Hand Gesture Vending Machine

Hussein A. Shelleh [Electrical Engineering, Photonics]
Jhamori Williams-Austin [Computer Engineering]
Logan Terrell [Computer Engineering]
Tyler Bornemann [Electrical Engineering]

Reviewer Dr. Sonali Das [Electrical Engineering]
Reviewer Dr. Peter Delfyett [Photonics]

Coordinator Dr. Aravinda Kar [Photonics]
Coordinator Dr. Chung-Yong Chan [Electrical Engineering]
Coordinator Dr. Lei Wei [Electrical Engineering]

University of Central Florida
EEL 4915
Group 1

TABLE OF CONTENTS

1. Executive Summary

The Hand Gesture-controlled Vending Machine (HGVM) is an alternative to in-use and modern vending machines without the use of touch controls. With the ripple effects of COVID-19 and increased use of ‘card-only’ machines, keeping ourselves clean of spots people often touch or grab only makes sense.

Vending machines, however, are operated with either a series of buttons or a touch screen, one used by any who want to purchase from it. It’s a hotspot of germs and a perfect jump point for infections and viruses to spread. So, in the place of this risk, we can instead show a prototype using hand gestures to select a product - cutting down on this risk significantly while still allowing snack companies an easy way to offer their products.

This project has a lens system for the camera with object detection linked into it. Then, once one of the right gestures are made, the desired item was vended for the customer. This did involve some mechanical elements, but primarily it would be coding a series of motors to push out the product down the chute - and making sure these motors are reasonably powered.

For the prototype, we intend to use a selection of three to five snack products for the machine to hold and allow a user to choose from. The testing did primarily involve confirming if the hand gestures work - and, following that, if the correct product is vended out. There also was testing done regarding out of stock snacks, both for the mechanical components and for the screen communicating with the user.

2. Project Description and Background

Vending machines have a long history dating back to ancient Greece, where holy water was dispensed automatically. A great example can be the small brass machines that dispensed tobacco, located in England around 1615 [1]. However, the modern concept of vending machines emerged in the nineteenth century. The first commercial coin-operated vending machines were introduced in London in the early 1880s, dispensing postcards and books. Over the years, vending machines have evolved to offer a wide range of products, including snacks, beverages, and even toys and electronics. They became ubiquitous in public spaces such as offices, schools, and transportation hubs, providing convenient access to goods. In 2016, there were even vending machines offering Ferraris and Lamborghinis in Singapore.

There was a point in time where instead of there being coins, vending machines were accepting notes, making things a tougher challenge in terms of accepting payment. The earlier version of the vending machine that accepted notes as a payment used a magnetic tape head style mechanism to detect if a dollar had the correct amount of iron content [2]. During the early 1990s, vending machines had then switched over to using low resolution digital cameras to verify a note's authenticity. In today's times we are even expanding on the way we're able to pay for things when it comes to vending machines.

Cash payments have been becoming less and less convenient for the customer, helping electronic payment options become more popular. In regards to electronic payments, one form can be through Apple Pay, which allows for a customer to pay for an item by tapping their iPhone to the reader. Market research projected that by 2020, twenty percent of all vending machines will be smart machines, with at least 3.6 million units knowing who you are and what you like [1]; this may not be the case quite yet, but we're getting there.

The COVID-19 pandemic significantly impacted the use of vending machines. With lockdowns, social distancing measures, and concerns about surface transmission of the virus, people became more cautious about using shared public facilities, including vending machines. The fear of touching surfaces that might harbor the virus led to a decline in the popularity of vending machines during the pandemic. Changes in consumer behavior and preferences during the pandemic influenced the types of products in demand. With more people staying at home, there was a shift towards online shopping and delivery services, reducing the need for on-the-go convenience items typically provided by vending machines.

Additionally, the closure of offices, schools, and other public spaces during lockdowns resulted in reduced foot traffic and, consequently, fewer opportunities for people to use vending machines. The shift towards remote work and changes in consumer behavior also played a role in diminishing the relevance of traditional vending machines. In response to the pandemic, some vending machine operators implemented touchless payment options and enhanced cleaning protocols to address hygiene concerns. People became more conscious of surface contact and potential transmission of the virus through commonly touched surfaces. Vending machine keypads, touchscreens, and product dispensing mechanisms were perceived as high-touch areas, contributing to a reluctance among consumers to use them. Overall, the impact of COVID-19 on the vending machine industry was a negative one, leading to a temporary decline in the use of said machines as people sought alternative ways to access goods while minimizing physical contact with public surfaces. With the vending machine industry facing many challenges during the pandemic, it managed to prompt new and different innovations.

2.2. Motivation

The motivation behind the HGVM stems from a proactive approach to address health and hygiene concerns, particularly in the context of illnesses like COVID-19. This innovative design aims to minimize the risk of virus transmission by reducing physical contact with surfaces, such as touch screens or keypads, which are commonly associated with the spread of germs. The primary focus with this project is to provide users with a touchless and contact-free experience. With customers being able to make selections and complete transactions using hand gestures, the vending machine did minimize the risk of bacterial and viral transmission, promoting a safer and more hygienic environment.

The HGVM is able to adapt to the realities of the pandemic. With the many challenges faced by the pandemic, this idea of design represents an adaptive solution to align with changing consumer behaviors and concerns. These changes in particular cater towards individuals who prioritize safety and hygiene in their interactions with public spaces. The integration of contactless payment options has also been a considerable factor in the HGVM adapting to the realities of the pandemic. Not only does this align with the growing trend of cashless transactions but also provides users with a comprehensive touch free experience from product selection to payment. Another critical aspect stemming from the HGVM is convenience and accessibility. This particular technology caters to individuals who may have concerns about using traditional vending machines due to hygiene concerns or those who prefer a quick and seamless transaction process.

Additionally, one must consider the operational efficiency of a machine with fewer mechanical components versus its competitors. We believe that there would be reduced wear and tear and maintenance on the machines; by having fewer physical components such as buttons, the machine would have fewer points of mechanical wear or failure, meaning the vending machine should require less maintenance compared to other, more traditional vending machines. With all of that in mind, this would result in improved operational efficiency, reducing downtime and costs associated with repairs.

Another huge motivational point to look at is the public perception in regards to using hand gestures over buttons or touchscreens. Public perception plays a huge role in building a brand; it is heavily influenced by the efforts businesses make to address health concerns. By choosing a more complicated technology and software in the place of traditional means, this allows an opportunity to show the commitment to the wellbeing of customers over profit, demonstrating a proactive approach to maintaining a safe and hygienic environment.

Some of the things mentioned further aid in building a brand for a company. Being in this particular era where health and safety are even greater concerns, by offering a touch free experience this would align with evolving expectations for clean and secure environments. We believe that this allows for trust to be built between a customer and for instance the company of the HGVM. The post pandemic has shifted the way people perceive public spaces and shared amenities. The HGVM can become a symbol of adaptability and responsiveness to current and future health challenges.

2.3. Goals and Objectives

Our overall goal for this project is to create a completely hands-free vending machine that utilizes object detection of hand gestures and a card reader to select the desired item. This machine is expected to be plugged into a standard electrical outlet, and due to its small nature as a prototype, set onto a table or countertop. The prototype should be a sufficient proof-of-concept for a hands-free vending machine.

Our basic goals are our absolute bare minimum. This machine must have a screen display that turns on when a person enters vicinity of the motion detection. Additionally, the vending machine needs to be able to take in information from a camera system and output commands to the system. If the system detects a customer, the output should be turning all the lights on in the vending machine, including the ones that illuminate the items and the lights on the display. If the system detects a gesture being made, it should then show the product in question on the screen and a confirmation message. If the customer gestures in agreement with the confirmation, it is ‘locked in’ for their purchase. If they gesture in disagreement, it does return to the selection process. Finally, the machine must be able to dispense the selected snack after ‘payment’ with a dummy card reader. To achieve these goals, our basic objectives are to pick a sufficient motion detector, camera and lens system, and the right learning algorithm to use in the process. We did also need to determine the right motors to use for both power use, needed rotational speeds, and size.

Our advanced goals are for alternative payment methods - specifically, the use of exact change cash and coin - and to increase the number of vended items from three to four. To achieve this, however, we did require additional research, software coding, hardware management, and likely some readjustment of spacing within the machine itself.

Our stretch goals are variable; they include a possible disability-oriented companion app, the ability to house cold drinks alongside snacks, and to use an alphanumeric system (A1, A2, B1, B2) with an increased number of snacks within the device. Accomplishing these each have their own extreme needs, however. For the companion app, there would be a significant increase in software workload and additional hardware elements necessary to connect the app to the device. For cold drinks, it would require insulation and a sufficient cooling system - in the case we do choose to do this, the complexity of the device would increase, but also make for an impressive final result. For the alphanumeric selection process, it would require more gestures for the object detection to notice, as well as a guide to show users the signs for the necessary letters.

2.4. Requirements and Specifications

In order to succeed with this project and accomplish as many goals and objectives as possible, we must start with a baseline. Our baseline - the requirements and specifications - are not particularly extensive, but every single one counts. Without taking these requirements and specifications into account, we actively ignore both the realistic limitations which we must be able to work around and any finer details of the device, resulting in an incompatible, impossible design that would not work.

When it comes to hardware, our requirements are many. First and foremost, our voltage requirements must remain within the one hundred and twenty volt (120V) standards of a traditional outlet - though due to the smaller nature of the prototype, we did be making such a requirement much more stringent: we must keep the voltage usage of the device within fifteen volts (15V). Additionally, we must keep the device within realistic operating temperatures; the use of an object detection algorithm with a consistent frame rate did result in significant heat from the AI development kit which we choose to work with. This runs the risk of damaging the components or ruining the food which did be held within the device, so as an additional restriction we intend to cool the AI development kit at or under ninety degrees fahrenheit - or thirty-three point three degrees celsius.

Additionally, less stringent hardware specifications involve reasonable sizing and weight for the machine, as well as keeping the initial development of the device within a small budget. For size, we expect it to be within a two foot by two foot by two foot cube - though likely smaller. For weight, it should weigh in at around twelve kilograms or less. For cost, we would like to keep it as cheap as possible, though for the moment that estimate rests somewhere around the six hundred to seven hundred dollar ballpark ($600.00 - $700.00).

When it comes to software requirements and specifications, the list is shorter but just as significant if not moreso. First, it must be coded in a language that the Computer Engineers of the team - Jhamori and Logan - are either skilled with or already somewhat proficient in prior to Senior Design. This is to mitigate unnecessary work given our limited amount of time. Second, the object detection must be accurate enough to limit the frustration of the users - somewhere around 80% accuracy or higher would be necessary given the nature of the machine being prototyped.

Included is a table of specifications - see Figure 2.4.1 - which is intended to help succinctly cover the requirements of the device as a whole. Of note are the rows bolded with red and orange; these are our critical requirements for the vending machine.

Overall Machine Requirements	
Maximum Outlet Voltage and Amperage	15V, 10.5A
Operational Temperature	< 90°F / < 33.3°C
Size	≤ 2’ x 2’ x 2’ / ≤ 61cm x 61cm x 61cm
Weight	≤ 26.5lb / ≤ 12kg
Cost	≤ $700.00
Coding Language	CpE **must** be at least familiar with it
Gesture Detection Accuracy	≥ 80%
Response Time	< 15s
Motion Activation Accuracy	≥ 80%

Figure 2.4.1: Table of Overall Machine Requirements

Additionally, we have put together a House of Quality within Figure 2.4.2; this portrays both the engineering requirements and the marketing requirements for the product, and is intended for the ease of comprehension with regards to our priorities and their respective hypothetical tradeoffs.

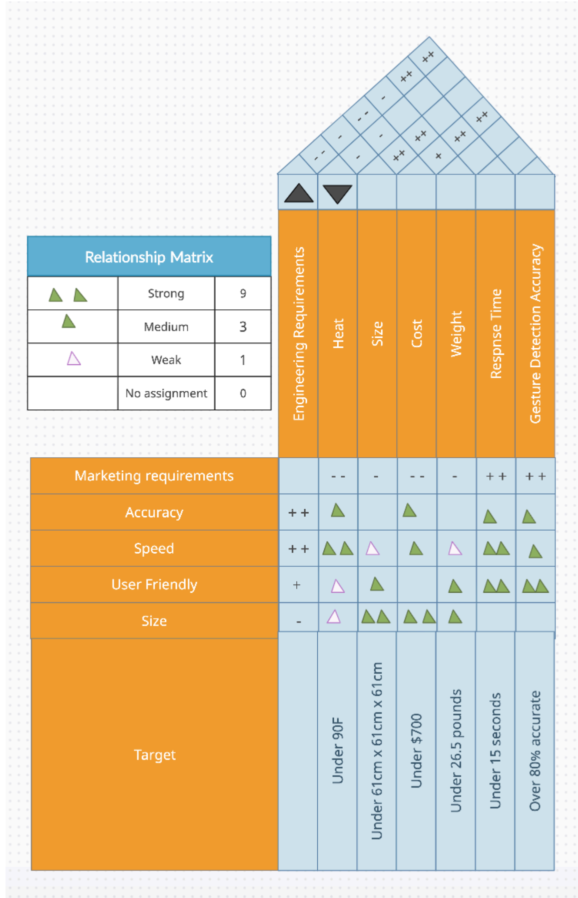


Figure 2.4.2: House of Quality.

2.5. Block Diagrams

The various work for this project was split according to the respective majors of our team. Jhamori Williams-Austin and Logan Terrell are both computer engineers; it was the two of them working on the machine learning algorithm alongside the code that was put into the microcontroller unit for the motor and screen UI control. As the sole photonics engineer, Hussein Shelleh did handle the motion detectors and the lens system with which the camera system did use. Finally, both Hussein and Tyler Bornemann, as the electrical engineers, did work on the outlet voltage regulation, PCB development, stock checking, and motor controls. To better show how this would be applied, figure 2.5.1 offers a demonstration. Additionally, figure 2.5.2 offers a more in-depth coverage of the device hardware, and figure 2.5.3 the same for software.

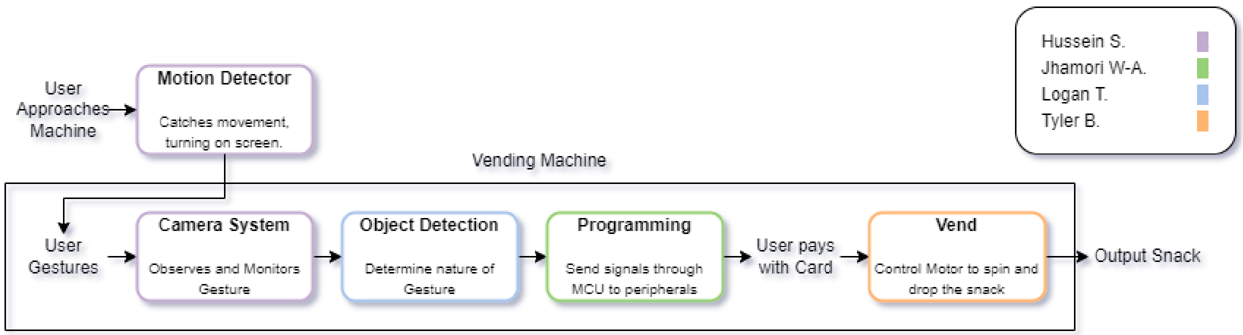


Figure 2.5.1: A pathway representing the group's individual roles and device 'action path'.

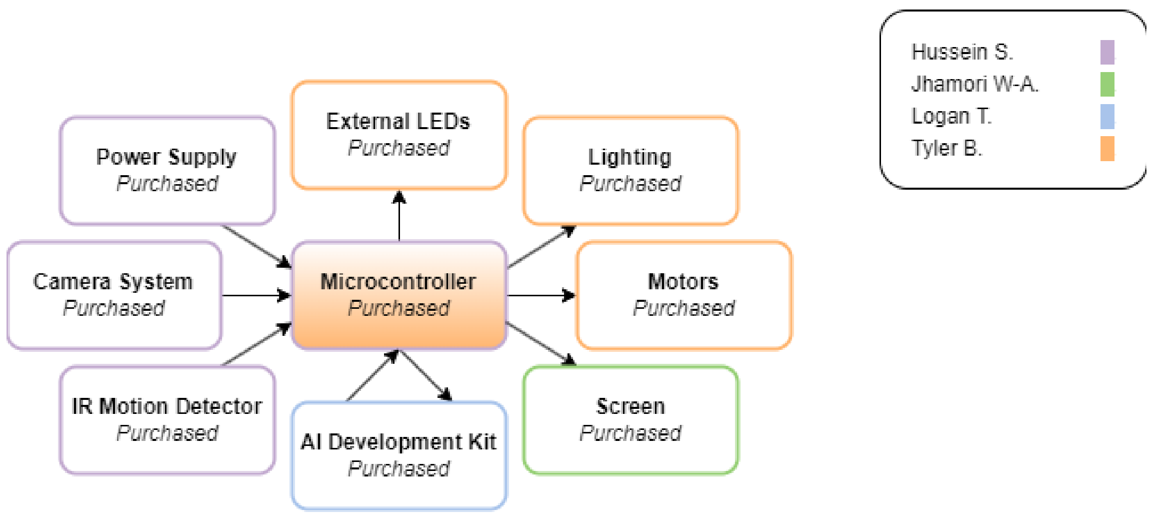


Figure 2.5.2: Hardware Diagram with group's divided Hardware workload.

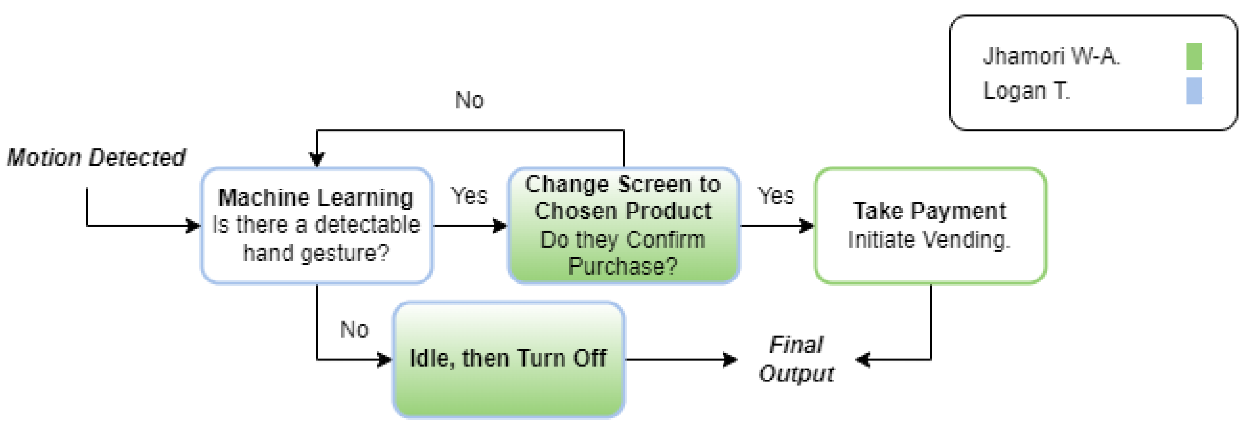


Figure 2.5.3: Software Diagram with the group's divided workload.

3. Related Project Research

The use of technologies for the sake of vending is surprisingly storied and varied, as are the technologies used to accomplish it. These technologies are or were once available commercially, meaning we have significant information regarding the underlying mechanics. Taking a look at this history and information did be the best way for us to determine the internal style of the machine - which did furthermore influence the individual components.

3.1. Vending Machines and Related Technology

The purpose behind the HFVM is, simply put, to make a vending machine with a different method of input. This does not mean that the insides of the vending machine are irrelevant; if anything, it makes the internal components all the more critical to research and study. There are dozens of styles of machine over the centuries, however - from ones powered by the weight of a coin and water to others using a two-axis control scheme to pull products.

Researching each major style of vending machine, or just similarly styled mechanics, could prove fruitful for determining the internal design. This is especially important given the lack of a mechanical engineer in our group - it further influences the design we seek out. Namely, simplicity and cheapness are the absolute most important parts for our chosen vending method.

3.1.1. Manual Analog Vending

A significant number of vending machines make use of quarter-based, insert-and-turn mechanisms in order to dispense a singular product, or in some cases, several small products. This is accomplished with one to two coins and a small handle that the user has to twist; the twisting motion is unlocked by the coin insertion, and is connected with an internal gearbox that releases the product or bulk products via a small chute. This style of vending machine has been around since the late 1800s, though the accreditation to the first company to use it is unfortunately unknown [3]. These machines initially didn't take quarters, but instead pennies or nickels.

In the modern day, we still see these machines frequently. Gumball machines, candy dispensers, capsule toy dispensers, and 'Gachapons' (Japanese capsule machines) all use this style of manual turn mechanism. There is a key issue that the mechanism is *manual* for our design concepts, but there are still multiple things of interest that we can take away from these designs.

Figure 3.1.1.1: Single-Quarter ‘Bulk Vending’ Machines.

One of the key takeaways from this style of vending machine is the focus on product shelf life and small-scale construction. Both of these are important to our prototype design, though the former is less important than the latter. We must keep our design relatively small, and these manual mechanical designs are a good reminder of just how small vending can be. Additionally, the ‘chute’ is of interest to us; given the nature of the design being a prototype, there were little to no ‘anti-theft’ features. As such, we’d likely make use of a ‘bowl’ attached to a chute near the bottom in the place of a more mechanically advanced pulley flap.

3.1.2. Automatic Analog Vending

For the most part, however, manual vending has been phased out in favor of automated vending systems. In other words, there is no need for physical effort beyond selection and payment - the machine does the work for you. This style of vending machine is actually significantly older than manual vending - albeit using water and the weight of a coin rather than electricity. In the times of the late Roman empire, estimated to the first century AD, there were machines used to dispense wine and holy water using the weight of the coin as a counterweight, pouring until the weight is balanced - then snapping the level back into place [4]. Counterweight-based vending machines have lasted beyond that, as well - up until only around the late nineteenth century.

Modern forms of automatic vending machines instead make use of electricity and motors, though with this increased complexity also comes several advantages. The most significant advantage automatic vending systems have over manual vending systems is the ability to offer multiple products from the same singular machine. Manual vending attempts to minimize this with multiple machines in parallel (see Figure 3.1.1.1 in the prior section for visualization) but a significant amount of space tends to be wasted in this process.

The means which these machines use for automatic vending varies from one to another. However, three forms of vending are most common. The first of the two is the use of a spiral attached to a motor - called a spiral motor in the vending industry - which spins to push products out. The second is usually for drinks, using a piston to push the item through a sloped choke point of plastic; without pressure, it would be too small for the drink to get through. The third is a modernized variant of the second, often seen in the newest models of drink vending machines. Instead of just using a piston, it also uses a motorized 'lift', which moves to where the drink was pushed out from. This minimizes jostling of the drink contents - a particularly good benefit for sodas.

There are several more takeaways from automatic analog vending versus its manual counterpart. Most important of them is the use of automation itself. The less the customer has to do with the machine in order to obtain the product, the less risks of misuse or improper wear and tear did pop up. Additionally, the automation itself also decreases points of contact - which is critical when it comes to the sort of design we are intending to produce.

3.1.3. Automatic Digital Vending

Even now, there have been further iterations on the vending machine that one can see. Namely, the use of touchscreen technology and tap-to-pay card readers. These technologies are relatively recent in the scheme of things, and significantly change the internal mechanisms when it comes to selection and payment for vending. Namely, the introduction of the need for more complex microcontrollers, as well as a need for an internet connection to facilitate the use of credit card payments.

This need for internet interfacing is both an upside and a hindrance, in truth. On one hand, by making use of internet connectivity, the vending machine can allow for payments via card - which most people tend to carry in favor of physical cash in the modern day. On the other hand, this effectively cripples the areas of operation said vending machines can work in - as without internet access, card payment becomes inaccessible. Not all places have wide-spanning internet networks like colleges or universities do, and not many places would put up the effort in connecting vending machines to a private network - especially if simpler vending machines would do the trick. As such, the demand for these machines is less than that of their analog counterparts.

There is still something of worth to learn from these machines, for this proof of concept. Specifically, the use of credit card tap payments. Unlike effectively all other forms of payment, tap payment has the least contact and no actual human contact onto the reader itself. This is significant for our design, and is as such something that was incorporated into it. At the same time, the same constraints other automatic digital vending machines possess did apply.

3.2. Optical Components

There are several critical optical components for this design, each with their own specialized purposes. First, there is the external infrared motion sensor, which did be the means to activate the camera system and object detection kit. Second, there are the optical lenses, which are intended to create a field of focused imaging that does allow for a sufficiently large image - but without wasting significant view space. Third, there are the infrared transmitters and receivers for stock-checking the products, which did keep customers from selecting a stock-free option.

3.2.1. Optical Motion Sensor

Photodetectors, also known as photosensors or light sensors, are sensors capable of detecting electromagnetic radiation. There are hundreds of applications for photodetectors, from power generation in solar panels, to automatic lighting, to photoresponse and photochemistry, to disability aids and more. Critically, photodetectors are used in data-dense storage technologies like compact disks (CDs) and compact disk read-only memory (CD-ROM), forming a backbone of optical data storage technology.

Some photodetectors are built with specific controls or data-management programs that can be used in conjunction with a desktop in order to get precise, high-quality data. These photodetectors, however, are fairly large and exceedingly expensive. Used high-quality data-measuring photodetectors go for over seven hundred dollars at the cheapest, increasing in price based on the specific operational range and the company which produced it. This is not what we want; our goal is to simply identify if there are sufficient optical emissions reaching it - this is to check if a person has come into range of the photodetector.

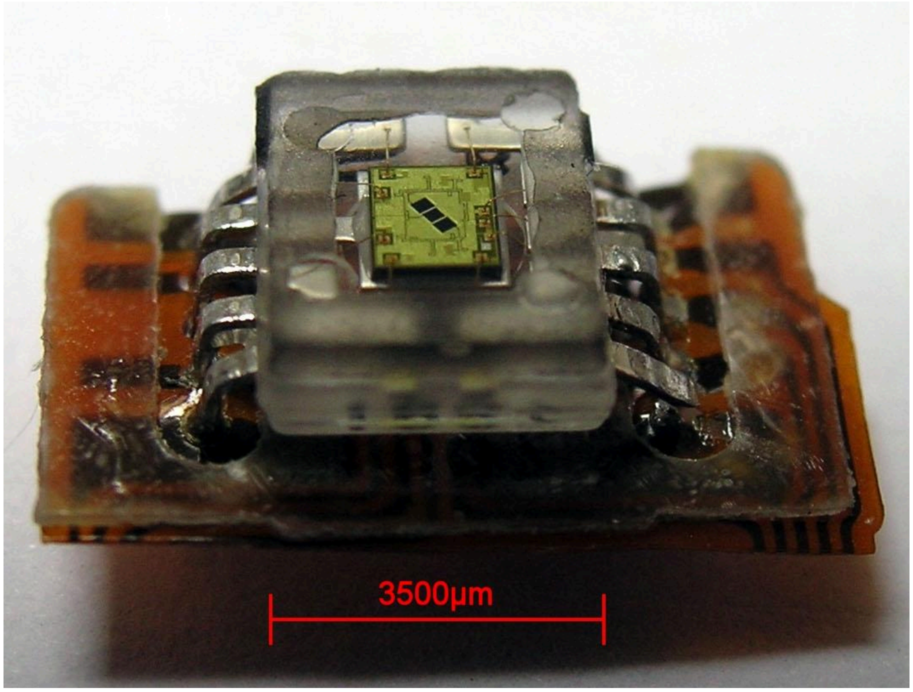


Figure 3.2.1.1: Close-up of a Photodetector salvaged from a CD-ROM. The three black squares in the center are Photodiodes.

For this project, we did not use such complicated or expensive photodetectors. Instead, we made use of a much simpler form of them - motion detection. Infrared motion sensors are a type of photodetector, one focused in the infrared - or IR - range of wavelengths. It works by measuring the incoming infrared light within its viewing range, looking for shifts in the black-body radiation emitted [5]. These kinds of photodetector usually operate with a Positive-Negative (P-N) Junction semiconductor, which converts the emitting photons into electrical current. This is accomplished via the photon energy forming electron-hole pairs in the depletion region of the P-N Junction. Certain forms of solar cell also make use of this form of photodetector.

When it comes to the Hand Gesture Vending Machine, the photodetector was mounted on the outside of the vending machine and used as an activator - when someone came within range of the photodetector, it did turn on the screen, camera, and the AI development kit. As such, there are a few factors we're specifically looking at. On the top of the list of factors is the range of operation - both in distance and in angle. Second to that is the power usage - if the voltage and current requirements are too high, then we cannot comfortably make use of the infrared photodetector. Lastly, we want to be careful regarding the size of the photodetector itself; if the photodetector is too large, mounting it would be extremely difficult.

The first photodetector we are considering is one the team has previous experience in: a HC-SR501 Arduino photodetector [6]. The HC-SR501 is a small, wide-angle infrared reflective-type sensor that is both low-power and cheap to purchase. The sensor can also be manually adjusted for sensitivity and range. However, with our previous experience comes the problem that the HC-SR501 is not very accurate.

The second photodetector we are considering is simply removing a photodetector from a door sensor. In this specific case, a Dioche Universal Wired Microwave Motion Sensor Detector [7]. Unlike the other options in this comparison, this photodetector is not operating in the infrared spectrum - instead, it's operating in the microwave spectrum. This change allows for significantly less electromagnetic noise from interfering signals, and for increased sensitivity. The cost for that, however, is an increased size, power cost, and actual cost.

The final photodetector we are considering is the E18-D80NK adjustable infrared proximity sensor [8]. This specific photodetector is more expensive than the HC-SR501 photodetector, but significantly cheaper than the Dioche microwave motion sensor. This is also the middle option for its size - larger than the HC-SR501 but smaller than the door sensor. However, it also has a fairly small angle that it can detect from.

Component	HC-SR501 Arduino PIR Motion Sensor [6]	Dioche Universal Wired Microwave Motion Sensor Detector [7]	E18-D80NK Adjustable Infrared Proximity Sensor [8]
Sensitivity	High	Very High	High
Approximate Size	1" x 1.5" 1.5"	6.2" x 4.5" x 2.4"	3" x 1" x 1"
Electrical Noise	Moderate	Very Low	Low
Operating Voltage	5-20V	24V	5V
Active Range	3m	2.5m	3cm - 80cm
Active Angle	120°	30°	~10°
Peak Wavelength	N/A	~12mm	N/A
Shipping Time	None	Days	Days
Cost	~$3.00	~$30.00	~$7.00

Table 3.2.1.1: Comparison of Infrared Photodetectors.

The final decision came down to a few key factors. Concerns of steering too far from more common optical frequencies, the large size and high power needs left us choosing to forgo the Dioche microwave motion sensor, which left the HC-SR501 PIR sensor and the E18-D80NK proximity sensor. The higher electrical noise and too-wide active angle of the HC-SR501 PIR sensor leads to a high chance of incidental activations - which is a major concern. As such, we did use the E18-D80NK proximity sensor as our photodetector for this project.

3.2.2. Optical Lenses

Lenses are transmissive optical devices, which refract light and warp the perspective of an observer. This allows for the resulting view to be bigger and smaller, warped, or skewed based on the lens and its position related to the observer. Lenses are used in all sorts of applications, and have been in use for thousands of years [9]. From eyesight correctives and image projectors to telescopes and magnifying glasses, even to screens and antennae, lenses have had an impact across all sorts of fields of study [10].

Figure 3.2.2.1: An example of how Lenses - in this case, a Convex Lens - can warp an image.

For the purpose of this project, however, we need lenses in order to form an optical system specifically intended for both a relatively small field of view and a close-range field of view. Without these lenses, the camera for the object detection system was unable to observe the gesturing hands. With this in mind, there are certain circumstances that must be considered.

Lenses come in all sorts of materials, manufacturers thicknesses, sizes, and price ranges - which are all important to consider. The most significant factor, in truth, is cost; optical lenses for personal use tend to be priced anywhere between thirty to a hundred and twenty dollars for singular lenses based on their size, material, and other circumstances. Cost aside, there is also the material of the lenses themselves to consider. There are three commonplace materials for lenses that would be best for our uses. These are Fused Quartz, N-BK7, and N-LASFN9.

The first material to consider is fused quartz. Also known as fused silica, this material is often a synthetic form of amorphous silicon dioxide rather than natural SiO_2, and is considered extremely pure because of it [11]. Unlike most other forms of optical lens material, fused quartz does not have any additives or additional materials put into the final lens creation. While this has an advantage in minimal warping from temperature, it has a downside of being less durable versus most lens options. These styles of lenses are also fairly cheap due to their material cost.

The second material to consider is N-BK7. N-BK7 is a variant of borosilicate glass, which is a type of glass that is primarily made of silica and boron trioxide [12]. Borosilicate glass is most commonly used as a material in reagent bottles and laboratory flasks, as well as household glassware such as food storage. This specific variant of borosilicate glass is borosilicate crown glass. Crown glass is specially-made for optical components, with a lower dispersion value and a relatively low refractive index. These lenses, similarly to the fused quartz lenses, are relatively cheap to produce.

The final material to consider is N-LASF9. N-LASF9 is a special type of flint glass that is primarily made of lanthanum - the fifty-seventh element and a metal in the periodic table [13]. Flint glass is effectively the inverse of crown glass; where crown glass has low dispersion values and a low refractive index, flint glass has high dispersion values and a higher refractive index. Additionally, in using lanthanum for the lens creation, the costs of these lenses tends to be significantly higher than the other two options.

Name	Fused Quartz [11]	N-BK7 [12]	N-LASF9 [13]
Wavelength Range	195-2100 nm	380-2100 nm	380-2500 nm
Index of Refraction	1.46019	1.51886	1.8
Surface Quality	20-10	40-20	40-20
Abbe Number	67.4	64.17	32.20
Density	2.51	2.51	4.44
Young's Modulus (GPa)	73	81.5	104.6
Shear Modulus (GPa)	31	34	40
Coefficient of Thermal Expansion	0.52	7.1	7.4
Relative Cost	Low	Low	High

Table 3.2.2.1: Comparison of Optical Lens Materials.

While there is the option to simply use multiple types of lenses, there is no real need to overcomplicate the design here - as such, we did solely be using N-BK7 borosilicate crown glass for the chosen lens material. Its relatively lower cost is a significant factor, as are its superior durability and surface quality. The slightly decreased abbe number (meaning increased dispersion) is a worthwhile tradeoff, overall.

There is another point to consider, material selection complete. Namely, the best optical manufacturer to pick from for our project. The biggest consideration here is cost - the cheaper these lenses can be obtained, the better. The simplest way to handle this is to determine the necessary lenses for the optical system based on available lenses, then check the respective stores of several optical manufacturers in order to compare which ones have the cheapest overall cost. At an initial glance, it seems that the best options are Edmund Optics and ThorLabs for purchasing our future optical lenses.

3.2.3. EM Transmitter and Receiver

Transmitters and Receivers are an old technology. In the past, we used simple fire in the night to signal others - the fire is the Transmitter, and our eyes the Receiver. But as times changed, we came to develop a much more advanced modern transmitter: radiowave-based communications. From here, modern transmitters and receivers evolve in all sorts of directions.

A transmitter, put simply, sends a signal of some kind through air or vacuum. A receiver, on the other hand, seeks out a specific signal. The underlying concepts can be compared to a mouth and an ear - one is speaking, and one is listening. This idea remains the same, regardless of the wavelength, but the means to accomplish it change. In the case of this project, we did use the infrared portion of the optical spectrum. The purpose for these transmitters and receivers is simple: they were checking for a product's availability. If it is available, the receiver will be blocked by one of the products. If the product is out, the receiver did catch the transmitter's signals and inform the microcontroller. This did stop customers from incidentally succeeding in purchasing a product that is out of stock.

Figure 3.2.3.1: An example of a commonplace Infrared Transmitter: a Television Remote.

When it comes to these transmitters and receivers, there are three critical factors we are considering. The first is size - if they are too large, then they might get in the way of product vending, or in avoiding that lead to a larger prototype size. The second thing to consider is cost. To help mitigate financial burden and emphasize the affordability of the design, we must avoid costly components unless they are deemed necessary where possible. The final critical factor is power usage. We do not want to overwork the microcontroller or power supply if we can help it, and being mindful of the power draws needed is the best way to avoid it.

The first transmitter-receiver pair to consider is making use of independently-set infrared Emitter LEDs and Receivers - like the EAILP05RDDB1 Infrared Emitter and the LUCKYLIGHT LL-503PDD2E Infrared Photodiode [14, 15]. These emitters and photodiodes are simple individual diodes, one an LED and the other a photodiode; this extremely small size would allow for ease of placement within the vending machine. Using the diodes alone also makes for an extremely cheap price point. However, these diodes coming alone is an advantage as much as it is a risk; the required resistors and capacitors have to be calculated and set into place in the PCB.

The second transmitter-receiver pair to consider is the HX-35 Infrared Transmitter and HX-M121 Infrared Receiver [16]. This is extremely similar to the first option, except it forgoes some of its smaller size and price in exchange for being prepared with the required components in advance. These are still relatively small and cheap, with the price increase mostly being a matter of convenience and current control. This larger size could cause issues with incidentally blocking the path of products or increasing the overall size of the vending machine, but with careful mounting of the components, that should be avoidable.

The final transmitter-receiver pair to consider is one with a much higher frequency than the other two - a three hundred and fifteen megahertz (315MHz) Radio Frequency Kit, in fact. This comes with a unique advantage of being able to hide the sensors within the shell of the vending machine rather than having them partially exposed - but it similarly comes with risks of the aluminum of the snacks being insufficient to diffuse the wavelengths of the kit. These transmitters and receivers are also notably larger and more expensive than the other options.

Name	**EAILP05RDDB1 / LL-503PDD2E [14, 15]**	**HX-35 / HX-M121 [16]**	**315MHz Radio Frequency Tx/Rx Kit [17]**
Size	~0.2” x 0.2” x 1”	1.5” x 1” x 0.5”	~3” x 0.5” x 2”
Operating Voltage	1.2V - 1.5V / 0V	5V / 5V	3.5V - 12V / 5V
Operating Range	Up to 5m	Up to 10m	Up to 200m
Operating Wavelength	850nm - 1050nm	850nm - 940nm	N/A
Peak Wavelength	940nm	~900nm	~95.2cm
Cost	~$0.50 Per Pair	~$2.00 Per Pair	~$6.00 Per Pair

Table 3.2.3.1: Comparison of Infrared Transmitter-Receiver Pairs.

On the offset, the three hundred and fifteen megahertz (315MHz) Radio Frequency Kit is both too risky and too far from what would be considered common optical frequency; as such, we stopped considering it as an option. Following that, it was difficult to decide which of the two remaining options we should go with. On one hand, working solely with the EAILP05RDDB1 and LL-503PDD2E diodes allows the most flexibility in the internal design of the Hand Gesture Vending Machine, which would be amazing to take advantage of. On the other hand, we are limited on time and resources - especially given our decreased time in the summer semester - so minimizing testing and workload with the premade HX-35 and HX-M121 components is an equally significant boon.

With both of these factors in mind, we had decided on the HX-35 and HX-M121 Infrared Transmitter-Receiver pair. We came to the conclusion that having more time to perfect the rest of the prototype would be more advantageous to us than simplifying the internal design - as the aesthetic, while nice, is simply not significant to the functionality.

3.3. Electronic Components

The electronic components within the Hand Gesture Vending Machine are some of the most important to the project; half of the team is made up of electrical engineering students, after all. As such, this specific subsection contains the most components of interest out of our related project research. First, there is the power supply. With the Hand Gesture Vending Machine being plugged in, a power supply is a must-have component to control electrical flow into the microcontroller and the other components. Second, there are the fans - which are necessary for cooling the AI development kit both during and after operation. Third, there are the four motors; these were used to push products within the vending machine, much like many vending machines on the market. Fourth is our selection of simple LEDs. This did allow for an affordable and out-of-the-way lighting option that won't be power-hungry. Fifth is the RFID reader, which mimics a card reader for the purpose of our prototype. Sixth is our screen, which can be used to visualize the selected product after hand-gestures - and show when a customer needs to pay.

3.3.1. Power Supply

In the United States, standard household electrical outlets typically provide an AC power at the voltage of around one hundred and twenty volts RMS at a frequency of sixty hertz [18]. Power supplies designed for use in the US are built to accommodate these voltage and frequency specifications. Most electronic devices operate on lower DC voltages, so the power supply needs to convert the higher AC voltage to the required DC voltage through a process of rectification and regulation.

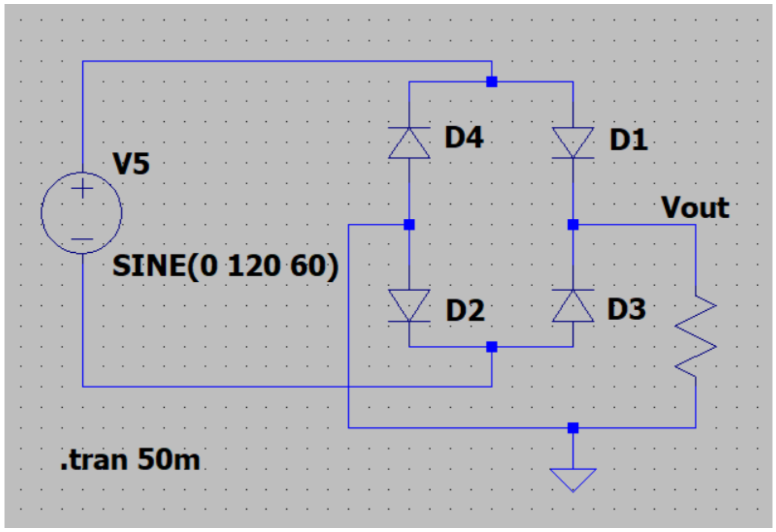


Figure 3.3.1.1: Full Wave Bridge Rectifier in LTspice.

The full wave bridge rectifier is the first step to converting AC to DC by using four diodes. It is a combination of two simple half wave rectifiers to utilize both halves of the AC input voltage waveform [19]. In the positive half cycle D3 and D4 are in reverse bias and the current flows through D1 and D2 to V_{OUT}. Then during the negative half cycle D1 and D2 are in reverse bias and the current flows through D3 and D4. The current is always going through the load in the same direction. This creates the rectified output waveform in Figure 3.3.1.2.

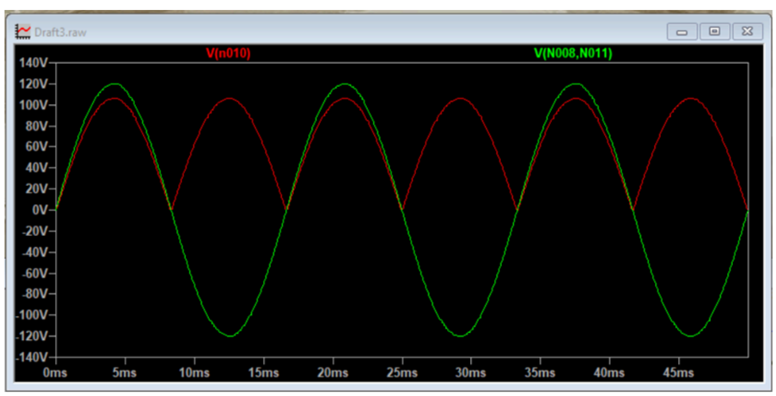


Figure 3.3.1.2: Full Wave Bridge Rectifier simulation in LTspice. The green line is the positive side of the voltage source with the probe reference on the negative side of the voltage source. The red line is the output of the full wave rectifier.

We can then put a smoothing capacitor in series with the load to smooth out the rectified output. The capacitor is charged to the peak voltage of the output DC pulse then discharges filling gap between pulses turning the rectified output into something closer to a DC output. Figure 3.3.1.3 shows the full wave bridge rectified with the parallel smoothing capacitor in parallel with the load.

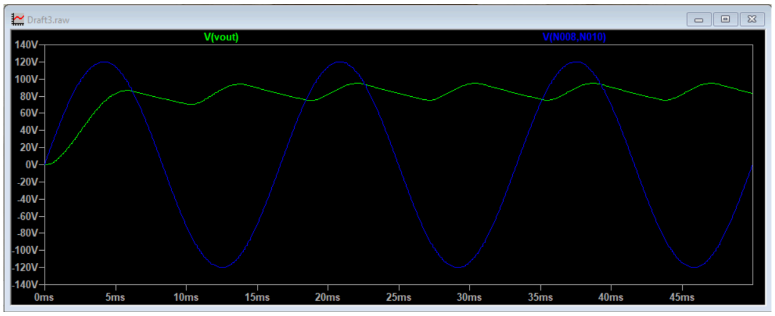


Figure 3.3.1.3: Full Wave Bridge Rectifier with smoothing capacitor simulation in LTspice.

Following our rough estimates from our overall machine requirements, in our overall machine requirements, we had a maximum outlet voltage and amperage of fifteen volts and ten and a half amps. One of the key features we want is to have the machine plug straight into the wall. We do not want to have multiple small power supplies connected to a power strip adding more points of failure, not to mention bad practice. You won't see your refrigerator having a power strip built in for your ice maker. In doing so we add simplicity to our machine and give ourselves ample opportunities to have sufficient PCB design.

When you look up power supplies on Google, the first result is about power supplies made for computers. There is an entire industry built around power supplies meant for render farms, servers, and computer gaming. Millions of people who build their own computers by picking out components have created a demand for companies to put out a variety of power supplies. These power supplies have different form factors, wattages, noise level, and efficiency. There are also power supplies meant for the server space; high wattage power supplies that have no need to be quiet. However, these power supplies are highly specialized being plugged into computer components. Even though we can find documentation for these power supplies, it is easier for us to have a power supply that isn't built for one purpose.

One of the potential power supplies on the market was a ALITOVE 24V 15A 360W Power Supply [20]. There is also a chance that someone could come into contact with the exposed wire or contacts and hurt themselves. This power supply is designed to be installed inside of equipment. Since we are only working on a prototype, things did move around constantly. Picking another safer option would be the better choice. Though using it in a finished project would be a good use case for this product.

The TRIAD WSU240-100-R is one of the cheapest options with twenty-four volts at $13.32 [21]. However, the maximum current is only one amp. Even with a buck converter increasing the current while lowering the voltage, it would be difficult to have sufficient current to control all of the components. One of our main goals is to have our device powered by a generic wall outlet with only one connection to the wall; to use this power supply, we would likely need to use multiple. We could achieve that by using a power strip mounted inside of the machine, but that runs the risk of electrical issues - or even a fire.

Our followup option was to seek a higher amperage power supply with the same voltage. This led us to the ALITOVE 24V 6A 144W [22]. Similar to the TRIAD, it also has an enclosed casing with just a smaller inner diameter barrel connector. These are smaller wall adapters called barrel connectors or barrel jacks; the TRIAD WSU240-100-R and the ALITOVE 24V 6A 144W use them, and they are fairly commonplace. Specifically, they are commonly used in laptops, routers, LED lighting systems, and audio equipment. The cylindrical shape of the connector also ensures a stable connection.

As an advantage, barrel connectors come in various sizes. We can source barrel jack connectors and incorporate that into our PCB so we can have an easily removable power supply, something we could have not done with the ALITOVE 24V 15A 360W so easily. Our two options have differing inner diameters so picking the right female counterpart for our board is important.

Name	**ALITOVE 24V 15A 360W [20]**	**TRIAD WSU240-100-R [21]**	**ALITOVE 24V 6A 144W [22]**
Input Voltage	[AC] 110V, [DC] 220V	[AC] 100V, [DC] 240V	[AC] 100V, [DC] 240V
Maximum Output	[DC] 24V, 15A	[DC] 24V, 1A	[DC] 24V, 6A
Connector	None	Barrel connector 5.5mm x 2.1mm	Barrel connector 5.5mm x 2.5mm
Safety Concerns	Exposed contacts are potentially dangerous.	Enclosed casing prevents contact with live components.	Enclosed casing prevents contact with live components.
Cost	$23.99	$13.32	$23.99

Table 3.3.1.1: Comparison of Power Supplies.

After some consideration, we have decided to use the ALITOVE 24V 6A 144W power supply. The ALITOVE 24V 15A 360W power supply, both lacking a connector and having exposed contacts, was something we decided was too risky to attempt and hotwire for our own uses. On the other hand, the TRIAD WSU240-100-R has too low a current output; as such, we could not use that power supply either. Through the process of elimination and convenience, we arrive at the ALITOVE 24V 6A 144W power supply.

3.3.2. Converters

None of our components have the same voltage requirements so we did need a way to safely provide power to each component without damaging them. To do that we did abide by each of the component's manufacturer's provided user manuals. Decreasing the voltage could be achieved with two resistors by voltage divider or using a voltage regulator but those methods dissipate heat. Dissipating power by heat significantly hurts the efficiency of our device. It is wasted energy. To avoid that we can use something called a buck converter or also called a step-down converter.

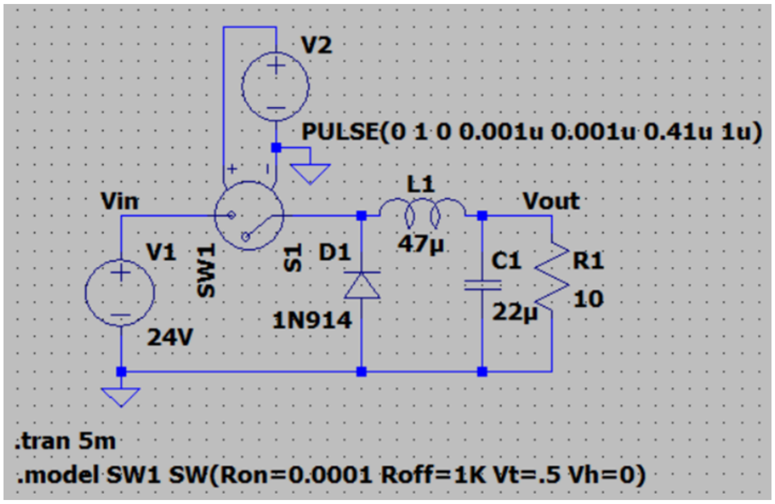


Figure 3.3.2.1: A Buck Converter in LTspice.

A buck converter is a DC-to-DC converter that decreases voltage, while increasing current. Figure 3.3.2.1 is the schematic of the buck converter built in LTspice. LTspice is a widely used circuit simulation software developed by Linear Technology Corporation. The capacitor maintains a relatively stable output voltage across the output resistor. The output resistor represents the component we are supplying power to, like our microcontroller or single board computer. The voltage controlled switch is controlling the duty cycle of the input. The actual circuit was controlled using a timer integrated circuit like a 555 timer IC. When the switch is closed, current does flow from the source through the load resistor, inductor, and capacitor. With some energy being stored in the inductor and capacitor. The diode is off in reverse bias mode.

When the switch is opened after being closed for some time, the inductor does become a current source resulting in the diode being on in forward bias mode. The inductor and the capacitor did discharge themselves through the load resistor. Then by adjusting the duty cycle of the input, we can adjust the output voltage of the circuit. The duty cycle is the percentage that the switch is closed over the period.

$$V_{OUT} = V_{IN} * D$$

Figure 3.3.2.2: Formula for Voltage Output (V_{OUT}), given Voltage Input (V_{IN}) and the Duty Cycle (D) assuming 100% efficiency [23].

For example, a component that requires nine volts would be damaged from a twenty-four voltage source.. A buck converter can then be used to take the twenty-four volts from the power supply and step it down to around nine volts that the microcontroller needs with a duty cycle of 37.5%. Using LTSpice, below is the result of simulating the buck converter seen in Figure 3.3.2.1.

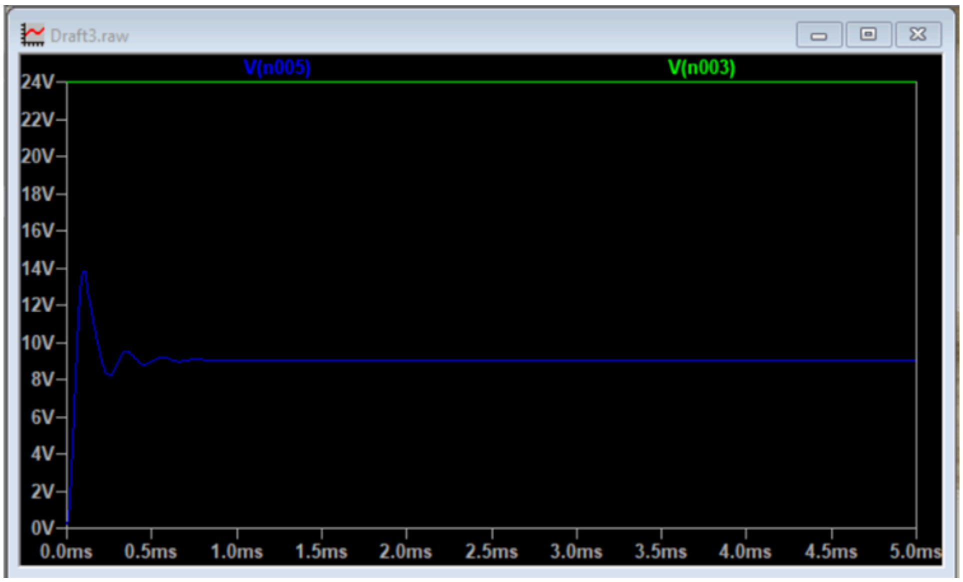


Figure 3.3.2.3: Buck converter simulation in LTspice. The green line is the input voltage, and the blue line is the output voltage.

As per Figure 3.3.2.3., the output voltage is in the voltage range of our microcontroller. From the one hundred and twenty volt AC wall outlet to twenty-four volt power supply to buck converter, the board is being supplied the power it needs to operate safely. The spike seen at the start in the Figure above can then be dealt with a voltage regulator. Inversely, a boost converter or a step-up converter is a DC-to-DC converter that increases voltage, while decreasing current. Since our power supply is a higher voltage than any of our components, we probably did not use any boost converters.

3.3.3. Fans

The Hand Gesture Vending Machine has components enclosed in a small space. Naturally the electronics did generate heat so we did use fans to cool down the components.. We won't have but it is a good idea to keep the air inside from being stagnant with food inside.

Much like power supplies, affordable and a great variety of general purpose fans have become a competitive industry thanks to the PC building market. We did be able to benefit from this because unlike the power supplies, the fans are still relatively simple with up to four pins. There are companies that produce fans that prioritize lighting features, size, speed, noise, power delivery, airflow, and static pressure. Right off the bat, lighting features are unnecessary. That is just cosmetics, which aren't important right now.

Fan size describes the horizontal or vertical size of the mount. Since we are designing the enclosure ourselves, picking fan sizes is arbitrary. If we find that we need smaller fans we can find options then. There are fans as small as 12mm. That fan could be used to move air through a tight space. One of the most common fan sizes used in computers is a 120mm fan. It strikes a balance between airflow, noise level, and size. Two different sizes of fans can produce the same airflow. However a smaller fan with smaller blades did produce higher frequency sounds that can be annoying to anyone around them. Larger fan blades can have the same airflow with a lower rotations per minute while being much quieter. A company known for their quiet fans is Noctua.

Noctua is an Austrian company that has been designing fans for almost twenty years [24]. They have spent years designing a new plastic compound to give their fan blades more rigidity. The Noctua NF-P12 redux-1700 PWM is a 120mm Noctua fan that has specially designed fan blades that are optimized for quiet operation, optimized airflow and static pressure [25]. The price of a 120mm fan from Noctua is $16.95 which makes them an expensive option.

Two common ways to power fans are by PWM. PWM or pulse width modulation is a technique used in electronics to encode analog information in a digital signal [24]. A fan that uses PWM did need another pin to send PWM signals. Fans with only two pins can only be controlled by changing the input voltage. By changing the duty cycle, PWM can effectively simulate an analog output voltage. Sending a signal with a 50% duty cycle would represent the halfway point of the maximum and minimum RPM. With a temperature sensor, the system could regulate its temperature and change the duty cycle to match the airflow the system needs.

The CFM-A225C-115-287 is a 120mm fan with no PWM pin so it only has two pins [25]. Changing the input voltage is the only way to change the speed of the fan. Rotational speed is lower but we will probably never need to be at maximum anyway. The fan also has an affordable price at $6.19.

Name	**Noctua NF-P12 redux-1700 PWM [24]**	**CFM-120C CFM-A225C-115-287 [25]**
Input Voltage	12V	5V
Size	120 mm x 120 mm x 25 mm	120 mm x 120 mm x 25.4 mm
Max input current	90 mA	240 mA
Acoustical noise	25.1 dB(A)	24.8 dB(A)
Pin Configuration	4-pin PWM	2 Wire Leads
Minimum Rotational Speed	450 RPM	0 RPM
Maximum Rotational Speed	1700 RPM	1500 RPM
Static Pressure	27.75 Pa	14.93 Pa
Airflow	$120.2\ m^3/h$	$88.87\ m^3/h$
Cost	$16.95	$6.19

Table 3.3.3.1: Comparison of Fans.

We did go with the much cheaper CFM-120C CFM-A225C-115-287. We can get almost three fans for the price of one Noctua NF-P12 redux-1700. PWM is going to add more complexity to our design. We did use the fans to keep air moving in and out of our machine. The Noctua datasheet says their fan has a higher acoustical sound however the Noctua fan can also go up to 1700 RPM so that is probably where the loudest noise comes from. We tuned something on our power delivery circuit to pick an RPM of the fan that doesn't have annoying high pitch frequencies and aim for a low ambient noise.

3.3.4. Motors

The electric motor is a technology that found its roots back in the 1740s, when it was first conceived by Benjamin Franklin [27]. However, the first electrical motor wasn't actually built until nearly a century later, by William Sturgeon in the year 1832. These first motors made use of Direct Current (DC), and were as such called DC motors. Just over half a century later, the first Alternating Current (AC) motor was made by Galileo Ferraris, shortly followed by Nikola Tesla with improved designs.

Since then, we've come to grasp exactly what makes these various motors tick - both upscaling and downscaling them with significantly less difficulty than we used to have. We've also made use of them in all sorts of technology, from doors to cars to drills and more. In our case, motors have also seen significant use in vending machines, giving us a place to build off of for our own research.

When it comes to our own purposes, we have to first decide which type of motor to use: AC, or DC. AC motors are more energy-efficient on startup, and tend to generally be more energy efficient in general versus DC motors. However, this comes with increased internal complexity, size, and cost. At a larger scale, this would certainly be a serious consideration for us to make. At the small scale we are working on, however, the efficiency difference is relatively negligible; as such, we have little reason to go with AC motors for our design. With this in mind, we did be selecting solely from DC motor options.

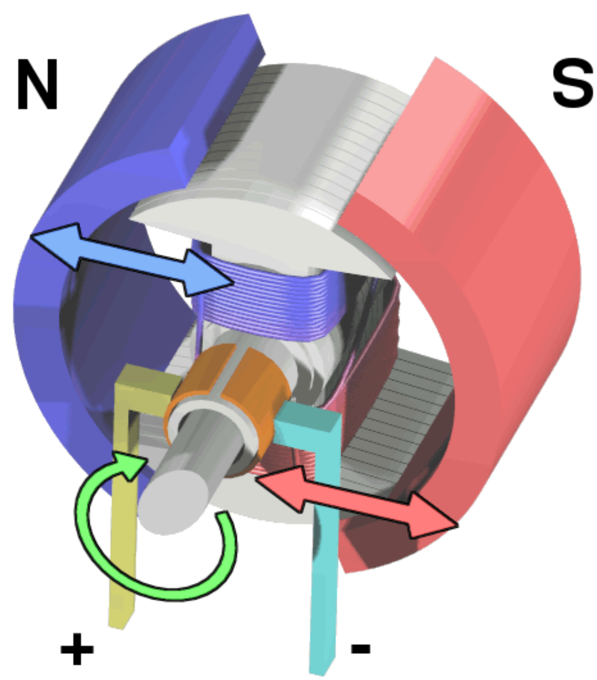


Figure 3.3.4.1: A Visualization of a brushed DC Motor.

For our purposes, we're looking for a few key factors in our DC motors. Ease of modification and cost are at the top of this list - as we did need to be able to mount the metal spiral for the motor to turn, and we would need to do so within relatively low costs. Power usage comes in after this; if the power requirements for these motors is too high, then it simply becomes infeasible to use the motors selected. The last major consideration is the size of the motor itself. One of our constraints is keeping the size of the prototype small, and we can only accomplish that by keeping in mind the size of all the components, backend ones included. One final note is that speed is not a significant factor - so long as the RPM is not too low, the motor is acceptable for use.

There is already a significant collection of vending machine motors available for purchase on the market. However, finding data specifications for any individual motor used within these vending machines has been a significant challenge, which has led to us mostly forgoing any market-available vending machine motors by virtue of lacking information. If we can locate a means to quantify the abilities and requirements of these existing vending machine motors, then we may return to this with more options to choose from.

The first motor we have considered are N20 High Speed Micro Motors [28]. These are available on the market under many names, but in short the N20 is an incredibly small high-speed motor - sitting under two centimeters in every dimension. They require six to twelve volts and only forty microamps to operate. The motor is fast, but lacks significant torque and a means to mount it - which means it would require a small gearbox to be fitted to it, and either a custom-made bracket or one purchased from online. These are similarly available, like the N20 Micro Motors are.

The second motor under consideration is a variant of the N20, the G12-N20 Geared Mini DC Motor [29]. These are very small, lightweight motors with a need of six to twelve volts and forty microamps - just like the N20. Despite their small size, they have an impressive torque of two kilogram-centimeters and a stall torque of sixteen kilogram-centimeters. For our purposes, we were barely moving a single kilogram at a time with the turning coil - so this is firmly within our tolerances. These tiny motors manage this impressive torque due to having a built-in gearbox - making adjustments easy and on the fly. However, these motors lack a mounting bracket - meaning one has to be made or purchased elsewhere.

The third and final motor that we have considered is the ASLONG JGY-370 12V DC Worm Gear Motor [30]. This motor is notably larger than the other two, but not too large to be unreasonable. The ASLONG Worm Gear Motor has the highest torque of the three options, with a baseline torque of ten kilogram-centimeters. Additionally, this motor both has an internal gearbox and mounting points - allowing us to set it into place with less hassle. Similarly to the other two, the input voltage is twelve volts. The input current, however, is just under a third of an amp - significantly more than the other two.

Name	N20 High Speed Micro Motor [28]	G12-N20 Geared Mini DC Motor [29]	ASLONG JGY-370 12V DC Worm Gear Motor [30]
Size	~8.5mm x 11mm x 20mm	~10mm x 12mm x 30mm	~8cm x 3.5cm x 3cm
Input Voltage	6V - 12V	6V - 12V	12V
Input Current	0.04A	0.04A	0.3A
RPM	4,600 - 20,000	10 - 100	5
Torque	<1kg*cm	2kg*cm	10kg*cm
Mounting Bracket?	No	No	Yes
Cost	~$2.00 Each	~$4.50 Each	~$16.00 Each

Table 3.3.4.1: Comparison of Motors.

Given the nature of the N20 Micro Motor, we disregarded it quickly; the G12-N20 is effectively the same motor with a gearbox already fitted on, after all. When it came to comparing the G12-N20 to the ASLONG JGY-270, the near quadrupled cost, the over quadrupled size, and the nearly tenfold increase in input current were enough of a reason to stick with the G12-N20 Geared Mini DC Motor.

3.3.5. LEDs

LEDs are the topic of discussion for the component in this section. An LED is a semiconductor device that emits light when an electric current is passed through it [31]. When the particles that carry the current combine together within the semiconductor material, then light is produced. The LED component is widely used in various applications, including lighting, displays, signage, indicators, and signage, due to their energy efficiency, durability, compact size, and versatility.

The availability of LEDs come in a wide range of colors, including red, blue, yellow, and green. The color comes from the process of inside the semiconductor material of the LED; the holes and electrons are contained within energy bands, with the separation of bands determining the energy of the photons that are emitted by the LED. The photon energy did then determine the wavelength of the emitted light, that is where the color is coming from. An LED did consist of a semiconductor material, typically composed of gallium arsenide (GaAs) and gallium phosphide (GaP). The low power consumption and compact size of LEDs make them ideal for use in portable electronic devices, illuminated signage, automotive lightning, and more.

Having the appropriate LED is crucial for the hand gesture vending machine project. The LED did play several essential roles that contribute to its functionality and effectiveness. A big thing needed from the LED is the illumination. The LED did provide much of the source of illumination for the vending machines display panel, making it visible and easily readable to users. When speaking on LEDs we can count on them being energy efficient. When compared to traditional lighting such as fluorescent sources LEDS are highly energy efficient. The Hand Gesture Vending Machine can minimize power consumption while maintaining bright illumination, by using LEDs. Lastly, using LEDs allow for customization to be done with the hand gesture vending machine. There is much flexibility to be offered in terms of brightness, color, and lighting effects, allowing for customizable display options tailored to the vending machines aesthetics.

Figure 3.3.5.1: A White-Light LED.

When selecting the right LED for this project there are several factors that did go into consideration to ensure that the chosen components meet the project's requirements effectively. Factors can range from brightness, power efficiency, color options, durability, and cost-effectiveness. We did present some LED options that are commonly used in vending machines.

The first LED option being presented is RGB LEDs, all RGB stands for is red, green, blue, respectively. RGB LEDs can adjust brightness levels, providing flexibility in achieving desired illumination levels for the vending machines display [32]. By mixing red, green, and blue light components, RGB LEDs can produce a wide range of colors. It is possible for RGB LEDs to be more expensive than single color LEDS due to additional functionality. RGB LEDs normally consume more power compared to single color LEDs due to the need for multiple light sources. The particular RGB LED we're looking at is the Kingbright WP154A4SUREQBFZGC [33]. Some features of this LED have uniform light output, low power consumption, and long life-solid state reliability. These LEDs are used for status indicator, illuminator, decorative and entertainment lighting.

The next option for LED options is the white LEDs. White LEDs are great solutions when looking for cost effective lighting solutions, offering a balance between performance, affordability, and reliability. White LEDs are suitable for providing clear visibility of product displays and user interfaces, being able to offer uniform and high-intensity illumination [34]. The color temperature varies with white LEDs. It can range from warm white (lower color temperature) to cool white (higher color temperature), allowing for customization based on application requirements. Overall, the white LED has many features that would be great with the hand gesture vending machine. The white LED that is being looked into is the CreeLED C513A-WSS-CW0Z015 [35]. This LED offers superior light output and provides stable light output over long periods of time. Since the LED is made of advanced optical grade epoxy it offers a high temperature and high moisture resistance performance in lighting and illumination applications.

Single color LEDs are another great option. They offer consistent brightness output in specific colors, making them bright. The power consumption involved with single color LEDs are relatively low, making them energy efficient and suitable for low power applications. These LEDs provide excellent value for your money in terms of performance and longevity. The LED we're looking into particularly is the SunLED XLUR12D [36]. This component has low power consumption, is reliable, and robust.

The last LED option we have is the surface mount LED. These LEDs are small LED packages that are mounted onto circuit boards using surface-mount technology [37]. The LEDs offer high brightness levels and energy efficiency, providing excellent illumination while minimizing power consumption. The Samsung LM301B offers a 0.3 W class middle power LED, and mold resin for high reliability [38]. This LED offers high efficiency and color consistency, making it ideal for this project.

Specification	**WP154A4SUREQBFZGC**	**C513A-WSS-CW0Z015 White LEDs [35]**	**XLUR12D Single-Color LEDs [36]**	**LM301B Surface-Mount LEDs [38]**

	RGB LEDs [33]			
Brightness (lumens)	200 mcd-3600 mcd (Depending on LED Color)	5400 mcd	39 mcd	3000 mcd - 5700 mcd
Color Options	460 nm - 465 nm, 515 nm - 525 nm, 630 nm - 645 nm	White*	575 nm - 675 nm	425 nm - 500 nm
Power Dissipation	75 mW - 102.5 mW (Depending on LED Color)	120 mW	75 mW	520 mW
Viewing Angle	50 Degrees	55 Degrees	30 Degrees	120 Degrees
Cost	~$2.00 Each	~$0.28 Each	~$0.28 Each	~$0.40 Each

*Table 3.3.5.1: LEDs Comparison. *No clear numerical data available.*

For this project when considering factors such as power consumption, brightness, and cost effectiveness, I believe the best option to pick is the white LEDs. The white LEDs offer high brightness levels, ensuring that the users can visibly view products on display. White LEDs did also offer a balance between reliability, performance, and affordability, all at a coast effective cost. Using these LEDs we'll be able to achieve optimal illumination for the vending machines display and user interface while minimizing power consumption.

3.3.6. RFID Reader

RFID, which stands for Radio Frequency IDentification, is used to identify a wide variety of items, such as books, food, and electronic devices through the use of radio waves. This is actually accomplished by the system scanning for and identifying an RFID tag, which is placed on or beside an item to identify itself. An RFID reader can be used for the Hands Free Vending Machine in order to emulate the vending machine reading a credit card for purchase; we did be using imitation credit cards for imitation payment.

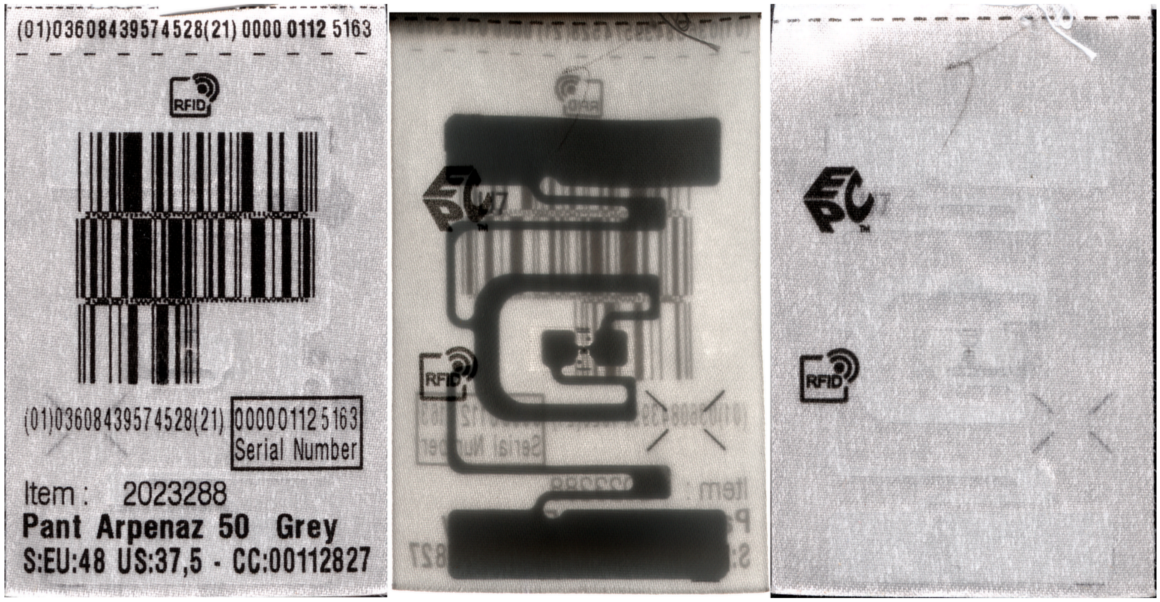


Figure 3.3.6.1: An example of an RFID label within a shirt tag.

One type of RFID system is a Low-Frequency RFID Reader. Its frequency range of communication ranges from thirty to three hundred kilohertz [39]. It specializes in access control and object tracking such as animals and livestock. Its range is six inches. A disadvantage of this system, however, is how slow it is; it operates at or less than ten kilobits per second. On the other hand, the low frequency RFID is very water resistant and has a high resistance to metal-based interference.

Another type of RFID reading system is a High-Frequency RFID Reader. Its frequency range of communication ranges from three to thirty megahertz, several magnitudes higher than that of the low-frequency option [39]. It specializes in access control, payments, and target object tracking. The high frequency RFID's range is twelve inches. A disadvantage of this system is how the high frequency RFID is not very water resistant, and metal sometimes can interfere with the signal. An advantage for the high frequency RFID is that it is more versatile; it has more uses. Another advantage for the high frequency RFID is that it is fast, when compared to the low frequency RFID; the high frequency RFID can operate at four hundred and twenty-four kilobits per second.

Specification	Low-Frequency RFID Reader	High-Frequency RFID Reader
Frequency Range	30 kHz - 300 kHz	3 MHz - 30 MHz
Active Range	6”	12”
Operating Speed	10 kbits/sec	424 kbit/sec
Moisture Resistance	High	Low
Frequency Noise	Low	Moderate

Table 3.3.6.1: Comparison between Low and High Frequency for RFID Readers.

In this project, we’ll be using the RFID to emulate the hands free vending machine reading a credit card for purchase. Therefore, the attributes prioritized are speed and resistance to metal-based frequency noise. Speed is prioritized because response time is very important in the grand scheme of this project. Resistance to metal interference is also prioritized for several reasons, from the fact that much of the Hands Free Vending Machine is partially or entirely metal and having noise runs a significant risk, to the fact that noise can lead to incidental purchases in products. Speed can be seen in High-Frequency RFID Readers; as said before, a High-Frequency RFID can operate at over four hundred kilobits per second. Resistance to metal, on the other hand, can be seen in a Low-Frequency RFID. On the other hand, a High-Frequency RFID Reader struggles to work around metal. Although the speed and response time are extremely important to this project, the resistance to metal-based frequency noise is the most integral to the project. For this reason, the RFID type we are selecting is the low frequency RFID.

3.3.7. Screen

A very important component that’s being researched for the hand gesture vending machine in this section is the screen. This project did encompass a variety of display technologies suitable for showcasing product information, user interfaces, and interactive content. The screen did serve as the primary means of communication between the vending machine and the user, providing visual feedback, and promotional messages.

The screen options we chose from did offer a range of features and capabilities suitable for different applications and user preferences. By doing research and evaluating the screen technologies, the team can come together to make an informed decision based on factors such as display quality, power consumption, cost, and suitability for the hand gesture vending machine.

When selecting the appropriate screen for the hand gesture vending machine project it is crucial that we select the one that fits our needs. The screen did directly impact the user experience, functionality, and overall success of the vending machine. I did list a few things we did need from the screen to produce a functioning hand gesture vending machine. One of the main things needed from the screen display is user interaction. The screen did serve as the primary interface for when the user interacts with the vending machine, providing complete transactions, visual feedback, and prompts to make selections. By choosing a screen that offers clear visual cues, the hand gesture vending machine can deliver a seamless and user-friendly experience.

Another thing needed from the screen is the information display. The screen did display information such as pricing, and product descriptions. This allows for users to be able to make informed choices and ensures transparency in product offerings. Having a sufficient enough of a screen did offer sufficient display area and legible text. Product showcase is another thing that is needed on the screen. The screen display did allow for the hand gesture vending machine to showcase product information and messages. When selecting a high-resolution screen with vibrant colors and clear visibility, this allows for the vending machine to effectively showcase products. By selecting the right screen needed for the hand gesture vending machine it is essential for creating an engaging, user friendly, and visually appealing interface.

When selecting the best screen for the hand gesture vending machine needed for this project, there are several key factors to consider to ensure it meets the project's requirements and objectives effectively. These are visibility, resolution, size, durability, power efficiency, integration capability, and documentation.

It is important that the visibility of the screen is clear and legible, even in various lighting conditions such as indoor and outdoor environments. In regards to visibility, it should have a high brightness level to ensure readability. With high resolution, we would be able to display the detailed product information, graphics, and sharpness. A higher resolution also enhances the visual appeal and professionalism of the vending machine’s user interface. When it comes to screen size, it should be large enough to display all necessary information and graphics effectively without compromising the vending machine’s design.

As said before, durability is also an important factor that we need from the screen. The screen should be durable and resistant to scratches, impacts, and environmental factors such as dust, moisture, and temperature fluctuations. A tempered glass screen can withstand even harsh conditions and ensure long term reliability. What is also needed is a power efficient screen; we did need for the screen to consume minimal power to optimize energy efficiency and prolong battery life or reduce power consumption in the vending machine. Energy efficient screen technologies such as OLED can help minimize power usage.

The integration with the screen, the vending machines hardware and software is additionally very important, and something we need for the success of the hand gesture vending machine. The screen did need to seamlessly integrate with software development tools, standard communication protocols, sensors, and microcontrollers for example. Some last major factors needed from the screen are the cost effectiveness and support/documentation. The screen should offer a balance between features, performance, and cost to meet budget constraints while delivering satisfactory functionality and quality.

Lastly, we did need a screen that is backed by documentation, technical support, maintenance, and a developer community to assist with implementation and troubleshooting. By prioritizing these factors and selecting a screen that excels in these areas, the hand gesture vending machine can deliver an engaging, user friendly, and visually appealing interface that meets the needs and expectations of its users.

There are various different screens to work with in regards to the hand gesture vending machine. The first screen we'll be talking about is a Liquid Crystal Display (LCD) screen. An LCD screen is made up of components such as a backlight and a liquid crystal layer [40]. LCD screens can be grouped into three categories, twisted nematic (TN), in-plane switching (IPS), and vertical alignment (VA). While these screen types do belong to the LCD screen type, they use thin-film-transistor (TFT) technology which is just a variant of the standard LCD screen type. The main features that differentiate between the three different types of LCD screens are brightness, viewing angles, color, and contrast.

To begin with, the TN LCD screen type consists of nematic liquid crystals sandwiched between two plates of glass, when power is applied to the electrodes, the liquid crystals twist ninety degrees. TN LCDs have a faster response time and refresh rate compared to the other LCD options. With a faster response time and refresh rate this allows for faster moving images or graphics to display across the LCD screen with extreme precision. The next LCD screen option is VA. The displays with VA screens can deliver wide viewing angles, high contrast, and good color reproduction.

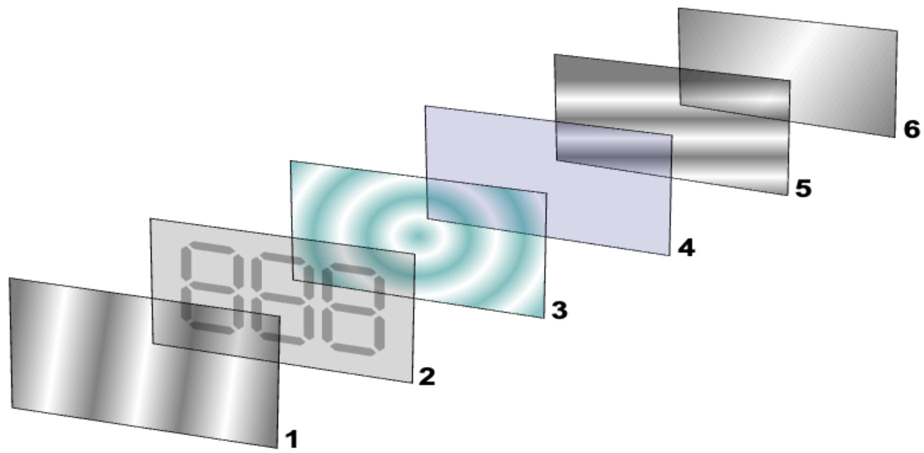


Figure 3.3.7.1: Layers of a reflective twisted nematic LCD.

Our last option for LCD screens is the IPS LCD. In terms of the features, it is superior in contrast, brightness, viewing angles, and color representation compared to TN screens. The technology associated with IPS displays improves image quality by acting on the liquid crystal inside the display screen, when voltage is applied the crystals rotate parallel rather than upright to allow light to pass through. If picking an LCD screen as the type of screen for this hand gesture vending machine, then the IPS option would be the best option.

Another option for types of screens to use for the hand gesture vending machine would be an OLED display. The way an OLED screen works is by generating images by applying electricity to organic materials inside the display [41]. There is no backlight, so each pixel gets its own illumination with the OLED. By placing a series of organic thin films between two conductors you're able to make an OLED. What makes an OLED organic is the fact that they are made from carbon and hydrogen. When looking for a display an OLED has a sharper image with high contrast, much bolder than an LED [42]. OLEDs have high color accuracy and are extremely versatile.

A touchscreen display is another option for the type of screen that can be offered for the hand gesture vending machine. The technology within the display offers multi-touch support and smooth gesture recognition [43]. The resistive touchscreen displays detect touch input based on pressure applied to the screen. The cost effectiveness for a touchscreen display can range from a moderate price to a high price, all depending on the technology. There are multiple types of touchscreens ranging from resistive touchscreens, capacitive touchscreens, or even infrared touchscreens to name a few. Touchscreens offer high accuracy and responsiveness.

Specification	TFT LCD	OLED Display	Touchscreen Display
Brightness	500 to 1500 nits	1000 to 1500 nits	500 to 1500 nits
Resolution	480 x 320 pixel	128 x 64 pixel	320 x 480 pixel
Viewing Angles	140 Degrees	160 Degrees	160 Degrees
Color Depth	16-bit to 24-bit	10-bit	24-bit
Contrast Ratio	1000:1 to 3000:1	1000000:1	1000:1 to 3000:1
Cost	~$38	~$55	~$70

Table 3.3.7.1: Screen Comparison.

When considering the best cost-effective screen needed for our hand gesture vending machine it was decided that the LCD screen was appropriate. The LCD screen offers sufficient brightness levels for indoor use, ensuring good visibility of product information and user interface elements. The LCD screen also provides good performance in terms of resolution, refresh rate, and color reproduction, meeting the requirements for displaying product information and graphics effectively. LCD screens are generally more affordable compared to OLED displays or even touchscreen displays, making them a cost-effective choice for projects with budget constraints. Overall LCD screens are a good balance between performance and cost-effectiveness, making them suitable for a hand gesture vending machine project where cost is a consideration.

3.3.8. Camera

The camera is basically the primary input for the hand gesture vending machine. There are several factors to be considered to ensure seamless compatibility and optimal performance. The resolution of the camera must be high enough for our AI-driven project. We don't believe it needs to be much higher than 1080p but it might be a good idea to find an HD camera at 720p. So our range of resolution was 720p to 1080p. Some cameras on smartphones today can have a resolution of 2160p and so finding a camera of that resolution is possible. However getting an image that defined might be counter productive at the cost of computational time and power.

A common camera connector uses the protocol Camera Serial Interface (CSI). CSI is a specification of the Mobile Industry Processor Interface (MIPI) Alliance [44]. The MIPI Alliance is a global organization that designs and promotes hardware and software interfaces that simplify the integration of components built into a device. The CSI specification is only available to MIPI Alliance members so companies that want to incorporate CSI or any other MIPI specifications would have to become members and pay membership dues. The camera we chose did use the CSI-2 protocol as that is a very common camera connector on single-board computers.

Raspberry Pi Camera Module 3 is a compact camera from Raspberry Pi [45]. The camera module uses a Sony IMX708 12 megapixel sensor with HDR also has phase detection autofocus. There are also Raspberry Pi Camera Module 3 variants that offer wide-angle variants and with or without infrared cut filter. The autofocus and wide-angle variants probably aren't going to be of use to us so just the standard option would be the one for us. The $25.00 price makes this a compelling option.

LI-IMX219 is a camera module made by Leopard Imaging [46]. It is a camera module designed for various embedded vision applications. The Sony IMX219 image sensor delivers a resolution of 8 megapixels. It uses the CSI-2 specification making it easy to integrate. With a price of $44.60, the price and resolution makes this a worse option compared to the Raspberry Pi Camera Module V2.

Arducam IMX477P Camera is a USB 3.0 option [47] Arducam offers a diverse range of USB-3.0 camera modules make for specific use cases like on-board ISP image enhancing, high resolution, ultra high resolution, ultra wide angle, HDR, ultra low light, and optical image stabilization. Most of the options are in the price range of $99 to $199. The Arducam IMX477P camera specializes in high resolution with the most listed video modes of our options. This option uses USB 3.0 rather than CSI-2 like our other options. USB 3.0 has a speed of up to 5 Gbit/s and the Arducam IMX477P Camera has a USB 2.0 mode with a resolution of 1280 x 720 at 10 fps but that is far too low for our applications.

Specification	**Leopard Imaging LI-IMX219 [45]**	**Raspberry Pi Camera Module 3 [46]**	**Arducam IMX477P Camera[47]**
Size (mm)	150 x 25 x 15	25 x 24 x 11.5	8.6 x 8.6 x 5.2
Still resolution	8.08 Megapixel	11.9 Megapixel	12.3 Megapixel
Video modes	3280 x 2464 21 fps	1920 x 1080 50fps 1280 x 720 100fps 854 x 480 120fps	4056 x 3040 10fps 3840 x 2160 20fps 2016 x 1520 50fps 1920 x 1080 60fps 1280 x 720 120fps
Sensor	Sony IMX219	Sony IMX708	Sony IMX477
Connection	CSI-2	CSI-2	USB 3.0
Cost	$44.60	$25.00	$149.99

Table 3.3.8.1: Comparison of Camera Properties.

We did use the Raspberry Pi Camera Module 3 due to its blend of affordability, versatility, and performance. It offers an exceptional resolution of 11.9 megapixels with a wide range of video modes for us to play around with to figure out the best performing video mode. The CSI-2 specification should allow seamless compatibility and ease of use.

3.4. Software Related Components

When developing the hand gesture vending machine, software plays a crucial role in enabling seamless operation, intuitive user interaction, and efficient management of machine functions. The firmware did control the microcontroller's operations, orchestrating tasks ranging from gesture recognition to motor control. By leveraging advanced machine learning algorithms our system is able to detect hand gestures captured by sensors or cameras, allowing for the user to interact with the machine. The different programming languages that we did use are dependent on the component. The software components involved with this project encompass firmware development, user interface design, communication protocols, and gesture recognition algorithms, working together to ensure efficient operation.

3.4.1. Primary Coding Language

The primary coding languages did vary for the hand gesture vending machine. There are many programming languages to choose from. The options for programming languages needed for this project were C, C++, Python, and Java. To start off, the programming language, C, is used for embedded systems and microcontroller programming. The programming language, C, can be used to write firmware for the microcontroller, including real-time control, peripheral management, and even low-level hardware interaction [48]. C++ is like C in the sense of it being used for microcontroller programming. C++ offers additional features such as Object-Oriented Programming (OOP), classes, and inheritance. Java is a versatile programming language, commonly being used for things such as web applications, server-side development, and enterprise software. Lastly, we have Python. Python is a powerful and versatile programming language known for its simplicity and readability. Python also has a large ecosystem of machine learning tools and libraries, making it ideal for training and deploying models. As we can see, each programming language has its strengths.

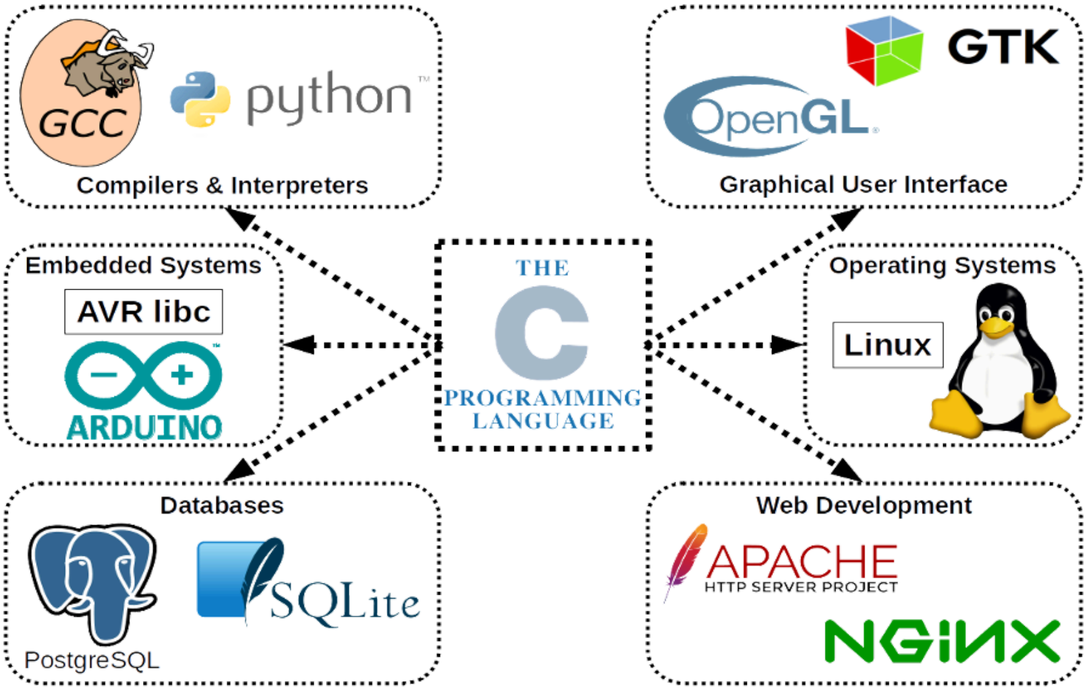


Figure 3.4.1.1: Various Software derived from C.

Each programming language additionally serves a specific purpose in the hand gesture vending machine, contributing to different aspects of its development and functionality. The programming languages, C, C++, Python, and Java are important software components for this project. There are a few things we're specifically looking for from these programming languages, and some things each language would need to best.

When working with C some things we're looking for are efficiency, hardware interaction, and real time responsiveness. C needs to excel in writing efficient, low-level code for the microcontroller firmware, ensure fast execution to meet real-time requirements, and optimize resource usage. Some things we're looking for from C++ is the modularity, abstraction, and scalability. C++ should facilitate the development of modular and reusable firmware components for the microcontroller; it did need to be supported by object-oriented programming (OOP) principles like encapsulation and inheritance to organize code into manageable units and promote code reuse. When dealing with C++ it should provide abstraction mechanisms to hide implementation details and expose high-level interfaces for interacting with hardware. The scalability revolved around C++ should enable the firmware to scale effectively as the project grows in complexity. Things that can be obtained from using the Java language is scalability, concurrency, and integration. Java would need to excel in building scalable backend services to support the vending machine system's growth and expansion.

Java would also need to support concurrent programming paradigms and provide concurrency control mechanisms for handling multiple concurrent requests and processing tasks asynchronously. Lastly, something that we would need Java to do the best for us is facilitating seamless integration with external systems, databases, and APIs. The final programming language is Python. Python did offer machine learning, flexibility, and interoperability. Python did need to be proficient in implementing machine learning algorithms for hand gesture recognition. Python should offer flexibility and expressiveness for prototyping, iterating, and refining machine learning models quickly.

C is a general purpose, procedural programming language developed in 1972 by Dennis Ritchie at Bell Laboratories of AT&T Labs, mainly developed as a system programming language to write the UNIX operating system [48]. It is mainly used for embedded systems, low level software programming, and system programming due to its portability, close to hardware abstraction, and efficiency. I did list a few specifications that the C language offers. C is a procedural paradigm language, making it a series of steps that forms a hierarchy of calls to its constituent procedures. The typing discipline is of type static weak. The memory management is manual due to having to do memory allocation. Another spec is the syntax. The syntax is simple and concise with minimal keywords and constructs. The portability with the C language is highly portable across different platforms and architectures. One of the last specs I would list is the performance. C is efficient and has fast execution speed, with low level access to hardware resources.

C++ is a general purpose, object-oriented programming language derived from C. The language was developed by Bjarne Stroustrup at Bell Labs in 1983 with the goal of adding object-oriented features to C while maintaining compatibility with C code. C++ is widely used around for developing applications, games, and large scale systems. I did follow up with a few specifications C++ offers. C++ is a multi-paradigm language which supports procedural, object-oriented, and generic programming. The typing discipline is of type static strong. The memory management is manual, allowing for manual and automatic memory management. The syntax could be complex by adding additional features like classes, inheritance, polymorphism, etc. In terms of performance C++ is comparable to C, with additional overhead for object-oriented features and abstractions.

Java is a high level, object-oriented programming language developed by Sun Microsystems in the year 1995, later acquired by Oracle Corporation [49]. It was designed to be platform independent, easy to use, and secure, with built in support for networked and distributed computing. When it comes to syntax Java is clean and readable syntax with similarities to C and C++, along with additional features for object-oriented programming. The portability is good when it comes to Java. Java is a write once and run anywhere (WORA), allowing compiled Java code to run on all platforms that support Java without the need for compilation. The performance with Java is generally slower than native languages like C and C++. Java is used for building enterprise-level applications, web servers, mobile apps, and web applications. The memory management related to Java is automatic.

Python is a high level, interpreted programming language that is known for its versatility, simplicity, and readability. Python was created by Guido van Rossum in 1991 with an emphasis on code readability, and its syntax allowing for programmers to express their concepts in fewer lines of code [50]. Python is widely used for data analysis, artificial intelligence, machine learning, automation, scripting, and web development. Python supports a multi-paradigm, supporting procedural, object-oriented, and functional programming. The memory management relationship with Python is automatic, there is a garbage collector for Python so that the user does not have to do manual garbage collection. Python is generally slower than programming languages such as C and C++. Python is highly portable, with interpreters available for various platforms and architectures.

Specification	**C [48]**	**C++ [48]**	**Python [49]**	**Java [50]**
Paradigm	Procedural	Multi-paradigm (OOP, procedural, generic)	Multi-paradigm (OOP, procedural, functional)	Object-oriented, concurrent
Typing	Static, weak	Static, strong	Dynamic, strong	Static, strong
Memory Management	Manual (explicit allocation and deallocation)	Manual and automatic (supports RAII, smart pointers)	Automatic (garbage collected)	Automatic (garbage collected)
Syntax	Simple, concise	Complex, additional features	Clean, readable	Clean, readable
Portability	Highly portable	Highly portable	Highly portable	Platform-independent (JVM)
Performance	Efficient, low-level access	Comparable to C, additional overhead for OOP	Generally slower, optimized implementations improve performance	Generally slower, optimized JIT compilation improves performance
Usage	System programming, embedded systems	Large-scale software systems, applications	Web development, data analysis, scientific computing, automation	Enterprise applications, web servers, mobile apps

Figure 3.4.1.2: Comparison of Coding Languages.

All in all there did not be one programming language used throughout the design and process of creating the hand gesture vending machine. This project did take multiple programming languages to get a successful hand gesture vending machine. In regards to what programming language we did use, it did depend on what exactly we are trying to accomplish. So for instance with any necessary embedded systems work being done we did programming in languages C or C++. These two languages are great for low level hardware programming. When dealing with machine learning we did use Python. Python is one of the easier languages to read and pick up on and it's one of the best languages to use in regards to machine learning.

3.4.2. Microcontroller Unit (MCU)

The component being researched for this particular part of the Hand Gesture Vending Machine is a microcontroller. A microcontroller is a compact Integrated Circuit (IC) that contains a processor core, memory, and various peripheral interfaces, all on a single chip [51]. It is designed to execute specific tasks within embedded systems, where it serves as the brain of the device. Microcontrollers are commonly used in a wide range of embedded systems, including consumer electronics, industrial automation, automotive systems, IoT devices, robotics, and more. They offer a cost-effective and efficient solution for controlling and interfacing with hardware components in embedded applications. When working with the microcontroller the processor core did act as the Central Processing Unit (CPU).

The CPU did execute stored instructions that are located in the memory and perform arithmetic operations, logic, and control operations to carry out tasks defined by the program running on the microcontroller. Microcontrollers normally have several different types of memory. There is Read-Only Memory (Flash or ROM), also known as program memory, which stores the program instructions that the microcontroller executes. This memory is non-volatile, meaning it retains its contents even when power is removed. There’s also Remote Access Memory (RAM), also known as data memory, which stores variables and data used by the program during runtime. Unlike ROM, this form of memory is volatile. Microcontrollers have several different peripheral interfaces for communicating with external devices and sensors. Common interfaces can be General-Purpose Input/Output (GPIO), Analog-to-Digital Converter (ADC), and even Serial Communication Interfaces (UART, SPI, I2C).

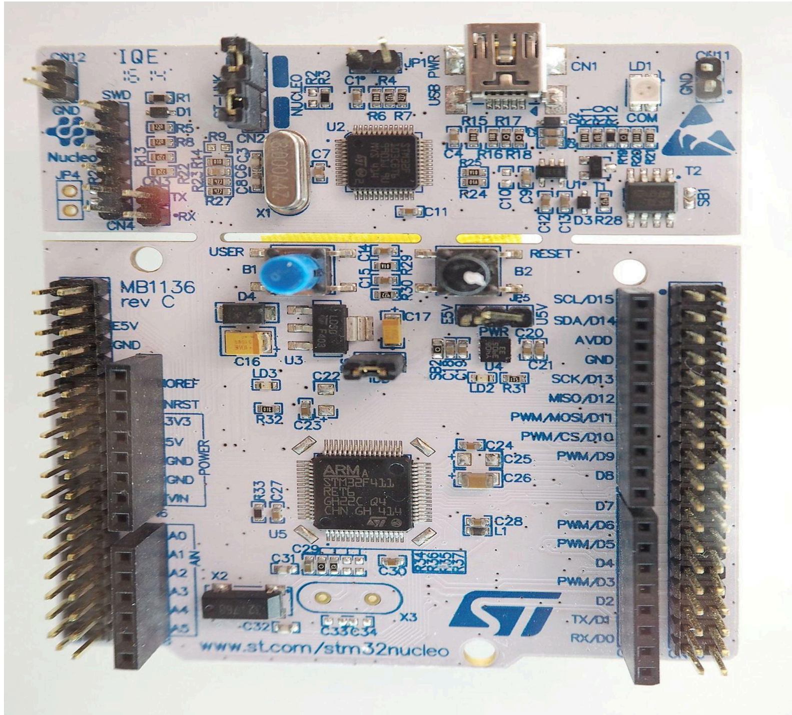


Figure 3.4.2.1: An STM 32 Microcontroller.

The microcontroller is an essential component towards the success of the hand gesture vending machine. This component is important for the project because it serves as the central processing unit that controls and coordinates the various components and functionalities of the system. An essential that's needed for this project and what exactly it did do for us is the gesture recognition processing. The microcontroller is responsible for processing the signals received from the gesture recognition system, analyzing hand gestures detected by the sensors or camera and interpreting them based on the programmed algorithm. Also, we did need a microcontroller to control the different peripherals.

The microcontroller did interface with peripheral devices such as the motors, sensors, displays, and communication modules dealing with our hand gesture vending machine, sending commands to the device to perform specific actions. Another thing the microcontroller did for us is execute the logic and decision-making algorithms programmed for the vending machine's operation. We did also need the microcontroller to manage the user interface of the vending machine, such as displaying menu options. Lastly, we would like for the microcontroller to handle power management and efficiency, allowing for optimization of power usage and managing energy consumption to ensure the vending machine operates efficiently.

In selecting a microcontroller for the hand gesture vending machine project, several key criteria need to be considered to ensure it meets the project requirements effectively. There are a few things we're looking for in this microcontroller and what it needs to be best. We did need gesture recognition performance, having the capability to efficiently process and interpret hand gestures. This function did require sufficient processing power and memory to implement gesture recognition algorithms accurately and reliably. We did require the best microcontroller in terms of peripheral interface compatibility, needing the microcontroller to support interfaces such as GPIO, UART, SPI, I^2C, ADC, and PWM.

In today's economy, one of the best things you can do is order electronics in your price range. So, we did need a cost-effective microcontroller, offering good value for its cost, performance, features, all while meeting the project requirements. Lastly, another task we did require from the microcontroller is real time responsiveness. The microcontroller should be capable of real time operation to ensure responsive interaction with the user and timely control of peripheral devices.

The first microcontroller we're looking at, the MSP430FR6989, is a very familiar component everyone in the group has knowledge on from the embedded systems lab. With the provided detailed documents from Texas Instruments (TI) this board is easier to implement. The MSP430FR6989 is a 16-Bit RISC Architecture up to 16-MHz Clock [52]. There is also a wide supply voltage range from 3.6 V down to 1.8 V. This microcontroller supports multiple serial communications such as UART, SPI, and I^2C. There are 83 GPIO pins with this microcontroller. The MSP430FR6989 provides 5 timers, up to 128KB of nonvolatile memory, and a 12-bit Analog to Digital Converter (ADC).

The second microcontroller we're looking at is the STM32F429ZI, or STM32 for short. The STM32 is the leading supplier of 32-bit ARM Cortex Microcontrollers [53]. Additionally, this microcontroller is a popular board in the industry and has been around for a while, so it does offer a range of helpful documentation and a community to help with certain tasks. The STM32 has a clock speed of up to 180-MHz, and the operational voltage for the microcontroller is a low voltage of 1.7 V to 3.6 V. Some of the serial communications that this board offers is SPI, I^2C, and USART. The difference between UART and USART is that USART can use both asynchronous and synchronous communication protocols. Finally, the STM32F429ZI offers 17 timers, 12-bit ADC, a RAM of 256KB and flash memory of 512K to 2056K.

The third microcontroller we're considering is the ESP32-WROOM-32. This microcontroller offers built-in Wi-Fi and Bluetooth connectivity [54]. There are clock speeds of up to 240 MHz. The different memory options this board offers are 384KB of ROM, and 512KB of SRAM. The operational voltage for the microcontroller is 3 V to 3.6 V. The ESP32-S3 also offers many different communication interfaces. This microcontroller has 4 SPI interfaces, 3 UART interfaces, and 2 I^2C interfaces. Lastly this microcontroller offers up to 45 GPIOs and a 12-bit analog to digital converter (ADC).

Last but not least in our considerations, the Arduino Due is a 32-bit ARM core microcontroller based on the Atmel SAM3X8E ARM Cortex-M3 CPU [55]. The Arduino

Due operates at a voltage of 3.3 V, so anything beyond that can damage the board. There is a clock speed of 84 MHz. The memory setup with the Arduino Due is a flash memory of 512KB and SRAM of 96KB. When working with this microcontroller there are 54 digital input/output pins, multiple peripherals such as UART, SPI, USART and I^{2}C. Also to our advantage we have multiple timers and a 12-bit analog to digital converter (ADC).

Factor	MSP430FR6989 [52]	Arduino Due [53]	ESP32-WROOM -32 [54]	STM32F429ZI [55]
Processing Power	MSP430FR5x, 16 MHz	ARM Cortex-M3, 84 MHz	Dual-core Xtensa LX6, 240 MHz	ARM Cortex-M4, up to 180 MHz
Memory	64 KB FRAM, 2 KB SRAM	512 KB Flash, 96 KB SRAM	320 KB SRAM	2 MB Flash, 256 KB SRAM
Peripherals	UART, SPI, I2C, ADC, DAC	USB, SPI, I2C, ADC, DAC, UART	USB, SPI, I2C, ADC, DAC, Ethernet, UART	USB, SPI, I2C, ADC, DAC, Ethernet, UART
Operating Voltage	1.8 V – 3.6 V	3.3 V	3.0 V - 3.6 V	1.7 V – 3.6 V
Development Tools	Code Composer Studio	Arduino IDE, Atmel Studio	Arduino IDE	STM32CubeIDE, Keil MDK
Size and Form Factor	VQFN-64, 14x14 mm	TQFP100, 14x14 mm	25.50 x 18.00	LQFP144, 20x20 mm
Operating Temperature Range	-40°C to +85°C	-40°C to +85°C	-40°C to +85°C	-40°C to +85°C
Compatibility	Energia, CCS, USB	Arduino Shields, USB	Arduino IDE	STM32CubeIDE, USB
Cost	$24.00	$48.40	$9.99	$24.47

Table 3.4.2.1: Comparison of Microcontroller Units.

Although the group has much more practical experience with the MSP430FR6989, we are leaning more towards the ESP32-WROOM microcontroller. The ESP32 offers high-performance ARM Cortex-M cores with ample processing power and memory, making them suitable for handling complex gesture recognition algorithms and controlling vending machine components. The ESP32 series is widely used in the embedded development community, so there is extensive documentation, tutorials, and community support available, which can be valuable resources during the development process. This microcontroller offers many peripheral interfaces, including ADCs, PWM outputs, and communication interfaces like UART, SPI, and I^2C, which are essential for interfacing with sensors, motors, and other peripherals used in the vending machine.

3.4.3. Single-Board Computer (SBC)

Single-board computers (sometimes called embedded computers) are complete computer systems that are built on a single circuit board. The necessary CPU, RAM, GPU, storage, I/O, and other components are all on one board. Microcontrollers often run on lightweight real-time operating systems while single-board computers typically run full-fledged operating systems, usually a flavor of Linux.

The single-board computer must possess strong AI capabilities while the microcontroller did handle real-time processing of sensor data. Utilizing the microcontroller's capability of communication interfaces like UART, SPI, and I^2C the microcontroller did communicate to the single-board computer. All of the boards we considered have these protocols. We did use a camera so having a board with a built-in 15-pin CSI connector is a priority.

The BeagleBone AI-64 is a single-board computer developed by the BeagleBoard.org Foundation [56]. The BeagleBoard.org Foundation is a non-profit organization dedicated to the development and promotion of open-source hardware and software for embedded computing and DIY electronics projects. The board features two sets of 46-pin GPIO headers, totaling an abundant ninety-six general purpose input and output. That is more than we did need. Since the board does not have a dedicated 15-pin camera connection, we did need to buy and use a 15 pin to 22 pin camera flex cable. The BeagleBone AI-64 has AI capability but has a few major drawbacks. At idle the board consumes five watts so the device is quite power hungry [57]. Then the cost of the board is at $190.00, making it the most expensive option.

The Odroid-M1 is a single-board computer created by Hardkernel Company, Ltd., located in South Korea [58]. The board features a 40-pin header leaving us with plenty of input and output options. It does have a 2-lane MIPI CSI camera port on board. Our other options use a MicroSD slot for storage but the Odroid-M1 is unique in how it has SATA3 and SATA Power on board. That could allow the use of a full-sized hard disk drive (HDD) or a full-sized solid-state drive (SSD). However, MicroSD cards in large capacity and speed have become very affordable so we did not have to worry about additional points of failure by putting in an HDD or an SSD. It is priced at $90.00 makes it a worse option compared to the Raspberry Pi 4 Model B.

The Raspberry Pi 4 Model B is from the fourth generation of the Raspberry Pi single-board computer series. Located in the United Kingdom, the Raspberry Pi Foundation developed the Raspberry Pi 4 Model B along with many other varieties of single board computers and microcontrollers [59]. For general purpose input and output pins, the board has a 28-pin header, the least amount in our comparison. It does have a 2-lane MIPI CSI camera port on board. A previous generation would suffice but older models that are not produced anymore are around the same price point, around $55.00. The latest model, the Raspberry Pi 5 is advertised to offer "2-3x speed of the previous generation" but at a higher price of $80.00 so the Raspberry Pi 4 Model B offers a healthy balance of features and price.

The Nvidia Jetson Nano is a developer kit designed by Nvidia for AI and robotics applications. Nvidia is a company widely known for producing high-performance graphical processing units (GPUs) for gaming PCs and consoles. They have also made many advances in AI with their own parallel computing platform, CUDA [60]. The Nvidia Jetson Nano has CUDA 10.2 and is able to use it to accelerate applications. However, due to the COVID-19 pandemic, a global chip shortage caused the Nvidia Jetson Nano to be priced at over five hundred dollars on the secondary market. This put it way over its manufacturer's suggested retail price (MSRP), but it has mostly subsided back to its MSRP and become affordable again. The Nvidia Jetson Nano offers a powerful single-board computer with great proprietary components and AI and capabilities at the cost of $99.00, assuming we can get it from Nvidia directly.

This was one of the more difficult decisions to make. While it was easy to eliminate the Raspberry Pi 4 due to its GPU being extremely limited (relative to the other options), the rest of the boards were much more evenly matched. This led to us getting nitpicky over what we thought was good or bad for our AI development kit. The two most critical factors were the camera connectors and the inclusion of a heatsink. In the case of the former, the need for a troublesome flex cable eliminated the BeagleBone AI-64 from our pool of options. In the case of the latter, the lack of a heatsink was what led to us excluding the Odroid-M1.

Of the above options, the Nvidia Jetson Nano is the only option to have a built-in heatsink to help offset overheating, a powerful GPU unit to quickly manage object detection, and a reasonable camera connector to allow for more camera options. If we chose to use another board, we would either have to consider adding our own heatsink or greatly increase the planned fans, simply because the single-board computer would heat up much too fast to be used for long.

Factor	BeagleBone AI-64 [54, 55]	Odroid-M1 [58]	Raspberry Pi 4 Model B [59]	Nvidia Jetson Nano [60]
GPU	3D GPU PowerVR Rogue 8XE GE8430	Mali-G52 MP2	Integrated	128-core NVIDIA Maxwell architecture GPU
CPU	Dual 64-bit ARM Cortex-A72	Rockchip RK3568	Quad core 64-bit ARM Cortex-A72	Quad-core ARM Cortex-A57 MPCore processor
Maximum CPU Clock Speed	2.0GHz	1.992GHz	1.5 GHz	2.0 GHz
Memory	4GB LPDDR4	8GB LPDDR4	4GB LPDDR4	4GB LPDDR4
GPIO pins	Two 46-pin headers	40-pin header	28-pin header	40-pin header
Camera Connectors	15 pin to 22 pin camera flex cable required	1x 15-pin 2-lane MIPI CSI	1x 15-pin 2-lane MIPI CSI	1x 15-pin 2-lane MIPI CSI-2
Power Consumption	5W-15W	1.25W-4.44W	2.7W-6.4W	5W-10W
DC Input	5V	12V 2A	5V 3A	5V 4A
Board Size	4” x 3.1”	90mm x 122mm	85mm x 56mm	70mm x 45mm
Heatsink?	No	No	No	Yes
Cost	$190.00	$90.00	$55.00	$99.00 - $165.00

Table 3.4.3.1: Comparison of Single-board Computers.

3.5. Machine Learning

Machine learning is a subset of artificial intelligence, and its objective is to recognize patterns using math, more specifically, statistical analysis to make informed decisions [61]. Furthermore, the objective of artificial intelligence, as a whole, is to produce a machine, or machines, that can reproduce the production of a human mind. Considering that this is artificial intelligence and not an actual human brain, a parallel between the two can be drawn; the more data and information that is fed into the body, the more accurate the output prediction did [62]. This is due, not only, to processing the data correctly and learning from that, but also from making mistakes in the processes and learning the seemingly insignificant differences between right and wrong. This process is defined as a self-learning algorithm, where the machine gains knowledge from the processed datasets to predict the outcomes of future related data. When considering machine learning the process is made up of multiple different phases [63].

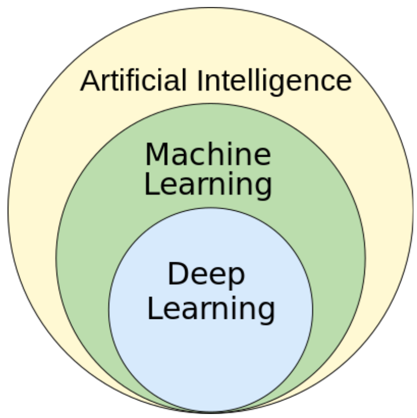


Figure 3.5.1: This illustration shows the connection between Artificial Intelligence, Machine Learning, and Deep Learning.

There is the Problem Declaration phase; in this phase, the problem that needs solving did be identified, the input/output data did be defined, and the objective of the machine learning project did be defined. In the case of this project, the problems that did be solved, through the point of view of the Vending Machine, are the hand gestures for the digits one, two, three, four, and five, as well as, the hand gestures for confirm and decline. The input and output data, for processing, consisted of hands, both left and right, that have one, two, three, four, and five fingers up, along with, a thumb up and thumb down to signify confirm and decline. The objective would be to learn the hand signals to the point where there are very minimal inaccuracies.

There is the Data Collection phase; in this phase, good and reliable input and output data is needed. More specific to the vending machine, the input data did have multiple different angles for the same digits and gestures; there did be top views and side views from both sides. There is the Data Preparation phase; in this phase, the data from the previous phase is split into the training data, which is the data used to train the artificial intelligence, and the testing data, which is used to test the trained machine.

There is the Model Selection phase; in this phase, there are selections for the model, which can be seen as the perspective for which the patterns, in the data, are recognized, and algorithms, which is how the model goes about receiving its input and output. There is the Training phase; in this phase, the machine learning model is passed the data that was reserved for training and as it goes through the data, the model gets better and more equipped to answer the problem that was defined in the problem declaration phase. In the case of the vending machine, as more data gets processed, it becomes more equipped to decide what hand signal is a thumb up or down and what hand signal is one, two, three, four, or five.

There is the Model Evaluation phase; in this phase, the model is reviewed for accuracies, inaccuracies, and speed. It is also reviewed for the case that the model is the incorrect model for this specific set of data. Finally, there is the Machine Learning Deployment phase; in this phase, all the adjustments have been made, the model has been trained and tested sufficiently, and it is ready for deployment.

A model, within the context of artificial intelligence, can be described as a set of equations whose output describes a real-world situation and possibly predicts future behavior of the given problem [64]. Models, in the realm of artificial intelligence, are the resultant system that remains, after the training and testing data is processed. This is important because the model existing to learn and predict is conceptually the basis of AI and machine learning. These models can be trained using Supervised Learning, Unsupervised Learning, Reinforcement Learning, and furthermore Deep Learning and Neural Networks [65].

For the hands-free vending machine, there are specific, mandatory attributes needed in a model to make the vending machine run as it should. The model needs to be able to take images and classify whether the sensor sees a thumb up, or thumb down and recognize if the sensor is seeing a digit one, two, three, four, or five. A training method that specializes in some sort of image classification is needed. The model does not have any time restriction on it as far as training the model. Therefore, the training phase of the model does not have to be very fast.

Supervised learning is one of the ways that machine learning models are trained and what makes supervised learning special is that it is used to train machines using labeled data, meaning that the machine already knows of the correct output of the algorithm before the machine fully processes the, also labeled, input data [66]; this provides a direct feedback, to the machine, between output and input. For the training strategy of supervised learning, the machine learning aspect stems more so from the algorithm investigating how and why the specific block of labeled input data results in the specific block of labeled output data.

The two types of problems that supervised learning is specifically fit to handle are classification problems and regression problems. A classification problem is a problem where the machine’s goal is to be able to predict what class, or group, a piece of data belongs to; an example would be a machine being able to predict if an apple is a fruit or vegetable given the apple's specifications such as size, acidity, and sweetness. A regression problem is a problem where the machine's goal is to be able to "understand the relationship between dependent and independent variables" [66]. An example of this would be a machine finding the relationship between education and the net worth of a person.

The aim of the supervised learning algorithm is primarily to forecast some sort of outcome; some of its real world uses include predicting if a phone call is a robot or a human, predicting stock values over time, and performing risk evaluations. Setting the correct parameters and picking the best algorithms for each specific problem is very important; each algorithm is very good at specific jobs and problems, however when an algorithm is chosen, but is not the most optimal, it can lead to a very inaccurate model [62]. Popular algorithms for supervised learning include linear regression, logistic regression, k nearest neighbor, and random forest. These all have their own specific strengths and weaknesses; the algorithm picked, in the case of supervised learning, is based, almost completely, on the size of the data sample, the amount of time readily available to train the model, amount of memory available, and accuracy requirement of the project [62].

An advantage for supervised learning is the highly accurate result of the model and the ability to predict new data, from the training. This comes from the labeled input and output data from the training phase. A major disadvantage also stems from the labeled input and output data in the training phase; the large amount of labeled data the model has to process can result in very slow processing of the input and output data.

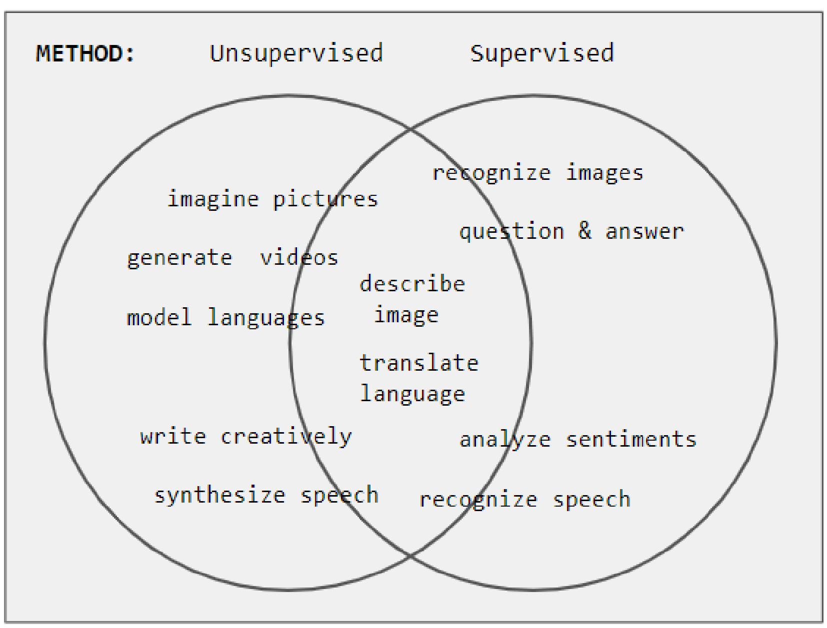


Figure 3.5.2: This image shows the similarities and differences between Supervised and Unsupervised Learning.

Unsupervised learning is another one of the ways that machine learning models are trained and what makes unsupervised learning unique, and differ from supervised learning, is that unsupervised learning is used to train machines without supervision and with using unlabeled data, meaning that the machine does not know of the correct output of the algorithm before the machine fully processes the, also unlabeled, input data [66]. A model that uses unsupervised learning tries to label the output data based on the features of the input. Furthermore, the way unsupervised learning models learn is by finding patterns between the input and output; the model attempts to label the output data based on features of the input.

The three types of problems that unsupervised learning is specifically fit to handle are clustering problems, association problems, and dimensionality reduction problems [66]. A clustering problem is a problem where data points are grouped together by similarities; an example would be a machine grouping together fruits by color. An association problem is a problem where the machine searches for relationships between data points. An example of this is market basket analysis. A dimensionality reduction problem is a problem where the machine's goal is to be able to reduce the number of variables while preserving the highest amount of information possible. An example of this would be a machine removing noise from a picture or video.

The aim of the unsupervised learning algorithm is primarily to find outliers in the dataset or to organize the datasets into groups [67]; a real-world use of unsupervised learning includes finding outliers in a dataset. Popular algorithms for unsupervised learning include k-means clustering, DBSCAN, and principal component analysis. These all have their own specific strengths and weaknesses; the algorithm picked, in the case of unsupervised learning, is based, almost completely, on the size of the data sample, the amount of time readily available to train the model, amount of memory available, and accuracy requirement of the project.

An advantage for unsupervised learning is how much less manual data labeling and hands on data preparation, in general, is needed. Whereas for supervised learning, there is a need for much more user supervision. A major disadvantage stems from how much less supervision there is when in comparison to supervised learning; in unsupervised learning, the little to no supervision can lead to inaccurate results unless there is validation by a person proctoring the training phase [66].

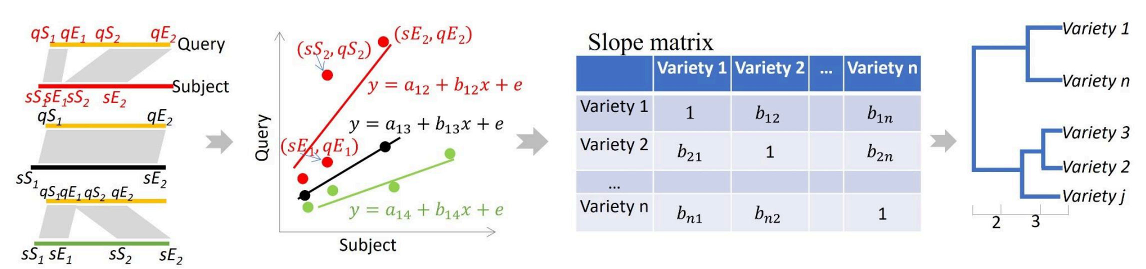


Figure 3.5.3: This illustration shows how Clustering works for Unsupervised Learning.

Reinforcement learning is another one of the ways that machine learning models are trained and what makes reinforcement learning unique, and differ from other training methods, is that reinforcement learning is used to train machines using an agent and an environment [68]. An agent is the learning, decision-making body being trained through trial and error and the environment is the body to which the agent interacts through simulated scenarios. The agent trains to earn as many rewards as possible and rewards are earned as the agent gets decisions correct within the environment.

On the contrary, the agent is not rewarded, but punished, every time the agent is incorrect on a decision. A model that uses reinforcement learning uses an agent and environment to iterate through multiple possible solutions to earn the most possible rewards with little to no supervision necessary. The primary type of problem that reinforcement learning is specifically fit to handle is a reward-based trial and error problem. Some examples of this would be a maze learning machine, a puzzle learning machine, and a game building machine.

The aim of the reinforcement learning algorithm is primarily for the agent to encourage a certain behavior and earn as many rewards as possible; a real-world use of reinforcement learning includes predictive text. Reinforcement learning algorithms fall under one of two different algorithms, model-based and model-free. It is said that, “if the agent can predict the reward for some action before actually performing it thereby planning what it should do, the algorithm is model-based. While, if it actually needs to carry out the action to see what happens and learn from it, it is model-free” [69].

The model-based algorithm is better for environments that stay the same and an example of a model-based algorithm would be a maze; the overall shape of the maze stays the same and the only thing that changes about the maze is where, in the maze, the agent currently resides. The model-free algorithm is better for environments that are more dynamic, and an example of a model-free algorithm would be a self-driving car. Popular algorithms for reinforcement learning include q-learning, SARSA, Monte Carlo, and Deep Q Learning. These all have their own specific strengths and weaknesses; the algorithm picked, in the case of reinforcement learning, is based on whether the environment is static or dynamic. Some advantages for reinforcement learning are how little hands-on data preparation is needed and the simplicity of selecting models. A disadvantage is how rigid and narrow the focus of the training method can be.

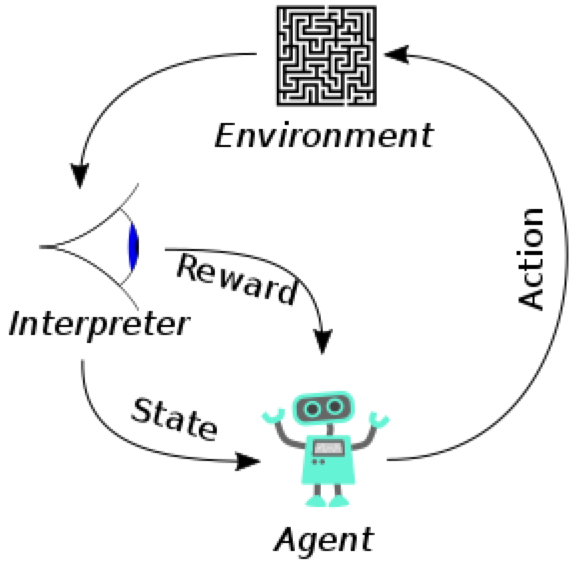


Figure 3.5.4: This illustration shows the process of the agent and environment for Reinforcement Learning.

For the hands-free vending machine, the stated mandatory attribute needed, in the model, to make the vending machine run as it should was specifically image recognition specialization; the model needs to be able to take images and classify whether the sensor sees a thumb up, or thumb down and recognize if the sensor is seeing a digit one, two, three, four, or five. The training method that fits the best, of the three stated, is the supervised learning method. This is because one of the problems supervised learning specializes in is the classification type of problem. Outside of this, supervised learning fits this project better than unsupervised learning because although unsupervised learning specializes in clustering, which could be useful for clustering different hand signals together, unsupervised learning lacks the human intervention aspect when there is an outlier in the set. The supervised learning also fits this project better than reinforcement learning because reinforcement learning severely lacks any attribute a model could possibly need for this project.

Deep learning is a subset of machine learning, and therefore a subset of artificial intelligence, and its objective is to recognize patterns using math, more specifically, statistical analysis to make informed decisions; by performing statistical analysis to recognize patterns, deep learning has the capability, and goal, of replicating certain, focused tasks as if it was a human brain making the decisions. So much so that deep learning refers to the depth of node layers in the neural networks [70]. Neural networks are a collection of node layers that make up a system of communication like one of the neural connections of the brain.

This system of communication consists of an input layer, a hidden layer, and an output layer. The input layer takes the input parameters, processes them, and passes its output to the hidden layer. The hidden layer takes the input from the input layer and further anatomizes the data that was received and learns the intricacies of each factor in the received data. There can be multiple layers within the hidden layer and this output from the hidden layer is passed to the output layer, where the output layer makes the final predictions for the input's question. Each node, in each layer, is its own linear regression model and the weights of each node determines how much of an impact that specific node has on the output [71].

Machine learning and deep learning are very similar in many ways. Both attempt to replicate a human mind using some sort of underlying statistical algorithm. Both require a very large sample of data to yield truly consistent and accurate results. However, a lot of the differences between the two lie within the margins. When examining deep learning, it can be seen that a deep learning model would need more data; deep learning is much more fit to handle unstructured data and more bandwidth is necessary to handle data that is not structured. This is a reason why deep learning is used in areas such as automation, image-based gesture recognition, and self-driving cars. Whereas with machine learning, less data is generally needed for the same reason; the data is structured and therefore needs less information.

For the hands-free vending machine, the stated mandatory attribute needed, in the model, to make the vending machine run as it should was specifically image recognition specialization; the model needs to be able to take images and classify whether the sensor sees a thumb up, or thumb down and recognize if the sensor is seeing a digit one, two, three, four, or five. It was seen that the supervised learning method, using machine learning, would be the best fit for this, but, when further examined, it can be deduced that deep learning is much more fit for this project. This is primarily due to machine learning being severely inferior in processing unstructured data as well as deep learning being vastly more fit to handle image-based gesture recognition specifically.

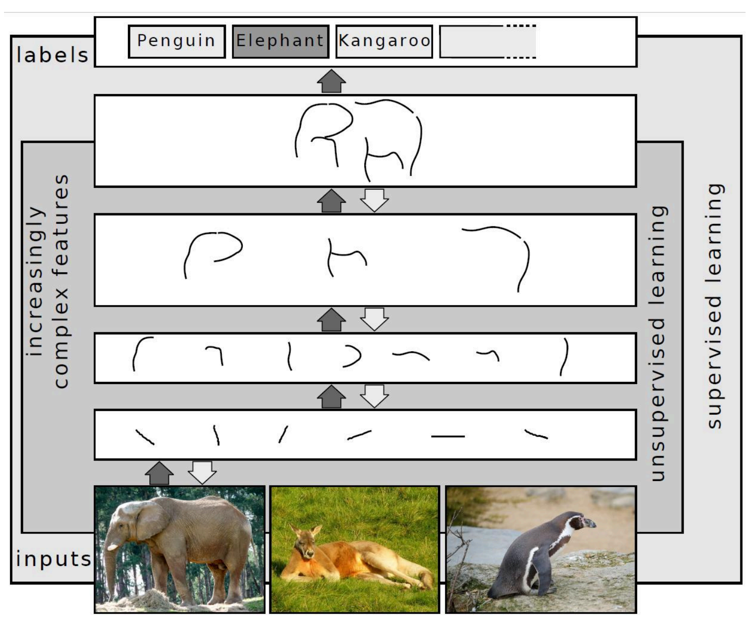


Figure 3.5.5: This figure shows the process of Deep Learning when it is processing images.

Although deep learning is its own entity, to make a model, it did still follow some sort of algorithm, as machine learning models do. An algorithm is described as “a process or set of rules to be followed in calculations or other problem-solving operations, especially by a computer” [72]. Neural networks and the neural network architecture are very important to deep learning algorithms. Furthermore, the differences in the way the two algorithms are shaped, make the difference in how they perform and react to data. The primary algorithms in the space of deep learning include Convolutional Neural Networks and Recurrent Neural Networks.

The convolutional neural networks algorithm is one of the ways that deep learning models are produced; models are the output of algorithms and these models “run on data and represent what was learned by the algorithm” [73]. As mentioned in a previous paragraph, neural networks are a collection of node layers that make up a system of communication like one of the neural connections of the brain, with the node layers including the input layer, the hidden layer, and the output layer. The convolutional neural networks are made up of a feedforward neural network, meaning that this type of neural network does not include a loop and it only goes one way.

The other two important features the CNN includes are the filters and the pooling layers. Filters are “small matrices of weights tuned to detect certain features in an image, such as colors, edges or textures” [74]. Multiple filters are used for each specific piece of data the CNN is examining and each filter the CNN uses helps it understand the piece of data more than it did before the filter. The pooling layers provide a way of compression of the data at the end of the CNN; the data can become heavy with all the different filters. The problems that the convolutional neural network solves are problems that include photos and videos; the CNN is used for tasks like facial recognition and image classification. Something that the CNN can do, after being trained, is input a picture, and predict the class it belongs to.

An advantage the convolutional neural network has is the ability to take images and classify them; this could possibly be very advantageous especially for a project that relies on picking up on what hand signs are being displayed. Another advantage stems from the ability of this algorithm to stay relatively light from a data aspect; the input and output data are fixed, and this makes the neural network faster. A disadvantage stems from the fact that the CNN is not greatly fit to handle any type of problem that includes predictive text tasks.

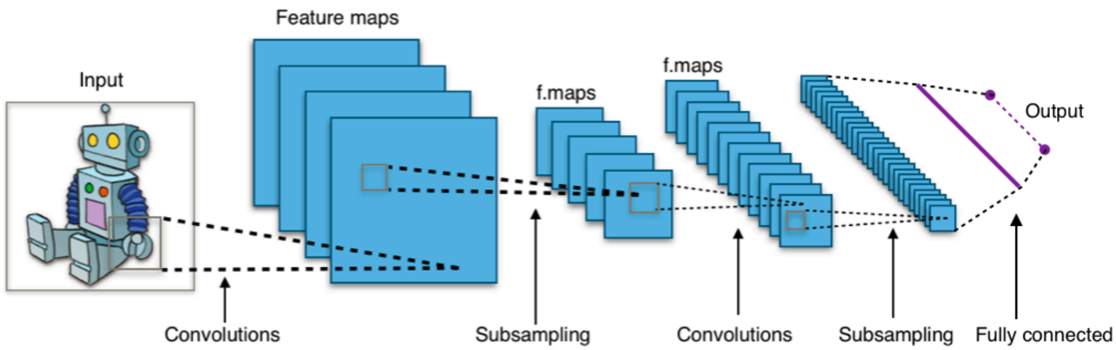


Figure 3.5.6: This image shows the full process of the Convolutional Neural Network.

The recurrent neural networks algorithm is one of the ways that deep learning models are produced; models are the output of algorithms. The recurrent neural networks are not made up of a feedforward neural network; there is a feedback loop and this loop feeds information back into the algorithm. This loop in the algorithm is used to detect patterns in the videos or text; each loop the algorithm learns. The algorithm learns and retains information due to recurrent cells.

Recurrent cells are "the units containing the feedback loops" and they "enable the network to retain information over time" [74]; recurrent cells use new data and prior steps to update the analysis of the input. The problems that the recurrent neural network solves are problems that include text or sequential data in general. The RNN is used for tasks like speech analysis. An advantage for the recurrent neural network is the ability to analyze sequential data such as text and speech recognition [75]. A disadvantage stems from the fact that the RNN has no sort of backend compression of the data and therefore the data can end up getting bigger and staying bigger.

For the hands-free vending machine, the stated mandatory attribute needed, in the model, to make the vending machine run as it should was specifically image recognition specialization; the model needs to be able to take images and classify whether the sensor sees a thumb up, or thumb down and recognize if the sensor is seeing a digit one, two, three, four, or five. The learning algorithm that fits the best, of the two stated, is the convolutional neural network learning algorithm. This is because one of the problems convolutional neural networks specializes in is the image-recognition type of problem. Furthermore, the CNN is considered supervised learning under the classification of neural networks and this meets a requirement stated earlier. Outside of this, the convolutional neural networks algorithm fits this project better than the recurrent neural networks algorithm because although the RNN learning specializes in video, which could be useful for the video of the different hand signals being used, RNN lacks the classification aspect that is needed for selection.

The hands-free vending machine's machine learning aspect required a couple of things: being able to classify different hand gestures, being very accurate when it comes to hand gestures, having quick outputs, and being able to handle unstructured data. A sector of machine learning met the criteria for classification, however it struggled with unstructured data. RNN met the criteria for unstructured data, but it did not specialize in image-based classification. Deep learning, supervised learning, and convolutional neural networks met all the different requirements set. Deep learning handles data that is unstructured very well. The supervised learning handles classification very well. Finally, the convolutional neural networks handles image-based classification very well, is very fast, and is very accurate.

3.6. Additional Components of Interest

3.6.1. Shell Fabrication

The first material to be examined is the material making up the shell, or outer casing, of the Hand Gesture Vending Machine. The outer shell serves as a strong foundation and structure for the snacks and technology to be housed in. This strong casing that was used for the hand gesture vending machine was referred to as shell fabrication. These materials play a big role in determining the durability and cost of the vending machine's outer shell and furthermore the vending machine.

One of the most common outer shell materials of vending machines is steel. This is because steel is cheap, easy to maintain, and easy to clean. However, with some vending machine companies looking to appear more modern, they have looked to different materials to keep a lot of the cheapness and cleanliness but increase the beauty. The decision shell fabrication material of the hand gesture vending machine is critical for several reasons; it directly impacts the durability, functionality, and safety of the vending machine. Obviously, the durability of the machine is directly affected by bad or brittle material and this would cause the machine to break down.

The less obvious effect of the shell's material is any sort of rusting or corrosion; iron is considered a strong metal; however it rusts, and this causes a lot of companies to steer away from using iron. Therefore, for this project, it is very important to use a material for the shell that has a lot of durable attributes. Lastly, we have the cost and affordability. Cost obviously plays a big role in the selection of materials for the outer casing; the casing is very important for creating a strong foundation and a strong base to house all the elements that rest inside like the glass panel or the motors. However, the casing is not important enough to spend too much money on it. The choice for material comes down to weight, cost, safety, and durability.

Plastic is the first option for the outer shell of the vending machine. The type of plastic, more specifically, would be Acrylonitrile Butadiene Styrene (ABS) plastic. ABS plastic has many uses in many industries from kids toys to plumbing pipes [76]. This versatility is because of how strong and light the plastic is. With the plastic being strong, light, and corrosive resistant [77]; this makes it a great choice for the outer shell for the vending machine. It is also cheap; it ranges from $1.70 to $2 per pound [78].The material is also cost-effective, offering a balance between performance and cost while providing durability and safety at a reasonable price point.

The second option, for the outer shell of the vending machine, is steel. Steel has many uses in many industries, hence why it is the most used metal in the world. This wide use is because of how strong, durable, and impact resistant the metal is. With steel being strong, cheap, and corrosive resistant, steel would be a great choice for the outer shell for the vending machine, if it wasn't for the density of it [79]. With only costing \$0.20-\$0.70 per pound [80], it is a very cost efficient metal and with it being as strong as it is, it would be great, but the overwhelming density is too much to look past.

Aluminum is the last option for the outer shell of the vending machine. Aluminum is also a very versatile metal ranging from the cell phone business to soda cans; it can be deduced that both industries use this metal, at least in part, due to its strength to weight ratio [81]. Since aluminum is light, strong, and corrosion resistant, it makes the metal versatile and usable in any industry. Furthermore, with it being cheap and ranging between \$0.13-\$1.53 a pound [82], aluminum becomes a very strong suitor for the vending machine. The material is also cost-effective, offering a balance between performance and cost while providing durability and safety at a reasonable price point.

The best option for the hand gesture vending machine would likely be aluminum. Aluminum offers a good balance between durability, performance, and affordability, making this material a cost-effective choice for the outer shell in various applications, including vending machines. Aluminum is the most cost efficient; it is not the cheapest, but the best value comes from it. Since aluminum is lightweight it allows for the ease of installation, and handling. With the ease of fabrication, it is able to easily be cut, drilled, and shaped to custom specifications, allowing for flexibility in design.

3.6.2. Glass And Glass Alternatives

Another thing to be considered is the material utilized for the display panel of the hand gesture vending machine, with an accentuation on glass or glass alternatives. The display panel did serve as the interface between the client and the vending machine, showing item data. Transparent or translucent materials that are utilized to cover and ensure the display panel while allowing for visibility of the content displayed underneath can be referred as glass or glass alternatives.

With these materials playing a significant role in determining the durability, safety, clarity, and aesthetics of the vending machines display. Normally glass is the more popular choice for display panels due to its transparency, scratch resistance, and ability to maintain optical clarity over time. However, with recent advancements in materials science this has led to the development of various alternatives to glass, offering comparable or enhanced properties while addressing specific needs or preferences.

The decision of glass or glass alternatives for the display panel of the hand gesture vending machine is critical for several reasons, it directly impacts the durability, functionality, and user experience of the vending machine. There are many important reasons why this component is important for the project. One of the reasons is the visibility and readability. The display panel did serve as the interface through which users interact with the hand gesture vending machine. It is imperative that the chosen material provides excellent visibility and readability of the displayed content, including pricing, and product information.

Another factor to consider is the protection and durability the glass or glass alternatives has to offer. The display panel is exposed to various environmental factors, including physical impact, moisture, UV radiation, and scratches. The material that we decide to go with for the vending machine must offer sufficient protection and durability to withstand these conditions. Safety and security play a role in the decision on glass or glass alternatives materials. For instance, in locations such as a retail environment or a park, safety is a primary concern. The display panel should be constructed from materials that are able to withstand an event of breakage or damage, prioritizing safety.

Lastly, we have the cost and affordability. Cost considerations play a crucial role in the selection of materials for the display panel. When it is all said and done the choice of glass or glass alternatives for the display panel of the hand gesture vending machine directly impacts its reliability, cost-effectiveness, usability, and safety.

Tempered glass is the first option for glass or glass alternatives to choose for the hand gesture vending machine. Tempered glass is a type of safety glass that undergoes a thermal tempering process, increasing its strength and resistance to impact [83]. By heating the glass to extreme temperatures and then rapidly cooling via blasts of air which causes the outside layers of the glass to solidify before the inside layers do. When tempered glass is struck, instead of shattering into dangerous pieces, it shatters into small, dull cubes making it safer for those nearby. Tempered glass is stronger than regular glass, with enhanced resistance to breakage and shattering. Tempered glass is scratch resistant due to the surface of the glass hardening during the tempering process. The material is also cost-effective, offering a balance between performance and cost while providing durability and safety at a reasonable price point.

Figure 3.6.2.1: Automobile windshield with "spider web" cracking typical of laminated safety glass.

Next glass or glass alternative option is acrylic glass. Acrylic is a transparent thermoplastic known for its lightweight, shatter-resistant properties, and optical clarity [84]. The acrylic material is significantly lighter than glass, making it easier to handle and install in various applications. Generally acrylic glass is more affordable than tempered glass, offering a cost-effective alternative for applications where cost and weight are considerations. Acrylic offers high impact resistance, making it suitable for applications where durability is essential. With acrylic glass being easily fabricated it allows for acrylic to be cut, drilled, and shaped to custom specifications, allowing for flexibility in design and installation. Another great specification acrylic glass offers is UV resistance. Some acrylic glasses offer UV resistance to prevent yellowing or degradation when exposed to sunlight.

Laminated glass consists of two or more layers of glass permanently bonded together with one or more polymer interlayers using heat and pressure [85]. Laminated glass can adapt to suit many architectural projects, those that require demanding performance in terms of increased security, safety, noise reduction, and even UV protection. The safety features consist of the laminated glass being able to retain fragments together in the event of breakage, reducing the risk of injury from sharp edges or flying debris. Acoustic insulating glass adds noise reduction as a benefit, helping to reduce noise levels. When considering the cost considerations, laminated glass may be more expensive compared to tempered glass or acrylic due to its additional safety features and manufacturing process. Lastly laminated glass is offered in various thicknesses to meet specific safety and performance requirements.

Glass-Ceramic such as Gorilla Glass combine the strength and durability of glass with the scratch resistance and optical clarity of ceramics. The materials used have the same chemical compositions as glasses but differ from them in that they are typically 95-98%

crystalline by volume, with only a small percentage vitreous [86]. This material is typically characterized by high strength, high impact resistance, and lightweight. Glass-ceramic materials offer exceptional strength and toughness, making them highly resistant to impact and scratches. Glass-ceramic can be produced in lightweight and thin forms, making them suitable for slim and sleek device designs. Although this material offers superior durability and scratch resistance, it might come at a higher cost compared to other materials like acrylic or tempered glass.

Finally, the last glass or glass alternative option we have is polycarbonate. Polycarbonate is a transparent thermoplastic sheet material often chosen for its advantage with impact resistance, insulation, flexibility, and flame retardance [87]. When it comes to other glass polycarbonate is virtually unbreakable, it is 250 times more resistant to impact. Despite its high impact resistance, it is lightweight, 30 times stronger than acrylic, and over 200 times stronger, and six times lighter than glass. Polycarbonate is exceptionally good in terms of durability, but with cost considerations in mind polycarbonate may be more expensive than acrylic and less cost-effective for some applications. Polycarbonate can provide softer lighting in building situations where it is preferred over direct sunlight, making it UV protected.

The best option for the hand gesture vending machine would likely be acrylic. Acrylic offers a good balance between durability, performance, and affordability, making this material a cost-effective choice for display panels in various applications, including vending machines. Acrylic is normally more budget friendly when compared to other materials such as tempered glass or glass-ceramic. Since acrylic is lightweight it allows for the ease of installation, and handling. With the ease of fabrication, it is able to easily be cut, drilled, and shaped to custom specifications, allowing for flexibility in design.

Specification	**Tempered Glass [83]**	**Acrylic [84]**	**Safety Laminated Glass [85]**	**Glass-Ceramic (Gorilla Glass) [86]**	**Polycarbonate [87]**
Impact Resistance	4	3	5	5	4
Scratch Resistance	4	2	3	5	3
Optical Clarity	5	4	5	5	3
Weight	½" glass weighs 6.5 lbs/ft	½" glass weighs 3.04 lbs/ft	½" glass weighs 6.36 lbs/ft	4.88 kg/m^2 at 2.00 mm	1.5 lbs/ft
UV Resistance	Blocks 60%-70% UVB rays	Blocks up to 90% of UV rays	Can block more than 99% UV rays	~	Blocks almost the entire relevant UV spectrum
Ease of Fabrication	2	5	3	3	5
Safety Features	Shatters into small, blunt fragments	None	Retains glass fragments together in event of breakage	Exceptional strength and toughness	Virtually unbreakable, may flex on impact
Cost	3	5	3	3	3

Table 3.6.2.1: Comparison of Glass and Glass alternatives, with 5 indicating highest performance and 1 indicating lowest performance.

4. Project Standards and Design

When it comes to project development, there is a significant number of variables one must address and control. Controlling a machine's temperature, determining the connection types, setting rules for battery interfacing and circuit board work, and more all address and control some of the variables a project works with, forming a set of Standards. In some cases, existing circumstances might control these variables for us. Financial limitations, size restrictions, manufacturing prerequisites and followup requirements, and more all fit under this umbrella; these control yet more variables, forming the project's Constraints.

It is through these standards and constraints that the final design may begin to take shape. Without standardization or constraints, designs would run wild with inconsistencies and blank values, could cost far more than they should, or simply not do what the consumer actually wanted in favor of what the engineers and designers choose. Standardization and constraints, while limiting factors, ultimately aid engineering design to offer consistency and financially reasonable solutions for interested parties.

These standards and constraints are split according to whether or not they are associated with the hardware of the Hand Gesture Vending Machine, or associated with the software of it. Similarly to Section 3. Related Project Research, this has been done for the sake of organization and ease of access. Splitting these standards and constraints also offers us the ability to delegate who did assure these standards and constraints are followed to the best of our ability.

4.1. Hardware Standards and Constraints

4.1.1. Connection Standards

Connection standards are something that we have to ensure to follow. Being that our device is utilizing many different components, we need to ensure that the proper connections between the devices and components can be made and function properly. For the camera and the AI development board, to ensure compatibility the camera's cord was taken into account and cross checked with the development boards to ensure that a connection between the two can be done. We also need to take into account the connection made between the microcontroller and the numerous IR sensors that were implemented in the vending chamber to ensure proper feedback. The battery that did power the system did also need to be able to connect and supply power to each hardware component within the system. This meant we also needed to not only keep track of how much power each component needs but the type of port that is required to power the board to ensure it can be implemented with our power supply. Connections also need to be considered and standardized for both the LCD display and LEDs.

To ensure that these standards are followed, as well for the PCB design, we did follow IPC standards for the connections within our PCB. IPC-2221 is a generic standard for PCB design that was followed to ensure proper quality and performance from our PCB design. This particular standard also has other underlying standards further within the IPC-2220 series of standards that cover rigid, flex, and other PCB designs which we did also utilize for our particular design to follow the standards for our PCB.

In this project, we did use a Camera Serial Interface 2 (CSI-2) connection between our selected camera and the Nvidia Jetson Nano AI Development Kit. This is a subtype of the Mobile Industry Processor Interface (MIPI) connector. MIPI standards have gone through several revisions over the decades; these revisions are maintained and supervised by multiple organizations, forming a larger organization called the MIPI Alliance [88]. These standards, set by the conglomerate organization, are in place to ensure that the connections are supported correctly and to keep connection speeds high.

4.1.2. Soldering Standards

J-STD-001 is the standard issued by the Institute for Printed Circuits (IPC) for soldering [89]. The institute first released this standard in 1992, and has since published revisions to the standard. The institute issued this standard in order to help train and certify individuals on both proper and safe processes to connect two conductive metal surfaces to one another. This standard goes into great depth on the material requirements, process, acceptable, and unacceptable outcomes. Material requirements cover the type of solder paste, solder system, cleaning media, and flux, as well as the target mediums to solder together. It is important to use the correct material, so no errors or defects may occur during the process of soldering.

With this in mind, the standard states the proper way to solder from the distance between two mounting parts to the method used for through-hole components. The standard goes on to demonstrate how a soldered part should look, and if any errors or defects occurred, the standard tells one how to fix it. Improper soldering is a quick way to ruin a project or inflict injuries. Therefore, in this project, we did follow this standard to avoid common problems found with incorrect soldering. Ultimately, J-STD-001 is an essential standard to follow the best practices in soldering.

4.1.3. Economic Constraints

In Section 2.4. Requirements and Specifications, various self-imposed limitations were set on this project. Among them was the financial constraint of keeping the entire budget of the Hand Gesture Vending Machine within seven hundred dollars ($700.00) to the best of our ability. This is critical in multiple aspects. First and foremost, an overall cheaper design allows for more freedom for any emergency replacements in components or additions to be made later on in the case of faulty or uncooperative components. Second, a cheaper design permits for less strain to be put upon us as a whole team; as university students, finances for the team are not exactly flush with wealth, so minimizing costs did ease our stress and our wallets. Third, a cheaper prototype in a practical scenario is critical to gather the interest of companies observing the work done; a cheap prototype would likely mean a cheaper final product, meaning they can profit more from the design.

The easiest ways to minimize cost for a project such as ours is threefold. Most critically, using components already in our possession. A good example of this is that we are re-using an Nvidia Jetson Nano instead of purchasing one - a cost cut of over a hundred and fifty dollars. Second, simply selecting cheaper components when they did do the job over high-performing options. Oftentimes, the best option is not the one that can do a lot, but the one that does just enough to work well for the project. The final thing to do is to double-check the overall design prior to committing to the components - to measure twice and cut once, as the saying goes. This did help minimize part replacement from our own purchasing errors, which by extension cuts down on the final cost. One example we are familiar with is the purchase of resistors in the wrong size; the resistors themselves are cheap, but the shipping is fairly expensive, so re-purchasing the components in the right size is much more expensive.

4.1.4. Weight and Size Constraints

Another self-imposed restriction set for the Hand Gesture Vending Machine had been the machine's weight and size. For this project, we have limited it to twelve kilograms (~26.5lb) and sixty-one centimeters in all directions (61 cm x 61 cm x 61 cm, ~2' x 2' x 2') despite being a vending machine; this is because the number of vendable options is only four instead of the wide selection most modern vending machines possess. As a small-scale vending machine, it did remain relatively boxy in shape and be designed with being moved around without a dolly or lift in mind.

Accomplishing all of the above requires some significant considerations. The constraint of size is relatively simple, when standing alone. Much of the work to accomplish it comes down to 3D modeling and properly setting and segregating the space for vending goods and internal components in such a way as to avoid cramping either half; given the dimensions of various snacks are easily accessed - either on the internet or via purchasing one - this can be done with some experience in a modeling program and discussion with a metalworking company that does custom requests. On the other hand, weight is a much more difficult thing for us to balance. A metal shell for the Hand Gesture Vending Machine is not light, and did likely be a majority of the project's weight on its own. Add in the components, and there is a genuine risk of going over the weight constraint we have set. While unfortunate, this is still acceptable so long as the weight is not so egregious as to need us to use a dolly to lift the project and move it around.

4.1.5. Temperature Constraints

Given we are working with an AI Development Kit and object detection in this project, one significant point of concern is the risk of overheating the expensive component from consistent use. As such, one of our self-imposed rulings is to keep the device at or under ninety degrees fahrenheit (90°F), which is about thirty three and one third degrees celsius (~33.3°C). This temperature may seem high for an indoor vending machine prototype, but object detection is a graphically intense process and heats up the processing unit significantly. As such, cooling it is a top priority. There is also the power supply to consider, which also requires space for its fans to intake cool air and output the hot air from its constant operation.

The best means to accomplish cooling for both is quality ventilation. This has to be taken into account with both fans and mounting spots for said fans on the external shell of the vending machine. Additionally, it is critical that there is open space for both the power supply and the AI Development Kit to 'breathe' with their fans; without clear access of both cool air to intake and space to ventilate the hot air, the Hand Gesture Vending Machine won't cool properly.

4.2. Software Standards and Constraints

4.2.1. Programming Languages

This section did go over the standards and constraints in programming languages. Programming languages are standardized to ensure consistency of output, scalability, and the quality of the product. Constraints in software development serve as an inadvertent bottleneck for the progress of a project.

4.2.1.1. C Programming Language

C is one of the most commonly used programming languages in the world. This is because C is a relatively lightweight language that many programmers know, and it can specialize in low-level programming and high-level programming as well. The C programming language has several standard versions, with the most commonly used ones being C89/C90, C99, C11, and C18 [90]. C18 (ISO/IEC 9899:2018) is the most recent standard for the C programming language. C is one of the most commonly used languages, more specifically, for microcontrollers and was used for the hands free vending machine.

4.2.1.2. Python Programming Language

Python is one of the most commonly used programming languages in the world. This is because Python is an object-oriented language that is generally considered easy to learn and it specifically specializes in automation and more specifically, in the case of the hands free vending machine, machine and deep learning. The Python programming language has several standard versions. PEP-8 is the most recent standard for the Python programming language. Python is one of the most commonly used languages, more specifically, for deep learning and was used for the hands free vending machine.

4.2.2. Programming Constraints

A constraint, within the context of software engineering, is a restriction on an aspect within the project at hand. The constraints can assist the programmers in making very consistent and connected code; with similar constraints in mind, individual programmers, on a team, can work on the same project at the same time and the code would not be as disconnected as it would be without constraints in mind. Constraints for software can include speed of the program, length of the program, cost, time to build the program, and anything else that can restrict the different possible variations of code.

Constraints, for any type of project, should be some of the more important focuses a group can make, early on. The effects of this intentional focus, did be felt by the group completing the project and the project itself, for the duration of the project. At the surface level, the importance of the identification of the constraint, or constraints, comes with the knowledge to be able to identify, within the project, what did be the problem that did take up the most, unavoidable time. The identification of the constraint, or constraints, also comes with the ability to exploit the constraints by making everything else much more efficient.

Similar to any software program, there are natural constraints to create the most efficient code. Every programmer wants to develop the quickest and most accurate code while also maintaining cleanliness and efficiency. Clean code helps programmers easily troubleshoot bugs and makes for further efficient code. This includes having optimal loops, omitting useless or unnecessary lines, and using correct and updated syntax.

The effect of having little to no constraints on a software development project is sloppy code amongst the software development team; this can include slower code, very long code, and the lack of constraints did ultimately result in a substantial increase in time spent on the program. The increase in time spent on the software development can be attributed to multiple individual things, however it can just be primarily attributed to unconnected code between multiple different people.

The lack of connectivity especially shows up for the debugging stage; at the point of debugging, when the code from all the different people is disconnected, all the code would need to be debugged individually, vastly reducing the efficiency of the debugging portion of the project and in turn the project.

A software constraint the group did adhere to is response time of the hand gesture recognition; this is important because with this in mind, we are able to make the main focus of the code to be speed. Another software constraint in the group is accessibility; with this constraint in mind, we are able to make the code more versatile.

4.2.2.1. Identification of Programming Constraints

Every sophisticated project needs the identification of constraints. Without the identification of constraints, there is no plausible ability to set goals; with unknown constraints, the project did not have a proper itinerary and an improper itinerary, leads to disorganization of the project, mismanagement of time, and, possibly, a late product. In the case of software development, and the hands-free vending machine specifically, a known constraint is learning new programming languages, such as the language for the STM32F429ZI. Without an identification of this constraint, or no assumption of constraint, this would consume more time than expected, leading to every other aspect of the project being late, as well as, creating a bottleneck at this juncture. This bottleneck would lead to the piling up of responsibilities centered around the assumption that the software is already known and thus would add an immense amount of time to the project.

Without the identification of constraints, changes in the system would have an unknown project-wide impact. With an unknown project-wide impact, there was no gauge of anything; there is no timetable, there is organization, and there is no knowledge of how the new system communicates with the constraint. In the case of software development, and the hands-free vending machine specifically, a known constraint is the researching of a new software, such as the software for an AI developer kit. If a change were to occur to the model of machine learning that is being used, this would create a bottleneck. This bottleneck would lead to the piling up of responsibilities centered around the assumption that the system is the same and thus would add time to the project.

Without the identification of constraints, it is difficult to figure out where to focus limited resources; without a complete understanding of where to focus limited resources, such as time and money, the project has a poor sense of time management and money management. A poor sense of time and money management did lead to a disorganized project process and project as a whole. With the time management issue, this would lead to a lot of stress, in terms of workload, on the programmers.

In regards to the money management issue, it could lead to a level of overspending and also a possible back up of workload due to the inability to work with the products needed, for the project, on time. This bottleneck would lead to the piling up of responsibilities centered around the assumption that the best product, for the project, was selected and thus would add time to the project.

Without the identification of constraints, the rules to optimize a system and the process of creating a product cannot exist; the way to optimize a system is to optimize the constrained area. Furthermore, the process of creating a product is only as efficient as the process of developing the constraints, thus the optimization of a system is completely reliant on the speed in which the constrained components are developed. With an unoptimized system, this can lead to missed deadlines due to not optimizing the constraints, inefficiencies of components, and inefficiencies of systems.

In the case of software development, and the hands-free vending machine specifically, a known constraint is the researching of datasheets, such as the researching of datasheets for a microcontroller. If the datasheets were not researched before reaching the lab, and the group came to the lab with no knowledge of the board, then the time in the lab becomes a waste; the time in the lab would become a time where the group researches the board, which is something that could have been done previously, creating a massive waste of time and a bottleneck. This bottleneck would lead to the piling up of responsibilities centered around the assumption that the board and datasheets have been researched and the assumption that there is a level of familiarization with the board.

4.2.2.2. Exploitation of Programming Constraints

To make a project as efficient as possible, there needs to be an exploitation of constraints. The exploitation of constraints occurs when the predefined and researched constraints are used in a way where instead of being the issue that brings down the whole project, or a disadvantage, they become a focal point and become the most important component in the project and, furthermore, become a component that can be taken advantage of in spite of the restricting nature of the constraint.

The purpose of exploiting a constraint is to make the system as efficiently as possible and to make the system as efficient as possible. Considering that a constraint can be a project's biggest weakness, but could be the same project's biggest strength, the efficiency of the project relies on the ability to maximize the throughput of the researched constraints; a chain, or the project in this case, is only as strong as its weakest link.

Considering that the efficiency of the project relies on the ability to maximize the throughput of the researched constraints, the impact that maximizing the exploitation of the constraints cannot be understated; the inability to efficiently exploit the constraints leads to multiple different unpleasant outcomes. Some examples of this are missed deadlines, an inefficient system, and a system that is incomplete.

Missed deadlines are an outcome of the inability to maximize the constraints because when constrained components are treated as if they are a regular component, the amount of time wasted is significantly increased. An example of this is when the software, which is the constrained resource, is not properly tested and it fails. What would stem from this issue would be a mass of resources, that required the software as a prerequisite, that would have to be placed on hold until the software is completely finished. An inefficient system is an outcome of the inability to maximize the constraints because, when considering the global constraint of time, whole sections of the project, that make the product run as smoothly as it should, did need to be removed. Thus, resulting in a product that is complete, but not as efficient as it could and should be.

The five ways the software development team, for the hands-free vending machine, plan on exploiting the constraints is evaluating the compatibilities of the software systems early and often, constantly reassessing, parallelizing the non-constraints, minimizing the interruptions for the constraints needed resources, and constant collaboration and communication with group members.

The way the software development team plans on evaluating the compatibility of the software systems early and often is by constantly researching how we plan on having the software interact with the hardware. However, this is not researching only a week ahead, more so multiple weeks ahead of the planned implementation. This is so that when the components researched have arrived, there is a smooth implementation and minimal bottlenecking.

The way the software development team plans on constantly reassessing is by constantly reevaluating all the different functions, regularly, and searching for a way to increase the productivity or efficiency of the function. This did include the deep learning aspect of the project and the software to hardware aspect of the project; both can become something that holds back the project in terms of the ability for it to be as efficient as possible, and in this case, as fast as possible.

The way the software development team plans on parallelizing the non-constraints is by programming different sections of the project at the same time. The way this exploits a common constraint is by exponentially increasing the throughput of a singular programmer by having multiple programmers work on multiple different sections of code.

The way the software development team plans on minimizing the interruptions for the constraints’ resources is by always having the correct parts in hand or coming in the mail while also having the matching code for these specific parts finished or almost finished. Completing tasks this way allows the software development team to help elsewhere where needed or to code sections for parts that have not come in yet. The way this exploits a common constraint is by never allowing the hardware to be waiting on the software.

The way the software development team plans on collaborating with the team for extra knowledge is by staying in constant communication. This constant communication is not always going to be asking for help with different sections of code; other reasons can be to delegate the programming responsibilities for incoming components and also for trading programming responsibilities if someone gets stuck on a problem. At this point is when sophisticatedly commented code becomes very important.

Cleaner, more understandable code leads to easier debugging, and easier maintenance, between multiple programmers. This leads to less time spent on debugging and more time helping in other areas in the project. Furthermore, when done correctly, greatly commented code, with appropriate variable names, leads to connectivity between programmers and therefore more connected and cleaner code. This can lead not only to code that is visually appealing, but more importantly, lead to code that can be seen, within the group, as written by not an individual, but the whole group, due to its readability.

4.2.2.3. Subordination of the Non-Constraints

To make a project as efficient as possible, there needs to be a subordination of non-constraints. The subordination of non-constraints occurs when the “other elements in the development process align their performance to support the constraint’s requirements” [90]. The purpose of subordinating the non-constraints is to make the system as efficient as possible, to make the system as efficient as possible, and to optimize resources. An effect of not subordinating the non-constraints is a slowly progressing project; at its core, the subordination of non-constraints exists to speed up the process of a project.

Considering that the efficiency of the project relies on the ability to maximize the throughput of the researched constraints, the impact that maximizing the subordination of non-constraints cannot be understated; the inability to efficiently subordinate non-constraints lead to multiple different unfortunate and avoidable outcomes. Some examples of this are missed deadlines and a finished product that is not as imagined. Missed deadlines are an outcome of the inability to subordinate non-constraints because when priorities are split between multiple different things, that do not include the constrained components, time cannot be used as efficiently as possible with the constrained resource and therefore more time did be spent on lower priority components and less did be spent on the higher priority components. An example of this is if more time was spent on UX, in a project that centers around hardware, than with motors, microcontrollers, and PCB design. What would stem from this issue would be a very underwhelming display of hardware implementation and an unfinished product.

The four ways the software development team, for the hands-free vending machine, plan on subordinating the non-constraints is making sure the externals are always readily available, keeping log of ready-to-test features, optimizing non-constraints, and regularly reassessing non-constraints.

The way the software development team plans on making sure the externals are always readily available is by constantly looking ahead, and researching, at what can possibly be the next component we can get in hand, ordering it, and how we plan on implementing the software onto it. With multiple people in the software team researching multiple components, ordering them, and brainstorming as a team how we plan on deploying the software onto them, at the same time, it can be imagined how efficient this can be. With all these actions occurring almost simultaneously, it leaves plenty of room for the unavoidable small, inevitable mistakes that did occur over the course of the project.

The way the software development team plans on keeping a log of ready-to-test features is by keeping a list that we all have access to, with names attached to each component that needs to be tested. This did help prevent any confusion or thinking, within the software team, about who has to do what test for what component.

The way the software development team plans on optimizing non-constraints is by focusing on the non-constraints secondarily as far as time and effort. Furthermore, the non-constraints, for the most part, did only change synchronously with constraints that exist in the same system, or subsystem. An example of this is of a three-pin IR transmitter and receiver, the non-constraint, only being researched and focused on when it exists in the same system as the microcontroller; the time spent on the microcontroller datasheet exists as a constraint.

The way the software development team plans on regularly reassessing non-constraints is by keeping track of what exactly takes the most time to do, how many times we have to change or adjust a certain subsystem in the project, or anything of that nature. This keeps from creating a new bottleneck without noticing it.

4.2.2.4. Training and Testing Constraints

After some research, by the software development team, it was found that two of the primary constraints are the constraint of training the deep learning models and the testing of the software. Firstly, these two constraints tended to show up in the same places, same systems, at the same time and, therefore, seemed foolish to treat them as if they were two separate constraints.

The impact that the training and testing have on the system is large and therefore cannot be removed; the only way the hand gesture machine can do its job is by using the model, trained by deep learning, and predicting the hand gesture that is being depicted by the user. Furthermore, the only way that the model can, consistently, correctly predict the hand gesture being displayed, by the user, is by being heavily tested. The result of these constraints, naturally, did be the new components taking some time to be fully implemented into the circuit. In the case of the training constraint, it is slightly less dependent than the testing constraint, from the fact that it only connects to the AI development kit. However, the process of training the model is rather cumbersome and every time development kit settings or the camera on the AI development kit are adjusted, the model needs to be adjusted to be able to take in less information or work with less information.

If this constraint was unknown or not acknowledged, it would vastly slow down the process of making the project. Some of the effects of ignoring this constraint are little to no understanding of how the AI development kit communicates with the model and minimal understanding of how microcontroller's software works and their complementing datasheets. Without knowing how all these subsystems work, the hands-free vending machine cannot exist.

Firstly, understanding how the AI development kit communicates with the model is very important; without the process of the development kit communicating with the model, the result, at the most, would be a vending machine shaped object that has no capabilities of a real vending machine. This includes no selection of snacks, no confirm payment, and no deny payment. Understanding how the microcontroller's software works and their complementing datasheets is very important; without understanding this, there is no way for the power, from the supply, to be used properly within the context of the project. The result from this constraint being ignored would be power not being correctly converted to work. In a vacuum, if any of these subsystems were to stop working, it would be catastrophic; if all these were to stop working, this would not bear any resemblance to a real project. Which is why it is paramount to acknowledge the constraints of testing and training and consider them when any decision is made in regard to the project as a whole.

A goal that has been set around the constraint of training and testing is adhering to the schedule previously set. This did allow the software development team to focus on the constraint and make sure that it is the priority for the vast majority of the project's span. The ways the software development team plan on exploiting the constraint of training and testing is constantly evaluating how the microcontroller communicates with all subsystems in the system, always reassessing the testing strategies, collaboration of teammates, and using external software to test the code for us, so we can complete other tasks. The ways the software development team plan on subordinating the non-constraints, relating to the training and testing, are keeping a log of ready to test features, continuously reassessing the system for new bottlenecks, and making sure the sensors, microcontrollers, and motors are always readily available.

4.2.2.5. External Dependencies

After some research, by the software development team, it was found that another one of the primary constraints is the constraint of external decency, on hardware. This constraint showed up more often than not and it especially showed up when examining the testing constraint for any inefficiencies or any linked constraints.

The impact that the external dependencies have on the system is big and cannot be adjusted or moved; the only way the machine can be made and taken care of is with getting the necessary components and putting them into the machine. Furthermore, the only section of the project that is possible without getting parts in would be the hand gesture detection software. However, the hand gesture software realistically means nothing if the parts, for the hand gesture software, don't exist to operate on; the highest capability of this software would be to just read the hand gestures displayed by the user. The result of this constraint was having to wait to test microcontroller code or any code that relies on an external component, such as motors, IR transmitter/receivers, or fans, until the external component is in hand.

If the constraint of external dependencies was unknown or not acknowledged, it would vastly slow down the process of making the project. Some of the effects of ignoring this constraint are undertested components and under-researched components. Without knowing how all these subsystems work and how they are related, the hands-free vending machine cannot exist. Under-tested components can be the reason why a whole project fails; in our case, an undertested component could be a microcontroller, an IR transmitter/receiver pair, amongst other components. under-testing these components would directly result in an unfinished product.

Research and development is one of the most important aspects of projects on any level. Under-researching components is never something that should considered, in turn, the researching of components and how they interact in systems is the reason why projects succeed; in our case, an under-researched component could be an AI development kit and the repercussions of a case like this would result in a vending machine system that does not receive input correctly. In a vacuum, if any of these subsystems were to stop working, it would result in an unfinished product; if all these were to stop working, this would not be a presentable project. Which is why it is paramount to acknowledge the constraint of external dependencies and consider them when any decision is made in regard to the project as a whole.

A goal that has been set around the constraint of external dependencies is to study and research incoming components and their compatibilities, to our board, weeks prior to receiving them. This did allow the software development team to have a good idea of the plan of action, and possibly even a template or theoretically working code, before the component is received from the mail. The way the software development team plans on exploiting the constraint of external dependencies is working as a team to research each component and its relevance to the system weeks prior to receiving the components.

4.2.3. Software Developmental Standards

Standards are a set of self-imposed rules and guidelines to adhere to when building a project. Standards help organize the project for efficiency purposes, but also for continuity amongst teammates building the project; they help make sure the project is built in a consistent and reliable way. The way standards impact the overall system is they provide a solid blueprint as to how the work gets finished, in the project, and more specifically creates a guideline to be followed through the project. In the specific case of software development, some standards that would be set would be standards for the formatting of the code, standards for the programming language, and standards for commenting the code. Furthermore, the discussion of the overall impact of upholding the selected standards, cannot be understanded; programming languages use standardized syntax to keep consistency in each system. If each system used different programming language standards, then systems would not be able to collaborate and cause massive errors to occur.

An example of a standard the software development team is working to uphold is how our code did include as little loops as possible to decrease the runtime of the code. To further explain, loops may take an extremely long time based on how long it takes for the conditions to be met. By programming loops to not include that extra time, the programmer can reduce the overall runtime of the system. Since a requirement for this project is to have a response time less than one minute, a great way to achieve this is to limit the number of loops used or optimize the loops. Another constraint is to not include any unnecessary lines of code. This only increases the runtime and creates bugs in your program. In addition, every programmer must follow the proper syntax for the language that they are utilizing.

For the Hand Gesture Vending Machine, the programming language used to allow the microcontroller to communicate is C and the programming language used to allow the aspect of deep learning to occur is python. Programming languages use standards to set an example of how statements should be structured in that particular programming language; standards tell the programmers where semicolons go, what can and cannot be placed in a loop, how to structure a loop, and anything else that is considered when producing a working program in that specific language. Whether the programming language is strongly typed, where specification of data types is needed, or weakly typed, where specification of data types is not needed, it has a standard of how to structure the code to make the program work. Without programming language standards, there would be no coherence or consistency between two different programs of the same language. This would not only cause confusion from the perspective of the programmer, but there would be a level of incoherence that would cause massive problems for any program written in any non-standardized programming language.

4.2.4. Communication Standards

Communication standards and protocols play a crucial role in enabling devices and systems to exchange data and interact with each other effectively. To achieve success with the Hand Gesture Vending Machine, we did use various peripherals. We did use the Serial Peripheral Interface (SPI), a synchronous serial communication protocol commonly used to transfer data between microcontrollers and peripheral devices at high speeds. When using SPI it did involve four signals, Master Out Slave In (MOSI), Master In Slave Out (MISO), Serial Clock (SCK), and Slave Select (SS).

An engineering requirement we have is to create a hand gesture vending machine with a response time of less than 15 seconds. The way we're trying to achieve this requirement is through the STM32F429ZI and the Nvidia Jetson Nano. The STM32F429ZI has efficient processing capabilities and optimized communication protocols, such as SPI, which ensure quick response times for the vending machine. It did rapidly process input signals from sensors and activate vending actions with minimal delay. The Jetson Nano, with its powerful GPU, did facilitate rapid data processing and decision making for complex tasks, further reducing response times.

The STM32F429ZI did communicate with sensors and peripherals using protocols such as SPI, and I2C. SPI was used for high-speed communication with devices like the SPI connected screen, while I2C did facilitate communication with sensors and other components. The Jetson Nano did communicate with the STM32f429ZI and other devices using UART, SPI, or USB interfaces. UART can be used for serial communication between the Jetson Nano and the microcontroller, while SPI or USB can be employed for connecting additional peripherals or external devices. Overall, the combination of the STM32F429ZI microcontroller and Jetson Nano enables efficient hardware/software integration, meets machine requirements, and adheres to communication standards, ensuring the successful implementation of the hand gesture vending machine project.

5. ChatGPT and Similar Algorithms

5.1. ChatGPT

ChatGPT is a conversational AI developed by OpenAI. The name comes from its output that tries to be a conversational response from what the user asks it and that it is based on the GPT architecture. GPT stands for generative pre-trained transformer. Transformers were developed by Google to address the limitations of recurrent neural networks [91]. Transformers can outperform recurrent neural networks by using attention mechanisms to focus on different parts of the input sequence when making predictions. ChatGPT is trained on vast amounts of publicly available text data from the internet to generate contextually relevant and diverse answers. New "chats" can be created to create context between user inputs, so you can ask for more information and ChatGPT did know what you have already asked in the same chat.

ChatGPT-3.5 is available to use for free, so we mostly used that. ChatGPT-3.5 has a knowledge cut off of January 2022. When asked about the future or topics past January 2022, the response did often say that it doesn't know information past that date. ChatGPT-4 In fact in Figure 5.1, when ChatGPT-3.5 is asked about ChatGPT-4, the model doesn't even know that its successor has already been released. The knowledge cutoff for ChatGPT-4 is April 2023 and presumably that did be periodically updated for the paid option.

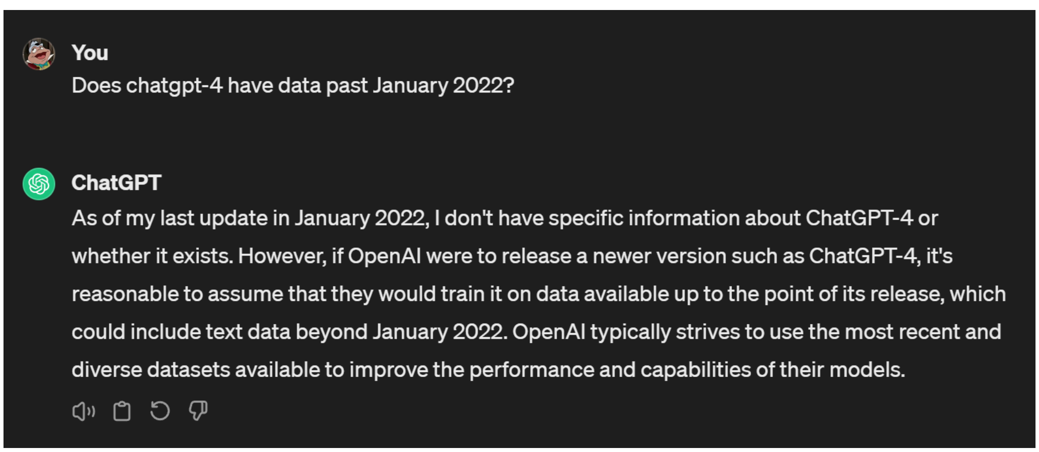


Figure 5.1.1: ChatGPT-3.5's response to ChatGPT-4.

By using ChatGPT, we were able to gather information, insights, or assistance regarding LED displays in various ways. We were able to leverage information from ChatGPT providing explanations and descriptions of LED display technology, including how LEDs work, their advantages over other display technologies, and the different types of LED displays available. Additionally, we were able to compare LED displays. This was accomplished by asking ChatGPT to compare different types of LED displays, such as

LED-backlit LCDs vs OLED displays, or different models and brands of LED displays. This allows for a gaining of understanding the differences in features, and performance.

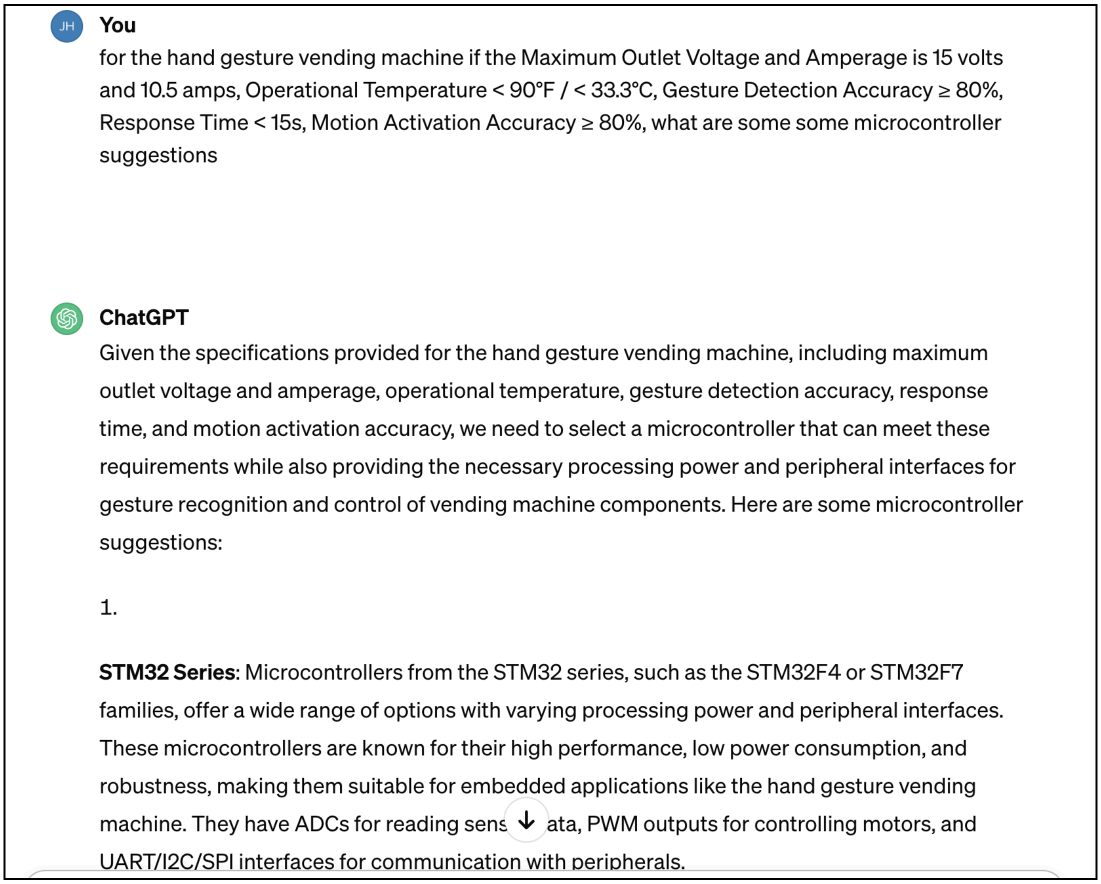
We were also able to take advantage of ChatGPT to create tables to compare the different LED options. There were a few times where we had to reiterate to ChatGPT about the tables, however, because the information was wrong. When gathering all of our information, we did have to go back through some of it to make sure it was all correct. This was the same for every time we made use of the ChatGPT algorithm - if only to assure its accuracy.

When entering the overall machine requirements and preferences, we were able to ask ChatGPT for recommendations on LED displays that meet the criteria. After providing information, such as the size of the Hand Gesture Vending Machine, alongside its estimated weight, budget, and operational temperature, ChatGPT was helpful in assisting with researching specific specifications of LED displays, such as power consumption and brightness levels. By leveraging ChatGPT's knowledge and capabilities, we were able to gain valuable insight, and information to help better understand, and utilize LED displays for my project.

There were many screen options to choose from, and ChatGPT was able to help breakdown and provide all the information that was needed. ChatGPT can explain various screen technologies such as TFT LCD, OLED, LED, and touchscreen displays. By asking ChatGPT certain things such as advantages and disadvantages of each technology to gain a better understanding. When asking ChatGPT to create a table to compare the different screen options, it categorized things by brightness, contrast ratio, color depth, viewing angles, cost, and performance. When mentioning the size of the entire hand gesture vending machine, ChatGPT was able to suggest suitable screens based on size, budget, and resolution.

The understanding of the MCU can be explained from ChatGPT. It can explain the fundamentals of microcontrollers, including how they work, their components, architecture, and typical applications. If someone is struggling on certain concepts such as CPU, memory, peripherals, and input/output ports then ChatGPT would be able to explain such topics. When telling ChatGPT how many numbers of I/O pins, memory size, processing power, communication interfaces needed, and any other specific features required, one can then be able to get a suggested list of different MCU.

ChatGPT can find suitable microcontrollers, as well; with the help of ChatGPT it can help you compare different families of microcontrollers such as STM32, Arduino, ESP32, and PIC. Each microcontroller has different features, capabilities, and programming languages supported. I was able to ask ChatGPT to compare all the microcontroller options on one table. The table was categorized by processing power, memory, peripherals, operating voltage, and development tools. ChatGPT can also be used for troubleshooting microcontroller issues. For instance, if you're encountering errors within your code, ChatGPT can suggest some corrections although it may not always be correct. Utilizing ChatGPT it can provide guidance on programming microcontrollers using languages such as C++, C, Python, or Arduino IDE. If you're stuck you can ask for explanations on programming concepts, code examples, debugging techniques, and best practices.



You

for the hand gesture vending machine if the Maximum Outlet Voltage and Amperage is 15 volts and 10.5 amps, Operational Temperature < 90°F / < 33.3°C, Gesture Detection Accuracy ≥ 80%, Response Time < 15s, Motion Activation Accuracy ≥ 80%, what are some some microcontroller suggestions

ChatGPT

Given the specifications provided for the hand gesture vending machine, including maximum outlet voltage and amperage, operational temperature, gesture detection accuracy, response time, and motion activation accuracy, we need to select a microcontroller that can meet these requirements while also providing the necessary processing power and peripheral interfaces for gesture recognition and control of vending machine components. Here are some microcontroller suggestions:

1.

STM32 Series: Microcontrollers from the STM32 series, such as the STM32F4 or STM32F7 families, offer a wide range of options with varying processing power and peripheral interfaces. These microcontrollers are known for their high performance, low power consumption, and robustness, making them suitable for embedded applications like the hand gesture vending machine. They have ADCs for reading sens ata, PWM outputs for controlling motors, and UART/I2C/SPI interfaces for communication with peripherals.

Figure 5.1.2: ChatGPT conversation on MCU suggestions (1/2).

2.

ESP32: The ESP32 microcontroller offers built-in Wi-Fi and Bluetooth connectivity, which could be useful for future enhancements or remote monitoring of the vending machine. It has dual-core processing, ample memory, and a rich set of peripheral interfaces, making it suitable for handling gesture recognition algorithms and controlling vending machine components.

3.

Raspberry Pi: Although technically not a microcontroller, the Raspberry Pi single-board computer offers significant processing power and flexibility for complex applications like gesture recognition and web app development. It can run machine learning algorithms for gesture detection and interface with peripherals via GPIO pins. However, it may consume more power compared to traditional microcontrollers, so power management considerations would be important.

4.

Arduino Due: The Arduino Due is based on the Atmel SAM3×8E ARM Cortex-M3 CPU, offering high performance and a rich set of peripheral interfaces. It has ADCs, PWM outputs, and communication interfaces like UART, SPI, and I2C. While it may not have built-in Wi-Fi or Bluetooth like the ESP32, it can be supplemented with external modules for wireless connectivity if needed.

Figure 5.1.3: ChatGPT conversation on MCU suggestions (2/2).

The last section that ChatGPT was used for was the Glass and Glass Alternatives Section (Section 3.6.2.). Our group as a whole was unfamiliar with the various options on the markets, their advantages, and their disadvantages, so we made use of ChatGPT to assist with initial research. ChatGPT was able to help us understand the different types of glass commonly used in various applications, such as tempered glass, laminated glass, acrylic, and polycarbonate. With this initial supplement, alongside other online resources, we accomplished our comparison with little trouble. ChatGPT was also capable of tabulating the glass and glass alternatives for use within the documentation.

5.2. Similar Algorithms

5.2.1. DALL·E 3

DALL-E 3 is a neural network model specifically trained for image generation from textual descriptions, excelling at generating highly detailed and realistic images based on the input text. The primary use of DALL-E 3 is used for tasks related to image generation, such as creating artwork, generating illustrations, designing products, and visualizing concepts described in text. DALL-E 3 is trained on deep learning methods, language processing, and machine learning algorithms, allowing it to learn the relationships between textual descriptions and corresponding images.

5.2.2. Google Gemini

Google Gemini - previously known as Google Bard - can generalize and seamlessly understand, operate across, and combine different types of information including text, code, audio, image, and video. Google Gemini is a family of multimodal large language models comprising Gemini Ultra, Gemini Pro, and Gemini Nano. Gemini Ultra is used for highly complex tasks. Gemini Pro was incorporated into Google Bard at the time. Gemini Nano is a very efficient version that could be integrated into Google's smartphones, the Google Pixel phones. With a score of 90%, Gemini ultra is the first model to outperform human experts on massive multitask language understanding (MMLU), which uses a combination of 57 subjects such as math, physics, history, law, medicine and ethics for testing both world knowledge and problem-solving abilities.

5.2.3. Google Generative AI Overview

Most people today look up many topics on Google everyday and sometimes at the top of results there is an experimental feature called Generative AI Overview. Generative AI is said to not do all the work for you but to help with the creative process. Likewise many times the AI Overview is completely wrong, so literally taking the top result is not always the best course of action. These results could be discontinued like many Google products and experiments but they can be very helpful in a pinch. Since the experiment was active and used by us during our research, it seems important to note this.

5.2.4. Microsoft Copilot

Microsoft Copilot - previously known as Bing Chat - is an AI powered code completion tool developed by Microsoft built upon OpenAI's GPT-4. As Bing Chat it would perform searches relevant to the user's question using Microsoft's own search engine, Bing and use the searches to give an informed response. It is designed to assist developers in writing code more efficiently by providing contextually relevant suggestions and auto completions. Microsoft Copilot represents a significant advancement in AI assisted coding tools, aiming to streamline the development process and empower developers to write code more efficiently and effectively.

6. Hardware Design

For the most part, much of the hardware design is directly interlinked with software or optical design. This does not, however, take away the hardware element of each component, nor does it take away the need for hardware connections for these components. With this in mind, the hardware design can be fairly easily broken down into smaller, simpler subsystems to explain the individual functionality of the components and how they interlink with the microcontroller on the micro-scale before coming back to the full hardware schematic.

There are a multitude of hardware subsystems, each with their own considerations to resolve and move forward to the next. There is the Power Supply and its need for additional voltage regulators, the LED subsystem, the front-facing Infrared Sensor subsystem, the AI Development Kit and Camera subsystem, the Fans subsystem, the Screen subsystem, the Infrared Transmitters and Receivers, and the Motors.

6.1. Power Supply Subsystem

Before we can talk about the rest of the components, it is critical to talk about what's powering it all. As the Hand Gesture Vending Machine is a vending machine, it is being plugged directly into the wall as its power source. Given outlets in the United States use one hundred and twenty volts (120V) at sixty hertz (60Hz) and a current maximum between fifteen and sixty amperes (15A - 60A) based on the cable type, this is significantly higher than the voltage requirements of a vast majority of our components. The only component capable of intaking such voltages, in fact, is our power supply.

However, our ALITOVE power supply is insufficient in lowering the voltage throughput, outputting between twelve and twenty-four volts (12V - 24V) and between one to six amperes (1A - 6A). This is significantly lower than the outlet's output, but it is still much too high for direct interfacing with the microcontroller. As such, we require voltage regulation in order to safely connect the two to each other.

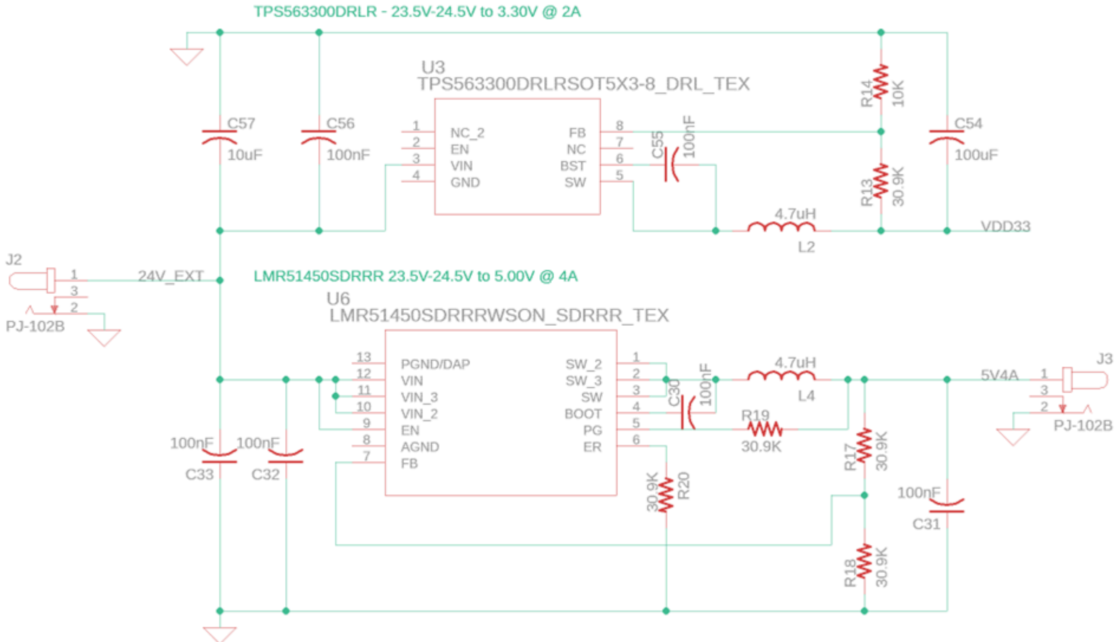


Figure 6.1.1: Power Supply Voltage Regulators.

Our microcontroller and the integrated circuits around it require both 3.3 and five volts (3.3V and 5V) and the Nvidia Jetson Nano wants five volts (5V). We can accomplish this by building two voltage regulators. Using the Webench Power Designer by Texas Instruments, we can come up with a multitude of solutions to this problem. Just put the input voltage of our twenty-four volt (24V) power supply and the output, and we get hundreds of designs instantly. Every integrated circuit in the design has datasheets and the Webench can print out a report on the design along with the bill of materials. Designs are graded by efficiency, bill of materials cost, footprint, component count, and integrated circuit cost. The most efficient design did use a lot of components which also drives up the footprint of the overall design. So, it just comes down to picking a design with a good balance of attributes.

Figure 6.1.1 shows our two finalized voltage regulators. The top half takes in twenty-four volts (24V) and outputs 3.3 volts (3.3V) with a maximum current of two amps (2A), while the bottom half takes in twenty-four volts (24V) and outputs five volts (5V) with a maximum current of four amps (4A).

The Nvidia Jetson Nano has a power option of using a barrel connector with five volts (5V) up to four amps (4A). The Nvidia Jetson Nano is an expensive piece of equipment, so we need to be careful. If the microcontroller is damaged and becomes unusable, the cost of replacing that is a fraction of the cost of replacing the Nvidia Jetson Nano. So, if we aren't confident in this hardware design, we have to find another option of powering it.

6.2. ESP-WROOM-32

Using the ESP32 Hardware Design Guidelines along with the ESP32-WROOM-32 Datasheet we can get the schematic for taking the microcontroller off the dev board and implement the microcontroller into our own board design [92, 93]. The only notable exception comparing this design to the ESP32-WROOM-32 is the exclusion of the antennae. We probably did not need to use that. If we find a use for it, we can easily revise our board design to reflect that.

6.2.1. Processor

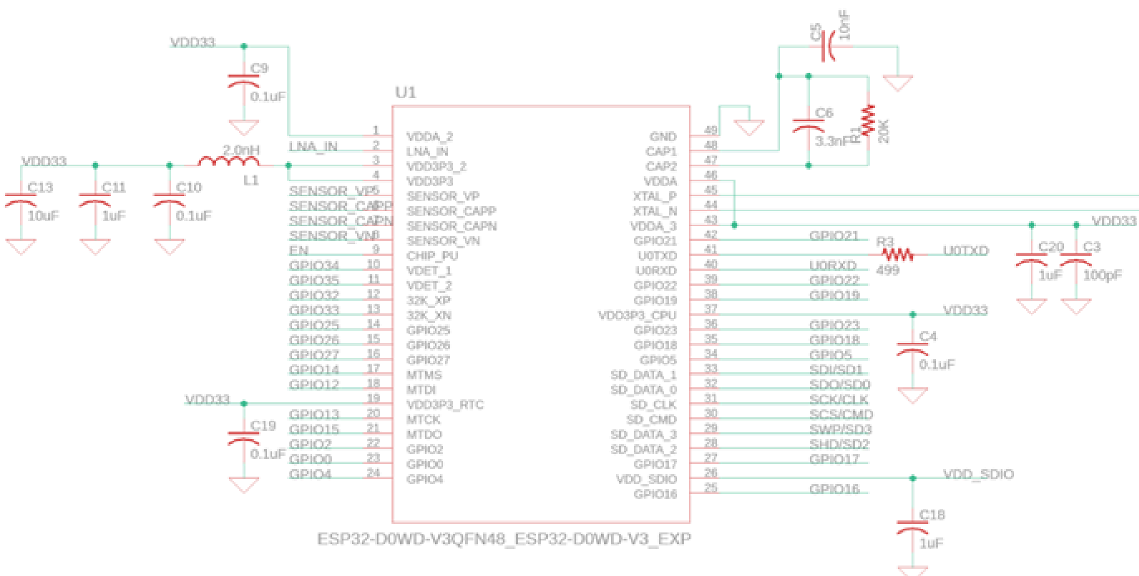


Figure 6.2.1.1: Schematic for the ESP32 Microcontroller.

The integrated circuitry of the ESP32 requires only twenty electrical components, a crystal clock, and SPI flash. All of the GPIO pins are present here, along with the power input for the microcontroller. They can be seen in Figure 6.2.1.1.

6.2.2 External Crystal Clock Source

The ESP32 firmware only supports forty megahertz (40MHz) crystal. Y1 in Figure 6.2.2.1 is the crystal clock. In the ESP32 Hardware Design Guidelines C1 and C2 capacitance values are listed as “TBD.” Those capacitance values can be calculated with this formula:

$$C_L = \frac{C_1 * C_2}{C_1 + C_2} + C_{stray}$$

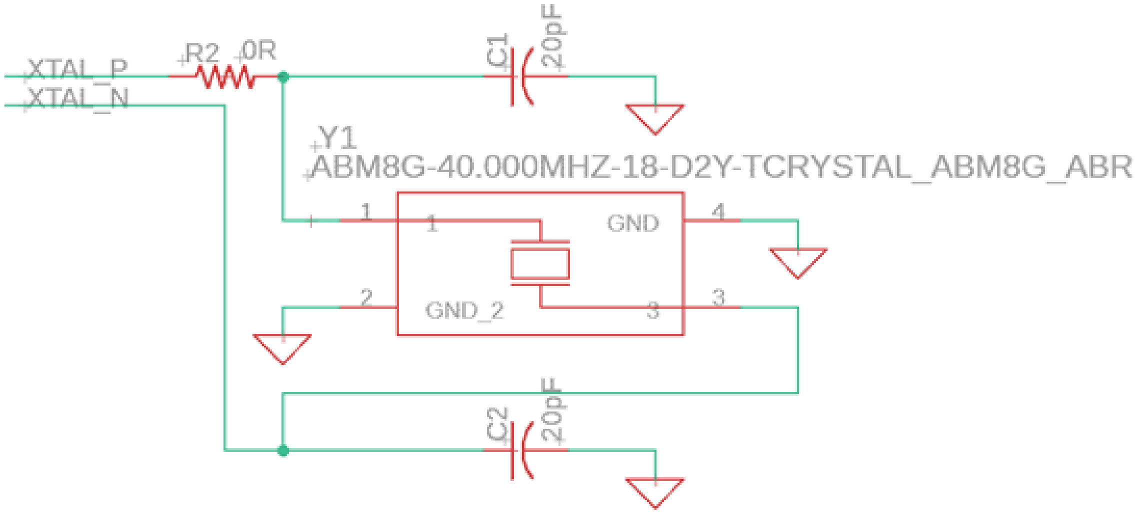


Figure 6.2.2.1: Schematic for the External Crystal.

C_L (load capacitance) can be found in the crystal's datasheet. According to the ABM8G's datasheet, the load capacitance is 10pF. Then we can get the C1 and C2 capacitance of 20 pF. Cstray is a little more complicated, C_{stray} refers to the PCB's stray capacitance. The datasheet lists a couple steps of finding C_{stray}. It involves using a radio communication analyzer or a spectrum analyzer to demodulate the TX tone mode to find the actual frequency offset. The goal is to adjust the frequency offset to be within 10 ppm by adjusting the external load capacitance.

6.2.3. Flash

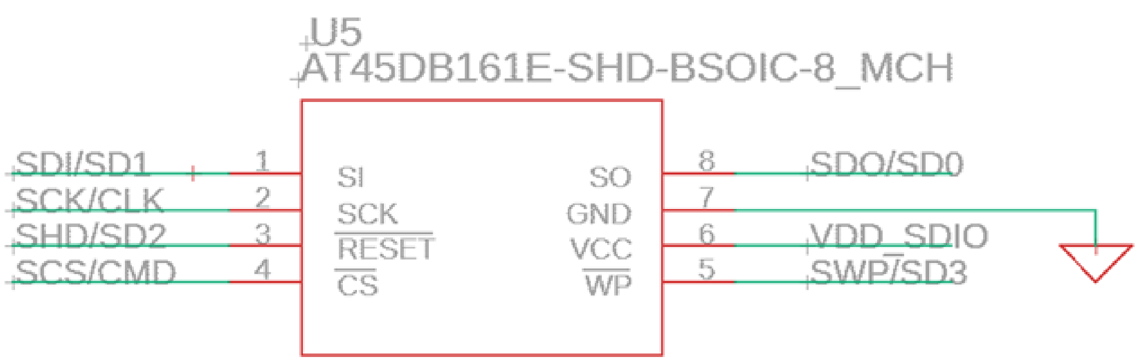


Figure 6.2.3.1: Schematic for In-Package Flash.

ESP32 requires in-package or off-package flash to store application firmware and data. The AT45DB161E is a sixteen megabit (16MB) SPI flash memory.

6.2.4. USB-UART and Switches

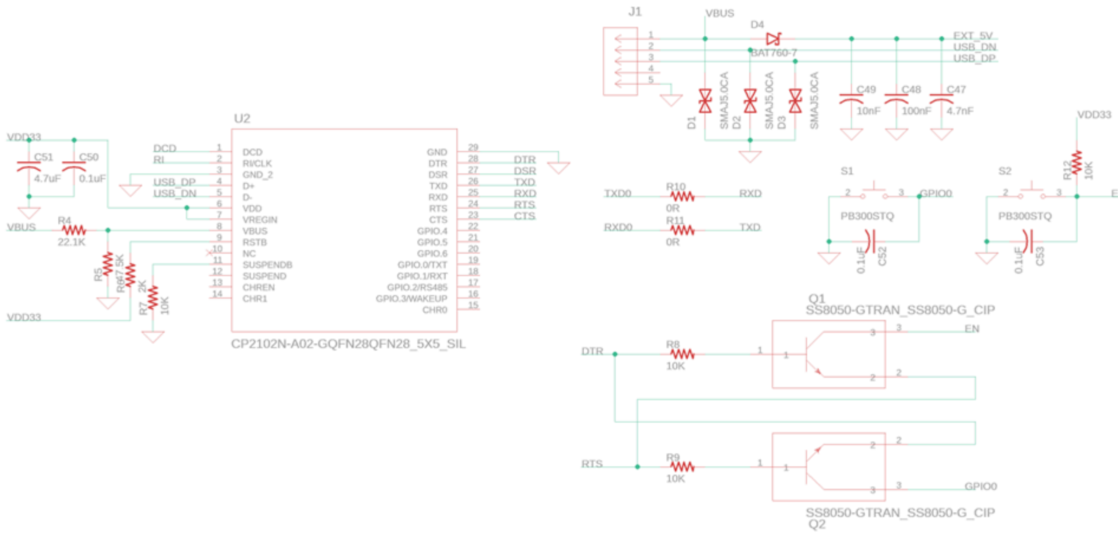


Figure 6.2.4.1: Schematic for USB-UART and Switches.

To program the ESP32, we need RS232C communication. Our development board has a USB to RS232 converter IC already included so our standalone board did need to have one of these RS232 converter ICs. Using a USB to UART Bridge IC accomplishes this. Additionally, the ESP32 needs two switches on pins GPIO0 and EN. Figure 6.2.4.1 shows the schematic of the USB-UART IC and connections with the Micro USB port.

6.3. GPIO Peripherals

There are a total of twenty-two analog GPIO pins on the ESP-WROOM-32. LEDs did require at least one of these pins. The Front-Facing Infrared Sensor did require another. The AI Development Kit won't require any, but it did use the I2C connection - a peripheral connection. The fans did likely use one each, totalling to two more analog GPIO pins. Next is the screen which needs a staggering six GPIO pins. Following that, the Infrared TX and RX pairs need two pins per pair, totalling to another eight pins in use. Finally, the motors need four GPIO pins. In total, this means that we are using exactly twenty-two out of twenty-two of the analog GPIO pins for the device.

This means that if something goes wrong with the board, there is no opportunity to re-solder to a backup connection. Instead, we had to solder everything to a wholly new backup board. This is perfectly acceptable, but it's something we must keep in mind while moving forward.

6.3.1. LED Design

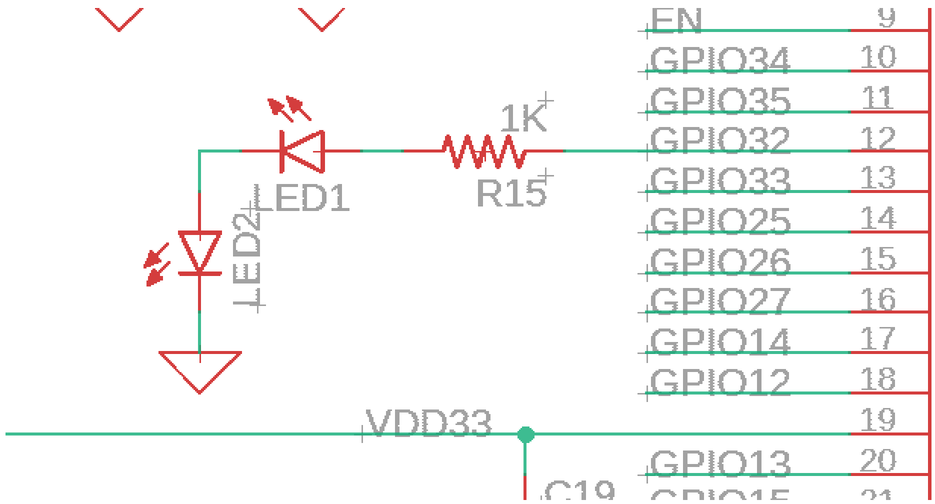


Figure 6.3.1.1: Cover LED Placement and Wiring.

The simplest GPIO use case is using it for LEDs. We want the inside of the machine to light up when the machine detects a potential customer, then to power down after a significant amount of time to save power. With just a single analog GPIO pin, we can accomplish that with a chain of connected LEDs.

6.3.2. Front IR Sensor Subsystem

When it comes to the infrared designs within the Hand Gesture Vending Machine, there are two independent subsystems. The first of which, the one that acts as the activator for the rest of the machine, is the front-facing infrared sensor. The E18-D80NK Adjustable Infrared Proximity Sensor is our sensor of choice, and it has three wires with which we connect to the power supply and microcontroller. These wires are an orange five volt (5V) input voltage cable, a blue ground cable, and a black signal cable.

The operation of this sensor is fairly intuitive. When powered with the five volt (5V) and ground cables, the signal cable sends out a pulse of power around three volts (3V) strong. This is acting as an idle signal, showing that it is running without a target entering its range. However, when something enters the range of the reflected infrared signal, the signal cable instead stops sending power through altogether.

Powering it did not be done through the microcontroller; simply put, it isn't necessary. The front-facing infrared sensor did be operating at all times while the device is plugged in, and wiring it up to the microcontroller runs a risk of taking too much of the five volt (5V) output pin current when taking directly from the power supply voltage regulators is an option. The black signal cable, however, was connected with the ESP32-WROOM-32 microcontroller to act as the signal to begin communicating with the AI Development Kit.

6.3.3. AI Development Kit Connections

The I2C communication protocol uses two wires to share information. One for the data line (SDA) and one for the clock line (SCL). For the ESP32, the GPIO Twenty One and GPIO Twenty Two are the default for SDA and SCL respectively. For the Nvidia Jetson Nano, the GPIO Three and GPIO Five are the default for SDA and SCL respectively. Connecting them should allow us to establish I2C communication.

6.3.4. Fan Subsystem

When it comes to the fans, we are currently undecided on one of three design options. The first of the three is using a singular fan, wiring it directly to the Nvidia Jetson Nano instead of the microcontroller. The second is, instead of connecting it to the Jetson Nano, we can build the PCB design with two fans as a part of the connectors available, allowing for increased airflow. The third option is to only have data connections to the microcontroller, wiring up the power supplies through the PCB directly to the fans to minimize microcontroller strain.

When it comes to the first option, the AI Development Kit can handle the fan without difficulty. However, the concerns about connecting up to there are twofold. The first concern is that the fan was limited to only operating while the Jetson Nano is operating; this is a severe limiter in how effective the fan can be, as when the device turns off, so too does the fan cooling it.

The second option, on the other hand, requires both of the fans to be taken into account and wired into the PCB, then set into place within the frame of the Hand Gesture Vending Machine. This would require the implementation of two analog GPIO pins and both power and ground connections to be wired into the board, which the fans would then connect to with their plugs for operating purposes. This allows for the fans to run at all times, which is the largest concern when it comes to the first option available to us. However, this particular fan placement has its own concerns regarding current needs and more complicated board design from using some of the limited number of analog GPIO pins; for the former case it's a matter of not wanting to burn out the microcontroller's five volt (5V) output pin, and for the latter case it's a matter of multiplexers being used in the board design.

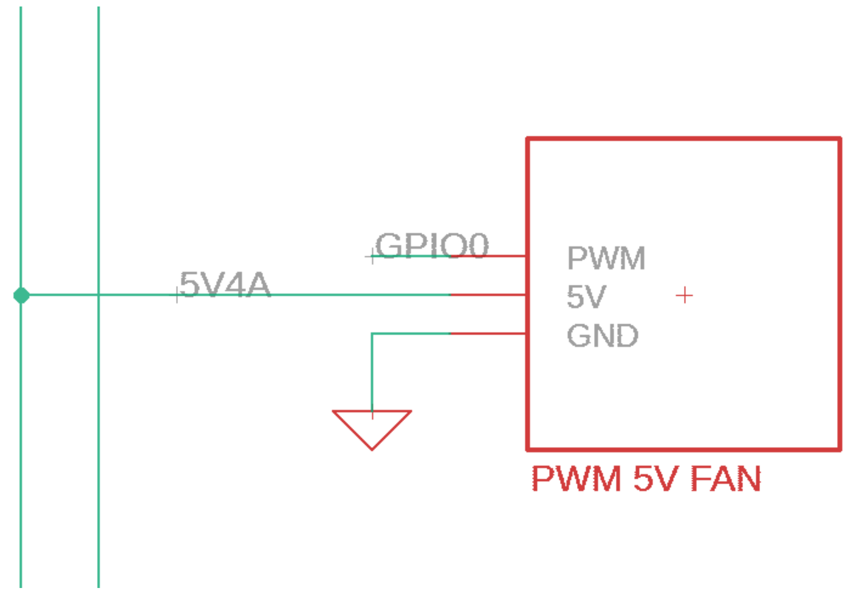


Figure 6.3.4.1: Visualization of a Simple Schematic for a Fan. Note the connection to a five volt (5V) rail and the analog GPIO Pin.

Then there's the third option, which is an alternative option to the second. Much like the second option, there would be two fans using the analog GPIO pins on the board. Unlike it, however, these fans did instead be powered in a separate line marked on the board itself from the outputting voltage regulators, skipping the ESP-WROOM-32 microcontroller entirely in the process. This still retains the analog GPIO pin issue the second option has, but it removes the concerns of too much current running through the microcontroller to try and power the fans alongside everything else. It also, much like the second option, still allows for the fans to be powered at all times, keeping the AI Development Kit cool regardless if it's in use or not.

We did, considering all of these options, most likely go for the third one. It covers the most issues while only retaining use of the analog GPIO pins - which we had just enough of to cover for every single component. With the concerns regarding these pins dealt with, the only other concern would've been over-drawing current from the ESP-WROOM-32. The third option, however, side-steps this issue in its entirety by wiring into a rail directly powered by the voltage regulator. This also means that we retain the higher overall airflow around the AI Development Kit, which did keep it and the other components safe from risks of overheating.

6.3.5. Screen Design

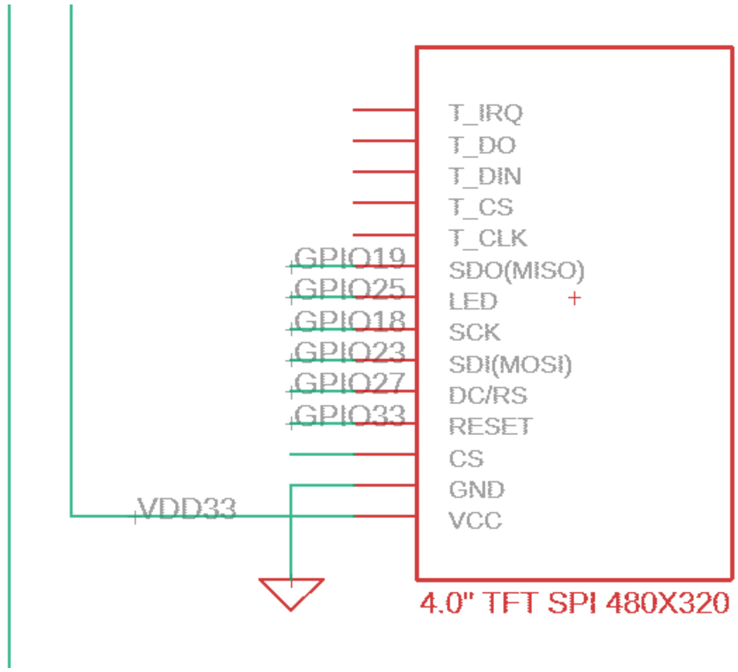


Figure 6.3.5.1: Simple Schematic of the Important Pins from the Screen for the PCB Design. Note that six of them are analog GPIO Pins.

The screen is the component that did use the most GPIO pins. Right off the bat in Figure 6.3.5.1, the pins used for the touch screen did not connect to anything as we did not need a touch screen. That would completely invalidate what is supposed to be a touchless machine. The challenge of picking out what GPIO pins are delegated to the screen comes down to which pins are able to input and output, and what pins are being used by other peripherals. Then the software was used to assign what part of the screen the pin is connected to. The manufacturer has supplied libraries and examples to help get the screen up and running on many different devices.

6.3.6. Internal IR TX and RX Subsystem

The infrared transmitter and receiver are simple three pin modules. As the machine has four options we did need to use four sets of modules. One pin is for power, one pin is for ground, and the last pin is for data. Meaning that we did need to use a total of eight GPIO pins for four inputs and four outputs. If we find we have run out of usable pins, we have to find and use a multiplexer to cut down on the amount of pins they use. Figure 6.3.6.1 uses GPIO pins that we could use. For the ESP32, pins 32, 34, and 35 can only be used as inputs so they are assigned to the transmitters.

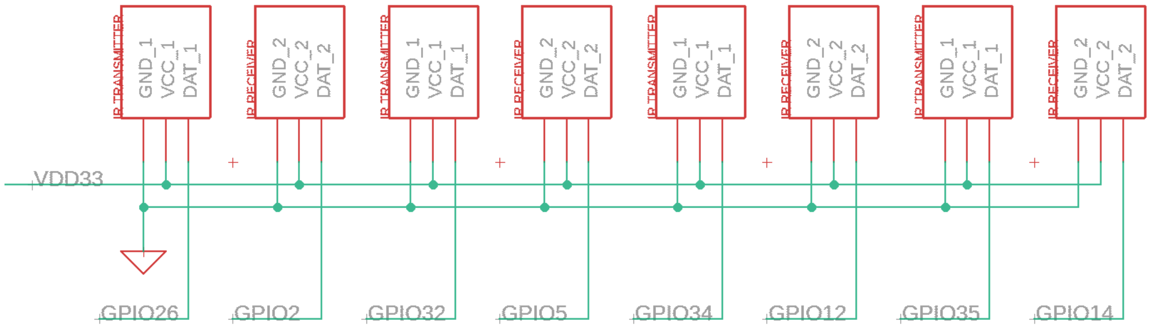


Figure 6.3.6.1: Schematic of the transmitters and receivers, using 8 GPIO pins.

6.3.7. Motor Design

Unlike many of the analog GPIO pin-using components thus far, the motors don't actually have any data pin for the microcontroller to communicate with for speed and power control. Instead, the G12-N20 Geared Mini DC Motors operate at a consistent pace controlled by their in-built gearboxes. As mentioned in Section 3.3.4., the torque is a nonissue for these motors because of this. However, an issue has indirectly arisen due to this: we cannot easily tell the motors when we want them to run. Controlling the vending process would be impossible without a real way to control the motors, so this is an unacceptable outcome.

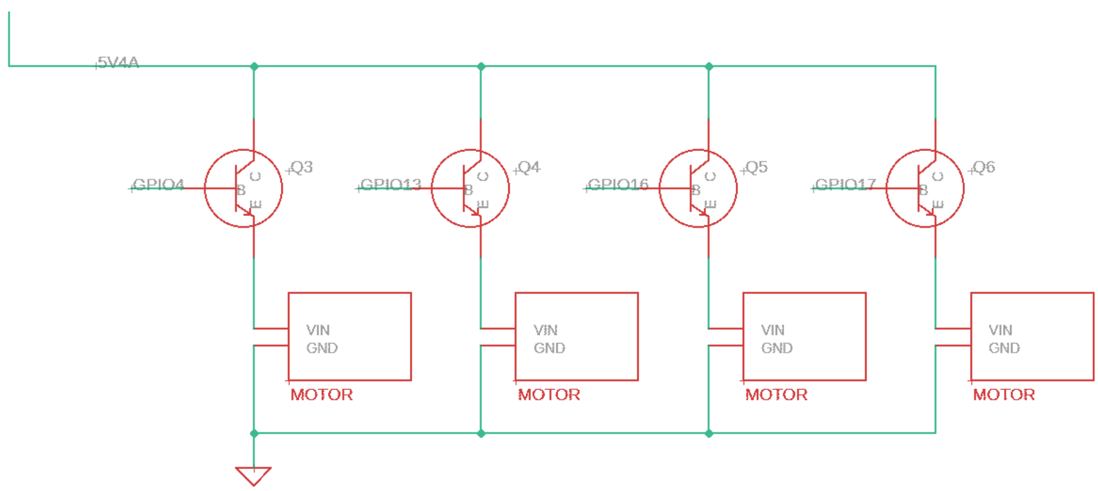
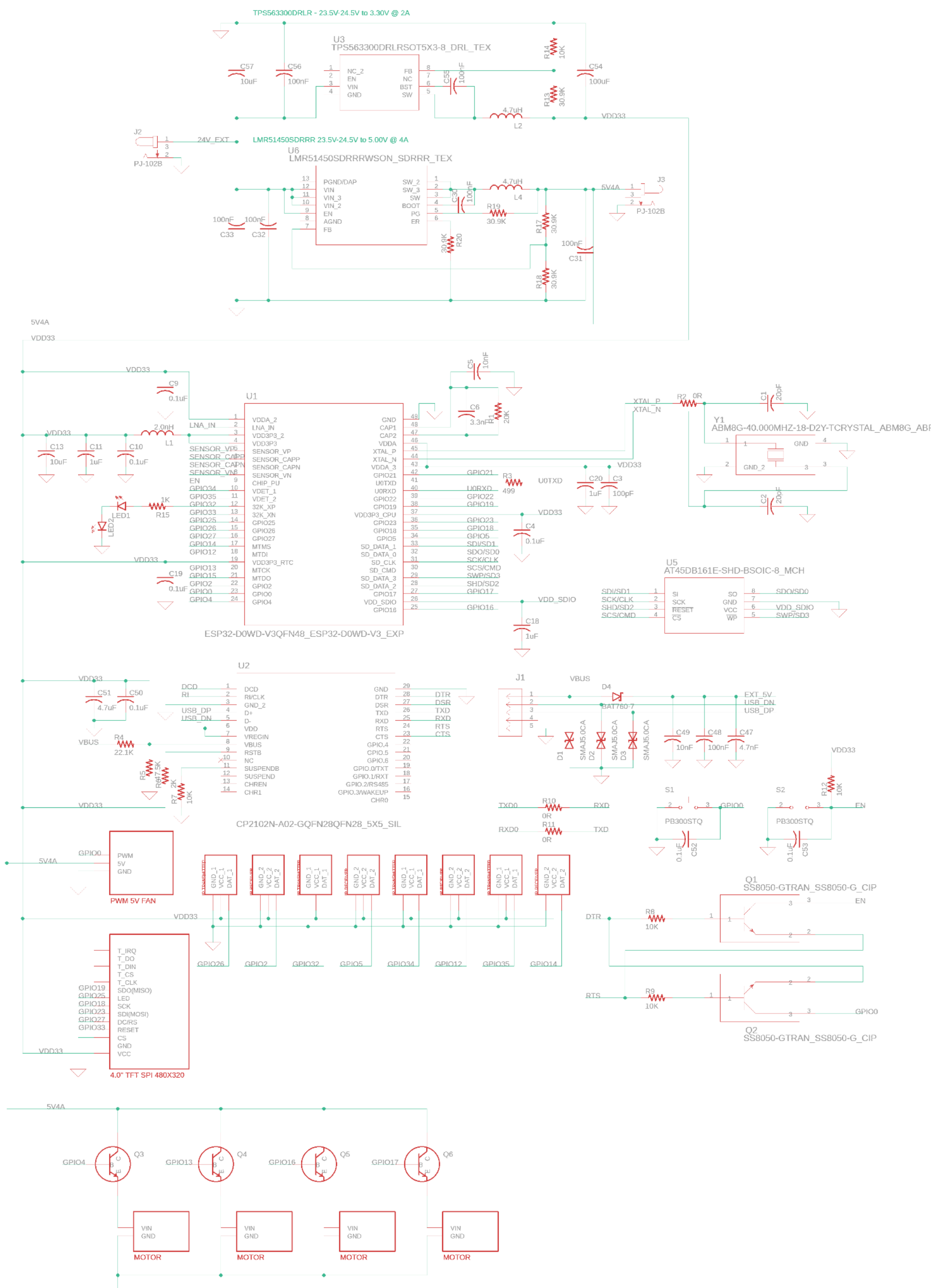


Figure 6.3.7.1: Visualization of the Motor Schematic. The microcontroller did control the switches to power them via analog GPIO Pins.

Our solution for this conundrum is relatively simple, all things considered, but did burn through yet more of our limited analog GPIO pins. As seen in Figure 6.3.7.1, connecting to the Voltage Input (VIN) pins for each of the motors was a set of pin-operated NPN gates. These gates did lock out voltage from the motors until the microcontroller sends a signal to them via analog GPIO pin, which did open the gate and allow voltage back through - and in doing so, power the motor and allow it to spin. This can then be closed back up to stop the motor, completing the vending process.

6.4. Full Schematic



6.5. Shell Design

While there is significant discussion regarding the individual hardware components and how they all come together, the casing that did hold all of these components is similarly important. Both when considering the proof of functionality and the professionalism in an aesthetically pleasing project, having the actual metal 'shell' frame for the Hand Gesture Vending Machine is important. However, finalizing a design for the external metal shell and the internal pieces to fill out the framework has been more difficult than anticipated, mostly due to the issue of working out space availability for the various components and the snacks.

The actual design in the abstract is fairly simple. Using a scaffolding of rounded square metal bars welded by a professional, additional sub-scaffolding would be welded as needed for the various components before metal sheets would be bolted on across most of the frame - excluding one side and the front. For the front, there did need to be a glass equivalent put into place for consumer observation, and for the side, there must be an access point for maintenance and showing that the electronic components of the proof of concept can comfortably fit within.

A vast majority of the electronic components were fitted into the right side of the Hand Gesture Vending Machine, much like they are with regular vending machines. With the additional sub-scaffolding and the setting of both thermal pads and slots for the components to either be snapped, slid, or screwed in, we can give all of the pieces both proper space and safety while moving or jostling the project from place to place. With the side panel access point, these components can also be just as easily removed for safe debugging and demonstration.

For the more decorative internals, there did likely be 3D-printed plastic to minimize any additional weight of the design; much of it, sans where the majority of the electronics did go, is more to ease the path of the snacks - or acting as the spirals in the place of traditional metal ones. Even the generic metal spirals for vending are unreasonably expensive, not to mention the fact that most of them simply would not fit the tiny motors we're making use of. As such, we did use 3D-printed plastic spirals for the shell design.

7. Software Design

7.1. ESP32-WROOM-32

During the testing of the microcontroller, it was acknowledged the many different problems being run into. Not having examples for things we needed hindered us tremendously in time, causing a lot of wasted time having to do extra research on certain things. One of the biggest things to come out of testing was realizing that the ESP32 would be a great substitution for our previous microcontroller, the STM32F429ZI. It was realized that the ESP32 would be a better board for this project once we realized how much online examples, documentation, and support was offered with this board. We needed certain things done like functioning fans, motors, LEDs, and LCD screens.

The ESP32 is the main microcontroller now. The software design behind the microcontroller was designed to be activated by the ESP32. The microcontroller did connect to our LCD screen to output a user interface for the hand gesture vending machine. The connection made between the microcontroller and the LCD screen was made through SPI.

Connections between the Jetson nano and the ESP32 were made, which were for running the Machine Learning algorithm to compute the hand gestures via object detection. Other connections such as the LEDs, fans, and IR sensors were also connected via analog GPIO pins. Each connection made between the component and the microcontroller is important to fulfilling each task; from the fans running when the ESP32 is turned on, to when the front IR sensor detects a motion, and to when the Jetson Nano is told to activate the connected camera, all of that is only possible with making use of these connections.

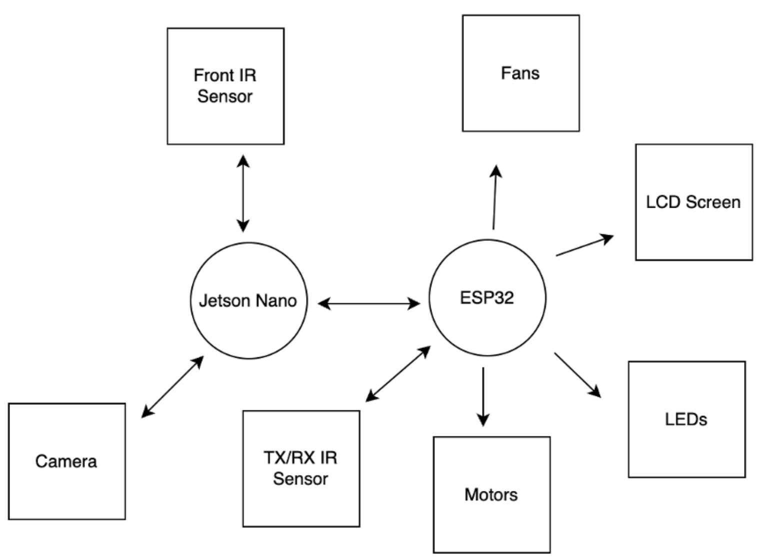


Figure 7.1.1: Microcontroller Component Connections Layout.

7.2. LED Software Design

For the LEDs to work correctly they did need to be programmed accordingly and connected to the chosen microcontroller. By programming the microcontroller, it was able to allow the LED to light the inside of the hand gesture vending machine. The LEDs we're looking to use are white LEDs. These LEDs did be able to brighten the entire vending machine, being able to display the different products inside of the machine. The software running on the MCU did include LED control functions to turn the LEDs on or off and adjust their brightness. These functions did be called based on specific events or conditions detected by the vending machine, such as a user hand gesture, vending operations, or system status.

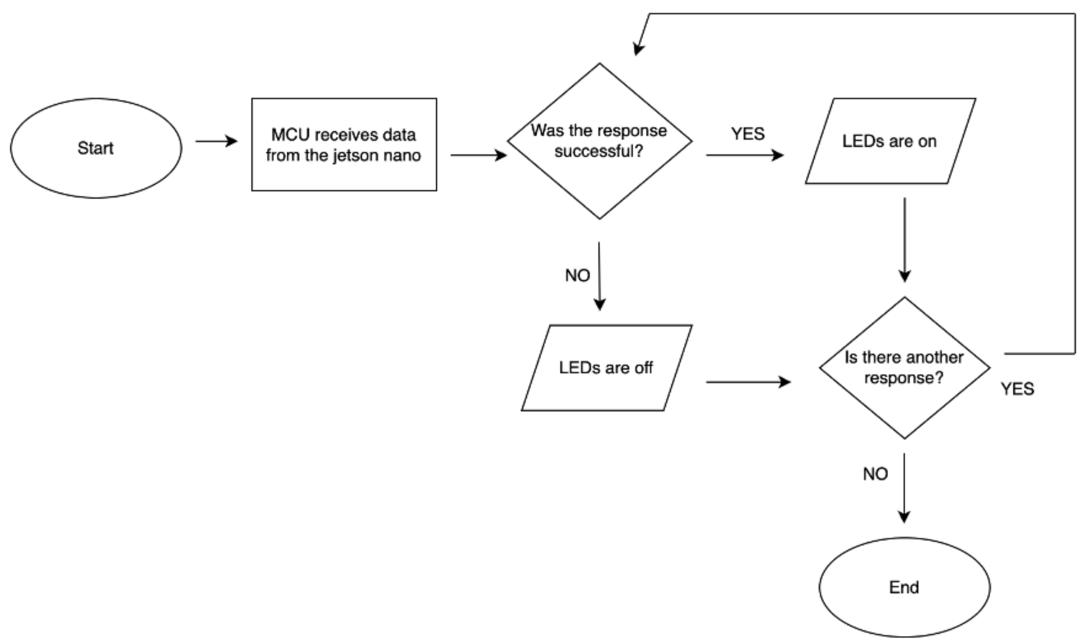


Figure 7.2.1: LED Software Flowchart.

7.3. Front IR Sensor Design

The front IR sensor did play a crucial role in the development of the hand gesture vending machine. The way the front IR sensor did work is by once it has detected that there is a gesture the machine did then turn on to start the machine learning algorithm.

As mentioned before the front IR sensor did detect a gesture before beginning the program. The way the software is designed for the front IR sensor is it does read high for when no gesture is being detected. It does read low when an object is being detected. When there is an object that is being detected this does signify with the microcontroller to know when to communicate with the Jetson nano.

When the front IR sensor can detect a gesture, this does then notify the Jetson nano to then power on, as well as the camera. Once the system is on it is then able to run the machine learning algorithm to then detect a hand gesture for choosing the item(s) inside of the hand gesture vending machine. The front IR sensor did act as a first responder to the rest of the machine, once a gesture is detected from the front IR sensor then the software did turn on corresponding components.

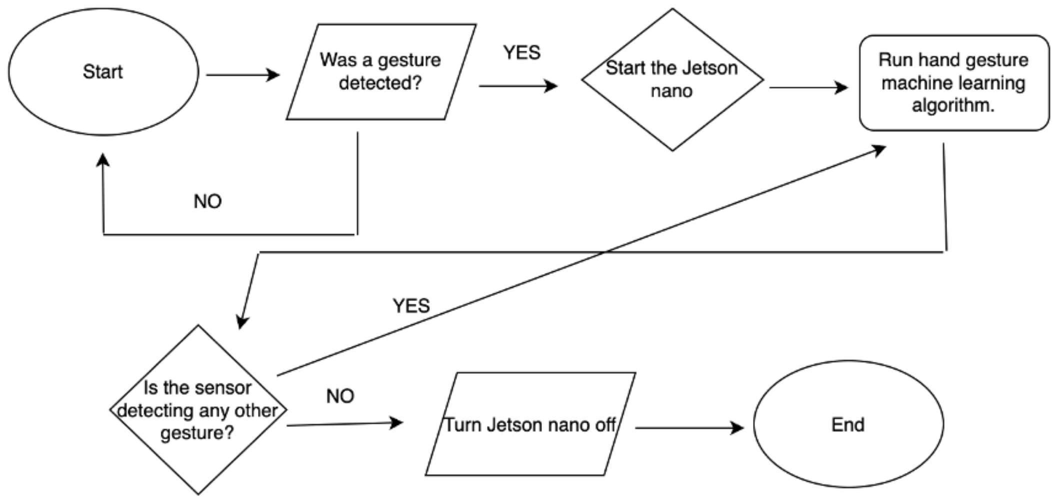


Figure 7.3.1: Front IR Sensor Flowchart.

7.4. LCD Design

The LCD screen software for the hand gesture vending machine did involve controlling the LCD display to provide visual information, feedback, and even user prompts. The software running on the microcontroller was initializing the LCD screen at the startup. Although the screen we do have is touchscreen there did not have any touchscreen capabilities incorporated into this project since this is a hand gesture vending machine. The vending machine has an idle state and active state. While in idle state the screen did be off until in active state to where the display turns on once the camera/sensor is able to detect an object.

The GUI was the graphical user interface for the hand gesture vending machine. It did allow for users to interact with the LCD screen using graphical icons and visual indicators, as opposed to text-based interfaces. A GUI did use menus, windows, and icons to carry out tasks, making it easier for users to operate the electronic devices, like computers, smartphones, and other devices.

While in idle state, it should display relevant information to attract potential users and provide guidance on how to use the hand gesture vending machine. The screen did prompt the user at a welcome screen on the display, followed by instructions. The LCD screen did display a welcome message, making it clear what the machine is and inviting users to interact with it. For example, the display could say, “Welcome to Hand Gesture Vending Machine”. The follow up instructions did be clear instructions displayed to guide users on how to use the vending machine. Instructions were concise, displayed prominently, and easy to understand.

Once in active state the LCD screen should provide feedback and guidance to users as they interact with the hand gesture vending machine. There was gesture detection feedback, allowing for when a user approaches the vending machine and makes a gesture the LCD screen provides feedback to acknowledge the gesture. An example of that can be “Gesture was detected! Please wait”. On the product selection screen once the user’s gesture is detected the LCD screen does display available products, allowing the user to select.

After a product is selected, the LCD screen then shows the user the selection and prompts the user to confirm the purchase. This confirmation prompt can be along the lines of, “You have selected Snack C. Confirm purchase.” At the end of this process, the LCD display did confirm the transaction success, the LCD screen did display a success message. The message could read, “Transaction was successful. Please grab your item.” One of the last things the LCD screen has is error handling. If any case of errors or malfunctions are detected by the vending machine during the active state, the LCD screen did display an appropriate error message to alert the user.

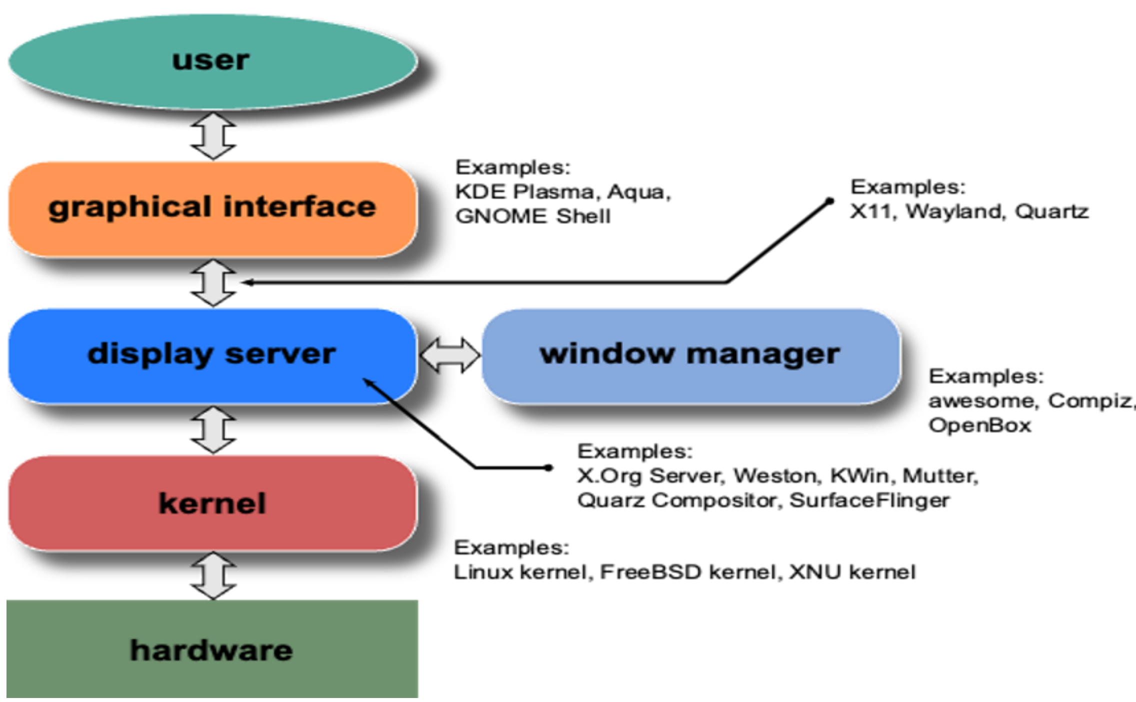


Figure 7.4.1: Layers of a GUI based on a Windowing system.

7.5. Fan System Design

The fan system for the hand gesture vending machine is yet another important piece of a component needed for the success of this project. The job of the fan(s) was to have air circulation inside of the vending machine unit in order to keep the insides of the various components cool. Having air circulating through the vending machine allows for components to not get overheated, especially the Jetson Nano. The Jetson Nano did run heavy and fast machine learning code needed to detect the hand gestures from the user.

The design for the fan system was based upon how we decide for the fan to operate. We can either set the fan up through the Jetson Nano or using the microcontroller. In terms of using the Jetson Nano with the fan (CFM-A225C-115-287-20) it did be a matter of just configuring connections, this did require minimal to no code. Setting up the fan with our ESP32 did require analog pins connection and coding. There are a few online resources for the software used to set up and control the fan. The fan system design using the ESP32 would consume less power compared to the Jetson Nano.

The design behind the fan system did fall on the code needed for it. The fan did run for as long as the hand gesture vending machine was functioning. When the vending machine is on and powered the fan is constantly running to provide air circulation and to make sure components are not overheating inside of the vending machine unit. Once the front IR sensor can pick up the initial gesture, the Jetson Nano then activates to start the machine learning algorithms while the fan is running. As stated, before the Jetson Nano did run very fast machine learning programs so it did need to be able to function correctly while.

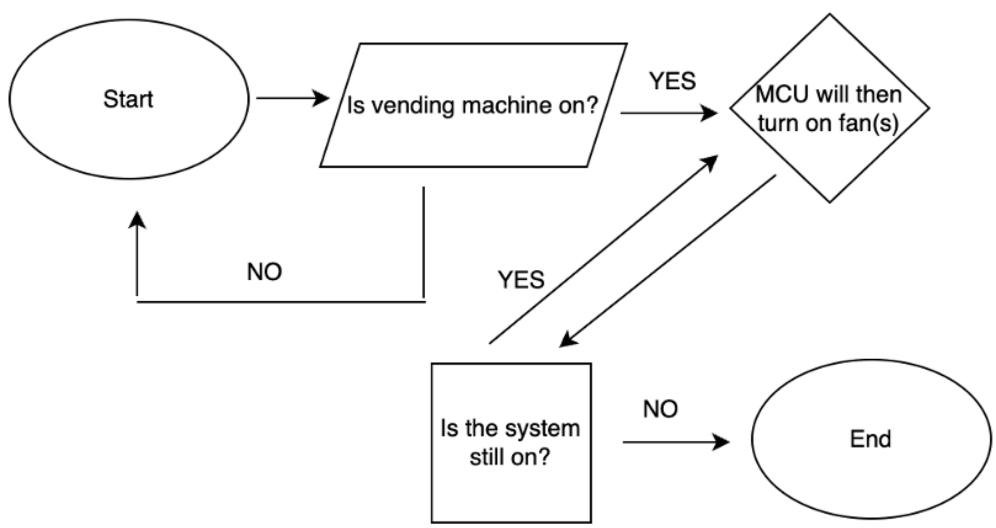


Figure 7.5.1: Fan System Flowchart.

7.6. Object Detection Design

The camera and object detection system for the hand gesture vending machine are very important components directly tied to the success of this project. The object detection system's job is to be able to distinguish between multiple different set hand gestures including one, two, three, four, a thumb up, and a thumb down. The camera is the eyes of this system; the way that all the hand gestures were captured did be on this camera.

After the front IR sensor detects a person, the system, including the camera and therefore the object detection, will be turned on. The user did be prompted to make an accepted hand signal and after the user does this, the accepted hand signal did be converted into a signal and this signal did be sent to the microcontroller to produce the corresponding output whether that be turning the motors on, restarting this process, or making the vending machine turn off.

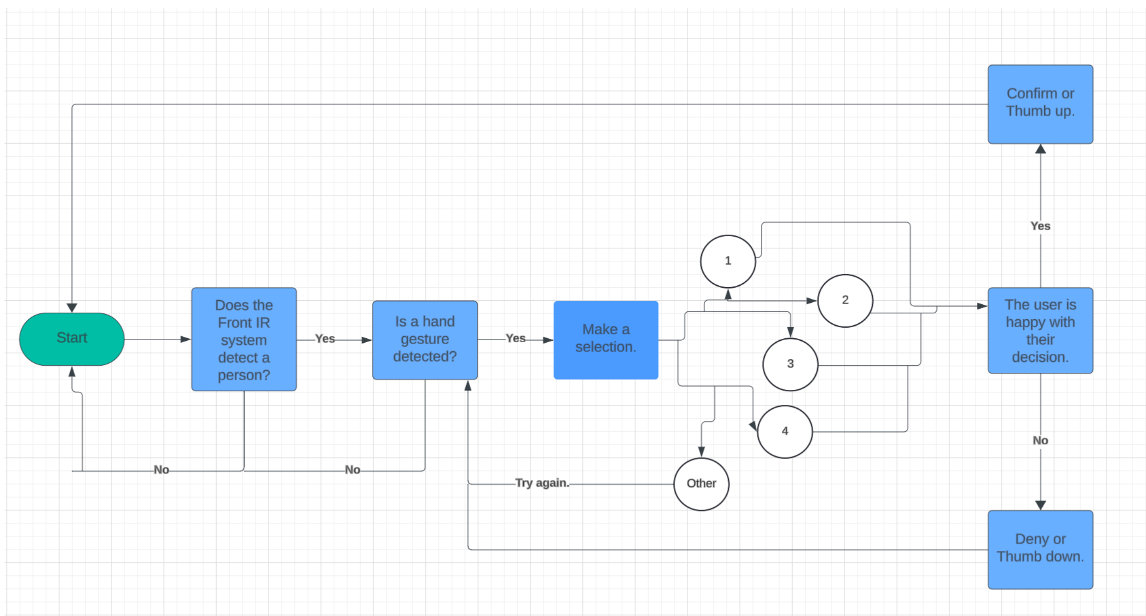


Figure 7.6.1: Object Detection Flowchart.

7.7. TX/RX IR Sensors Design

The original software design concept we had for the TX/RX IR sensors involved the STM32F429ZI. The concept involved programming the microcontroller to enforce for the transmitter to send a high signal to the receiver and if it does for it to blink a red LED. There were a few issues that arose such as limited online examples, and too many GPIO pins just to name a few. Changes had to be made.

With some research we found that the best microcontroller for our project was an ESP32-WROOM-32. This board has enough pins needed for the success of this project. With a CPU speed of 2.4 GHz and supporting Arduino IDE, this allows for us to utilize the online resources related to the TX/RX IR Sensors. The ESP32 has much documentation and a community of people to help online.

The TX/RX IR sensors software design with the ESP32 is with the purpose of checking for a product's availability. If there is an item available, the receiver will be blocked by one of the products. If the product is not available, the receiver did catch the transmitter's signal and inform the ESP32. To achieve this the code did require some if statements with conditions in regards to whether or not there is an available product.

7.8. Motor Design

The motors used for this software design were the G12-N20 Geared Mini DC Motor. With a torque of two kilogram-centimeters and a stall torque of sixteen kilogram-centimeters these motors did be able to move the vending spirals with little trouble, as they would barely weigh a single kilogram.

The software design for the motors did also probably require a motor driver. The motor driver allows us to drive two DC motors simultaneously. With the code required to run the motors there are a few things we're looking to do, control speed and direction. By declaring motor pins we'll then be able to manipulate the motors. With some line of code we'll be able to move the DC motor forward at a maximum speed, stop the DC motor, and move the DC motor forward with increasing speed.

7.9. Software Used

Software	Description
Arduino IDE	Arduino IDE is an open-source integrated development environment used to write and upload code to Arduino and compatible microcontroller boards. It simplifies the process of code writing for Arduino projects. This was used for the programming of the ESP32 microcontroller using the C language.
Autodesk Eagle	Autodesk Eagle is a PCB design software widely used for designing schematics and circuit boards. It provides a comprehensive, easy-to-use suite of tools for PCB design.
Diagrams.net	Diagrams.net is cross-platform graph drawing software developed in HTML5 and JavaScript. Its interface can be used to create diagrams such as flowcharts, wireframes, UML diagrams, organizational charts, and network diagrams. Some of the flowcharts were created using this software.
Draw.io	Draw.io is much like Diagrams.net, and was also used in earlier flowchart creation.
Solidworks	3D Modeling software to use for 3D printing.
STM32CubeIDE	STM32CubeIDE is a free cross-platform Integrated Development Environment (IDE) to configure, develop, build, and debug STM32 microcontroller applications. It simplifies STM32-based development. This was used during the first half of our project when we were using the STM32F429ZI.
Ultra Librarian	Ultra Librarian is a software used to create and manage CAD models of electronic components. It provides access to a vast database of electronic components for use in PCB design.
VS Code	Visual Studio Code is a source-code editor developed by Microsoft. It includes support for debugging, embedded Git control, syntax highlighting, intelligent code completion, snippets, and code refactoring. Visual Studio was used to create a GUI, build our website using HTML and CSS, and for machine learning algorithms.

Table 7.9.1: Software used to create the Hand Gesture Vending Machine.

8. Optical Design

For the Hand Gesture Vending Machine, there are several optical components that are important to talk about and put into position. Chief among them is the lens system, which is critical for maximizing the efficiency of the object detection's graphical processing power. Following that are the front-facing infrared detector and the internal infrared transmitters and receivers, which both are similarly important - though debatably less so than the lens system proper.

8.1. Camera Lens System Design

The lens-based optical system attached to the camera is intended to make it so the small, sub-centimeter sensor can efficiently observe hand gestures while minimizing wasted pixel space and maximizing the size of the image. By cutting down on wasted pixels, the Object Detection did operate faster than it normally would, and by maximizing the image size it should make it easier for the Object Detection to determine the nuance of the gesture.

To help, the lens system was mounted near the top right corner of the device - to the perspective of the observer - to allow the consumer to simply raise their hand while looking into the Hand Gesture Vending Machine rather than having to awkwardly position themselves elsewhere. Additionally, this allows for most people to easily lift their arm at the elbow rather than at their shoulder to get their hand within range of the camera.

8.1.1. The Lenses

Within Section 3.2.2., the decision regarding the material of the lenses themselves was made and selected as N-BK7. However, functionality details regarding this material were limited within that particular section for the sake of discussing the design in more detail here. Below is a set of tables to show the exact properties of NBK-7, as well as the specific lenses that we have come to use for our final lens system design.

Common Specifications (Uncoated N-BK7, A-Grade) [12]	
Wavelength Range ($\Delta\lambda$)	350 nm - 2.0 µm
Design Wavelength (λ_{Peak})	633 nm
Index of Refraction (n)	1.515 (@ 633 nm)
Surface Quality (Scratch)	40
Surface Quality (Dig)	20
Surface Flatness (Plano)	λ / 2
Surface Curvature (Concave/Convex)	3 * λ / 2
Surface Irregularity (Peak to Valley)	λ / 4
Abbe Number	64.17
Focal Length Tolerance	± 1%

Table 8.1.1.1: Mathematical Specifications for Uncoated N-BK7 Optical Lenses.

Item ID	Manufacturer	Lens #	Diameter (mm)	Lens Type	Focal Length (mm)	Back Focal Length (mm)	Radius of Curvature
32-014	Edmund Optics	1	12	Bi-Convex	12	10.79	15.34
48-682	Edmund Optics	2	9	Bi-Concave	-18	-18.49	-18.86
45-292	Edmund Optics	3	9	Plano-Convex	45	43.98	23.26
32-968	Edmund Optics	4	9	Bi-Convex	+18	17.12	18.15

Table 8.1.1.2: Table of Lenses to be used in the Hand Gesture Vending Machine.

8.1.2. Optical Design Formulae

The fundamentals of lens system design lie within Geometric Optics. Or to be more specific, they lie within the optical design equations which came together from the study of Geometric Optics. The study of Geometric Optics is primarily interested in the formation and modification of the image formed from an object through optical mediums like lenses, apertures, and more.

Most critical to lens system creation is the Lens Maker's Equation. The Lens Maker's Equation is a relatively simple formula that relates the physical properties of a lens to Focal Length, finding one or the other based on available data. Every lens system makes use of this formula, the data critical to ray tracing the final design. The formula is as follows:

$$\frac{1}{f} = (n - 1)(\frac{1}{R_1} - \frac{1}{R_2})$$

Where f represents the Focal Length of the lens, R_1 and R_2 represent the respective Radii of Curvature of the two Lens Surfaces - in the order of light impacting it - and n is the Refractive Index of the lens. We can then use the calculated Focal Length to find the position of the image related to the object, via the Image Position Equation.

The Image Position Equation, known as the Gaussian Lens Equation, is used to locate the Real Image relative to the Position of both the Object and the Focal Length of the lens itself. This equation is fundamental to calculating image formation for just about every optical lens system, as without knowing the location of the Real Image, there is no way to know where to position the Object and lens without guesswork. The formula is as follows:

$$\frac{1}{S_1} + \frac{1}{S_2} = \frac{1}{f}$$

Where f is the Focal Length of the lens, S_1 is the distance from the Real Object to the lens, and S_2 is the distance from the lens to the Real Image formed by said lens. There are many uses for this simple and intuitive formula, but for our purposes we did only make use of it technically for two reasons. The first reason is to help position the initial lens with the camera. Following that, the second reason is for the final system's Focal Length, following the final system calculations and lens placements.

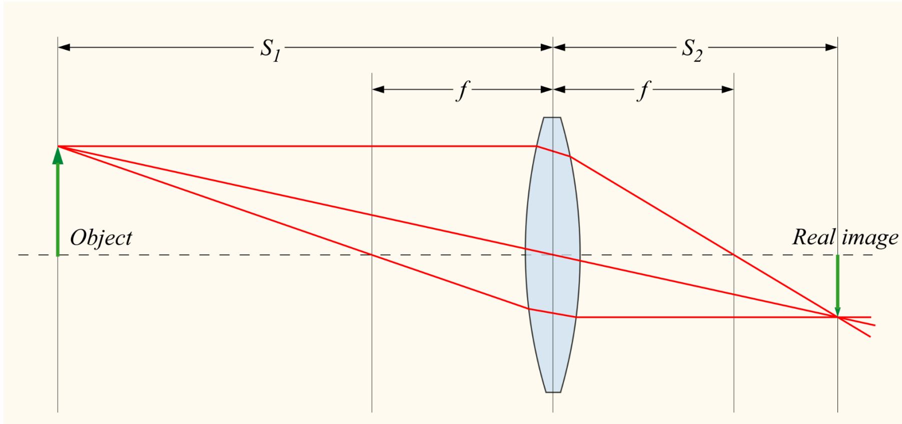


Figure 8.1.2.1: Visualization of the Gaussian Lens Equation.

The final equation particularly significant to the optical lens system design for the Hand Gesture Vending Machine is the Magnification Equation. When an Object is observed through a lens system, the resulting Real Image is rarely the same scale as the Object itself; this is due to the shifting of light via the different Refractive Indexes and the Focal Length(s) of the lens(es) in use. This is how a magnifying glass offers a close-up view, as well as how telescopes and microscopes are both capable of incredible zoom - but on different scales and for different purposes. The Magnification Equation makes use of the relationship between the distance between the Real Object and lens, as well as the distance between the lens and the Real Image in order to calculate the Magnification Factor. The formula is as follows:

$$M = \frac{-S_2}{S_1} = \frac{f}{f - S_1} = \frac{h_i}{h_o}$$

Where M represents the Magnification Factor - which can be either positive or negative - f is the Focal Length of the lens, S_1 is the distance from the Real Object to the lens, S_2 is the distance from the lens to the Real Image, h_i is the Image Height, and h_o is the Object Height. For our purposes, we actually know our Magnification and a range for S_2, and are instead working backwards from M towards what S_1 would equal within the range of desirable S_2 values.

8.1.3. Preliminary Data Calculations

From the offset, we had a list of constraining data parameters that influenced the final optical lens system design. The Hand Gesture Vending Machine, as it's being designed with the idea of actually functioning as a vending machine in mind, would specifically be operated within a limited distance range. As such, various values were put together from both available empirical data and general personal experiences with vending machine use.

Data	Data Type	Value	Source/Reasoning
Operating Range ($S_{2,\text{ Final}}$)	Length	~25 cm - 50 cm	Personal Experiences
Object Height (h_o) [Camera Size]	Length	6.45 mm (Vertical) 3.63 mm (Horiz.)	Camera Datasheet Calculation [46]
Image Height (h_i)	Length	~17 cm - 25 cm (Vertical) ~7 cm - 10 cm (Horiz.)	Average to Large Size of Adult Hand [94]
Magnification (M)	N/A	~30 - 35	Formula Calculation
Estimated Required System Focal Length	Length	~11.4 mm @ $S_{2,\text{ Final}}$ = 40 cm	Formula Calculation [95]

Table 8.1.3.1: List of Preliminary Data for Lens System Creation.

Given the value for Operating Range, we set that aside and worked backwards from the pixel size and pixel count of the camera sensor to calculate the exact length and width of the sensor area itself. From there, research was done to calculate the average size of an adult male hand - as well as the average large size, just in case [94]. From there, Magnification could be calculated given both Object Height in the camera sensor dimension and Image Height in the hand dimensions. Making use of existing calculators to help determine system parameters, the final pieces were able to come together for a required System Focal Length at around 11.4 millimeters [95].

From here, lens system testing was started with a twelve millimeter (12mm) Focal Length Bi-Convex lens. Due to limitations in the manufacturers, this lens is also twelve millimeters (12mm) in Diameter - though this was unnecessary. A simple two-lens system using a large negative Focal Length Bi-Concave lens with the larger Bi-Convex lens managed the scale, but the Image formed on the same side as the Object rather than opposite of it. So the second lens was supplemented by two more, intended to help focus the design. This is where our success lies, with the four-lens system. A significant advantage of this many lenses was the ability for micro-adjustments - so long as a proper casing could be made, a few slight adjustments in the design could allow for significant clarity. Of particular note are the second, first, and fourth lens; these are the three lenses with the most significant change in final image position, from most to least significant.

8.1.4. Final Lens Design

Object	Part #	Distance	Distance from Last Object
Camera	N/A	0 mm	0 mm
Lens 1	32-014	6.5 mm	6.5 mm
Lens 2	48-682	11 mm	~0.65 mm
Lens 3	45-292	13mm	~0.5 mm
Lens 4	32-968	16 mm	~0.5 mm

Table 8.1.4.1: Data on Lens Part IDs and their Positions in the Lens System.

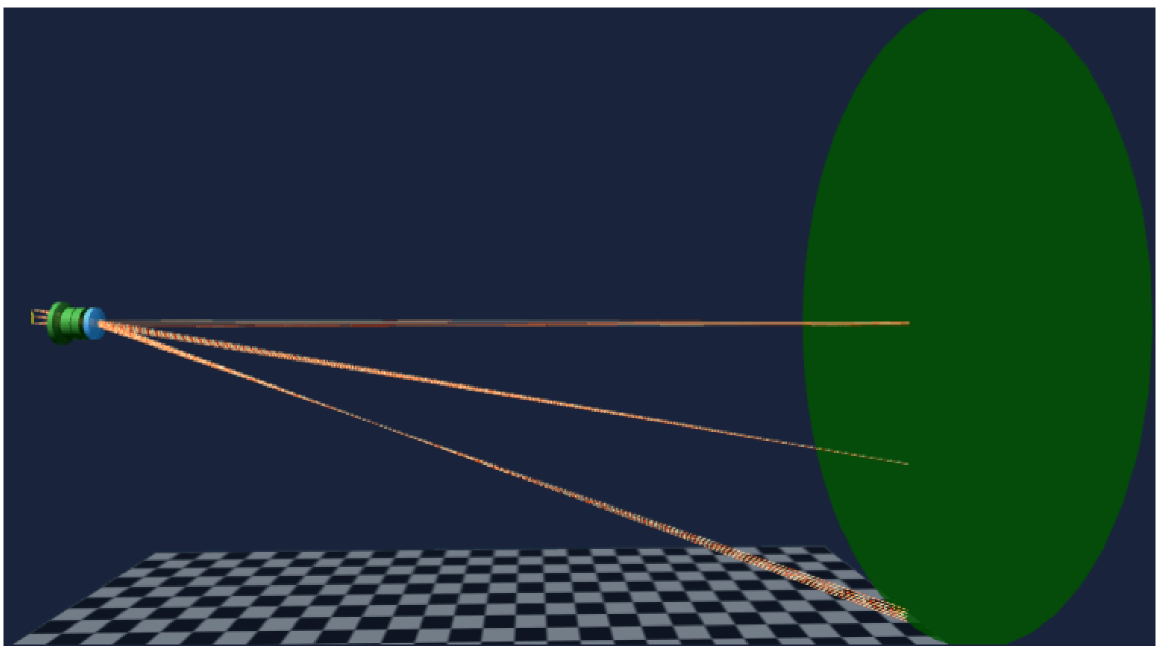


Figure 8.1.4.1: Simulation of the Hand Gesture Vending Machine's Lens System [96].

Operating Range ($S_{2,\ Final}$)	Image Height (h_i)	Magnification (M)
25 cm	18.8 cm	~29.147
30 cm	22.6 cm	~35.039
35 cm	26.39 cm	~40.915

Table 8.1.4.1: Comparison of Hand Distance from frontmost lens to Real Image Height and Approximate Magnification Factor.

8.2. Infrared System Designs

In comparison to the complicated four lens system, the Infrared design is significantly simpler to cover. It can be split into two parts: the external front-facing infrared sensor, and the internalized infrared transmitters and receivers. These components do not require significant optical calculations or background to be used, but their functionality as optical elements is still important to expand upon with reference to this project.

8.2.1. Front-Facing Infrared Sensor

The first component of the two is the front-facing E18-D80NK Adjustable Infrared Proximity Sensor. This sensor works off of a diffuse reflection style detection system, making use of modulated infrared signals in order to even work in environments with heavy presence of existing infrared sources, such as the Sun in the case of outdoor operation. Diffuse Reflection means that it is specifically detecting when the electromagnetic wavelengths the sensor emits are reflected back towards it, which is particularly effective against large targets (like a body) or smooth surfaces. This reflection did then trigger the sensor to send a signal to the microcontroller, which did in turn activate the camera and Object Detection.

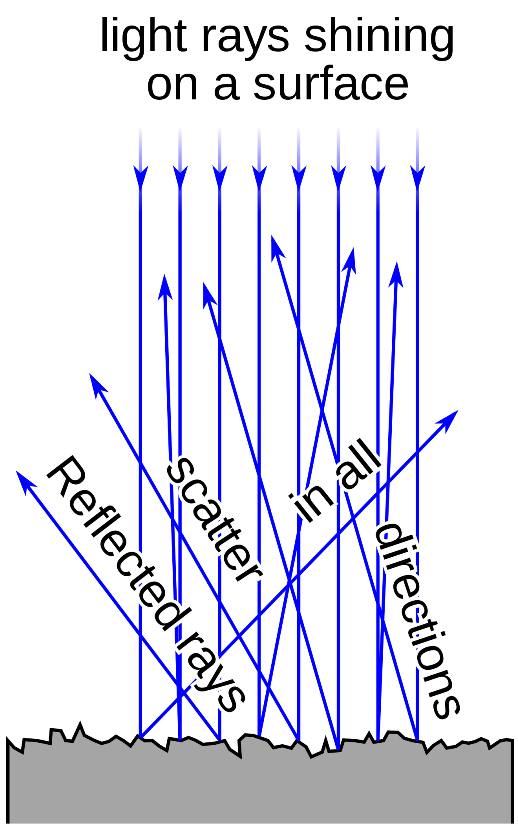


Figure 8.2.1.1: A Visualization of Diffuse Reflection.

8.2.2. Internal IR Transmitters and Receivers

The other half of the two infrared portions of the design are actually a pair of components acting in tandem. The HX-35 Infrared Transmitter and HX-M121 Infrared Receiver are a duo of components designed to transmit and receive infrared light from one another. The light is transmitted via LED, and the light is received via photon absorption through a photodiode. The intention behind using these is to place a transmitter-receiver pair for each snack item ‘tunnel’; if placed at the front, they can read if there’s a snack blocking their line of sight, marking the stock as ‘full’. On the other hand, if there is nothing blocking the line of sight, the snack would be considered ‘out of stock’ and marked as such when a person tries to ‘buy’ it from the device.

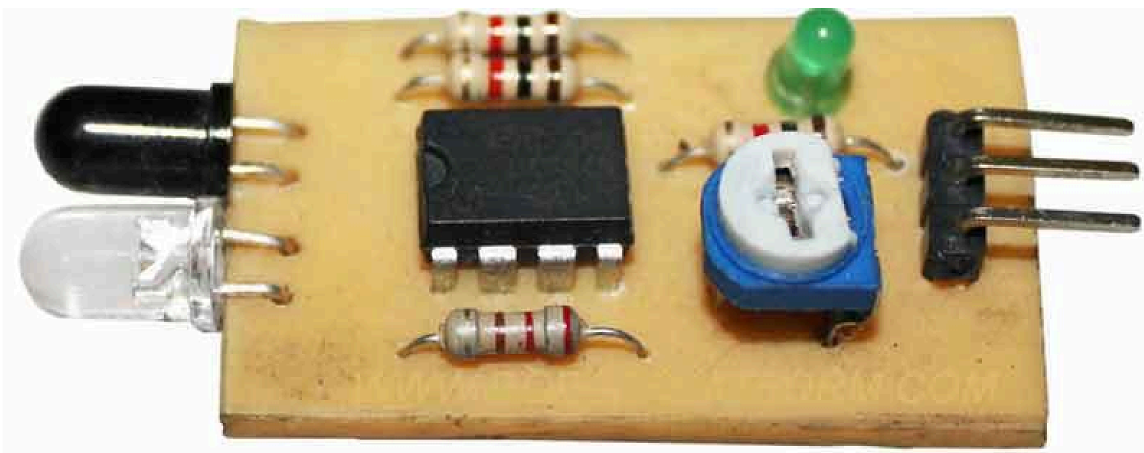


Figure 8.2.2.1: An example of an Infrared Transmitter-Receiver combination.

9. Testing and Quality Assurance

Before and even while we put together the components of the project, there are important steps we must take to protect ourselves from wasted time and resources. Namely, testing the components and confirming that they can consistently do the tasks they are built for. For each component and each branch of the project, this is reflected differently, but the fact remains that all of them require testing and quality assurance in their own ways before the whole design can be brought together.

Even when the design fully comes together, final functionality and benchmark tests must be done to confirm the Hand Gesture Vending Machine well and truly works properly. But no matter what, we should not begin work on the final design until after we have confirmed that the individual components can properly communicate and work with what they are built to.

9.1. Hardware Testing

When it comes to hardware testing, it’s a matter of working with not just the components on an individual level, but also confirming the the components work with the microcontroller proper as well. The easiest way to accomplish this is to start with the root of all the components and work from there. As such, testing starts with testing the power supply and the voltage regulator boards that have been built for the express purpose of stepping down the still too-high voltage of the power supply. This far, we have accomplished the former, but lack the boards to confirm the latter beyond simulation tests. However, given the simulations work, we have decided to continue on with testing the various components. As an aside, when given the chance, we did go back to confirm the functionality of the voltage regulator boards directly.

Following this initial test is a battery of tests for the components that don’t necessarily need to be connected to the microcontroller or the AI Development Kit to prove functionality. This includes the front-facing infrared sensor, the LEDs, and the motors. For all of these, tests have been - not just independently, but also while in the same circuit to prove cross-component functionality. Given all of them operate solely off of voltage and without need of data connections, however, that degree of accessibility is to be accepted. Future testing has to be done with using these components with a microcontroller, but it all looks very promising.

Then there is the AI Development Kit and the components that link directly to it. The Nvidia Jetson Nano needed a special microSD card with proprietary software to allow the storage chip to be used for booting off with just the Jetson Nano itself - and the chip. Following this bootup, code could be run to show the AI Development Kit’s functionality. This includes the ability to run a fan autonomously - and more importantly, it could successfully run the camera module. The IMX708 is a newer module of camera, and we were initially uncertain as to whether or not it would work with the Jetson Nano, so the immediate success has been significant for us. Arducam, the manufacturer of the camera module, provided detailed instructions to install the necessary drivers to ease the difficult process. After that, by using a command-line camera capture application, ngstcapture we got an image from the camera module through the SPI port.

Following this is testing on the components that *do* require microcontroller support to function; the screen, the fans, the infrared TX/RX, and the microcontroller-Dev Kit intercommunication link. Out of all of these, we have tested two and only one of the two functions at this moment. The fans work exactly as intended, spinning on command. The screen has yet to be put through its paces, and the microcontroller itself has not been set up for communicating with the Nvidia Jetson Nano quite yet. As for the infrared TX/RX, it simply did not cooperate at the time.

For the recent future, we intend to test the yet-to-be tried screen and communication link. We additionally intend to work through the difficulties we've faced with the infrared TX/RX, and follow up with the results of those tests. Finally, when the PCB is designed and within our grasp, we did begin full-scale testing with the components - including practice runs of full functionality.

9.2. Software Testing

The Jetson Nano is an affordable option to AI projects, there exist many open source applications and demos that use object detection, with the Jetson Nano and even hand gesture detection. We can use these already established algorithms as a template to get the AI software section of the project done faster; downloading and installing the prerequisite software can be time consuming and learning all the special cases and nuances of the python programming language, with the deep learning focus, would prove to be rather cumbersome. However, when considering the magnitude and impact of the AI software on the project, a high level of understanding is needed. This same logic cannot apply to the microcontroller; the microcontroller is much less complex and slightly more niche and therefore, the microcontroller did need to be mastered in the next progression of this project.

Classification Report

	precision	recall	f1-score	support
0	1.00	1.00	1.00	211
1	0.99	1.00	0.99	212
2	1.00	0.99	1.00	179
3	0.99	0.99	0.99	136
4	0.87	1.00	0.93	141
5	1.00	1.00	1.00	158
6	1.00	0.91	0.95	249
accuracy			0.98	1286
macro avg	0.98	0.98	0.98	1286
weighted avg	0.98	0.98	0.98	1286

Figure 9.2.1: The hand gesture model accuracy

When considering the testing of software, the software development team mulled over all the different subsystems in the project, including, but not limited to, the camera subsystem, the IR sensor subsystem, the transmitter and receiver subsystem, the AI subsystem, and the fan subsystem. The software development team came to the conclusion that we were grouping multiple subsystems together, in the testing phase, and testing them in conjunction. In one group, the notable members are the camera and the AI software. In this group, the testing did include the distance at which the camera, and object detection software, did be able to pick up on hand gestures. This is important because the person displaying the hand gestures should be able to be a reasonable distance away from the machine; not too far and certainly not too close. In the next group, the notable subsystems are the fans and the IR sensor subsystem. In this group, the testing did include the ability to get a pulse from both subsystems. The testing also includes adjusting the component-specific traits for both. For the fan subsystem, adjusting the component-specific trait would include adjusting the speed of the fan to our liking. For the IR sensor subsystem, adjusting the component specific trait would entail changing how far or close someone would have to be for the IR sensor to pick it up. The last notable subsystems receiving software testing were the IR transmitters and receivers. This testing, for the software development team, did entail performing communication protocols on the transmitters and receivers and, firstly, seeing if they work, but more importantly, testing if they can communicate through a folded piece of paper. This is important because the purpose of the transmitters and receivers, in this project, are to keep track of stock and if they are able to transmit and receiver through a piece of paper, they are not communicating properly.

9.3. Optics Testing

When it comes to testing the pure optical elements of the Hand Gesture Vending Machine, it's fairly simple. A vast majority of the testing is simply a matter of confirming the mathematics and simulations, which is done via putting the optical lens system together, mounting it into a casing, and confirming that the resulting images coming from the camera sensor are within the right range and magnitude. If it's slightly off, then all that needs to be done is remake the casing with slightly different ranges.

At the current moment, testing is impossible due to the lenses having shipped late. However, testing did begin once the lenses are in and the 3D printed mold is confirmed to snugly fit them all. Once the quality of the system is assured, it will be set aside for full-system tests and full-functionality run-throughs once we have done sufficient work with the PCB design iterations and software development. Once more information on the functionality of the system is gathered, this will be updated accordingly.

9.4. Complications in Testing

During the testing phase of our project our main objective was to be sure that our components were working accordingly. During testing a lot was learned about certain components, particularly the microcontroller. The microcontroller has a lot of extra GPIO pins which is great, but at the same time there are a lot of data sheets to have to go through to pinpoint the exact pin you need to use. Another thing presented during the testing phase was the lack of examples for certain objectives the group was trying to satisfy. For instance, there were not many videos or tutorials on certain mechanics we wanted to achieve with the STM32F429ZI, our chosen board. For simple mechanics such as using the microcontroller to move the motors, connect to a LCD screen, and transmit data between an IR transmitter and IR receiver.

Through testing and advice from Dr. Weeks, it was best to switch our microcontroller to the ESP32, which was already a microcontroller we were interested in based on the research that was done. A lot of trouble that was run into when trying to test the STM32F429ZI was when we wanted to transmit data from the IR transmitter to the IR receiver. Logic wise the code did seem as if it was correct and the components on the circuit board were properly set up but we were still running into error transmitting that data. The logic behind the code was to connect the transmitter to an GPIO output pin, then connect the receiver to a GPIO input pin.

Once that was done then put a 1 in the transmitter and you should get a 1 in the receiver and the logic behind that was if that happened there would be a red LED that would toggle on the microcontroller. The red LED seemed to always toggle no matter what, even if the transmitter or receiver were unplugged from the breadboard, concluding that no actual data was being transmitted between the two components. One thing that was accomplished with the STM32F429ZI was a simple program to blink two LEDs with a delay.

Another complication in this project stemmed from the Jetson Nano as a whole; the Jetson Nano had a dependency issue. For the software team's object detection app, they needed a certain set of available packages and versions of those packages. For example, a certain version of Tensorflow and Tensorflow Lite was needed, a certain version of Python was needed, and a certain version of Keras was needed. However, to download even one of these, the team had to attempt to go against what the Jetson Nano operating system allowed. For example, the software team, first, attempted to download Ubuntu 20.04 on the Jetson in order to get Python. However, to do this, the whole Chromium browser had to be completely removed from the system, and when the Chromium browser is removed, the apps on the desktop basically lock themselves. This aspect of the apps locking themselves caused there to be problems with accessing the terminal and problems installing any new package. This same idea of downloading one package only to essentially lock out all the other packages was the whole issue with the Jetson nano. This issue even caused problems with the camera; when the Ubuntu version was updated, that caused the camera to become an issue as well. Due to all the issues connected to the Jetson Nano, it was decided that the best course of action was to switch to the Raspberry Pi.

Communication was another point of complication that we failed to solidify in Senior Design 1. We knew the ESP32 had the SPI, WiFi, and I2C protocols built in so we knew we were probably going to use one of those methods. However each one gave us unseen complications. First, the screen used SPI so our SPI pins were already being used. Secondly, we got WiFi working very well. It was honestly really cool, we were about to control the motors through a web server. However, we soon found out that using WiFi disables some of the pins used for SPI and we decided we really needed the screen so we abandoned WiFi. Lastly, I2C gave us the most trouble because theoretically it should have worked but it was incredibly unreliable. It would work perfectly for a few minutes then crash the software on both sides randomly and since the gesture recognition code takes a considerable amount of time to start up, we can't have it crashing. We even verified that the signal was clean and defined on an oscilloscope so the whole issue could have been something with that particular ESP32 but we couldn't waste any more time on that. The solution we did find worked was to use the GPIO pins on the Raspberry Pi connected to free GPIO pins on the ESP32. Using just three wires (and a ground wire), the Raspberry Pi can send 2^3 different signals barring the no signal, bringing us to 8 binary-like signals. Four of the signals are used for sending the final selection of the user to the ESP32 to vend.

Our first PCB designs had issues, which can be seen in Figure 9.4.1. The biggest of which were airwires. Soldering wires to the floating ground planes did get it to run and be programmable. However there was an airwire on the regulators very close to the regulator integrated circuit that was very difficult to fix and once that regulator stopped working we needed new designs. They stopped working the next day.

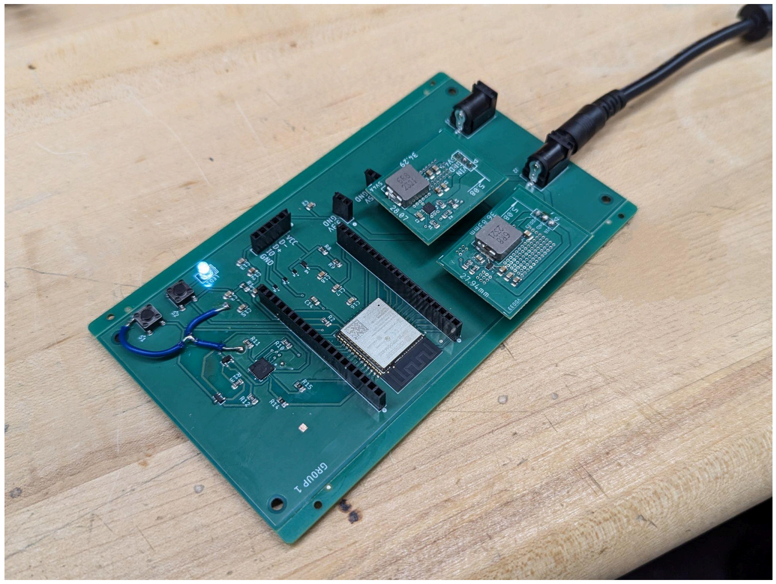


Figure 9.4.1: The first working PCB design

At this point we decided to drop down our input voltage from 24V to 12V. It became very apparent that this was the best decision when finding a suitable 24V to 3.3V became very difficult. Also it seemed like we were the only group using 24V. All new designs were made along with a new secondary board too. It was made to be a very cheap board that has the connection points for all of the motors and sensors that go back to the ESP32.

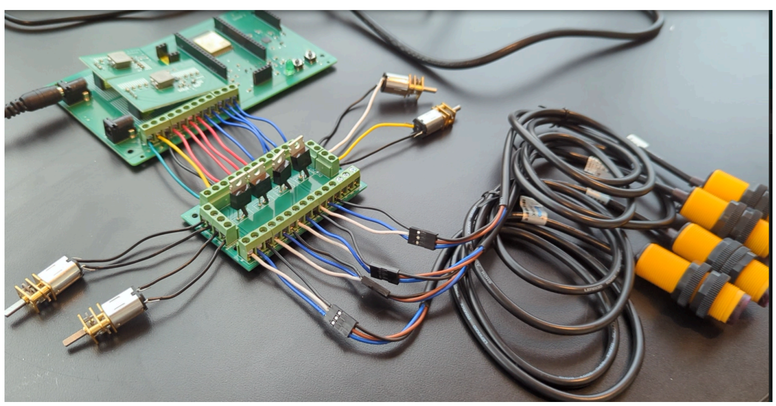


Figure 9.4.2: The final PCB design

These new designs worked extremely well. None of the regulators went bad, in fact we used the same regulators from first testing the new designs to our showcase day. Another new feature of the new main board was that the screen connections were all next to each other to match our screen perfectly. That made it so we could connect the screen in seconds rather than minutes.

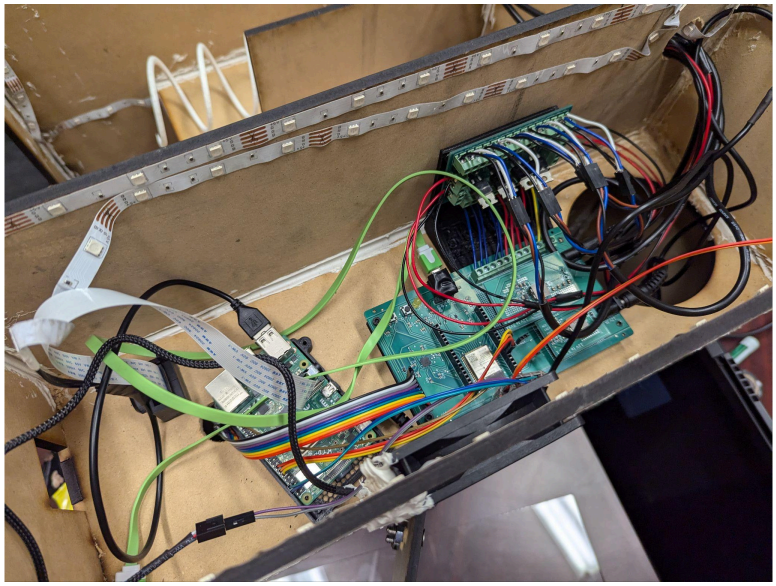


Figure 9.4.3: Inside of the machine

The inside of the machine along with the Raspberry Pi 4 (left) and the main board and secondary board can be seen in Figure 9.4.3. A 3D printed mount was created for each part to not have the PCB be touching the chassis. The Raspberry Pi is being powered by our 5V regulator through a barrel jack to USB C cable like originally planned.

Figure 9.4.4: Final product

Figure 9.4.4 is a picture of our final product of our project. The screen on top of the machine is only displaying the HDMI out of Raspberry Pi so we can verify the gesture recognition code is running.

10. Administrative Content

10.1. Milestones

Milestones for the project need to be set in order for a smooth completion of the project. To keep in line with these Milestones, individual deadlines are to be set so that falling behind does not occur. As of yet, no deadlines have been missed, and we intend to keep it that way. There were at least weekly meetings - with the possibility to expand into bi-weekly meetings - in order to discuss progress in person to help facilitate this, as well as constant team communication availability via a discord server. These, alongside peer review of the work done, did ensure that things did not go awry - or at least, not for long.

Tasks	Start Date	Planned End Date	Actual End Date
D+C 10 Page	1/18/2024	2/02/2024	2/01/2024
60 Page Draft	2/01/2024	3/29/2024	3/29/2024
Purchase All* Components	4/1/2024	4/15/2024	4/15/2024
120 Page Final Document	3/29/2024	4/23/2024	4/23/2024
PCB Design	03/29/2024	4/30/2024	6/29/2024
PCB Ordering	4/30/2024	5/7/2024	6/29/2024
PCB Testing	5/14/2024	6/29/2024	7/19/2024

Table 10.1.1: Milestones.

10.2. Bill of Materials

Part Name	Manufacturer	Part ID	Part Type	Quantity	Cost
24V Adapter Converter	ALITOVE	ALT-2406	Power Supply	1	$25.55
12mm Dia. Bi-Convex Lens	Edmund Optics	32-014	Lens	1	$31.00 + $28.50 + $29.50 + $29.50 + Ship. + Tax = $135.20
9mm Dia. Bi-Concave Lens	Edmund Optics	48-682	Lens	1	
9mm Dia. Plano-Convex Lens	Edmund Optics	45-292	Lens	1	
9mm Dia. Bi-Convex Lens	Edmund Optics	32-968	Lens	1	
840 Pcs Jumper Wires Kit	/	/	Breadboard Jumper Wires Kit	1	$7.99
38khz IR Sensor Module	/	HX-35 / HX-M121	IR Transmitter and Receiver Pair	10 Each	$17.99
Adjustable Infrared Sensor	NOYITO	E18-D80NK	Diff. Reflection IR Sensor	1	$6.99
Camera Module 3	Arducam	IMX708	Camera	1	$26.61

Figure 10.3.1: Bill of Materials, Part 1.

Part Name	Manufacturer	Part ID	Part Type	Quantity	Cost
ESP-WROOM-32	HiLetGo	WROOM-32	Microcontroller	1	$9.99
Geared Mini DC Motor	Handson Technology	G12-N20	Mini DC Motor	5	$22.45
Jetson Nano	NVIDIA	/	AI Dev Kit	1	$108.11
Solder Wire	TOWOT	SW-0850	Solder Wire	1	$6.99
Solder Wick	TOWOT	SW-3030	Solder Wick	1	$6.99
TFT LCD Screen	Hosyond	/	LCD Screen	1	$20.22
Raspberry Pi 4	Raspberry Pi	/	AI Dev Kit	1	$65.76
Total	/	/	/	/	$460.84

Table 10.3.2: Bill of Materials, Part 2.

11. Conclusion

The Hand Gesture Vending Machine is a project designed with three majors, some disciplines, and many skills in mind. Some skills and disciplines we have come into the project with, others have come through the trials and tribulations this project has presented us with - or did present us with. Even as the first half of the project comes to a close, it's been a great learning experience for us all.

We are a team of two Electrical Engineers - one of which is dual-majoring for Photonic Science and Engineering - and two Computer Engineers. A majority of the fields cluster together under the same, closely knit umbrella of blending hardware, component design, and software for their goals. However, this umbrella has expanded, and with it has come new strengths and pieces of interdisciplinary education that we did take with us as we move forward.

Speaking of moving forward, our skills when it comes to design and practical application of the theoretical have already been repeatedly put to the test - both proving ourselves in the paper-writing and in component functionality - yet we've only just landed our feet on the ground. The running comes next, taking these recent but rapidly improving skill sets and putting them to the test over the next semester. We've already started taking those first steps; components and design philosophies we haven't considered before are being taken into account, and we are adapting instead of floundering.

As a group, we have been united from the beginning. Coordination has been a strong suit and that has, without a doubt, kept us ahead of the game on more than one occasion. Our willingness to trust one another to get our parts done and to pick up slack when one of us struggles has also been a significant factor, making sure that any difficulties we may individually have won't hurt us all in the long run. We're sure to be successful if we can keep it together through the next semester.

To surmise, the Hand Gesture Vending Machine did be the work of four students working as one, with many smaller pieces coming together and becoming something far greater - just as we have. Just as we will continue to do.

A References

Section 3.1.

[1] “The History of Vending Machines,” ThoughtCo, 2019.
https://www.thoughtco.com/the-history-of-vending-machines-1992599

[2] “From holy water to luxury cars, vending machines dispense all,” *GovTech Singapore*, Feb. 18, 2022.
https://www.tech.gov.sg/media/technews/history-of-vending-machine

[3] “History of Gumball Machines | OK Manufacturing,” web.archive.org, May 12, 2008.
https://web.archive.org/web/20080512073919/http://www.okmfg.net/history-of-gumball-machine.html

[4] S. Magazine and E. Jaffe, “Old World, High Tech,” *Smithsonian Magazine*.
https://www.smithsonianmag.com/science-nature/old-world-high-tech-141284744/

Section 3.2.

[5] F. Stöckmann, “Photodetectors, their performance and their limitations,” vol. 7, no. 1, pp. 1–5, May 1975, doi: https://doi.org/10.1007/bf00900511

[6] “HC-SR501 datasheet,” *datasheetspdf.com*.
https://datasheetspdf.com/mobile/775434/ETC/HC-SR501/index.html

[7] “Amazon.com: Dioche Universal Wired Microwave Motion Sensor Detector, 12-24V Automatic Door 24.125Ghz Microwave Motion Sensor Autodoor Sensors, Monitor and Protect Outdoor Indoor Property : Electronics,” *Amazon.com*.
https://www.amazon.com/Universal-Microwave-Detector-Automatic-24-125Ghz/dp/B07RWNDJ91

[8] Handson Technology User Guide E18-D80NK Adjustable Infra-Red (IR) Proximity Sensor.” Available:
https://handsontec.com/dataspecs/sensor/E18-D80NK%20IR%20Sensor.pdf

[9] “inlay | British Museum,” *The British Museum*.
https://www.britishmuseum.org/collection/object/W_-90959

[10] K. Ray, S. N. Sharan, S. Rawat, S. K. Jain, S. Srivastava, and A. Bandyopadhyay, *Engineering Vibration, Communication and Information Processing*. Springer, 2018, p. 420.

[11] “Optical Materials,” *MKS Newport*. https://www.newport.com/n/optical-materials

[12] “Optical Glass | Edmund Optics,”*Edmund Optics*.
https://www.edmundoptics.com/knowledge-center/application-notes/optics/optical-glass/

[13] “N-LASF9 | SCHOTT Advanced Optics,” *SCHOTT*.
https://www.schott.com/shop/advanced-optics/en/Optical-Glass/N-LASF9/c/glass-N-LASF9

[14] “EAILP05RDDB1 5mm Infrared LED Datasheet,” *Mouser Electronics*.
https://www.mouser.com/datasheet/2/143/EAILP05RDDB1-708504.pdf

[15] “With Flange Type Silicon PIN Photodiode Technical Data Sheet,” 2005. [Online]. Available:
https://www.tme.com/Document/4d064856518b7fe1676be4d7bfb07091/LL-503PDD2E.pdf

[16] “IR Sender HX-53 5V Arduino IR remote Control.”
https://www.androegg.de/shop/ir-sender-hx-53-5v-arduino-ir-remote-controll/

[17] “RF Transmitter and Receiver Link Kit - 315MHz/433MHz.” [Online]. Available:
https://media.digikey.com/pdf/Data%20Sheets/Seeed%20Technology/RF_Transmitter_Receiver_LinkKit_315MHz_433MHz_Web.pdf

Section 3.3.

[18] “Electricity 101,” *United States Department of Energy*.
https://www.energy.gov/oe/electricity-101

[19] Electronics Tutorials, “Full Wave Rectifier and Bridge Rectifier Theory,” *Basic Electronics Tutorials*, Feb. 24, 2018.
https://www.electronics-tutorials.ws/diode/diode_6.html

[20] “DC 24V 15A 360W Power Supply Universal Regulated Switching Transformer Adapter LED Driver 110V/220V AC Input for LED Strip CCTV Radio,” *Alitove*.
https://www.alitove.net/index.php?route=product/product&product_id=90

[21] “WSU-Series - WSU240-1000-R,” *Triad Magnetics*.
http://catalog.triadmagnetics.com/Asset/WSU240-1000-R.pdf

[22] "ALITOVE 24V 6A Power Supply Adapter Converter 100-240V AC to DC 24 Volt 144W 6Amp 5.5A 5A 4A Transformer with 5.5x2.5mm Plug for LED Strip Light CCTV Camera LCD Monitor Massage Chair," *Alitove*. https://www.alitove.net/index.php?route=product/product&product_id=87

[23] "Basic Calculation of a Buck Converter's Power Stage," Texas Instruments. https://www.ti.com/lit/an/slva477b/slva477b.pdf?ts=1711703501230

[24] Fenlon, Wes. "Noctua Spent Four and a Half Years Designing Its Quietest, Strongest Fan Yet." Pcgamer, PC Gamer, 5 June 2017, www.pcgamer.com/noctua-spent-four-and-a-half-years-designing-its-quietest-strongest-fan-yet/

[25] "NF-P12 redux-1700 PWM Specifications," Noctua. https://noctua.at/en/nf-p12-redux-1700-pwm/specification

[24] Jordandee. "Pulse Width Modulation." Pulse Width Modulation - SparkFun. learn.sparkfun.com/tutorials/pulse-width-modulation/all

[25] "CFM-A225C-015-287 Datasheet - Axial Fans," Cuidevices. https://www.cuidevices.com/product/resource/cfm-120c.pdf

[26] "Session 4 - Airflow and Static Pressure," Tech Compass https://techcompass.sanyodenki.com/en/training/cooling/fan_basic/004/index.html

[27] O. D. Jefimenko and T. Valone, *Electrostatic Motors: Their History, Types, and Principles of Operation*. Beltsville, MD: Integrity Research Institute, 2011. pp. 22-45.

[28] "Micro Metal Gearmotors." Available: https://www.pololu.com/file/0J1487/pololu-micro-metal-gearmotors_rev-5-1.pdf

[29] "Handson Technology Datasheet G12-N20 Geared Mini DC Motor." Available: https://www.handsontec.com/dataspecs/motor_fan/GA12-N20.pdf

[30] "ASLONG JGY-370 12V DC Worm Gear Motor." [Online]. Available: https://media.digikey.com/pdf/Data%20Sheets/Seeed%20Technology/108990007_Web.pdf

[31] LEDs Magazine, "What is an LED? | LEDs Magazine," *Ledsmagazine.com*, Sep. 01, 2004. https://www.ledsmagazine.com/leds-ssl-design/materials/article/16701292/what-is-an-led

[32] R. Treat, "How Do RGB LEDs Work?," *The Hyperspace Lighting Company*, Jul. 14, 2023. https://hyperspacelight.com/blogs/news/how-do-rgb-leds-work

[33] Kingbright, "Kingbright WP154A4SUREQBFZGC," Apr. 08, 2021.[Online]. Available: https://www.kingbrightusa.com/images/catalog/SPEC/WP154A4SUREQBFZGC.pdf

[34] J. Cho, J. H. Park, J. K. Kim, and E. F. Schubert, "White light-emitting diodes: History, progress, and future," *Laser & Photonics Reviews*, vol. 11, no. 2, p. 1600147, Mar. 2017, doi: https://doi.org/10.1002/lpor.201600147

[35] "PRODUCT FAMILY DATA SHEET C513A-WSS/WSN/MSS/MSN: 5-mm Round White LEDs PRODUCT DESCRIPTION." [Online]. Available: https://www.mouser.com/datasheet/2/723/HB_C513A_WSN_WSS_MSN_MSS-3402001.pdf

[36] "Part Number: XLUR12D,". [Online]. Available: https://www.sunledusa.com/products/spec/XLUR12D.pdf

[37] "A Journey through the Evolution of LED Lighting Technology," *Matluxe*. https://matluxe.com/blogs/learn/from-the-first-leds-to-todays-led-era-a-journey-through-the-evolution-of-led-lighting

[38] "Middle Power LED Series 3030 LM301B CRI 80" Available: https://download.led.samsung.com/led/file/resource/2022/04/Data_Sheet_LM301B_CRI80_Rev.10.2.pdf

[39] S. Smiley, "Low Frequency RFID vs. High Frequency RFID: The Top 8 Differences," *atlasRFIDstore*, Jun. 15, 2022. https://www.atlasrfidstore.com/rfid-insider/low-frequency-rfid-vs-high-frequency-rfid/

[40] "Types of LCD," *Newhaven Display.* https://newhavendisplay.com/blog/types-of-lcd/

[41] "Display types and technologies explained: LCD, OLED, miniLED, microLED, and more," *Android Authority*, Oct. 03, 2022. https://www.androidauthority.com/display-technology-lcd-oled-led-3030739/

[42] "OLED vs LCD," *Newhaven Display*. https://newhavendisplay.com/blog/oled-vs-lcd/

[43] "Touchscreen Types, History & How They Work," *Newhaven Display*. https://newhavendisplay.com/blog/touchscreen-types-history-how-they-work/

[44] "CSI-2 Specification," MIPI Alliance. https://www.mipi.org/specifications/csi-2

[45] “Raspberry Pi Camera Module 3 Product Brief,” Raspberry Pi. https://www.mouser.com/datasheet/2/635/camera_module_3_product_brief-3241376.pdf

[46] “LI-IMX219-MIPI-FF-NANO SPECIFICATION,” Leopard Imaging Inc. https://www.mouser.com/datasheet/2/233/LI-IMX219-MIPI-FF-NANO_SPEC-1596177.pdf

[47] “12.3MP USB3.0 Camera Module Datasheet,” Arducam. https://www.arducam.com/downloads/datasheet/B0459_12MP_477P_USB3.0_Camera_Datasheet.pdf.pdf

Section 3.4.

[48] “C Language Introduction,” *GeeksforGeeks*, Feb. 12, 2014. https://www.geeksforgeeks.org/c-language-introduction/

[49] GeeksForGeeks, “Introduction to Java,” *GeeksforGeeks*, Nov. 07, 2020. https://www.geeksforgeeks.org/introduction-to-java/

[50] “Introduction To PYTHON,” *GeeksforGeeks*, Nov. 06, 2015. https://www.geeksforgeeks.org/introduction-to-python/

[51] B. Lutkevich, “What is a Microcontroller and How Does it Work?,” *IoT Agenda*, Nov. 2019. https://www.techtarget.com/iotagenda/definition/microcontroller

[52] “MSP430FR698x(1), MSP430FR598x(1) Mixed-Signal Microcontroller | TI.com,” *Texas Instruments*. https://www.ti.com/document-viewer/msp430fr6989/datasheet

[53] “STM32F429ZI - High-performance advanced line, Arm Cortex-M4 core with DSP and FPU, 2 Mbytes of Flash memory, 180 MHz CPU, ART Accelerator, Chrom-ARTAccelerator, FMC with SDRAM, TFT - STMicroelectronics,” *www.st.com*. https://www.st.com/en/microcontrollers-microprocessors/stm32f429zi.html

[54] “ESP32-WROOM-32D & ESP32-WROOM-32U Datasheet,” Espressif Systems. https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32d_esp32-wroom-32u_datasheet_en.pdf

[55] “Arduino Due,” Arduino Documentation. https://docs.arduino.cc/hardware/due/

[56] “BeagleBone® AI-64,” *BeagleBoard*. https://www.beagleboard.org/boards/beaglebone-ai-64

[57] “BeagleBone AI-64 based Bot in Go,” *hackaday.io*. https://hackaday.io/project/194796-beaglebone-ai-64-based-bot-in-go

[58] "History [ODROID Wiki]," *wiki.odroid.com*. https://wiki.odroid.com/history

[59] R. P. (Trading) Ltd, "Buy a Raspberry Pi," *Raspberry Pi*. https://www.raspberrypi.com/products/

[60] "CUDA Zone," *NVIDIA Developer*, Apr. 30, 2019. https://developer.nvidia.com/cuda-zone

Section 3.5.

[61] IBM, "What is Machine Learning?," *IBM*, 2023. https://www.ibm.com/topics/machine-learning

[62] M. Banoula, "The Complete Guide to Machine Learning Steps," *Simplilearn.com*, 2023. https://www.simplilearn.com/tutorials/machine-learning-tutorial/machine-learning-steps

[63] A. C. Müller and S. Guido, *Introduction to machine learning with Python: a guide for data scientists*. Beijing: O'reilly, 2017.

[64] "What is an AI Model?," *YouTube*. https://www.youtube.com/watch?v=8gLdN8WoMak

[65] edureka, "Supervised vs Unsupervised vs Reinforcement Learning | Data Science Certification Training | Edureka," *YouTube*. Jan. 29, 2019. https://www.youtube.com/watch?v=xtOg44r6dsE

[66] J. Delua, "Supervised vs. Unsupervised Learning: What's the Difference?," IBM Blog, Mar. 12, 2021. https://www.ibm.com/blog/supervised-vs-unsupervised-learning/

[67] "What is Unsupervised Learning?," SearchEnterpriseAI. https://www.techtarget.com/searchenterpriseai/definition/unsupervised-learning

[68] S. AI, "Reinforcement Learning algorithms — an intuitive overview," Medium, Feb. 18, 2019. https://smartlabai.medium.com/reinforcement-learning-algorithms-an-intuitive-overview-904e2dff5bbc

[69] IBM Technology, "Neural Networks Explained in 5 minutes," YouTube. May 24, 2022. [YouTube Video]. Available: https://www.youtube.com/watch?v=jmmW0F0biz0

[70] IBM, "AI vs. Machine Learning vs. Deep Learning vs. Neural Networks: What's the difference?," IBM Blog, Jul. 06, 2023.
https://www.ibm.com/blog/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks/

[71] E. Odemakinde, "Model-Based and Model-Free Reinforcement Learning: Pytennis Case Study," *neptune.ai*, Jul. 21, 2022.
https://neptune.ai/blog/model-based-and-model-free-reinforcement-learning-pytennis-case-study

[72] Merriam-Webster, "Definition of ALGORITHM," *Merriam-webster.com*, 2019.
https://www.merriam-webster.com/dictionary/algorithm

[73] "Difference Between ML Algorithm and ML Model | ML Algorithm vs Model," *www.youtube.com*. https://www.youtube.com/watch?v=eYlbg0TqcNk

[74] "CNN vs. RNN: How are they different? | TechTarget," Enterprise AI.
https://www.techtarget.com/searchenterpriseai/feature/CNN-vs-RNN-How-they-differ-and-where-they-overlap

[75] "What are recurrent neural networks and how do they work?," SearchEnterpriseAI.
https://www.techtarget.com/searchenterpriseai/definition/recurrent-neural-networks

Section 3.6.

[76] "What is ABS Plastic and Why Do We Use It? | Mueller Corporation," *Mueller Corporation*. https://muellercorp.com/abs-plastic-and-how-we-use-it/

[77] Mike, "Acrylonitrile butadiene styrene (ABS) USA price index," *BusinessAnalytiq*, Mar. 22, 2021.
https://businessanalytiq.com/procurementanalytics/index/acrylonitrile-butadiene-styrene-abs-usa-price-index-2/

[78] "Acrylonitrile Butadiene Styrene (ABS Plastic): Uses, Properties & Structure," *omnexus.specialchem.com*.
https://omnexus.specialchem.com/selection-guide/acrylonitrile-butadiene-styrene-abs-plastic

[79] "MatWeb - The Online Materials Information Resource," *www.matweb.com*.
https://www.matweb.com/search/DataSheet.aspx?MatGUID=abc4415b0f8b490387e3c922237098da

[80] "Stainless Steel Scrap Price in Florida , United States Scrap Yards - Price USD/LB," *ScrapMonster*.
https://www.scrapmonster.com/scrap-yard/price/stainless-steel-scrap/florida/6/3374

[81] AZoM, "Aluminium - Specifications, Properties, Classifications and Classes," *azom.com*, May 17, 2005. https://www.azom.com/article.aspx?ArticleID=2863

[82] "Scrap Metal Prices in Orlando, Florida Scrap Price Index,United States Scrap Yards," *ScrapMonster*. https://www.scrapmonster.com/scrap-yards/prices/orlando/3380

[83] M. Ford, "How is tempered glass made?," *Scientific American*, Jan. 22, 2001. https://www.scientificamerican.com/article/how-is-tempered-glass-mad/

[84] "What is PMMA? | Acme Plastics," *Acme Plastics*. https://www.acmeplastics.com/what-is-pmma

[85] "Laminated glass: What is it? How is it made? Benefits & thickness," *Guardian Glass*, 2022. https://www.guardianglass.com/us/en/our-glass/glass-types/laminated-glass

[86] "What is a Glass Ceramic?," *AZoM.com*, Apr. 19, 2019. https://www.azom.com/article.aspx?ArticleID=11110

[87] "Polycarbonate vs Glass," *Piedmont Plastics*. https://www.piedmontplastics.com/blog/polycarbonate-vs-glass

Section 4.1.

[88] R. Merritt, "Mobile chip interface gets real," *EE Times*, Feb. 13, 2006. https://www.eetimes.com/Mobile-chip-interface-gets-real/

[89] T. S. Team, "IPC J-STD-001 Standard for Soldering," *Sierra Circuits*, Sep. 23, 2021. https://www.protoexpress.com/blog/ipc-j-std-001-standard-soldering-requirements/#:~:text=J%2DSTD%2D001%20is%20a

Section 4.2.

[90] Carruego, Miguel. "The Theory of Constraints in Software Development." *Medium*, 30 June 2023, mikecarruego.medium.com/the-theory-of-constraints-in-software-development-7e37cb0911db

Section 5.1.

[91] A. Vaswani *et al.*, "Attention Is All You Need," 2017. Available: https://arxiv.org/pdf/1706.03762v7.pdf

Section 6.2.

[92] “ESP Hardware Design Guidelines - ESP32 - — ESP Hardware Design Guidelines latest documentation,”*Espressif Systems*. https://docs.espressif.com/projects/esp-hardware-design-guidelines/en/latest/esp32/index.html

[93] “ESP32WROOM32E & ESP32WROOM32UE Datasheet.” Available: https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32e_esp32-wroom-32ue_datasheet_en.pdf

Section 8.1.

[94] K. E. Aboul-Hagag, S. A. Mohamed, M. A. Hilal, and E. A. Mohamed, “Determination of sex from hand dimensions and index/ring finger length ratio in Upper Egyptians,” *Egyptian Journal of Forensic Sciences*, vol. 1, no. 2, pp. 80–86, Jun. 2011, doi: https://doi.org/10.1016/j.ejfs.2011.03.001

[95] “Imaging System Parameter Calculator | Edmund Optics,” *Edmund Optics Worldwide*. https://www.edmundoptics.com/knowledge-center/tech-tools/imaging-system-parameter-calculator/

[96] “Lens Design Cloud | LightMachinery,” *LightMachinery*. https://lightmachinery.com/optical-design-center/lens-design-cloud

B Copyright Permissions

Section 2.4.

Figure 2.4.2
Attribution: Logan Terrell

Section 2.5.

Figure 2.5.1
Attribution: Hussein Shelleh

Figure 2.5.2
Attribution: Hussein Shelleh

Figure 2.5.3
Attribution: Hussein Shelleh

Section 3.1.

Figure 3.1.1.1
Attribution: Nate Enyedi
This file is made available under the Creative Commons Attribution-Share Alike 3.0 Unported license.

Section 3.2.

Figure 3.2.1.1
Attribution: H0dges
This file is licensed under the Creative Commons Attribution 3.0 Unported license.

Figure 3.2.2.1
Attribution: Anton Croos
This file is licensed under the Creative Commons Attribution-Share Alike 4.0 International license.

Figure 3.2.3.1
Attribution: RockMancuso
This file is licensed under the Creative Commons Attribution-Share Alike 3.0 Unported license.

Section 3.3.

Figure 3.3.3.1
Attribution: Abnormaal
This file is licensed under the Creative Commons Attribution-Share Alike 3.0 Unported license.

Figure 3.3.4.1
Attribution: Maheep Singh24
This file is licensed under the Creative Commons Attribution-Share Alike 4.0 International license.

Figure 3.3.6.1
Attribution: Polarbear24
This file is licensed under the Creative Commons Attribution-Share Alike 4.0 International license.

Figure 3.3.7.1
Attribution: Ed g2s
This file is licensed under the Creative Commons Attribution-Share Alike 3.0 Unported license.

Section 3.4.

Figure 3.4.1.1
Attribution: Максим Пе
This file is made available under the Creative Commons Attribution-Share Alike 4.0 International license.

Figure 3.4.2.1
Attribution: Ordercrazy
This file is made available under the Creative Commons CC0 1.0 Universal Public Domain Dedication.

Section 3.5.

Figure 3.5.1
Attribution: Lollixzc
This file is made available under the Creative Commons Attribution-Share Alike 4.0 International license.

Figure 3.5.2
Attribution: Numiri

Figure 3.5.3
Attribution: KIWSUMAP

Figure 3.5.4
Attribution: Megajuice

Figure 3.5.5
Attribution: Sven Behnke

Figure 3.5.6
Attribution: Aphex34

Section 3.6.

Figure 3.6.2.1

Section 6.

ALL Figures
Attribution: Tyler Bornemann, Hussein Shelleh

Section 7.

Figures 7.1.1, 7.2.1, 7.3.1, 7.5.1
Attribution: Jhamori Williams-Austin, Logan Terrell

Section 7.4.

Figure 7.4.1
Attribution: Shmuel Csaba Otto Traian

Section 8.1.

Figure 8.1.2.1
Attribution: DrBob

Section 8.2.

Figure 8.2.1.1
Attribution: Jeff Dahl