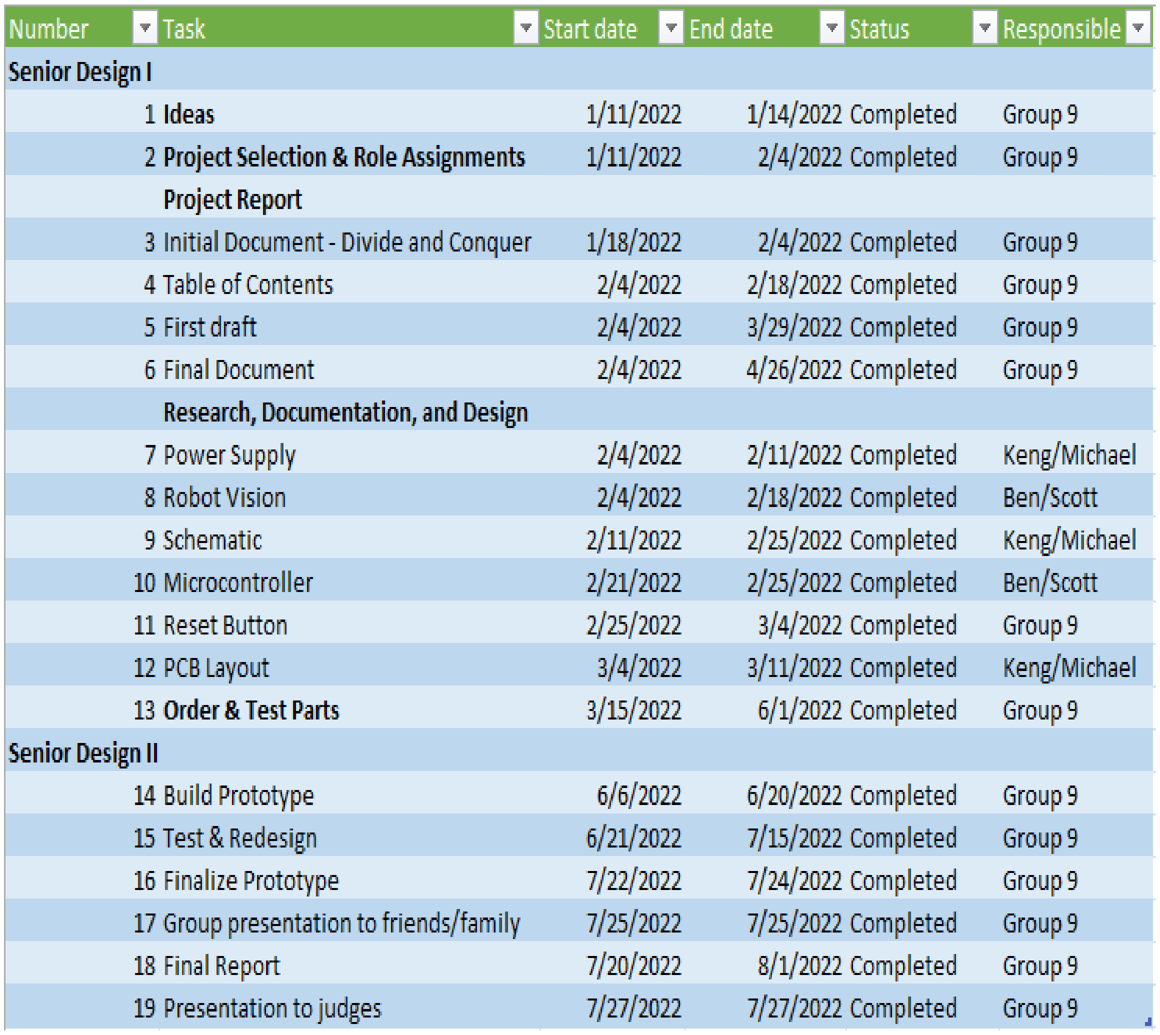
12. Administrative Details

This section will contain the management portion of the project. Firstly, we will be looking at and discussing the budget plans of the ArticuLight design, setting up milestones for the project. Following that, we will compare two budget tables. One that consists of the specific chosen parts and one that focuses on the generalized cost of the components.

12.1 Initial Project Milestones

The milestone start and end dates were set based on the corresponding tasks needed and with accordance with each team member’s schedule and responsibilities. Most of the milestones that are in the first half of the implementation are based on research and familiarization with the system’s components, while functionality will be the focus of the second half during senior design 2.

We were able to meet all of our initial project milestones, but did have a minor setback due to complications mentioned above with the flexing in the case. This required a quick redesign and build which impacted testing and tuning the servo controls.



Number	Task	Start date	End date	Status	Responsible
Senior Design I					
1	**Ideas**	1/11/2022	1/14/2022	Completed	Group 9
2	**Project Selection & Role Assignments**	1/11/2022	2/4/2022	Completed	Group 9
	Project Report				
3	Initial Document - Divide and Conquer	1/18/2022	2/4/2022	Completed	Group 9
4	Table of Contents	2/4/2022	2/18/2022	Completed	Group 9
5	First draft	2/4/2022	3/29/2022	Completed	Group 9
6	Final Document	2/4/2022	4/26/2022	Completed	Group 9
	Research, Documentation, and Design				
7	Power Supply	2/4/2022	2/11/2022	Completed	Keng/Michael
8	Robot Vision	2/4/2022	2/18/2022	Completed	Ben/Scott
9	Schematic	2/11/2022	2/25/2022	Completed	Keng/Michael
10	Microcontroller	2/21/2022	2/25/2022	Completed	Ben/Scott
11	Reset Button	2/25/2022	3/4/2022	Completed	Group 9
12	PCB Layout	3/4/2022	3/11/2022	Completed	Keng/Michael
13	**Order & Test Parts**	3/15/2022	6/1/2022	Completed	Group 9
Senior Design II					
14	Build Prototype	6/6/2022	6/20/2022	Completed	Group 9
15	Test & Redesign	6/21/2022	7/15/2022	Completed	Group 9
16	Finalize Prototype	7/22/2022	7/24/2022	Completed	Group 9
17	Group presentation to friends/family	7/25/2022	7/25/2022	Completed	Group 9
18	Final Report	7/20/2022	8/1/2022	Completed	Group 9
19	Presentation to judges	7/27/2022	7/27/2022	Completed	Group 9

Figure 46: Initial Project Milestones

12.2 Budget - Estimated Cost

The total estimated budget is outlined in table 30. Each major component that is expected to be purchased is outlined in table 31 and includes the quantity needed and the price that each component costs. This project is self – financed by the group members. These estimates were developed at the very early stages of conceiving of this project; as is evident, we undershot the amount of money required to build the ArticuLight system. Table 30 shows the predicted parts that we planned on using for our design when we first started planning our project in Senior Design 1 and Table 28 shows the updated components we used for our actual design in Senior Design 2.

Part	**Quantity**	**Price**
Camera	1	$25 - $29.15
NVIDIA Jetson Nano Developer Kit	1	$99
Arduino Uno Rev3 Microcontroller	1	$23
Motors	2 - 4	$11.99 each to 29.99 (4pack)
Motor Controller	0 - 4	$6.89 - $9.99
Wire Loom	2 feet	$12.60 (39 inches)
Gear/Wheel Wire Guide	2 - 4	$8.98 - $17.96
Light Socket	1	$3.88
Power Regulator	1	$9.90
Relay	2	$18.79
LED Light	1	$12.98 (4-pack)
Wire (electrical, 24AWG)	4 feet	$11.98 (50 feet)
Wire (mechanical, 3mm)	2 feet	$8.99 (7 feet)
Frame/Case (material)	Unknown at this time	N/A
Memory Card	0 - 1	$0 - 6.66
TOTAL	N/A	$265.97 - $294.87

Table 28: Estimated Budget

The next section, true cost of components, reflects the cost of the components after research has been fully completed. Table 29 shows each specific component and their cost that were used for our project.

12.3 Budget - True Cost of Components

Part	**Quantity**	**Cost**	**Buying from**
NVIDIA Jetson Nano Developer Kit	1	$99 (group member bought 2 years ago)	Nvidia
CanaKit 5V 2.5A Raspberry Pi 3 B+ Power Supply/Adapter	1	$9.95	Amazon
PNY 128GB Micro SD Card	1	$13.99	Amazon
Raspberry Pi Cam V2	1	$26.57	Amazon
D956WP Servo Motor	2	$109.04 ea	Amazon
ATMEGA 328P Microcontroller	1	$25.99	Ebay
E-26 Light Socket	1	$8.99 for 5-pack	Amazon
LED bulb	1	$2.50	Amazon
Filament (PLA)	1	$30	Amazon
Frame/Case material	1	$15	Lowes
Total		$450.07	

Table 29: Cost of Materials

13. Conclusion

Throughout the ArticuLight design, we encountered various engineering challenges that needed to be overcome when it came to the researching and development phases. The system requirements that were shaped during the ArticuLight's conceptual design stage have been taken into consideration for each and every component, as well as line of code, that has been integrated into the system.

ArticuLight's design consisted of many simple components such as a voltage supply source, relays, microcontroller, servo motors, motor controller, and an LED bulb as well as complex components such as the NVIDIA Jetson Nano. When these components are integrated properly, they deliver a marketable function to an industry in need of an updated lighting system that can be controlled via hand gestures and can also rotate and position up to 180 degrees, to get the best lighting possible for their workbench. Considerations went into choosing each component for the design phase to make our product be the most efficient, reasonable cost, and end user's safety as a priority for our ArticuLight system. Moreover, other factors such as movement space, camera angle and lighting (to see hand gestures), and testing of the ArticuLight system arm, have all been accounted for to ensure our finished product is the most demanded type of movement light system technology of its kind.

The ArticuLight system turned out to be a well – balanced design. The way the components and software interact with one another allowed the ArticuLight design team to mesh together interests, technical skills, and concept ideas. It is important that the team focuses not only on each individual technical detail, but how each specific detail impacts every other aspect of the ArticuLight system.

One of the biggest accomplishments of ArticuLight's design is the successful integration of using a Jetson Nano to act as the master of the system. Having the Jetson Nano allowed us to design our system more cost efficient. Since the Jetson Nano is able to act as a master or "brain" of our system, we didn't need a motor controller to control the motors or relays. Using this specific microcontroller (Jetson Nano), can become an ever - growing engineering masterpiece, which gives motion light systems such as the ArticuLight, an opportunity to shine compared to other light systems that are only available in today's society.

14. Index

List of Figures

List of Tables

15. Citations

Constraints and Standards

https://support.jlcpcb.com/article/64-certifications
https://www.rohsguide.com/

Inspiration for ArticuLight
https://www.youtube.com/watch?v=5PgLXFf4avU

Microcontroller:
Arduino Mega 2560 REV3
Arduino Uno R3
MSP430 FR6989 Launchpad

Computer Vision Module:
Raspberry Pi 4 Model B
OpenMV Cam H7 R2 (includes camera)
NVIDIA Jetson Nano Developer Kit-B01
Arducam IMX219-AF Programmable/Auto Focus Camera

Camera:
Raspberry Pi Camera Module 2
Raspberry Pi High Quality Camera
IMX219 Fixed Focus Camera
OpenMV Cam H7 R2
Arducam IMX219 Low Distortion
USB vs MIPI
Global Shutter vs Rolling Shutter

Convolution Neural Networks

Software:
OpenCV
SimpleCV
Scikit-image
TensorFlow
Keras
PyTorch
Machine Learning Models
trt_pose_hand

Fan:
Noctua NF-A4x20 PWM Fan

Motors:
Short Body Nema 17 Bipolar Stepper Motor
(Pack of 4pcs) Stepper Motor High Torque Bipolar DC Step Motor Kit
Servo Motor

Motor controller:
DC Motor Drive Shield Stepper Motor Drive Shield Expansion Board
Adafruit 16-Channel 12-bit PWM/Servo Driver - I2C interface [PCA9685] : ID 815 : $14.95 : Adafruit Industries, Unique & fun DIY electronics and kits

Gear/Wheel Wire Guide:
Wire Rope Wheel, Deep U Groove Pulley, Double Bearing Wheel Block Wire Rope Lifting Guide (diameter28mm)

Light Socket:
Project Source Plastic Keyless Lamp Socket, Black in the Light Sockets department at Lowes.com
E-26 Standard Socket

Power Regulator:
DC 5V 2A 10W Power Supply Module AC110V 85V-265V 50 60Hz to 5V output

Relay:
AC100V-250V 30A High Power 2-Channel Relay DC5V High-Low Level Trigger Switch Module
Electromagnetic Relay

LED Light:
GE Basic 100-Watt EQ A19 Daylight LED Light Bulb (4-Pack) in the General Purpose LED Light Bulbs department at Lowes.com
Dimmable A19 LED Bulb

AC Bulb Dimmer Module:
AC Bulb LED Dimmer Module

Wire (electrical, 24AWG):
50 feet 24AWG Stranded Electrical Wire, red and black

Wire (mechanical, 3mm):
Lumintrail 3mm Braided Steel Coated Security Cable Luggage Lock Safety Cable Wire Double Loop

Plastic Wire Loom:
Plastic Wire Loom
Tinned Copper Braided Sleeve

Gooseneck Pipe (Metal Wire Loom)

PCB Board:
Double sided PCB Board Kit

Memory Card:
128GB SD Card
Communication Protocols