Smart Mirror

University of Central Florida

Department of Electrical Engineering and Computer Science

Dr. Lei Wei
Senior Design 1

Group 5

James Timothy – Electrical Engineering
Matthew Caserto – Electrical Engineering
Greg Eugene – Computer/Electrical Engineering
Menashe Mordachai – Photonics/Electrical Engineering

Table of Contents Page

[TABLE OF CONTENTS WILL BE FILLED OUT FOR THE FINAL REPORT, AS THE SECTONS ARE STILL CHANGING AND MORE ARE BEING ADDED IT WILL BE LEFT BLANK FOR NOW]

1.0 Executive Summary

The following document will show the processes taken to produce the Smart Mirror. This document will show the depth of research that goes into designing this rigorous project as a part of the Senior Design curriculum. This team of electrical engineers, some of which are also majoring in computer engineering and photonics, will take on this project and develop the technology that has yet been introduced into this product. Despite it being a product that has already made an appearance in the technological world, this group is hoping to develop some new and interesting features that would make this project stand out. The innovation of this project seeks to develop teamwork with the group as well as to nurture creativity and out-the-box thinking, all of which will be required in future careers.

The Smart Mirror project seeks to advance the technological world, as well as to ease everyday lives. The purpose of our Smart Mirror is to be placed in the bathroom, so when getting ready to tackle the day people may check the weather, emails, watch shows, or just browse the internet for entertainment. It can do all that and simply function as an ordinary mirror when needed.

The Smart Mirror will require the use of a two-way mirror to function the way that it's intended. For the display to look as if it's floating freely in the mirror, using a two-way mirror which only lets the light of the display show through is critical. It's also necessary for the Smart Mirror to function as a regular mirror if the technological functions are not in use, which is another reason why the two-way mirror will be required.

For ease of access, the Smart Mirror will be able to function using gestures which will be processed by the sensors and translated into specific actions that was required by the user. The other method of access to the mirror is using an IR-frame, which will provide a touchscreen-ability to the Smart Mirror. Bluetooth will also be added to this Smart Mirror, so that the user can use their phones to access the mirror. Mainly accessing the data specifically for that phone.

2.0 Project Description

This section of the paper will explain in greater detail, the purpose of the Smart Mirror. It will also show the functions that we intend to put on this Smart Mirror given the allotted time. This will provide a clear objective on what is required of the mirror and how it will affect the daily functions of the user.

2.1 Project Background

In the life of an average person, our morning routine can be one of the most stressful parts of our day. There's a lot to do in the morning that's important to starting the day off right and that can have an effect on the outcome of your entire day. That's why it's important to condense everything to one place and simplify the morning or even evening routine to one place, the bathroom.

A majority of people start their day off by going to the bathroom and getting started on their morning routine, such as showering, brushing teeth, shaving, etc. These tasks require spending a lot of time in front of the mirror but what if it could do more than just displaying your own reflection? Such as displaying the weather, so you know what you'll wear that day; Showing news reports; The next event on your schedule and even ordering an Uber to be there as soon as you leave the house.

We want to create a mirror that will allow anyone to accomplish basic daily tasks while simply getting ready in the morning and the goal is to make it as minimalistic as possible. When it isn't being used, the tech should be completely invisible, and it'll look just like any other mirror. No visible camera's, sensors, and wires anywhere to be seen. This is important because the best kind of technology is the one you never notice. It's noninvasive and simple and will feel seamless in the use and functions.

Another important feature for us is making a navigation scheme that lets users easily navigate the OS but without actually touching the mirror. This is important because nobody wants fingerprints all over their bathroom mirror. We will accomplish this using an optical frame that surrounds the mirror and will act as a touch input device, sensing where a finger is place near the screen. The second input method will be a gesture sensor, most likely IR LEDs, that will sense hand movements such as a swipe left, right, up, and down. The combination of these will allow someone to easily get around the interface and use the mirror.

The features intended to be implemented are facial recognition, an LCD display, sensors for sensing touch and gestures for navigating the user interface, and Bluetooth connection for retrieving information from user's phones.

2.2 Objectives

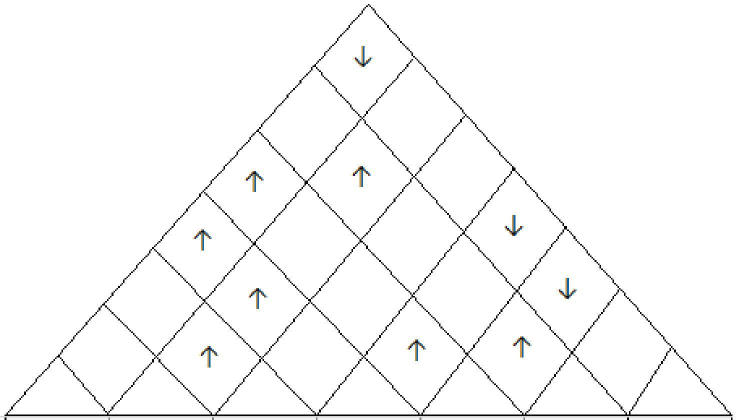
The objective for this project is to use our combined skills and the education we've gained at the University of Central Florida to work as a group and build something together. It will challenge us in many ways from testing our team work to our problem solving. This is a very valuable experience as it will prepare us for graduation and finding a job afterwards to kickstart our future careers.
The objective of the project itself is to simplify the everyday lives of the user. The efficiency of everyday activities should be increased, as the user can get ready and check emails, etc. at the same time. The overall features of this project should be innovative and easy to use for the users. The secondary objective of the project is to create a smart mirror that can provide the best ease of access for the user. The use of gestures will make the mirror innovative and easy to use for those that have their hands full while getting ready. The use of Bluetooth, will allow the user can use their phones to access the smart mirror and their own personal data on said smart mirror, making it quick and efficient to access their data. The IR frame allows the user to use touchscreen to access the smart mirror applications, for those that enjoy the hands-on experience. Lastly, we may possibly put in a camera for facial recognition, making it easy for the user to access their own personal data with their face.

2.3 Requirements Specifications

- The mirror will have an LCD display that fits the mirror size as closely as possible
- The mirror will be mounted to the LCD panel as closely as possible to maximize display fidelity
- The mirror will have a quick turn-on time when someone stands in front of it
- The electronics will consume much less power when the mirror is not being used
- The electronics shouldn't be visible at all, besides the camera.
- The user interface will use mostly white to maximize the display contrast as much as possible
- The mirror processor and microcontroller will communicate over serial to relay information between the two devices
- The microcontroller must have enough I/O to connect to all the sensors and communicate to the processor
- The electronics will include Bluetooth for getting information such as weather and calendar events from a user's phone.
- The electronics will include a Wi-Fi module for getting information for when a Bluetooth connection isn't available.

2.4 House of Quality

Table 1: House of Quality



		Optical touch grid	Motion control	Facial recognition	Social media connectivity	Phone connectivity	Environmental measurements	Installation
		+	+	+	+	+	+	-
Cost	-	↑↑	↑	↑	↑	↑		↓
Usability	+	↑	↑	↑	↑	↑	↑	↓
Power consumption	-	↑			↑	↑		
Mirror size	+	↑↑						↑↑
Targets for Engineering Requirements		Precision (<5% variation)	Hand gestures	>99% accuracy	Twitter, Facebook, Netflix	Bluetooth functionality	Temperature, humidity (1% accuracy)	>5 minutes

The house of quality of this product shows the basic requirements that consumers want the product to have. To make the product marketable, it's important to meet the consumers expectations as much as possible. This house of quality chart shows how each function that are placed on the smart mirror will meet the expectations for the product overall from the consumer perspective.

One of the most crucial component for consumers is the price factor of the product. The more features we put on our smart mirror will obviously raise the price factor of the product. However, if the product was to be marketable, it is essential to have these functions that have become a norm for the users in today's technological world.

The next important factor is ease of access, which is exactly what we have done with the several functions placed on our smart mirror. We have designed our product in such a way that it should satisfy most of the users in terms of accessibility of the use of the product. From gestures to facial recognition, our smart mirror provides ease of access fit for most if not all consumers.
Another important component for consumers is the power consumption of the product. Consumers dislike products that will greatly increase their monthly electric bills. Some of the function we placed on the smart mirror will influence the power consumption. However, with most electric products, in terms of power consumption, it all depends on the amount of time the consumer uses the product overall. We have taken steps to help reduce power consumptions by making the display go offl when not in use.

The last factor that is important to some of the consumers is product dimensions. For our case, our smart mirror will not be too large as it would be too difficult to carry. However, from marketability sense of view, if we made our smart mirror larger it may provide a better experience for touchscreen users, as it would reduce the chance of the user to accidentally touch and open applications that are not intended, thus reducing stress. However, if we made our smart mirror small, it may be useable to more consumers, because most consumers do not have a large surface area in their bathroom walls to mount said mirror.

3.0 Research related to Project Definition

This section of the paper will examine research that was done on the different technologies and other projects that exists closely related to our project. This includes any software or hardware components.

3.1 Existing Similar Projects and Products

Many other smart mirrors exist, especially custom projects done created by others online. All of them have similar technology and features and while we will use some of those, we want to make something a bit different.

Mirror Design

The common approach to the smart mirror is just fitting a display behind the mirror and securing it in a frame to keep the two attached. An important step some projects neglect is removing everything but the LCD itself and the driver/power board to allow the display as close to the mirror surface as possible. This will allow better viewing angles and visual fidelity overall.

3.1.1 Mirrocool

Mirrocool is a company that has already created their own version of the smart mirror. They offer their customers a choice of two types of smart mirrors, the unframed option is meant to be mounted on the bathroom and the framed option is meant for hallways. The mirror that’s meant to be mounted on the bathroom is IP65 water resistant certified, this rating means that the product is protected from dust and water that may harm the equipment. This product is like ours as it is also meant to be used in the bathroom. The screen is only visible when activated and it also allows the user to connect to their home WIFI. Their smart mirror also has a built-in high definition camera for the user to be able to user gestures and facial recognition. In our case, we have decided to go with a sensor to allow the users to use gestures to activate the smart mirror. They have also thought through the ethical issues in having a camera in the bathroom of the user and added on a privacy mode which essentially disables the camera all together. We have yet decided if we should add a camera in our smart mirror to be able to add in a facial recognition function. However, if we include this function in our camera, it would prove beneficial to our project. Mirrocool has also added in an app for smartphones so that the user may access the smart mirror functions remotely. This gives them the capability of using their smart mirrors as security cameras that takes pictures of guests that are unrecognizable by the mirror and sends a notification to the smartphone along with the picture. As we are building our smart mirror mainly for use in the bathroom, this function may prove unnecessary as most home intruders would most likely not go to the bathroom areas

3.1.2 Evervue MirrorVue Smart Touch Mirror

Evervue has innovated multiple different kinds of smart mirror variations. They have created smart mirrors for the bathroom, salon, conference room, retail, as well as showrooms. Their smart mirror has several different choices of applications depending on the operating system that the user requires (Windows, Raspberry, Android or Linux). Instead of just being able to see the daily weather and access the internet, it has other functions that would make it useable in most situations. Their smart mirror also provides the user with touchscreen capabilities, just as our smart mirror does. The differentiating factor is that we will use an IR frame to create this touchscreen capability, which mimics a resistive touchscreen, allowing only a single touch point on the mirror. Evervue has included a 10-point touchscreen in their smart mirror, which like capacitive touchscreens, allow for multiple touch points in the mirror. Also similar to our design, Evervue has included Bluetooth capabilities in their smart mirror to allow connections with external devices.

3.1.3 Savvy SmartMirror Electric Mirror

The company Savvy SmartMirror has created a smart mirror mainly for hospitality to be used in hotels. Besides the general functions like having a weather, time and date displays, they have added custom functions that would allow the hotel guests to see hotel advertising and promotions, as well as, amenities and services. They have included two different types of mirrors on their smart mirrors, one is the DuraMirror and the other is the Iris mirror. The Iris mirror will accomplish what we are trying to do with our mirror in that it will function as an average mirror when the display is turned off. The DuraMirror on the other hand goes and appears dark gray if the display is turned off. Just as Evervue did, the Savvy SmartMirror Electric Mirror also has touchscreen capabilities. This company also makes a voice smart mirror, which just as the name suggest, it responds to the hotel guest's voice. Voice technology is widely popular in the electronics market, just as shown by the innovations of the Amazon Echo and Google Home. As a stretch goal with our own project, our instructor has recommended us to have our smart mirror be able to function with the use of one of these voice technologies. This would give our smart mirror voice functionality which would increase the difficulty level of the project and set it apart from other senior design projects.

3.1.4 Biometric Verification

In researching technologies for the smart mirror, we looked at all types of products that implemented some form of facial recognition. From that researched what components would be necessary to build a functioning biometric system.

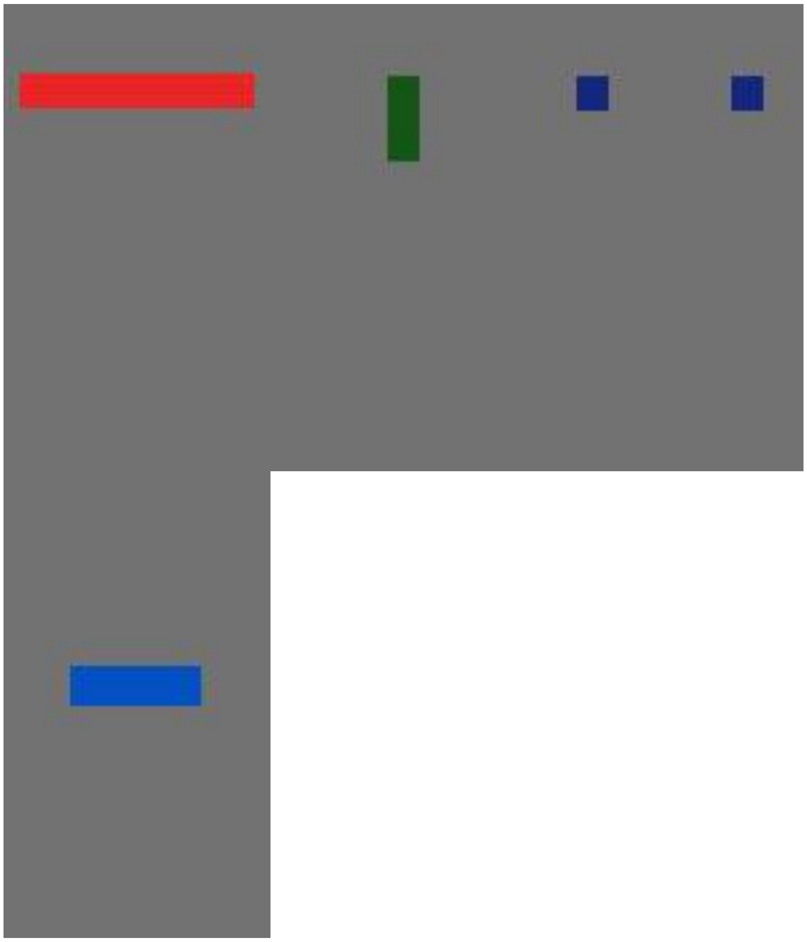
Facial recognition software applications are biometric based algorithms designed to identify the unique features of an individual by comparing segmented facial patterns to verify the identity of the person. By using deep learning algorithms, the software can analyze live capture features or use stored digital imprints to verify the identity of an individual by measuring their overall facial structure. For our particular system it was necessary to utilize a biometric technique of non-contact nature to limit physical contact with product to prevent residue and preserve visibility. Being one of the least intrusive and fastest forms of biometric technology, facial recognition as form of authentication has become one of the most viable options for security purposes. The feature measurements that are gathered by the algorithm are given a numerical code and stored in a database and then compared to other collected datasets of faces whenever a person stands in front of the imaging device. In today's market there are many consumer electronic companies that offer some form of facial recognition for both authentication and identification.

In 2009 Apple released its iPhoto application with both facial detection and recognition using the facial detection algorithm to determine if an objects features were at first consistent with those of face, and then facial recognition to distinguish people based on their unique features.

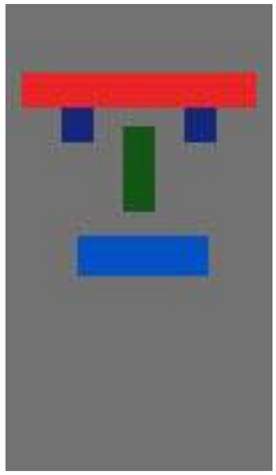
In 2017 the iPhone X was released with a facial recognition system for unlocking devices using a 3D mapped facial imprint stored in a local part of the processor.

3.1.5 Face Detection

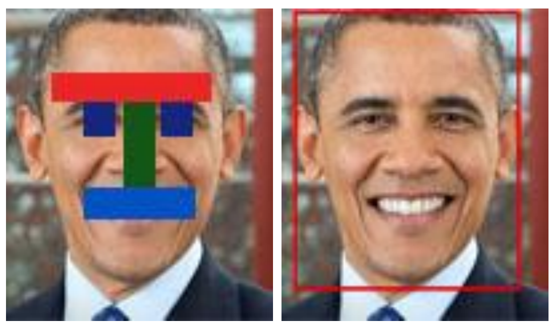
Face detection software extrapolates human faces from images and/or videos based on a set of mathematical rules used to define standard identifiers for human faces. Being that human faces do not drastically vary from one another we can compose a general structure for a face to use as a standard comparator: A forehead, two eyes, a nose and a mouth. For example, consider the following figures:



Each figure can be used as the representation of an individual feature. When we combine each figure, we have a construct that closely resembles the general structure of a human face.



Using this composite as a template for the algorithm to determine if objects captured in an image/video have features similar to the model, we can establish whether or not the object is a face. Because we are matching a template, we are not so much concerned with accurately matching the shapes of the model to image features; rather, we just need to determine whether or not the features exist.



The algorithm first compresses the image to reduce the amount of information being analyzed. When given an input the algorithm must detect human faces while at the same time disregarding peripheral objects. By iteratively matching features we are able to detect areas of an image or video that does not match a face. This process is repeated multiple times increasing the probability of accurately detecting face features. By continuously rejecting portions of the image/video that do not match the model we are essentially determining what is not a face. The elimination of features can be quickly compared to relative position validation.

Face Detection - Color

This method requires an image/video capture to be in color. The algorithm searches for colors that are typical of human a skin tone, it then searches each object to see if it has an aggregation of features that resemble a human face based on a preset algorithmic structure defined by the program.

Face Detection - Motion

People move in real time so when using video images or any live capture method movement can be used as a guide to capture the area of the image that is moving. Aside from the subject of the image there are peripheral elements within the frame that also exhibit either direct or relative motion. The software will need to create reference points that act as markers to determine whether or not the object in motion is actually face that is moving.

3.1.6 Face/Feature Extraction

One way to reduce memory consumption is to extract image features and use them we can use it. Features can be defined as basic patterns, distinctive attributes or parts of an image that we can use to describe the things we see in an image. For example, a mouth or a nose can be used as a feature on an image of a person. The human eyes and mind work in conjunction with one another to accomplish this However with physical images, using basic image pixels is not descriptive enough. The importance of the use of features in computer image processing is being able to convert visual information into digital information. This allows algorithms to use mathematical approximations for comparing various features between similar vectors. With representation learning, the idea is to apply a set of filters that are able to distinguish one classification of images from another classification through the use of specific algorithms. For example, Google's Image search uses the basic qualities of a given set of data with low reconstruction error. The specific variables capture different commonalities within images. For example, one unit might be if there is a hat in a given image or another might be a particular kind of shape present in an image.

Face Recognition

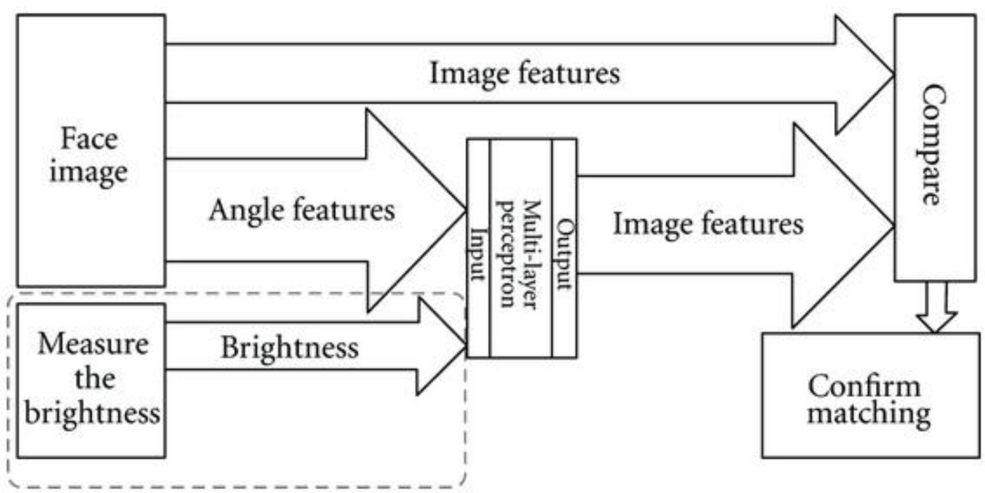
Facial recognition software is a type of software that can be used to instantaneously identify individuals from either the use of video or digital image capture. Most facial recognition software uses some type of algorithm that analyze particular features on an individual's face. This can either be accomplished using such things such as the relative position of points on a face or size and shape of an individual's features. Unlike fingerprinting and voice recognition, facial recognition software yields nearly instant results because subject consent is not required. Facial recognition software is primarily used as a protective security measure and for verifying personnel activities, such as attendance, computer access or traffic in secure work environments.

An advantage of live capture is that the appropriate action can be taken whenever the biometric data is gathered. Like other biometrics solutions, face recognition technology processes and identifies the distinct characteristics of an individual that can be used for identification or authentication. Often times using some form of digital or video/live capture device. Face recognition algorithms have the ability to identify faces in images/videos and can compute their features, and then validate them based on stored datasets within in a user database.

The are many live capture methods used in consumer facing devices. For example, there are some ATM's that use face recognition to verify that the individual making the transaction at the machine is the individual whose information matches the card being used at the machine. The individual has to look in the direction of the live capture device; images are captured for verification and transaction continues otherwise, the transaction is not cancelled.

Face recognition software requires a live capture device to use identification instead of passwords to secure user devices such phones, computers or and laptops. For example, whenever individuals sit in front of their computer, the camera uses the image which the software then processes the face to identify the person in front of the device. If the individual in front of the computer is a registered user, then he will be automatically logged on to the computer.

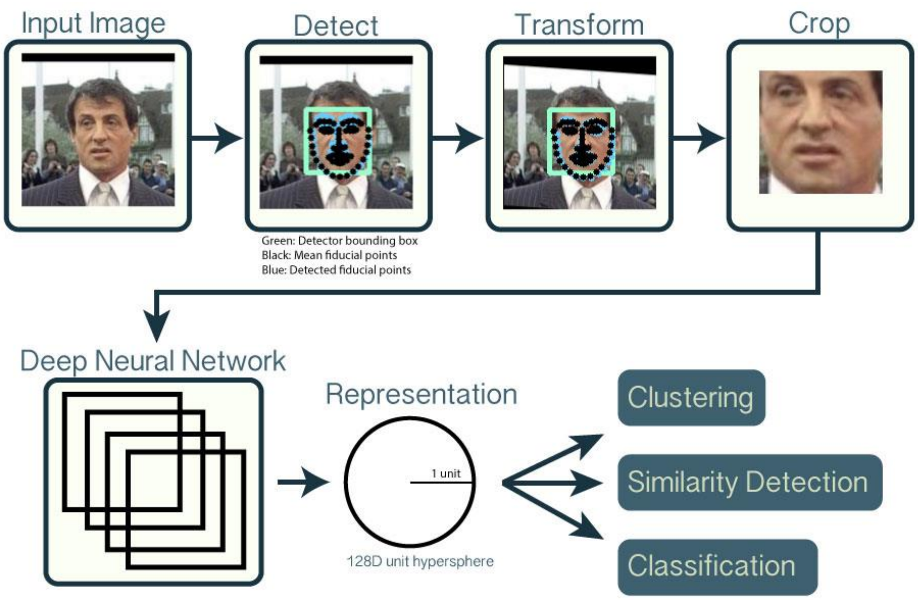
The main difference between detection and recognition is that with former the only thing we need to determine is whether or not there is an actual face in the image or live feed in front of the camera. With face recognition the goal is to determine the face and verify whether it grants access to the user. In the example above we used pattern matching to determine if the image had features consistent with a face this is detection. When the system is trained it would be able to should be able to recognize the person in the photo as the Barack Obama.



3.1.7 Face/Feature Extraction

Using OpenCV there are three methods implementing face recognition: Eigenfaces, Fisherfaces and Local Binary Patterns Histograms (LBPH).

Each of these three methods performs face recognition by comparing scanned faces to be recognized with some training dataset of already known faces. In the training set, we supply the algorithm with faces and tell it to what person the face belongs. When the algorithm is presented with a face to recognize and it is foreign to the system, it references the data set determine whether or not it recognizes the face. Each of method uses the information in the set a different way.



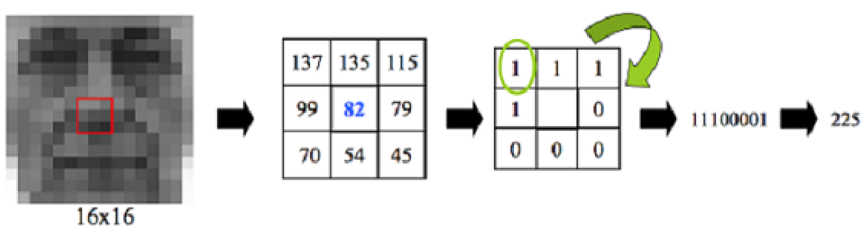
Local Binary Patterns

LBP’s are image descriptors that need to be trained using multiple images. Our faces are comprised of features that are micro visual patterns and can be extracted to create vector of features to classify faces and non-faces. Every image that is collect for training is divided into set number of blocks. For each block within the vector, the LBP segments 9 pixels at any given time, using the pixel located in the center of the frame. To determine the value for the centermost pixel we need a reference point. We can start from just about any pixel and move either forward or reverse, once we pick our ordering method it must remain consistent for all other block pixels in our image and any other images in our dataset.

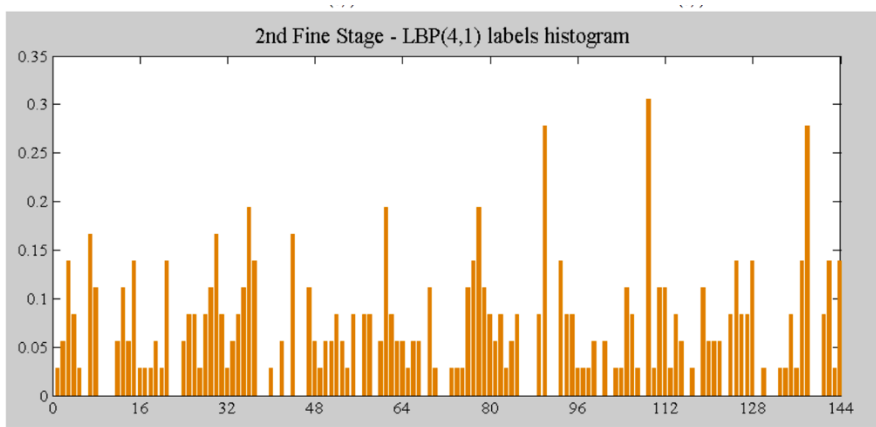
$$D = \sqrt{\sum_{i=1}^{n} (hist1_i - hist2_i)^2}$$

Euclidean distance formula for calculating distance

Given a frame of data, the algorithm must perform a binary test on every block within the frame . The results of this binary test are stored in an 8-bit array, which is then converted to decimal. The algorithm then compares the value of the center pixel with every pixel each adjacent value within the frame moving outward accordingly. For each adjacent pixel that is greater than the center pixel, it is a assigned value to 1, and for every pixel with value lower than the center they are set to 0. Once that process is complete, the algorithm then reads the updated value of the pixels in a clockwise direction and generates a binary number. It then converts the binary number into a decimal approximation, and the resulting decimal number is assigned as the new value for the center pixel. this is done for every pixel in a frame.



OpenCV also implements LBPs, but strictly in the context of face recognition — the underlying LBP extractor is not exposed for raw LBP histogram computation.



LBP Histogram. Source: López & Ruiz; Local Binary Patterns applied to Face Detection and Recognition.

EigenFaces

Eigenfaces are images that can be added to a mean (average) face to create new facial images. We can write this mathematically as,

$$F = F_m + \sum_{i=1}^{n} \alpha_i F_i$$

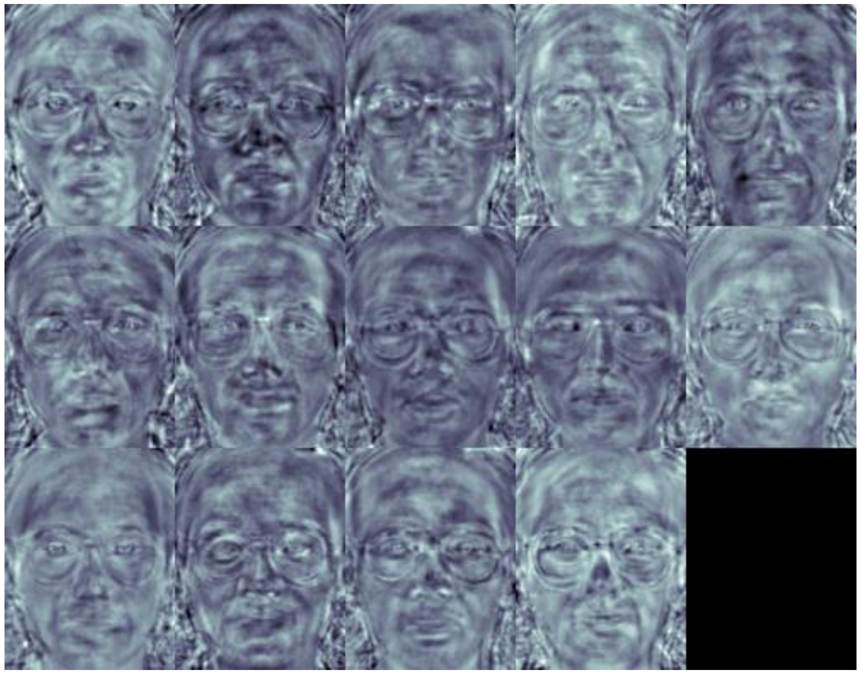
where,

- F is a new face,
- F_m is the mean or the average face,
- F_i is an EigenFace,
- α_i are scalar multipliers we can choose to create new faces. They can be positive or negative.

To calculate EigenFaces, we need to use the following steps.
Obtain a dataset: generate a collection of f images comprised of different kinds of faces.

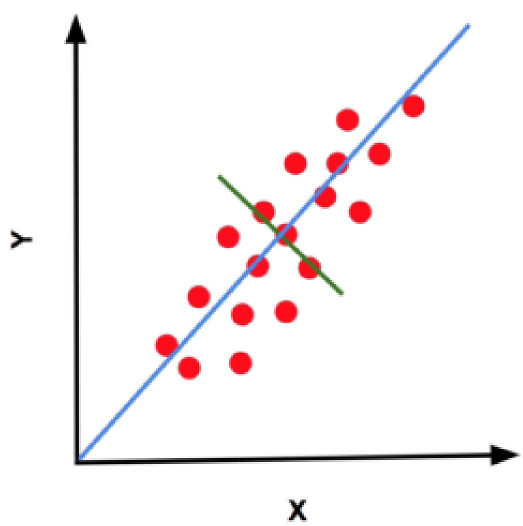
Align and resize images: align and resize all images to create standard center for the face so that all of the images are aligned. By using facial landmarks this calculated.

data matrix: Create a data matrix containing all images as a row vector.



Mean Vector: calculate the mean by averaging all the rows of the data matrix. The calculation is not a necessary component of the algorithm because OpenCV actually calculates mean using other linear algebra packages.

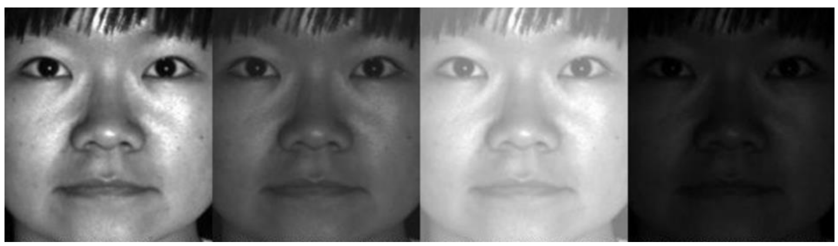
Principal Components: The principal components of this data matrix are calculated by finding the Eigenvectors of the covariance matrix. Fortunately, the PCA class in OpenCV handles this calculation for us. We just need to supply the data matrix, and out comes a matrix containing the Eigenvectors.



Reshape Eigenvectors to obtain EigenFaces: The Eigenvectors so obtained will have a length of 30k if our dataset contained images of size 100 x 100 x 3. We can reshape these Eigenvectors into 100 x 100 x 3 images to obtain EigenFaces.

FisherFaces

This particular algorithm is an improved version EigenFaces. EigenFaces inspects all the training dataset at one time and finds principal components from all of the training images combined. By doing this, it doesn't place emphasis on any feature that might distinguish one image from another. Rather, it focuses on the features that are representative of all the faces in the training dataset in its entirety. EigenFaces also uses light as a part of its calculation, it uses this difference as an aspect of face recognition. These variances that are extracted represents just a singles individual's features.



If we tune the EigenFaces algorithm to extract useful features from the face of each individual separately rather than extracting features from all them combined, regardless of the changes in illumination in any image it will not affect the other individuals feature extraction process. FisherFaces face recognizer algorithm extracts principal components that differentiate one person from another. In that sense, individual features do not become more dominant over the others.

3.1.9 Android Studio

We will be developing our interface using the Android platform currently more than 75% of Smartphone's use Android as their operating system. Android Studio uses the Gradle build system and uses modules to manage and organize codes. Every project has its own Gradle build files which means projects can state their own dependencies. The suite consistently updated and has very little bugs and provides a more stable performance guarantee. The Android device emulator uses multi-core processing, that makes development a lot faster. The emulator itself is faster than real smartphones, and new features can work best with the new emulator. Also, the new interface helps with fine tuning application by imitating interactive screen features. Once the app is completed the Google's Cloud Test Lab tests against a portfolio of physical devices. It works with proprietary tests developed by Google, that check for and correct any conflicts and other bugs.

Developers can also preview the GPU Developer, which lets developers debug issues related to graphics.

Software system

Our software system will be written in C++. This language was chosen because it is a partial Object-Oriented Programming Language. It is a compact language and executes quickly. It has a well-developed system of libraries which makes it a very versatile language. It also supports graphics and image processing (OpenGL/OpenCV) which will be a fundament component for our system.

3.1.10 Raspberry Pi Compute Module

The Raspberry Pi Compute Module (CM1), Compute Module 3 (CM3) and Compute Module 3 Lite (CM3L) are DDR2-SODIMM-mechanically-compatible System on Modules (SoMs) containing processor, memory, eMMC Flash (for CM1 and CM3) and supporting power circuitry. These modules allow a designer to leverage the Raspberry Pi hardware and software stack in their own custom systems and form factors. In addition these module have extra IO interfaces over and above what is available on the Raspberry Pi model A/B boards opening up more options for the designer.

The CM1 contains a BCM2835 processor (as used on the original Raspberry Pi and Raspberry Pi B+ models), 512MByte LPDDR2 RAM and 4Gbytes eMMC Flash. The CM3 contains a BCM2837 processor (as used on the Raspberry Pi 3), 1Gbyte LPDDR2 RAM and 4Gbytes eMMC Flash. Finally the CM3L product is the same as CM3 except the eMMC Flash is not fitted, and the SD/eMMC interface pins are available for the user to connect their own SD/eMMC device.

Note that the BCM2837 processor is an evolution of the BCM2835 processor. The only real differences are that the BCM2837 can address more RAM (up to 1Gbyte) and the ARM CPU complex has been upgraded from a single core ARM11 in BCM2835 to a Quad core Cortex A53 with dedicated 512Kbyte L2 cache in BCM2837. All IO interfaces and peripherals stay the same and hence the two chips are largely software and hardware compatible.

The pinout of CM1 and CM3 are identical. Apart from the CPU upgrade and increase in RAM the other significant hardware differences to be aware of are that CM3 has grown from 30mm to 31mm in height, the VBAT supply can now draw significantly more power under heavy CPU load, and the HDMI HPD N 1V8 (GPIO46 1V8 on CM1) and EMMC EN N 1V8 (GPIO47 1V8 on CM1) are now driven from an IO expander rather than the processor. If a designer of a CM1 product has a suitably specified VBAT, can accommodate the extra 1mm module height increase and has followed the design rules with respect to GPIO46 1V8 and GPIO47 1V8 then a CM3 should work fine in a board designed for a CM1.

Booting

The 4GB eMMC Flash device on CM3 is directly connected to the primary BCM2837 SD/eMMC interface. These connections are not accessible on the module pins. On CM3L this SD interface is available on the SDX pins.

When initially powered on, or after the RUN pin has been held low and then released, the BCM2837 will try to access the primary SD/eMMC interface. It will then look for a file called boot code. bin on the primary partition (which must be FAT) to start booting the system. If it cannot access the SD/eMMC device or the boot code cannot be found, it will fall back to waiting for boot code to be written to it over USB; in other words, its USB port is in slave mode waiting to accept boot code from a suitable host.

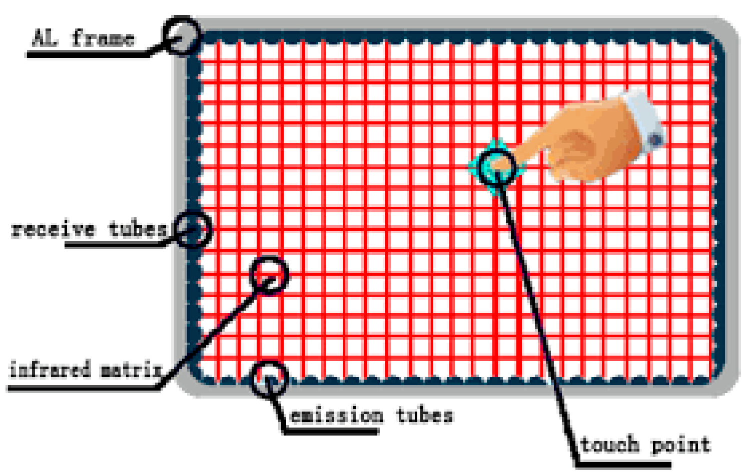
A USB boot tool is available on GitHub which allows a host PC running Linux to write the BCM2837 boot code over USB to the module. That boot code then runs and provides access to the SD/eMMC as a USB mass storage device, which can then be read and written using the host PC. Note that a Raspberry Pi can be used as the host machine. For those using Windows a precompiled and packaged tool is available. For more information see here.

The Compute Module has a pin called EMMC DISABLE N which when shorted to GND will disable the SD/eMMC interface (by physically disconnecting the SD CMD pin), forcing BCM2837 to boot from USB. Note that when the eMMC is disabled in this way, it takes a couple of seconds from powering up for the processor to stop attempting to talk to the SD/eMMC device and fall back to booting from USB.

Note that once booted over USB, BCM2837 needs to re-enable the SD/eMMC device (by releasing EMMC DISABLE N) to allow access to it as mass storage. It expects to be able to do this by driving the EMMC EN N 1V8 pin LOW, which at boot is initially an input with a pull up to 1V8. If an end user wishes to add the ability to access the SD/eMMC over USB in their product, similar circuitry to that used on the Compute Module IO Board to enable/disable the USB boot and SD/eMMC must be used; that is, EMMC DISABLE N pulled low via MOSFET(s) and released again by MOSFET, with the gate controlled by EMMC EN N 1V8. Ensure you use MOSFETs suitable for switching at 1.8V (i.e. use a device with gate threshold voltage, Vt, suitable for 1.8V switching).

3.1.11 The Optical Grid

Many devices today use a capacitive touch screen for input, making this form of interaction with devices come naturally to most people. However, when the main form of input for a device requires directly touching the screen, this results in oils being deposited on the display and smudges appearing. With an optical grid we can simulate the input of a touch screen which has become ubiquitous with modern technology but also allow people to hover their fingers just above the point they wish to interact with.



This allows the user to both interact with the device in a seamless manner while simultaneously preventing smudging on the screen. Our optical grid will consist of an array of optical sources, either laser diodes (LD) or light emitting diodes (LED) and opposite of them an array of photodetectors. Whether we choose LDs or LEDs will be determined after testing with both technologies to determine which is easier to work with from a packaging and calibration perspective.

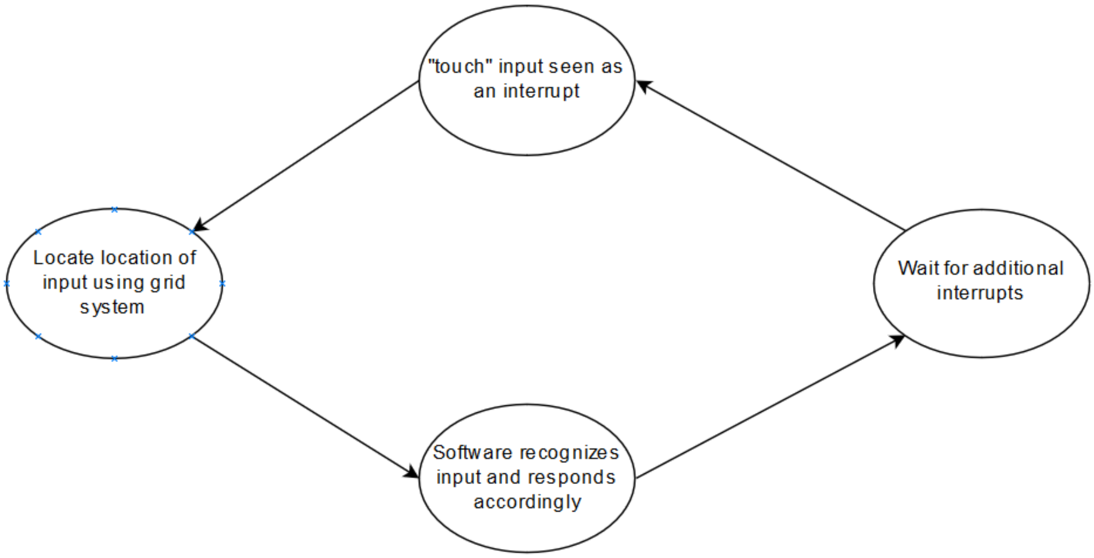
Theory of Operation

The light source and detectors will be embedded within all four sides of the mirror frame indicated in the figure to the left. Two sides will contain the light sources and the two opposing sides will contain the detectors. When aligned properly the light from each source should contact an opposing photodetector inducing a current to flow. When an object, such as a finger, is placed in between the light source and the photodetector, due to there being less light making contact with the photodetector, we can use this as an interrupt after calibration. Because of the arrangement of sources and detectors we can consider the surface of the mirror to be an x-y grid, thus allowing us to pinpoint the location of the "touch". This, in turn, allows the user to interact with the mirror as if it was a touch screen.

red arrows indicate sides with light sources

and blue arrows indicate sides with detectors

When this technology is implemented correctly, the optical touch grid will function as indicated by the following flowchart.



Functional block diagram of optical touch grid

Wavelength

Additionally, the wavelength we choose to use greatly affects the usability of the device itself. If we were to use light sources in the visible spectrum, between wavelengths of 400 nm to 700 nm, then the radiating light could affect both the visibility of the screen behind the semitransparent glass as well as cause additional eye strain from the extra light. For this reason, we have chosen to use light sources in the infrared region, which spans from the very edge of the visible spectrum at 700 nm to 1 mm wavelengths. There are multiple reasons and advantages to choosing infrared light sources instead of ultraviolet, which spans from wavelengths as small as 10 nm to 400 nm (the other edge of the visible spectrum).

The first reason we chose to use infrared light sources is it is a light source which emits less energy. The energy of a photon, a particle of light, can be determined by the following equation:

$$E = hf$$

Where E represents energy, h represents Plank's constant, and f represents frequency. We can determine the frequency of the light by dividing the speed of light, approximately three hundred million meters per second, by the wavelength of the emitted light. Using these known quantities, we can see that each individual photon in an ultraviolet light source contains more energy than both a photon from an ultraviolet light source and the visible spectrum. Since these spectrums of light are largely outside the visible spectrum, our eyes are not capable of dilating appropriately when exposed to large quantities of this light since it cannot detect it, which can result in eye strain and, in extreme circumstances, burning of the retina and permanent eye damage. So, by choosing a light source that emits less energy than the visible spectrum this issue is largely prevented.

3.2 Relevant Technologies

The following section of the paper will talk about all the relevant technologies that are available in the world today and how it relates to our project. It will also explain any current uses for this component, as well as how we will use it or something resembling it in our project.

3.2.2 Gesture Sensor

The gesture recognition system will use an Infrared LED and four photodiodes that sense reflected IR energy from the LED. There's many ways to do this but we're going to use a package from Broadcom called the APDS-9960 that has everything needed integrated into a small module. The detection range is 4-8 inches which will be perfect for what we want to do with it. It will allow us to easily detect hand gestures like swiping left, right, up, and down, and even more can be programmed ourselves.

3.2.3 Camera Module

The power to the camera will be switched using a MOSFET for security concerns. Connected to the power feeding the camera will be a status LED that will turn on when the camera is receiving power. Privacy is important, especially in a bathroom, so this lets users know when the camera is being accessed. This can't be disabled in software and that is important in keeping it safe and secure.

The camera we will use will most likely be a generic webcam because they're cheap and easy to setup. But it's also worth noting that the raspberry pi has a special high-speed camera interface that is just for that use. Raspberry pi cameras are available online that make use of this interface as opposed to USB which is slightly slower. The only negative to the Pi Camera is that it's more expensive but also comes with the advantage of not wasting a USB port so the sacrifice in cost might be worth it.

3.2.4 Serial Communication

Our project will have both a main processor and a microcontroller. The main processor will run the Mirror application, drive the LCD display, and monitor the Bluetooth and Wi-Fi connections. The microcontroller will be used to monitor the various sensors and relay that information to the main processor but also take actions from it was well.

Serial communication can be both synchronous and asynchronous. In synchronous communication methods, data is sent as alternating voltage levels on a wire and a second wire is used as the clock signal which signals when the data is valid at a certain moment in time. Asynchronous communication uses a oscillator that is required on both ends which determines exactly when data is to be written and read.

I^2C

I^2C is a very powerful and popular communication bus used for communicating between multiple devices. It was invented in 1982 by Philips Semiconductor, now NXP Semiconductors. It's usually used for attaching multiple low speed devices to processors and microcontrollers for data transmission across short distances such as a single circuit board.

The typical composition of a transmission is between a master and slave device. The interface consists of both a serial clock (SCL) and serial data (SDA). Both SDA and SCL are connected through a pullup resistor to the supply voltage. The communication happens by using a certain set of commands that are defined by the voltage on the two lines. The first command used to start a transmission is the START condition. A typical transaction between a Master and Slave will look like this:

Sending Data

- Master sends a START condition and addresses the slave to be communicated
- Master sends data to slave receiver
- Master terminates the communication with a STOP condition

Receiving Data

- Master sends a START condition and addresses the slave to be communicated
- Master sends the register to be read
- Master receives data from the slave
- Master terminates the communication with a STOP condition

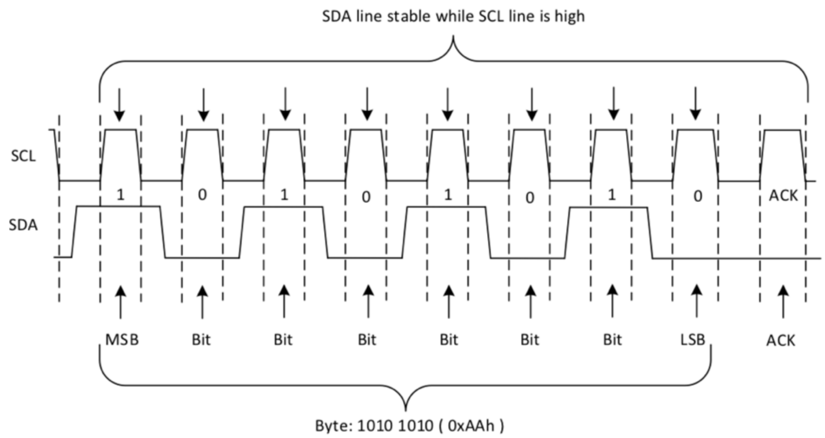


Figure 1: Single Byte Data Transmission

Because speed isn't a priority, our plan will be to use I^2C for communication.

SPI

SPI stands for Serial Peripheral Interface. It differs from I^2C in that it's a high speed synchronous serial communication method and also requires four wires instead of just two. So, this is the option we'd want if speed is an important factor in our needs for communication between the main processor and microcontroller. The four pins used are:

Pin	**Type**	**Function**
SCLK	Output	Serial clock output by master
MOSI	Output	Master output slave Input
MISO	Input	Master input slave output
SS	Output	Slave select(chip select)

Table 2: SPI Pins

The names for these data signals can vary depending out implementation but this is most common. In simplified words, SPI is using two shift registers to shift data in and out from the master to the slave and vice versa. It's a lot more complicated than that but typically specialized hardware included in a processor or microcontroller can make this easier. Although it's easy to see that not only does this use two more data lines than I^2C, but it's more complicated to implement.

UART

UART stands for universal asynchronous receiver-transmitter. It usually comes on an integrated circuit that is included in the microcontroller itself. It's similar to I^2C in that it can transmit data using two data lines but doesn't need a clock signal. Instead it sends start and stop bits to signal the start and stop of the data transmission. When a start bit is received, it starts reading at a specific frequency called the baud rate. The baud rate is a measure of the data transfer speed and it's unit is bits per second or bps. Both communicating UART modules must communicate at the same baud rate or close to it. A disadvantage to UART is that it doesn't support multiple slaves but in our case that's fine since the communication is happening between just two devices.

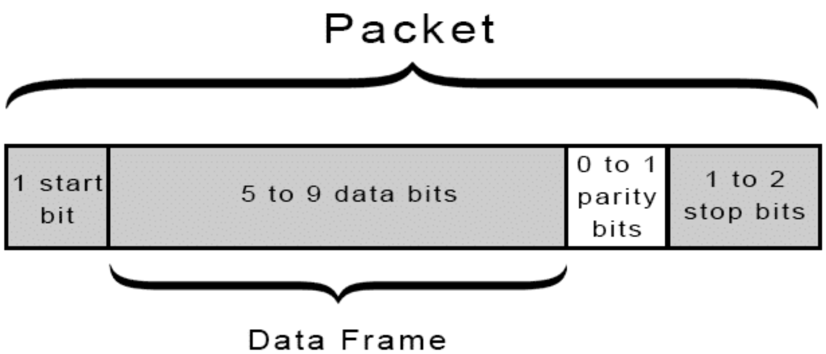


Figure 2: UART Data Packet

3.2.5 Multiplexer

Considering the limited number of analog inputs on our microcontroller, a solution will be needed to connect the large number of photodiodes used by the IR frame to the analog inputs. We will be doing this with two or more multiplexers connected to create one large multiplexer. A multiplexer will allow us to connect over 30 analog inputs with a sacrifice of 5 digital output pins. We have two options:

1. Use an individual multiplexer for the y and x axis
2. Use one large multiplexer instead

The benefit to the first option is that the board will be more organized and have an input for the x-axis LEDs separate from the input from the y-axis LEDs. This will also make coding the two a bit more organized and simple, but it comes at the expense of twice as many digital pins. If we're using two 16x1 mux's for example, each will use 4 digital pins making for a total of 8. While one 32x1 mux will use only 5. Currently we don't have room to spare 8 digital output pins but if we do find that there's extra pins we will use a second mux to simply software development and have one MUX for each axis. It's also important for us to use an analog mux since they come in a digital variety and analog. We will be reading values from a photodiode which is just a diode that starts conducting when light hits it. This means a voltage across a resistor will increase or decrease depending on the amount of light hitting the photodiode and that will be an analog value.

Analog Multiplexers:

Manufacturer	**Type**	**Supply Voltage**	**Package**	**Cost**
Texas Instruments	16:1	2-6 volts	24-SOIC	$0.08
Analog Devices Inc.	32:1	1.8-5.5 volts	48-TQFP	$10.8
Nexperia USA inc.	16:1	4.5-5.5 volts	TSSOP	$2.88
Maxim Integrated	02:1	4.5-18 volts	8-DIP	$4.08

Table: Available Multiplexers

I included a variety of Multiplexers that we might use and the relative specifications that will affect our project. Ideally, we want a large multiplexer for the most Photodiodes but 32:1 was the largest reasonably packaged available. The issue with this is that multiplexors of this size come in a very small TQFP package which can be quite difficult to solder by hand. So, it might be a better option to use two 16:1 multiplexers if we want a 32:1 device. I included the specs for a 2:1 mux for this exact purpose. It comes in a through hole DIP package which will be very easy to solder. The 16:1 mux still come in a surface mount SOIC or TSSOP package, but the pin pitch is larger, and they will be easier to solder overall. Although if we decide that a 64:1 mux is needed the 32:1 mux will be necessary. Unless we can find a 64:1 mux in a package we can reasonably solder to our board. The process for connecting two mux's is easy and is shown below:

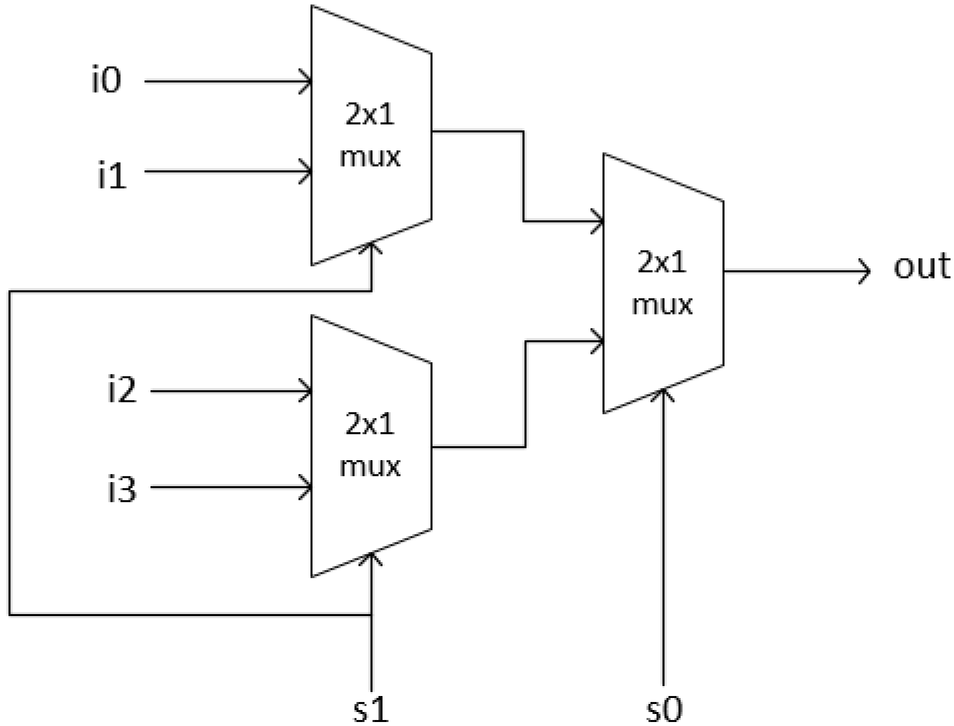


Figure: 3 Multiplexers used to create a larger one

3.2.6 Printed Circuit Board (PCB)

The PCB is found in most, if not all, electrical devices, such as, beepers or pagers. The PCB is a way to connect all the electrical components of a project together. PCBs are made by using a conductive material, typically copper foil, on a non-conductive surface, typically fiberglass. There are three different types of PCB constructions available as of today, single-sided, double-sided, and multi-layered. As the names suggest, the single-sided PCB has the conductive material printed only on one side of the circuit board, whereas the double-sided PCB have the conductive material printed on both sides. However, the last type, the multi-layered PCB, is different from the other two types. It consists of several layers of printed circuits with insulation separating the different layers. Despite the meticulous construction, the multi-layered PCB simplifies circuit patterns.

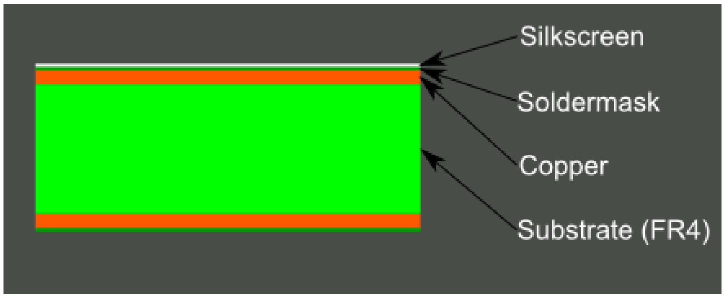


Figure 3: double-sided PCB construction

As for the components mounted on the PCB itself, there are two different methods to do so. The much older and outdated method is to simply place the component through the board and solder or glue the pieces in place so that it meets the conductive material. While this is sufficient method, it raises a few issues because if the components should become non-functional it would be more difficult to replace. Also, with solder it would increase the likeliness of a short to occur if the solder should come into contact with each other. The newer method involves surface mounting components into the PCB. This method is much more effective as it simply allows the user to replace non-functional components with another without the need for unsoldering and re-soldering the pieces into place. It also reduces the risk for solders to come into contact with each other and shorting out connections, thus killing two birds with one stone.

3.2.7 Power Supply

Research into the power supply is important for this project because we have a lot of different components and voltages to take into consideration. The first step of the power supply will be AC to DC conversion because our circuits will run on DC current as do most electronics. The AC will be stepped down to a lower voltage using a small transformer and then a full bridge rectifier will rectify the AC to a DC supply.

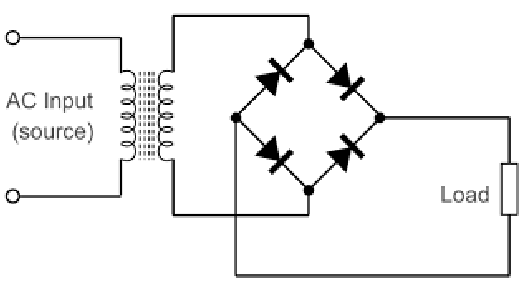


Figure 4: Full Bridge Rectifier

Some parallel capacitors will be added to the output to the rectifier to give a stable DC, otherwise there would be noticeable drops in the supply. In some circuits this wouldn't matter but for a microcontroller it's important to have a stable DC input voltage. The rectified DC will be most likely 12-24 Volts depending on what our LCD display operating voltage is and that will be powered straight from there. Next will be dropping this voltage to 5 Volts to power out microcontroller, Raspberry Pi, and the various sensors and camera we will include. We have multiple options of doing this, one being a linear regulator, and another being a switching one. We're going to be using a buck converter or DC/DC converter that will step down the 12-24 volts to 5 volts for our other electronics. The reason we will be using a switching regular is because it is more efficient, and we want less wasted energy. It also produces less heat which means we can get away with not using a heat sink for the regulator.

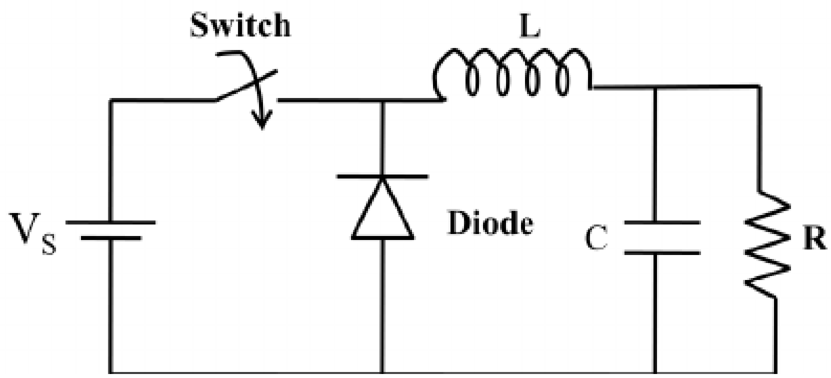


Figure 5: Buck Converter schematic

A buck converter typically consists of two semiconductors, a diode and a transistor which acts as the switch. Two timing components are also used such as a capacitor and inductor.

While these take care of the power regulation another aspect of our power design will be power MOSFETS that will switch on or off the power leading to the display and camera module. This will have to be incorporated either into our power supply or somewhere on the Microcontroller board. If we do decide to build our own power supply, that will increase cost as an extra PCB will be needed but the advantage is that it will most likely decrease size and allow us to have a smaller form factor in our final design as opposed to just throwing an off the shelf power supply onto the board. We could also go with the option of buying a 24 Volt AC to DC power supply off the shelf and then just doing all the DC regulation on our existing circuit board to minimize cost and that might be the best option for us at this time. This means we could simplify everything to one, cheaper, board and not worry about the AC step down and rectification.

Components	Required voltage
Gesture sensor APDS-9960	3.3v
Raspberry pi	5v
Atmega328	5v
IR Frame LEDs & photodiodes	3.3v

Table 3: Voltages required for various components in the project

3.2.8 Wireless Technology

The wireless technology refers to the capability of remotely connecting other technologies to this project, as well as, the capability this project has to connect remotely to other technologies. In our case, we are using Bluetooth to have the user's smartphone to connect wirelessly to our product. This product is also having the capability to connect to the internet via WIFI technology.

The use of wireless technologies makes our smart mirror innovative and give it a sleek and neat design. It also increases the overall ease of access of the product, which is what we are aiming for. Having less wires like ethernet cables in the bathroom, which is where our product intended use area is, minimizes the chance of accidents to occur. Therefore, also tackling a safety concern while remaining innovative.

Bluetooth

One of these wireless technologies we plan to use in our project is Bluetooth. Bluetooth technology use short radio waves of mobile devices to create personal area networks (PANs). It was made as a wireless alternative to the RS232 data cables, making it more convenient and capable of connecting to several devices. Simply put, this technology acts like a wire to connect two technologies together. We will repurpose this widely popular technology in use for our project to allow users to connect their smartphones wirelessly to the smart mirror. The connection would in turn access user specific data, making it convenient to check on things like their personal emails and social media. In the case of our smart mirror project, the smart mirror would have the master role and the user's smartphones would have the slave role. The smartphone works like a key to accessing user data stored in the smart mirror. The different keys open different sets of data.

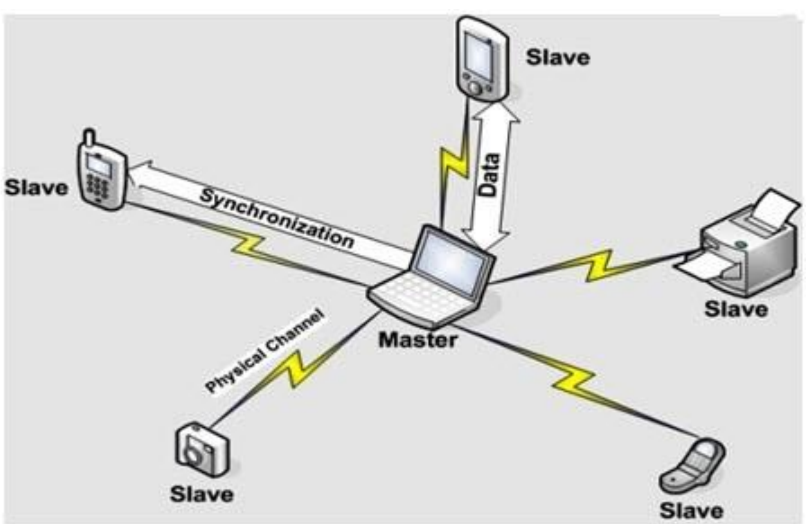


Figure 6: Visualization of how Bluetooth works

WIFI

WIFI is a networking technology, also using radio waves, that allow high-speed data transfer over a short distance. Its innovation allows local area networks (LANs) to operate without the use of cables. The use of internet is probably the most popular thing amongst today's population. By integrating this wonderful technology into our smart mirror, it would increase the versatility of our project. The addition of this technology would allow users to check live weather conditions for the day before going outside. It would allow them to access their social media and emails, allowing the Bluetooth technology mentioned before to have a use in our project.

3.3 Strategic Components and Parts Selections

This section of the paper will explain the different parts and components that will be used in the project. It will explain the specs of each part or component and explains why that specific part or component will be used in the project.

3.3.1 Microcontroller Considerations

The microcontroller in our project has two roles. One of which is to monitor the device inputs such as the IR touch sensor and also gesture sensor and then relay that information to the Raspberry Pi. This also includes receiving commands from the Pi, such as to turn power to the camera on and off when it is needed.

The second use is to monitor activity in proximity to the mirror itself. This will be a power saving feature so that when nobody is using the mirror, it will cut off power to the display to save power. Neither of these is too intensive so a basic low power microcontroller will work for our needs.

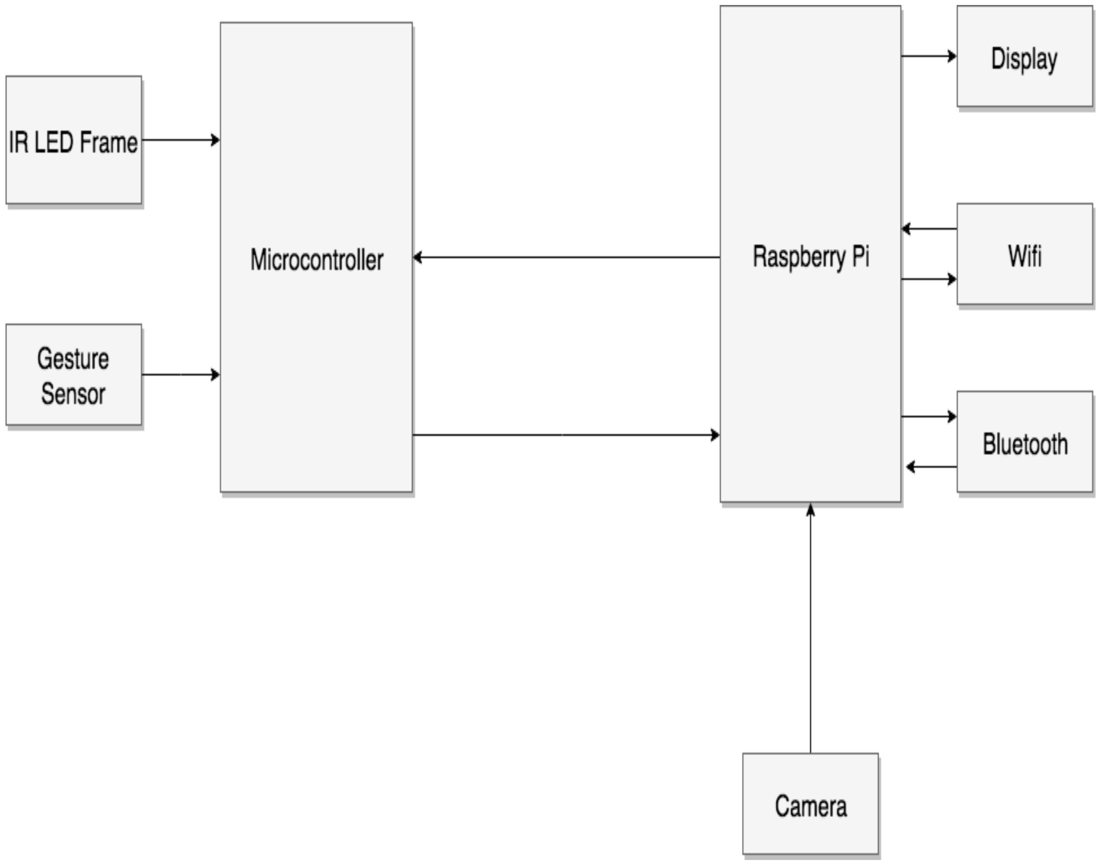


Figure: Complete connections between microcontroller and microcomputer

Texas Instruments MSP430FG461x

The MSP430 microcontroller from Texas Instruments is the first microcontroller we considered based on our familiarity with it from previous classes and its low power modes of operation. The positive’s to using the MSP430 are that it’s programmed in C which is simple and familiar to the group as a whole. It also has a JTAG interface that allows debugging of the microcontroller at a closer level. The only negative I see to the MSP430 is that it doesn’t have as many available libraries as some competitors do. Although its biggest advantage is the multiple low power modes it has built into the system. These would really be useful for our project since when the mirror isn’t being used, we want it to be consuming very little power.

Specification	Value
CPU Architecture	16-Bit RISC
Operating Voltage	1.8-3.6 Volts
Clock Speed	Multiple Oscillators allow the microcontroller to have different clock speeds ranging from a few KHz to 25MHz
Flash	92-120 KBytes
ROM	0-120 KBytes
RAM	4-8 KBytes
Communication	Enhanced UART, SPI, and I^2C
Inputs/Outputs	80

Table 3: MSP430FG461x General Specifications

Atmel ATmega328

The second microcontroller considered is the ATmega328. It's an 8-Bit low power microcontroller that I would say is the most user-friendly option on the market. It also meets all the requirements our project needs for our current implementation. Considering our needs are simple, a powerful microcontroller really isn't needed.

The most important features for our project will be serial communication and low-power modes of operation. Since the microcontroller will just be reading sensors and communicating back to the main CPU, those needs are easily met. The only issue here might be the availability of I/O pins. Our IR touch sensor will have a large amount of output pins to be read by the microcontroller, but a multiplexor can easily solve this problem. We also must consider the MOSFET outputs needed for switching the power to the camera and display as discussed above.

The greatest advantage that the ATmega328 has over some of its competitors is huge community support and a plethora of libraries available online for Serial communication, LCDs, Wi-Fi and Bluetooth devices, various sensors, and many more. Installing the Arduino bootloader onto this microcontroller will even allow it to be programmed with the Arduino IDE. This will make our prototyping stage much easier and cheaper than any other option because we will be able to just use an Arduino Uno device and Raspberry Pi to prototype our system.

Specification	Value
CPU Architecture	8-Bit RISC
Operating Voltage	1.8-5.5 Volts
Clock Speed	0-20MHz @ 4.5-5.5 Volts
Flash	32KBytes
EEPROM	1KBytes
RAM	2KBytes
Communication	UART, SPI, and I^2C
Inputs/Outputs	23

Table 4: Atmega328P General Specifications

Texas Instruments Tiva C Series

The Tiva C Series from Texas Instruments is a microcontroller based on the Arm-Cortex M4 architecture. These days ARM is a name we hear all too often so it's hard not to take a look at what they have to offer. ARM's architectures are used in everything from smart phones, tv's, laptops, and smart watches. I've seen them in almost every new piece of technology today, so they were heavily considered for use in our project. Early on in planning our project, the microcontroller was going to be the main processor and we were looking into running an operating system called RTOS on one for managing processes and possibly using an FPGA for driving the display. I've seen this done in other products before and it looked promising so that was the early idea. After more research though it looked like using OpenCV would be difficult on this microcontroller and so we decided on using a Raspberry Pi as the main processor and the microcontroller was dropped to just managing smaller systems in our project. So, with that change, using an Arm M4 based microcontroller was overkill in terms of what we wanted to accomplish with it.

Specification	Value
CPU Architecture	32-Bit RISC
Operating Voltage	1.5-5 Volts
Clock Speed	80 MHz
Flash	256KBytes
EEPROM	2KBytes
RAM	32Kbytes SRAM
Communication	UART, SPI, and I^2C
Inputs/Outputs	43

Table 5: Tiva C Series specs

So now we must compare these microcontrollers and choose one that fits our needs the most.

Property	TI MSP430	ATMega 328P	Arm Cortex-M4
Low Power	Yes	Yes	Yes
Arduino Compatible	No	Yes	No
I/O	80	23	43
Max Clock Freq.	25MHz	20MHz	80MHz

Our set of requirements are simple and so the features of each microcontroller boil down to these key factors.

3.4 Low Power Consumption
3.5 Arduino Compatibility
3.6 I/O
3.7 Max Clock Frequency

Low power consumption is important in our list because ideally the Microcontroller will be the only device consuming power when the mirror is off and that will be the bottleneck in our power consumption. Arduino compatibility is important to our project because it will simplify the coding process very much and open a world of libraries for use from sensors to Bluetooth modules. This will allow us to focus more on the system design and less on how to implement all the small details. I/O is the most important because we need enough inputs/outputs to interface with our various sensors and control electronics from multiplexers to MOSFETS. The last feature we are observing is the clock rate of the microcontroller. The clock rate will determine how fast our programs can execute and that can affect how quickly our operating system on the Raspberry Pi will be able to update the user interface. If the microcontroller clock is too low, our program will noticeably skip as it takes input because it’s constantly waiting on updates. Our choice for microcontroller will be the ATMega328P for its many online resources, tutorials, and libraries available.

3.3.2 IR Light Sources

In order to embed our light sources into the frame of the smart mirror we have narrowed our options down to two different technologies: the LED and the LD. The reason we have chosen these two options as candidates for integration into our optical grid is due to their small size, low power draw, and most importantly, their availability in the desired wavelengths. In the following text we will discuss the merits of both technologies and the reasoning for choosing the LED over the LD.

Laser Diode

The LD holds one distinct advantage over the LED, that being its significantly narrower bandwidth. The narrower bandwidth of the laser diode, in theory, would make it much easier to calibrate the photodetectors and filter out unwanted noise and detect only the center frequency. The disadvantage of using a LD, however, is the alignment. One of the distinct differences, other than the bandwidth, of a laser and a LED is the rate at which the light from each source diffracts. Since a laser diffracts very little in comparison to a LED the alignment must be very accurate to ensure adequate contact between the light and the photodetector. In addition to this, we must ensure that, once aligned and assembled, we have secured both the photodetectors and the LDs well enough to prevent any misalignment occurring due to transportation.

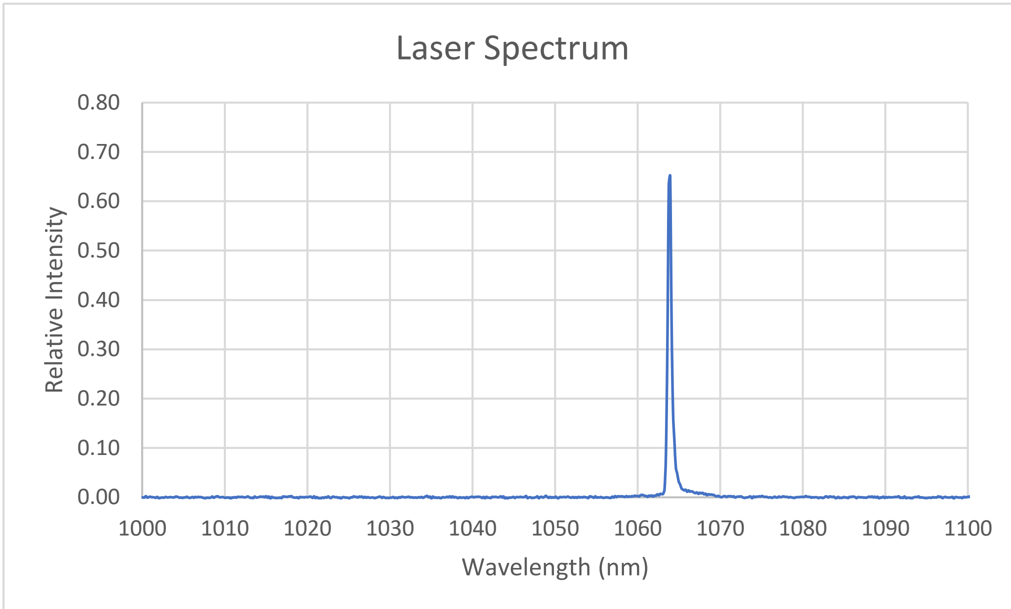


Figure 7: Spectrum of a laser diode

Light Emitting Diode

While cross contamination and noise concerns may favor the LD, the LED offers a wider variety of packaging options which make it favorable. LED strips can be purchased allowing for simpler assembly. Additionally, due to the LED emitting light in a nearly omnidirectional manner, alignment becomes less of a concern. However, this results in another issue taking its place. Specifically, the highly diffracted light being emitted will result in excess noise on the adjacent detectors, which ideally would only be receiving light from the corresponding LED opposite of it. This is both the most significant advantage and disadvantage of using LEDs as our light source.

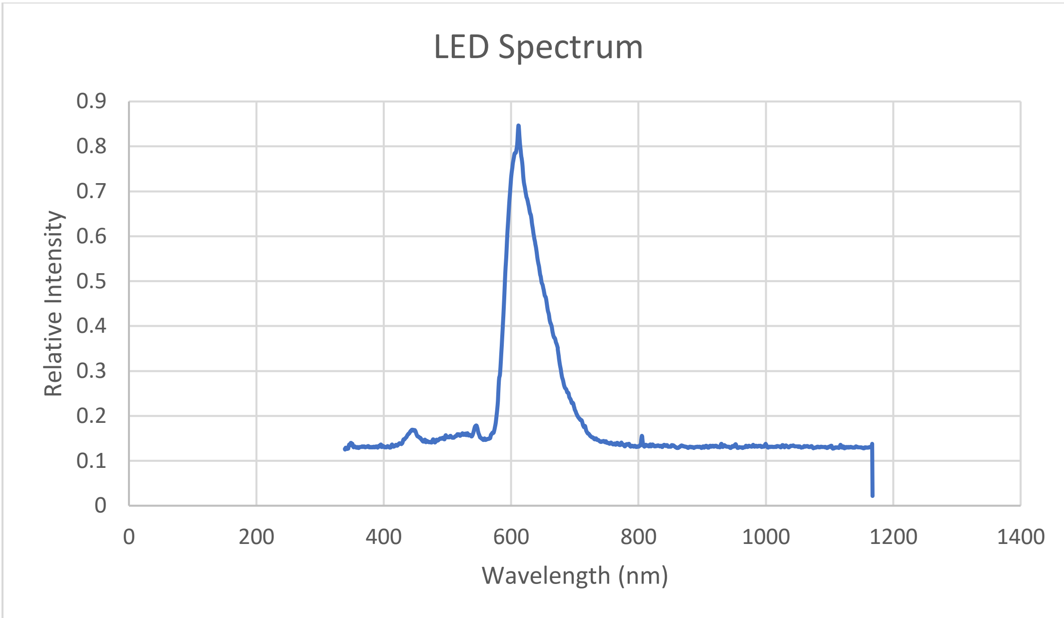


Figure 8: Spectrum of red LED

Noise

Since it is likely that the photodetectors will pick up a range of wavelengths around the specified frequency, greater care will be needed if using LEDs since their emitted spectrum is far greater than that of the laser.

The LD, due to the inherently low diffraction and narrow spectrum of a laser is unlikely to produce a significant amount of noise or cross contamination if implemented into the optical grid when compared to the LED, thus noise would effectively be a non-issue.

A possible solution to the excess noise which would be Introduced when using LEDs as the light source would be to confine the photodetector with a straw as shown in the following figure.

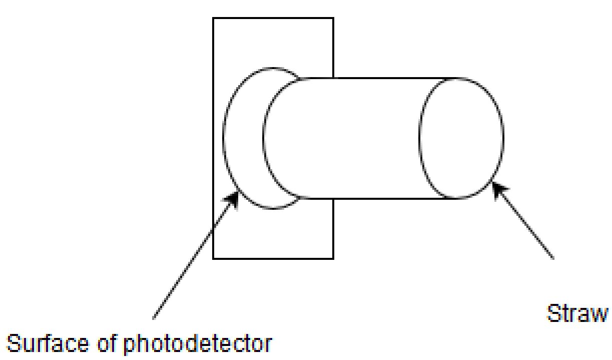


Figure 9: Illustration of noise limiting straw

By placing a straw in front of the surface of the photodetector we can limit the detected light that comes at an angle. This, in theory, would reduce the amount of noise from adjacent LEDs and allow us to better calibrate and filter out noise for the optical grid.

An alternative solution to dealing with noise is by using LEDs of differing wavelengths and photodetectors that detect at the appropriate wavelengths. Two commonly available IR LEDs are those that emit 850 nm light and 940 nm light. By setting them up in an alternating pattern such as the one illustrated in the figure below, we can use photodetectors that specialize in detecting their corresponding wavelengths.

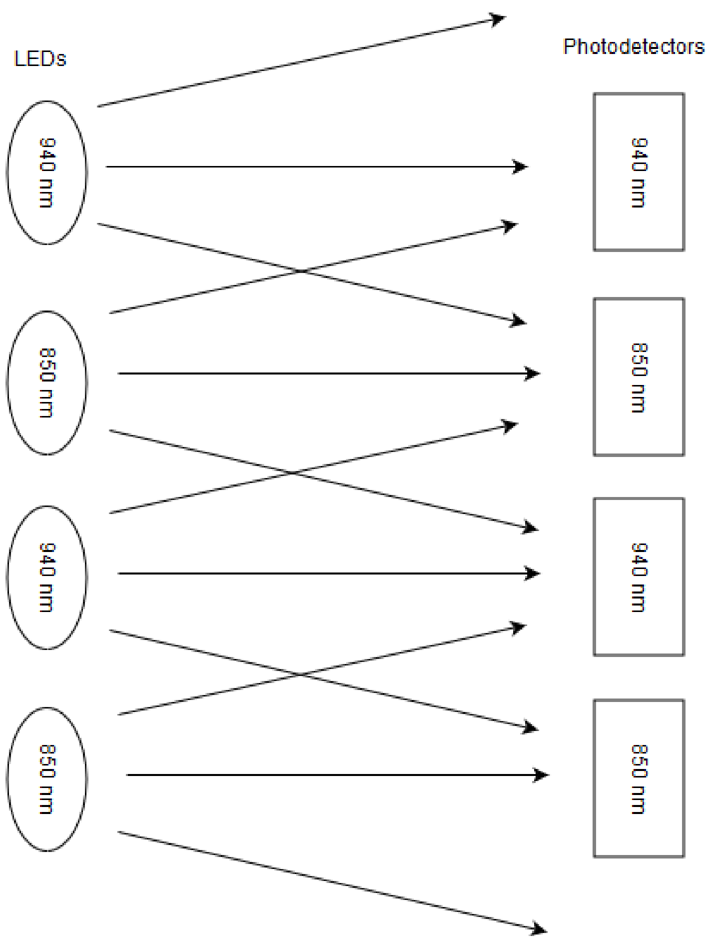


Figure 10: Alternating pattern of LEDs and photodetectors

While this method will still result in cross contamination, it will not have a significant impact or increase in noise due to the different wavelengths being detected by the adjacent photodetectors. With regards to packaging this method will be the simplest to implement since no additional components are required other than the light sources and detectors, which are required regardless of the optical grids implementation methodology.

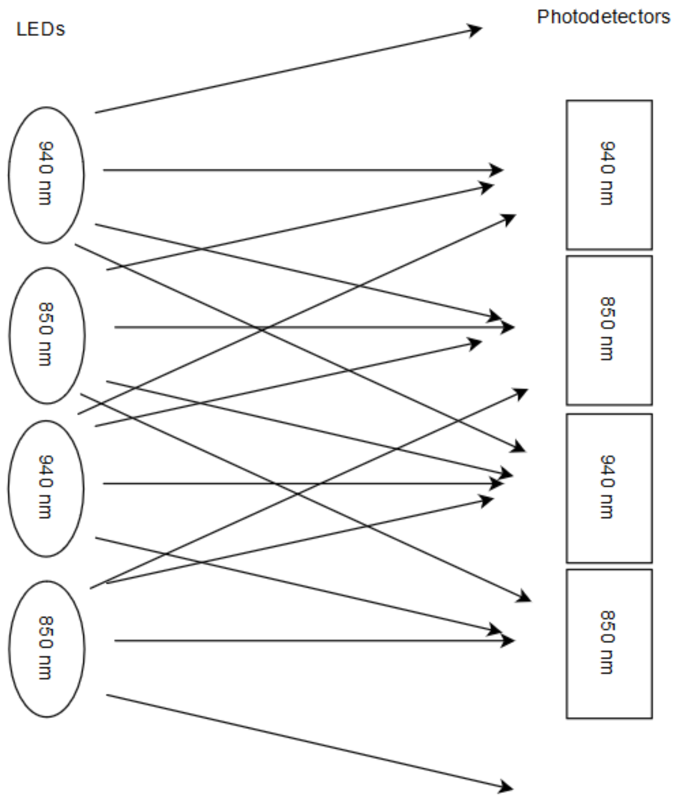


Figure 11: cross contamination due to increased "touch" resolution

While this method will reduce the amount of cross contamination from adjacent IR light sources, it will not completely eliminate them. The biggest factor in how effective this method is will be the resolution of the "touch" input. In order to increase the resolution of the input the number of detectors and light sources must increase. This results in the spacing between the IR light sources and photodetectors decreasing. By decreasing the space between adjacent components, we may end up with a situation such as the one illustrated in the previous figure where the spacing is close enough that a significant amount of light from LEDs emitting the same wavelength of light provide a significant amount of cross contamination.

Ultimately this method leads to a tradeoff, increased resolution in exchange for increased cross contamination between the light sources and detectors.

Choosing the LED

Ultimately, we decided to use 940 nm LEDs with photodetectors that are sensitive to that specific wavelength. Our reasoning for this is due to the LEDs we have decided to use have an emitting angle of 30 degrees. While this will still result in more noise than a laser diode, the excess noise on the adjacent photodiodes can largely be prevented by including straws as depicted in Figure 9 in front of the photodiodes, making the need for LEDs with two different center frequencies unnecessary.

Additionally, using LEDs instead of laser diodes allows for simpler assembly of the mirror frame. Since a laser diode requires some type of heatsink to prevent overheating and failure of the laser diodes. In contrast, an LED puts out minimal heat, allowing for more options on assembly.

The final reason for choosing LEDs instead of laser diodes is cost. Cheap laser diodes can commonly be found in packs of 10 for about the same price as a pack of 100 LEDs. By using LEDs we are able to reallocate funds to other parts of the project.

Table 1 – Summary of the pros and cons of the two IR emitter technologies being discussed

LED vs Laser Diode		
	Pros	**Cons**
LED	Low heat output and power draw	Requires straws to prevent cross contamination on photodiodes
	Cheap	Wide bandwidth
	Wide emitting angle allows for some misalignment	
Laser Diode	Narrow bandwidth	Hot, requires heatsink
	narrow emitting angle, no cross contamination	Expensive
		Requires near perfect alignment with photodiodes

3.3.3 Photodetectors

Through research we have studied various types of photodetectors and their typical applications. After coming to an understanding of which type of photodetector is best suited for interaction with a microcontroller, we have come to the conclusion that, between the photoresistor and photodiode, the photodiode best suits the requirements for the smart mirror.

Photoresistor

The photoresistor is a light-dependent variable resistor. The resistance decreases as the light shining on the photoresistor increases. A photoresistor is a passive component without a PN junction and is less sensitive to light then the other two options which will be discussed. While a photoresistor was considered for use in this project, it was decided that the alternatives provide a potentially better solution since they are active components. As a result of the photoresistor being a passive component without a PN junction the change in resistance is gradual, making it more difficult to digitize the resulting signal.

Photodiode

The photodiode is a device which converts light into current. Electrically it is connected to a circuit the same way a normal diode is, the difference is how it is turned on. A photodiode must be exposed to enough light in order to conduct current.
When discussing photodiodes, there are two operating modes that must be defined: photovoltaic mode and photoconductive mode.

When a photodiode is operating in zero bias, meaning no external voltage is applied to the diode, the flow of current is restricted, and voltage gradually builds up. This phenomenon is due to the photovoltaic effect and is thus called the photovoltaic mode.

The photoconductive mode is used when the photodiode is reverse biased. When in the photoconductive mode the amount of current is linearly proportional to the irradiance, which is defined as the amount of power per unit area provided by the incoming light. For the purposes of this project it is likely that we will use the photoconductive mode as it seems to be common to convert the resulting current into a voltage that can be digitized.

3.4 Possible Architectures and Related Diagrams

Given the allotted time that we have, we had to trim down this project to fit our timeline. However, should there be a change of pace that allows us to finish our project in an earlier time than projected, we would like to add certain things to this project to advance it even further. This collection of different components/architectures is called our stretch goals.

3.4.1 Power supply

Currently, we have decided to go along with any available power source that would fit the necessary constraints of our project. We are aiming for our smart mirror to be used in the bathroom to ease access for morning activities. Since most bathrooms only have sockets, the power supply we need to design needs to be an AC/DC power supply. Currently each component would require their own AC to DC converter, which are quite bulky, therefore requiring a power strip. Since our Smart Mirror is meant for use in the bathroom, it would pose a safety concern if a power strip is anywhere near the sink. However, should time allow, we have considered constructing our own power supply that would be custom fit to work with this project. The custom power supply would allow us to have less components having to use their own individual power supplies thus cluttering the socket area. We have considered the idea of adding a built-in battery in our smart mirror to combat this situation. However, since we are building this smart mirror with ease of access in mind, the fact that the users will have to charge the mirror when it runs out of battery seems slightly inconvenient. If the user also has the power continuously charging to combat this inconvenience, it would affect the battery life of the product. As for things like batteries, it would require the user to continuously remove and place the smart mirror back on the wall, which once again is inconvenient. Also, it would greatly increase the risk of the user of breaking the mirror and injuring themselves. With all these in mind, we have decided that a normal AC/DC power supply would be the best thing to use for our smart mirror.

There are two different types of power supplies, one is regulated and the other is unregulated. The regulated power supplies provide a constant voltage with a range of values for current, while the unregulated provides constant power with varying voltages and current. With the regulated power supplies, multiple regulators can be used to offer multiple output voltages to operate different devices. Voltage regulators are used to put the voltage source to a certain amount that is required by the device. There are two types of voltage regulators as well, there's the linear and the switching regulators. The common topologies used with these switching regulators are buck (step down), boost (step up) and buck boost (step up/ step down), it basically refers to what the regulators can do to the given input voltage. The linear regulator on the other hand just keeps the output voltage at a constant rate based on the input put into it. Efficiency wise, the switching regulator is supposedly 90% efficient while the linear is only 50%. In terms of cost, the unregulated power supplies are cheaper than the regulated power supply. For our purposes, we probably would want the regulated power supply because each of the different components that we have require different types of voltages. For the purposes of energy efficiency, we may use the switching regulator. However, due to higher cost ranges for the switching regulator we may have to use the linear regulator instead.

Table 5: Linear vs. Switching regulators

	Linear	**Switching**
Efficiency	Low with high voltage difference between input and output and high between low voltage differences	High
Complexity	Low	High
Size	Larger at high power	Smaller at high power
Cost	Low	High
Ripple / Noise	Low	High due to high frequency

3.4.2 Compatibility with Voice Technology

During our divide and conquer paper discussion with the instructors, they suggested for us to make our projects to be able to work with existed voice technologies, such as, the Amazon Echo. From past senior design groups, the instructors have seen a trend of popularity amongst the judges to see senior design projects that are integrated to work with voice technology. The integration of our smart mirror with these voice technologies would require more than a simple Bluetooth connection. The voice technology device would have to have access to all the functions of the smart mirror for it to be able to use those functions. To do this, it would require for us to tweak the software in our smart mirror in such a way that would allow for this connection to occur. When done successfully, this would give us yet another method to access our smart mirror, thus furthering its ease of accessibility. This would also add depth to our project and thus would most likely greatly impress the senior design judges.

3.5 Parts Selections Summary

The following section of the paper will explain in detail the parts that are chosen for this smart mirror project. It will give explanation as to why those parts are chosen over the other parts. Important aspects, such as, price, functionality and other limitations will be included.

3.5.1 Voltage regulator selection

Based on the voltages required of each component from table 3, we need a 5v DC voltage regulator for the Atmel ATmega328 microcontroller and the Raspberry Pi 3 model B. There are two different choices to use for the voltage regulators, switching and linear. The Raspberry Pi is more demanding in terms of current, with the regulators it may be better to use a switching one to avoid overheating. However, some testing was to be done to see if the switching regulator was necessary. Based on that, the regulators we have chosen are, the LLM7805CV and the LM2598.

Table: comparison of 5v voltage regulators

LM7805CV	**LM2598**
Linear regulator	Switching regulator
Input voltage: 35v max	Input voltage: 7 – 40v
Output voltage: 4.9 – 5.1v	Output voltage: 4.8 – 5.2v
Output current: 1.5A max	Output current: 1.2 – 2.4A
Price: $0.50	Price: $2.81

Due to the high output current of the switching regulator, we have decided to go with it. As mentioned before, the Raspberry pi requires a lot of current. The LEDs and the photodiodes will also be drawing current. Using the linear regulator will result in overheating even with the heatsink and thus will break the voltage regulator. Even though the pricing is more expensive, it is an essential component to power all the things in our project. Also, we do not require many regulators, so the pricing won’t really affect our choosing of the best bang for the buck. For the 3.3v that are required from the same table 3, resistor(s) will be used to obtain this output voltage. Doing this will prevent the need to buy a linear/switching regulator for the 3.3v DC output and thus save us money. However, should the resistors fail to create us the desired voltage value, we will get the required 3.3v regulator.

3.5.2 Microcomputer selection

Initially, we have considered using the Raspberry Pi Compute Module 3 for our project. Simply, this compute module would plug in to our PCB, just like a DDR2 SODIMM RAM module to a laptop. This would involve difficult routing work and would take a lot of time to implement. The senior design class is too short to implement this to our project and would not really add any overall benefits as opposed to using a normal Raspberry pi 3 model B or the model B+.

Table: Comparison of the two Raspberry Pi microcomputers

	Raspberry Pi 3 model B	**Raspberry Pi Compute Module 3**	**RaspBerry pi 3 model B+**
Processor	Quad-core Broadcom BCM2837 64-bit Speed: 1.2 GHz	Quad-core Broadcom BCM2837 64-bit Speed: 1.2 GHz	Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC Speed: 1.4GHz
Memory	1 GB RAM	1GB RAM	1GB
Storage	MicroSD	4GB eMMC flash device	MicroSD
Network	100Mbps ethernet BCM43438 wireless LAN Bluetooth Low Energy (BLE)	100Mbps ethernet BCM43438 wireless LAN Bluetooth Low Energy (BLE)	300Mbps ethernet Gigabit Ethernet over USB2.0 2.4GHz and 5.4GHz IEEE 802.11.b/g/n/ac wireless LAN Bluetooth 4.2 (BLE)
GPIO pin	40	N/A	40
USB port	4x USB2.0	N/A	4x USB2.0
Power	300 – 600mA, 5v	300 – 600mA, 5v	2.5A, 5v
Power supply	MicroUSB, GPIO	powered by PCB	microUSB, GPIO
Price	$34.99	$30.00	$35.00
Software	Raspbian Occidentalis	Raspbian Occidentalis	Raspbian Occidentalis

Despite cheaper pricing, overall it does not hurt the project to use a Raspberry Pi 3 model B instead of the Compute Module 3. While using the Compute Module 3 would have given the project more sense of difficulty, it would also have taken more time to implement into the PCB. The Raspberry Pi 3 model B also has a larger storage capability than the Compute Module 3. Using the Raspberry Pi 3 model B allows us to use a Raspberry Pi hat to plug straight into the GPIO pins which would make our design more compact. Ideally, the Raspberry Pi hat in our case would be our PCB circuit design. Therefore, switching to the Raspberry Pi 3 model B would do the opposite of our initial idea of placing the Raspberry on our PCB using the DDR2 SODIMM slot input, now the PCB is placed on the Raspberry pi. Lastly, two of our members already own the Raspberry Pi 3 model B, so it would make the cost for this component technically free. The overall benefits of the Raspberry Pi 3 model B+ are relatively small compared to the regular model B to invest another $35 on it. The only things that really changed are a slightly faster processor and faster ethernet speeds. The processor speed at this point is not worth the money investment for a small change. As for the ethernet connection that all varies on the speed of the user's internet to begin with. Even if the user has 300Mbps internet download speed, the overall speed would not make much of a difference on our smart mirror project.

3.5.3 MicroSD card selection

The purpose of the MicroSD card in the Raspberry Pi would be of that of a hard drive in a computer. It stores the operating system and other important data that are required for the Raspberry Pi to operate properly. The Raspberry Pi 3 model B supports up to 64GB storage space. There are more aspects than just storage space to keep in mind when choosing the microSD card. The cards themselves have varying read and write speeds that could affect the project greatly. Just like SSD technology in computer hard drives, which are much faster than the HDD variant at loading various applications and the operating systems of computers. Ideally, we would want to have the largest storage device capable of being used in the Raspberry Pi 3 model B, while also being the fastest in the read/write speeds. The higher the storage space in the microSD does not just affect the project in terms of storing more applications and whatnot. It also promotes longevity of the microSD because wear leveling of the object would be spread out across more memory space. Essentially, having a storage space that is the bare minimum to fit the operating system and other essential files would mean that the read/write load would be spread on smaller memory which would degrade the device faster. Lastly, the Raspberry Pi requires a FAT32 format with the microSD cards. Even if the cards do not come in this required format, it can be reformatted using a simple process. With all this in mind, the microSD cards we have chosen are the SanDisk Ultra 64GB MicroSDXC, the SanDisk Ultra 16GB MicroSDHC and the Samsung EVO MicroSDXC 64GB.

Table: Comparison of the different MicroSD cards

	SanDisk XC	**SanDisk HC**	**Samsung**
Storage space	64GB	16GB	64GB
Speed	Transfer: Up to 100MB/s	Transfer: Up to 98MB/s	Read: Up to 100MB/s Write: Up to 60MB/s
Price	$18.99	$7.82	$19.99

With every technological feature being pretty much similar, it just comes down to pricing and needed capacity. The 16GB variant from SanDisk may prove large enough for storing the required things for this project, as well as having a decent wear leveling. Since cost will be a large issue with this self-funded project, the SanDisk Ultra 16GB MicroSDHC is the obvious choice for now.

3.5.4 Gesture sensor selection

This component is one of the major components in our project that would affect accessibility of the smart mirror. Certain actions from the user will do certain things on the smart mirror. The first gesture sensor that comes to mind was the Xbox 360 Kinect. Since one of the group members had this item available it would have lowered costs for the project. The Kinect sensor is one the first sensors used for wireless gaming and promote user movement. The components included in the Kinect would allow us to harvest not just a simple gesture sensor but also a RGB camera used for facial recognition and a microphone if we chose to add voice recognition to the project. The other gesture sensor we reviewed was the APDS-9960 from Broadcom, as well as the ZX Distance Gesture Sensor. The following table compares the two components more clearly.

Table: Comparison between the two Gesture Sensor components

	XBOX 360 Kinect	**Broadcom APDS-9960**	**ZX Distance Gesture Sensor**
Price	Free	$2.72	$24.95
Detection range	0.7 – 6m	100mm without customer calibration	Up to 12 inches
Motion	Presumably 360°	Up-down-right-left	Distance (Z-axis) and side to side (X-axis)
Communication	-	I^2C	UART / I^2C

Even though the Kinect would lower the costs of the project even more, the lack of knowledge with the hardware components would make it difficult to implement in the project. There was no telling whether it would require special software for it to work or if it would work all together. Also, the required motion for accessing the mirror should be simple. Comparing the ZX Distance Gesture Sensor against the Broadcom APDS-9960, they both use I^2C to communicate, which is what we had planned to use to connect to it. It mainly comes down to pricing, so it would overall be better to use the Broadcom APDS-9960 because it’s cheaper.

3.5.5 Display selection

The display of the smart mirror needs to be large enough to fit all the necessary information, while also being cheap. Undoubtedly, the display of our smart mirror project will be the costliest item to get. Also, having high definition displays have become a norm for all technologies. Consumers tend to seek the best high definition display available for anything that involves a screen, such as, television or laptops. Also, ideally it would be better to use computer monitors than a television for our project because we are trying to get the most barebone components. By choosing a computer monitor, there are less components to remove to get just the monitor and the required input connections. Also, it would be cheaper because there are less components included. To reduce costs even further, two different monitors were made available for us to use on this project provided by group members. One of the monitors is an eMachines E17T4W and the other is a Dell E228WFP.

Table: Comparison between the two available display monitors

	eMachine E17T4W	**Dell E228WFP**
Price	Free	Free
Resolution	1280*720	1680*1050
Size	17”	22”
Connection	VGA	DVI, VGA
Input voltage	100 – 240vAC	100 – 240vAC

Keeping everything in mind, the clear choice was the Dell E228WFP because it has a larger screen size, which would mean we have more space for the UI. The Dell monitor is also a higher resolution than the eMachine monitor, which would mean the picture would be clearer.

3.5.6 Power supply selection

The power supply for the smart mirror will power the monitor, microcomputer, microcontroller, IR LEDs and the photodiodes. Based on our research into the monitor we have obtained the monitor will require 12V. The other components required voltages are listed on Table 3. Using the components made available to the group, from other unused personal electronic devices, we were able to salvage from an old portable DVD player a 12V AC/DC converter and a 20V AC/DC converter from an old laptop charger. Currently, we have decided on the 20V converter to power our project. To power the monitor, we would just have to add another voltage regulator to change it to the required value. However, as one of our stretch goals, we mentioned building a power supply of our own for this project, so this decision may be changed in the future.

3.5.7 Mirror selection

Ideally for this project, any surface that is relatively reflective would work great. Since we have decided on a display, the Dell E228WFP, which have the dimensions 24”x17”, we searched for a surface that would be ideal for the mirror aspect of the project. In our search, we found a two-way glass mirror 18”x24” with thickness of 0.25” on Amazon. There are both mirropane and mirroview selections from the same seller. Both of them do the same things, but the mirroview is slightly more reflective and blocks out more light than the mirropane. There was also an acrylic see through mirror available on Amazon of the same dimensions. The following table compares the three mirrors more clearly.

Table: Comparison of Mirropane vs Mirroview two-way mirrors

	Mirropane	Mirroview	Acryllic
Reflectivity	16%	66%	-
Visible transmittance	11%	20%	30%
Price	$129.99	$159.99	$36.99

Ideally, we wanted to use an actual glass mirror for this project, however other than just the cost other potential factors turn us away from choosing either the mirropane or the mirroview two-way mirrors. It would be nerve-wrecking should the mirror break in transit for the actual demo of the project and it would also weigh down the project. Most mirrors are wall mounted, the extra weight could affect whether it stays on the wall. The biggest thing that definitely turned us away from the mirrors are definitely the pricing, as mentioned exhaustively, the project is self-funded. The pricing of both mirrors go well beyond the allocated budget we first allocated for this project.

4.0 Related Standards and Realistic Design Constraints

The following section will explain in detail the standards and constraints that corresponds to this project. For the sake of marketability, standards need to be used. Also, the various constraints show the limiting factors that affect this project.

4.1 Economic Constraints

The biggest issue that we currently have with this project is the fact that it is self-funded. This means that this project will require low cost parts that may not function as good as a higher cost component. One example being the overall size of the Smart Mirror. The larger design requires not just a larger two-way mirror but also a larger monitor to take advantage of the newly acquired surface area. Another example would be the overall design of the Smart Mirror. Since it would be expensive to have a custom-made housing for this project it would mean that we would have to handmake the casing ourselves. While this won't necessarily affect function, it will affect the overall aesthetics of the project.

4.2 Time Constraints

The time limit for this project is the combination of the summer semester and the fall semester. Since the summer semester is mostly comprised of brainstorming and writing the documentation for the project, realistically the only time we will have to spend on the physical component of this project is most of the fall semester. Our project can be easily adapted to new technological features that would take it the next level, however, that will all depend on the amount of time we require to complete the main bulk of this project before adding any new components to it.

4.3 Environmental and Social Constraints

As far as environmental constraints go this project should not be affected by changes in the environment. Nor will it drastically cause change in the environment. In terms of social constraints, there is the concern that with the fact that the Smart Mirror can connect to the internet, other users may be able to access personal data from another user for harmful means. However, with the way that the design of our Smart Mirror works, is that it will only allow that specific user to access his or her own data.

4.4 Ethical Constraints

As of right now, we are still deciding if our Smart Mirror should have a camera for the use of facial recognition. Depending on placement, such as the bathroom as our Smart Mirror is intended for, it would develop into a fairly serious issue if the mirror was somehow hacked into and another person is able to monitor the user activities. It would pose an issue in general regardless of placement, if another person is able to observe the daily activities of the user. Such is the reason why people have been covering their laptop cameras with tape. We may come up with a way to combat this by simply designing an LED to go on to show that the camera is being used, as suggested by our instructor. We could also come up with a simple cover for the camera, when not in use by the user. This would make sure that the camera does not pick up any user activities regardless of if the malicious hacker can access the camera in the first place.

4.5 Safety Constraints

As mentioned before, our Smart Mirror is intended to replace the bathroom mirror so that the user may be productive while getting ready to tackle the day. With this intention in mind, there is a concern that should any parts of the hardware come to contact with water that it would raise some safety issues. However, we will design our Smart Mirror in such a way that it should cause no issue when it encounters water/steam.

4.6 Related Standards

The standards of a project are what makes the item marketable. There are certain standards that are widely used around the country/world for certain specific things. The following of these given standards ensures product safety and compatibility and it also keeps things consistent with other technologies that abide by those standards. Therefore, as engineers, it is important that those standards are followed, not just for marketability, but in generality.

4.6.1 Lead Solder Safety

Solder is the metal used to make solid electrical connections between components and the pads on a PCB. It comes in many different forms and varieties but an issue that has come up lately is the safety of it including Lead. Lead is a soft metal with a low melting point and that’s why it has been used in Solder for so many decades. The issue with lead of course is the dangerous effects it can have on people.

The dangers of lead these days are well known and that’s why it is used less now, and many lead-free solder options are available. The largest disadvantage here is that lead-free solder is typically harder to work with because it has a higher melting point than the lead equivalent. Lead can be especially dangerous when hand soldering because the fumes from the solder is hard to avoid breathing unless you have a fan or vacuum that can displace it.

Because of these obvious dangerous we will be working with Lead-Free solder since its overall safer and not terrible to work with for the most part.

4.6.2 Laser Safety

Since lasers were considered for use in this project, it is vital to know the various precautions one must take when handling them. When dealing with lasers care must be taken depending on the classification of the laser. When classifying a laser, the maximum permissible exposure (MPE), which is the highest power or energy density of the light source considered safe. This is measured at the cornea or at the skin for a given wavelength and the length of time exposed. The classifications are as follows:

- **Class 1**: A class 1 laser is eye safe. This is due to the maximum permissible exposure cannot be exceeded when viewing the laser with the naked eye or magnifying optics.
- **Class 1M**: This is a laser that is eye safe except for when viewing through magnifying optics.
- **Class 2**: A class 2 laser is considered safe due to what is known as the blink reflex, which is when one averts their eyes due to bright lights. The blink reflex typically limits exposure to no more than a 0.25 seconds and only applies to 400-700 nm lasers. These lasers are limited to a continuous power output of 1 mW. If the emission time is less than 0.25 seconds or the emitted light is spatially incoherent then the 1 mW power threshold may be exceeded. If one suppresses the urge to blink than this can result in eye injury.
- **Class 2M**: A class 2M laser is similar to a class 2 laser except it cannot be viewed through optical instruments without high risk of eye damage.
- **Class 3R**: A class 3R laser is only considered safe when handled with care and limited viewing. The MPE can be exceeded, however, with little risk of injury. Lasers in the visible spectrum classified as 3R are limited to 5 mW of continuous emission. Wavelengths outside of the visible spectrum and non-continuous outputs have other power limitations applying to them.
- **Class 3B**: Class 3B lasers are considered hazardous of directly exposed to the human eye but can be safely viewed from diffused reflections from matte surfaces. The acceptable emission limits (AEL) for a continuous laser between the wavelengths of 315 nm to far infrared is 0.5 W. If the laser is not continuous in the visible spectrum the limit is 30 mW. Protective eyewear is required when there is a possibility of direct viewing of these

lasers. Additionally, these lasers must have a key switch in addition to a safety interlock.

- **Class 4**: This is the highest class of laser and the most dangerous. A class 4 laser can burn the skin or cause permanent eye damage from direct or indirect viewing of the beam. Additionally, it is possible for a class 4 laser to ignite combustible materials. This may occur from direct or indirect reflections of the laser beam.

5.0 Project Hardware and Software Design Details

This section of the paper will explain in detail the hardware and software that will be used on the project. It will explain why those components are used and how they will connect. The software design sections will explain how the software will be designed and how they will be used in the project.

5.1 Microcontroller

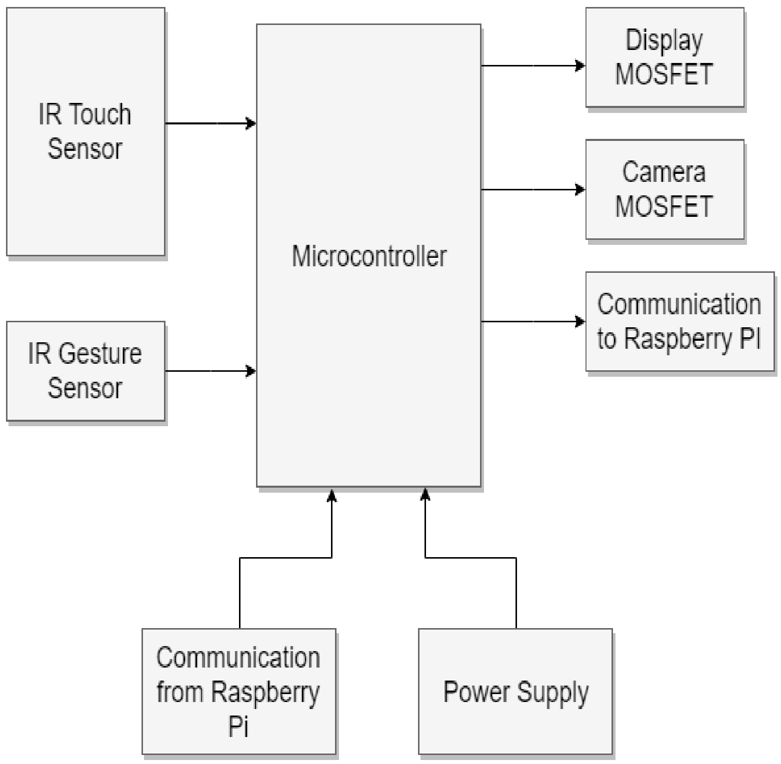


Figure 12: Microcontroller Block Diagram

The microcontroller system will be setup as in the block diagram shows above. The microcontroller we'll be using is the Atmel 328P since it's simple and has more than enough features for our needs.

5.2 Communicating with the Raspberry Pi

The microcontroller in our implementation will be the Master and the Raspberry Pi will be the slave. When it monitors input from any of the various sensors it will start sending data to the Pi immediately, so the display can be updated according to the input. Since the Pi is a slave, normally it won't be able to initiate communication with the microcontroller, but we will work around this by using interrupts. If the Pi wants to send a command, such as activate the camera, it will pull an interrupt pin High on the microcontroller. Once this happens the microcontroller will then request bytes from the Pi. The first byte will be the command such as turning on the camera or a sensor info request. If the command requires more bytes to be sent, the microcontroller will request it.

5.3 Gesture Sensor

The Gesture sensor as stated above will be an APDS-9960 from Broadcom. Ideally this will be mounted somewhere convenient such as the bottom of the smart mirror and pointing upwards to be used in the middle of the display. Our idea is to have it hidden behind a translucent piece of black plastic or glass that it can still function behind, yet not be too obvious to anyone using the mirror.

Figure 13: Broadcom Gesture Sensor

This tiny device has everything we need built straight into it for the gesture sensing we want to do. It’s an all in one RGB, ambient light, proximity, and gesture sensor. Most of these features will come in handy for making our smart mirror work the way we want it to. The proximity sensor will be especially useful when trying to save power and detect if anyone is around or using the mirror. Using that sensor, the display can be turned off when nobody is using the mirror. It also has an I^2C interface for communicating with it and other devices easily. We will use a breakout board for this device that provides access to all the necessary connections and also has all the support circuitry so that we don’t have to design a second PCB or implement superfluous components onto our own. This will simplify the design and leave space for more important items like our multiplexers, microcontroller, headers/connectors, and the logic level shifters for communication between the Pi and microcontroller. Since we’re aiming to have a small PCB, space is a precious commodity and we need to make the most use of it as we can. We will accomplish this by using surface mount components where we can and by packing everything together as tight as possible.

6 Project Prototype Construction and Coding

This section will explain the processes that will be taken when designing the hardware prototypes of the project. This includes any software that will be used to design said prototype. This section will also exhibit and explain the coding that will be involved for this project.

6.1 Bill of Materials

Figure: Parts Selected for the prototype of the project

The above picture shows all the required essential components to use for building the prototype of the project. Some parts may be changed in the future to better the overall finished project. The parts above correspond to the table below.

Table: the names of the items to the corresponding letters based on figure

Letter	Item
A	Raspberry Pi 3 model B
B	Arduino Uno Rev3 (for testing in place of our PCB)
C	IR LEDs
D	Photodiodes
E	Broadcom APDS-9960 gesture sensor
F	CD4051 DIP-16 single 8 channels analog multiplexer
G	L7805CV 5volts linear voltage regulator
H	12v, 10A relays

6.2 Printed Circuit Board (PCB)

Printed circuit boards (PCB) have become synonymous with modern electronics. A PCB allows for electronics to take on form factors which were previously impossible and allows for the electronics themselves to shrink since most mechanical stress burdens the PCB instead of the electrical components. Additionally, due to the rigidity of the PCB the assembly of electronics has been automated to reduce human error and improve efficiency. Electrically, the PCB has embedded copper paths connecting the various components of a circuit together.

However, even though the PCB has revolutionized electronics, designing a PCB is still a challenge. One must take many factors into account when designing a PCB.

The routing of the traces must be taken into consideration when presented with area constraints. The width of the traces must be wide enough to avoid burning of the traces when current passes through them. Spacing between traces must be accounted for in order to minimize unwanted coupling. The desired cost to manufacture the PCB must be made clear, as this directly influences both the footprint of the PCB as well as the number of layers the final board will have. All of these must be taken into consideration in addition to the quality of the substrate and they type of substrate to ensure that the finished product works as intended and is as cost efficient as possible.

6.2 The PCB Design Software

Computer aided design software (CAD) has become an essential part of designing any electronic circuit. Many CAD software have built in schematic editors in order to create the electrical connections for the circuits. Additionally, the software which was considered for use in this project typically had the capability for seamless transition from creating a schematic which deals with the electrical connections, to the physical layout of the PCB. This is due to the availability of the various libraries which house the all of the necessary information regarding the electrical and physical properties of each component. When specifically designing PWBs there are a multitude of options available, each with their own distinct advantages. The following table lists the various CAD software that was considered for use in this project.

Table 6: CAD software which was considered

Software	Specs	Cost
Autodesk EAGLE Standard	99 schematic sheets, 4 signal layers, 160 cm^2 board area	$15/month
Autodesk EAGLE Premium	999 schematic sheets, 16 signal layers, unlimited board area	$65/month
Autodesk EAGLE Free	2 schematic sheets, 2 signal layers, 80 cm^2 board area	Free
Xpedition xDX Designer	Hierarchical search/edit capability, simulation support for various circuit types	$5000+ depending on package
DipTrace Starter	300 pins, 2 signal layers	$75
DipTrace Freeware	300 pins, 2 signal layers, non-profit use only	Free

The options listed in the table were considered because of both price point, except for xDX Designer, and their respective capabilities. xDX Designer was considered due to its use in large corporations, such as Lockheed Martin, making it a desirable software to be familiar with. However, due to the price it is not a viable option for student use.

After taking into consideration both the familiarity of our team members with the various software packages and the requirements for our project, it was decided that Autodesk EAGLE Free is best suited for the purposes of this project.

6.3 Autodesk Eagle

As mentioned in the research section of this paper, there are multiple different methods to designing a PCB, such as having a single-sided or a double-sided design. The EAGLE program has taken these into account and included these decisive components into the program. There is a function within the program to add the conductive foil within the top or the bottom of the circuit board. One of the most important issue with designing a PCB is routing. Basically, it’s how the conductive materials are connected from components to components. There is an option within the EAGLE program for auto-routing, which simply puts your required conductive connections from the schematic unto the board for you in a way that different routes do not accidentally connect with each other. This function will save countless hours trying to think of the path for yourself. It's also worth noting that even though this function exists, it will be used with caution. The best way to layout a PCB is by hand and while the auto-routing function is great, it's not perfect and will be used sparingly if at all. With the addition of conductive paths, there are also copper pours included in the program. This method is essentially used to create a ground in the circuit. There is also an option to add a surface mount for the microcontroller of our choosing. This includes doing a more advanced method in the program, but the basics are basically the same, the different connectors of the surface mount still need to be routed to its correct place. There is an added option to add silkscreen to the PCB design, which simply adds wording onto the circuit board as to what should go in each specific area. This would make construction of the board easier, as well as being more user friendly for those trying to get a grasp of the board design. After the design is complete, we generate the Gerber files, which is a universal language for PCB design. The Gerber files simply contain the design description for each layer of the PCB.

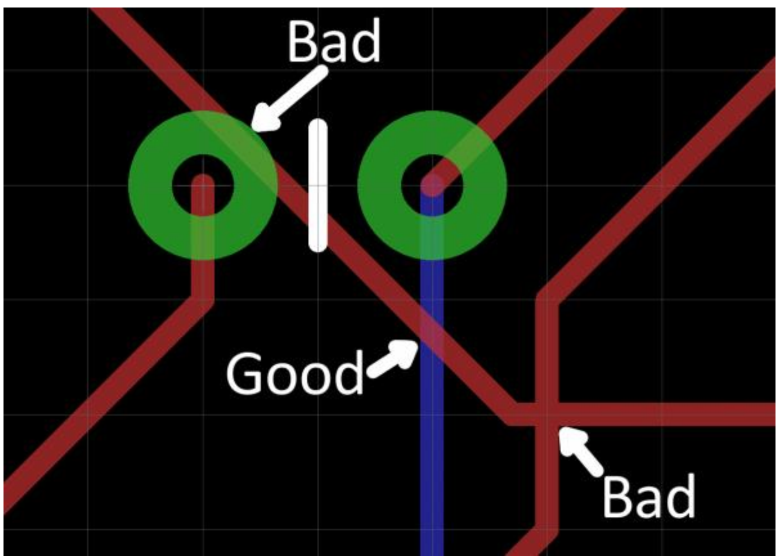


Figure 14: PCB pathing example on EAGLE

6.4 Manufacturing the PCB

After a series of thorough examinations on the design, the board will have to go through a manufacturer to get the physical component made. There are multiple decisions to make to decide which manufacturer to go with for our project. The first important factor is the overall quality of the PCB manufacturer. Should the product we receive be poorly made and non-functional it would drain both our time and money. The second important decision is cost, different manufacturers will have differentiating costs, as this project is funded from our own pockets, this will probably be one of the most important decisive factors. The third important decision to make is time, as in the length of time it would take for the manufacturer to ship the product to us. As we are already way limited in time for the creation and building of this project, we cannot afford a delay in our process from a lengthy production and shipping time. After all these factors are considered and a manufacturer is decided, then we simply send the required Gerber files to the manufacturer for production.

Table 7: PCB manufacturers with pricing and time

Manufacturer	**Price**	**Time**
OSH Park (3 PCB per order)	2-layer prototype: $5/in.^{2} 4-layer prototype: $10/in.^{2}	>12 days shipping
Advanced Circuits (3 PCB for 2-layer) (4 PCB for 4-layer)	2-layer prototype: $33 4-layer prototype: $66 (max 60 in.^{2})	5 days turn
Gold Phoenix	2-layer prototype: $85 4-layer prototype: $200 (max 75 in.^{2})	7 hours turn

6.5 Schematic Design

The most crucial component of the smart mirror is the custom PCB which will be used to take input from the optical grid for the user to interact with the smart mirror. As the optical grid will be the main method with which users provide input to the smart mirror, special care was needed when designing the circuit.

From the 12V input which will be supplied by a standard laptop charger LM7805 voltage regulators are used to provide a constant 5V output to power both the ATmega328 microcontroller and the 940 nm LEDs which will illuminate the photodiodes so we can properly process input.

Due to constraints on the available I/O on the ATmega328 microcontroller two multiplexers are connected to the photodiodes to allow enough photodiodes to be connected to the smart mirror. Each multiplexer will provide input to the ATmega328 corresponding to one of our two axis, horizontal or vertical. By connecting the output of the CD4051 multiplexers to an analog I/O pin, due to the photodiodes providing an analog signal, and the select pins to digital pins, we are able to poll the multiplexer to constantly check for input. With the digital pins on the ATmega328 assigned to the select pins on the CD4051 we are able to count through the various inputs and check the signal of the corresponding photodiodes.

When a photodiode is not receiving adequate light and thus no signal is passing through, the ATmega328 will determine where the user is providing input by correlating the location on the mirror to the location of the photodiodes.

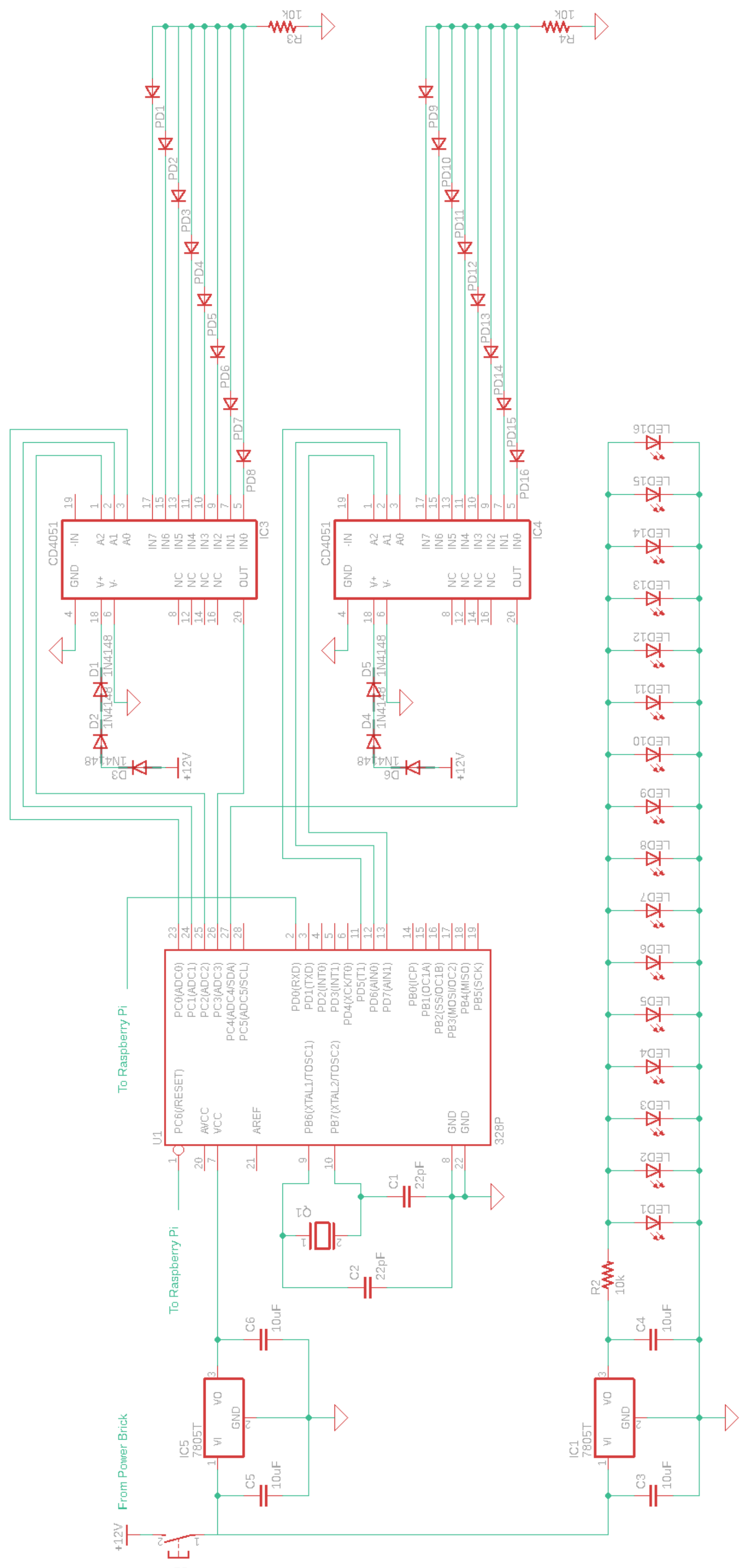


Figure: Schematic design for the smart mirror

Table: Use of the ATmega328 I/O

Smart Mirror ATmega328 Pin Outputs	
Pin #	**Description**
1	Communication with Raspberry Pi
2	Communication with Raspberry Pi
7	Vcc +5V input
8	GND
9	16 MHz oscillator
10	16 MHz oscillator
11	Multiplexer select line
12	Multiplexer select line
13	Multiplexer select line
22	GND
23	Multiplexer select line
24	Multiplexer select line
25	Multiplexer select line
26	Multiplexer output
27	Multiplexer output

6.6 Crystal Oscillator

When using microcontrollers, ideally it will need a clock source to determine at what speeds instructions can be implemented. To get this clock source, a Piezo-electric oscillator is typically used, and the most common being a crystal variant. There are internal and External clocking mechanisms, the one we will use, the crystal oscillator is an external clocking mechanism. The picture from section 5.4 shows how we plan to connect the crystal oscillator and the corresponding load capacitors. Pin 9 and 10 connects the crystal oscillator to the microcontroller and pins 8 and 22 connects the oscillator to its load resistors which are then grounded.

Despite all the other clocking mechanisms, crystals are by far the most precise and stable. The microcontrollers usually have a built-in oscillator but to run faster clocks than available from the internal clocking mechanism we must use an external oscillator, in the case of the Atmega328p the max clock speed is 8MHz. In our case, based on the datasheet, the maximum external clock that we can use for the Atmega328p microprocessor is 20MHz. However, it seems that the safe value to use for clock speed is 16MHz. The clock speed doubles using an external clock mechanism. The following equation will be used to find the required load capacitance value.

$$C_{LTOTAL} = \frac{C_{L1} \times C_{L2}}{C_{L1} + C_{L2}} + C_P$$

Figure: Equation used to find the load capacitors

The C_{LTOTAL} is the capacitance of the crystal oscillator we will be using, which we will assume is the Crystal 16MHz from Sparkfun. In our case the C_{LTOTAL} = 20pF, as for the C_P, that refers to a parasitical capacitance and usually it is safe to assume a value of 5pF for that. After doing the math, you get load capacitance C_{L1} and C_{L2} to be (16pF – 5pF) x 2 = 22pF. Sparkfun also has a 22pF ceramic capacitor that is recommended along with the crystal oscillator.

6.7 Clock Configuration

Now that we have decided on the required crystal oscillator and ceramic capacitors to potentially increase the clock speed, we still must choose a clock configuration that would best fit our project. There are two types of clock configuration for the crystal oscillator, the low power and the full swing.

The low power configuration, as the name suggest would help to lower power consumption. Ideally, since we are connected to an outlet as a power source this doesn't really matter as much. Also, the clock speeds of the crystal are susceptible to noise. As with any electrical devices there are bound to be noise included with it, usually it would not matter much, however since we are dealing with the clock speeds the noise would affect it in a way that would make it less than the most efficient setup.

The full swing configuration does exactly the opposite as the low power configuration. It is not susceptible to noise, but it consumes a lot of power. Even though this configuration consume way more power than the last configuration, this configuration would keep the clock speed running at the most efficient, which would mean things get processed as fast as intended. Not only that, the overall stability is also improved, since the device isn't susceptible to noise it runs as intended therefore keeping its stability. This makes the full swing configuration the best choice to use for our project.

6.8 Hardware Testing

Hardware testing of the various subsystems of the smart mirror will allow us to know everything is working before we finalize our circuit board design and send it off to be shipped to the fabrication house. This step is debatably the most important step because it's essential to have our circuit board design correct or our project could be set back weeks if not more by waiting for another circuit board. Hardware testing will be done in Multisim first if possible and then on a breadboard.

Objective:

The objective for testing is verifying that every subsystem in the smart mirror design works and if it doesn't work as expected, a different part will be ordered and tested.

Environment:

The hardware testing will be performed in the UCF Senior Design lab where there is access to a variety of power supplies and measuring equipment including:

- Tektronix MSO 4034B Digital Mixed Signal Oscilloscope, 350 MHz, 4 Channel
- Tektronix AFG 3022 Dual Channel Arbitrary Function Generator, 25 MHz
- Tektronix DMM 4050 6 ½ Digit Precision Multimeter
- Agilent E3630A Triple Output DC Power Supply

Procedure:

1. Power Arduino microcontroller and wire various connections to the power supply and sensors.
2. Write code to test gesture sensor and light up LEDs depending on the direction swiped.
3. Write code to illuminate LED from Raspberry Pi to test GPIO output.
4. Lastly, connect Raspberry Pi to the Microcontroller and make sure both can communicate over Serial UART.

<u>Microcontroller Testing</u>

First step in testing the microcontroller was writing some code to blink the included LED on the Arduino Uno board. This was to make sure that the board is working correctly, and code can be uploaded to the microcontroller without any issues. This was simple and worked the first try so I know the microcontroller is good and now I can write code to talk to the gesture sensor.

<u>Gesture Sensor Testing</u>

The Gesture sensor we have communicates over an I^2C bus to the microcontroller to send various commands. The sensor can be used to detect ambient light, gestures, and distance up to 4 inches. Our use is for the gesture detection alone.

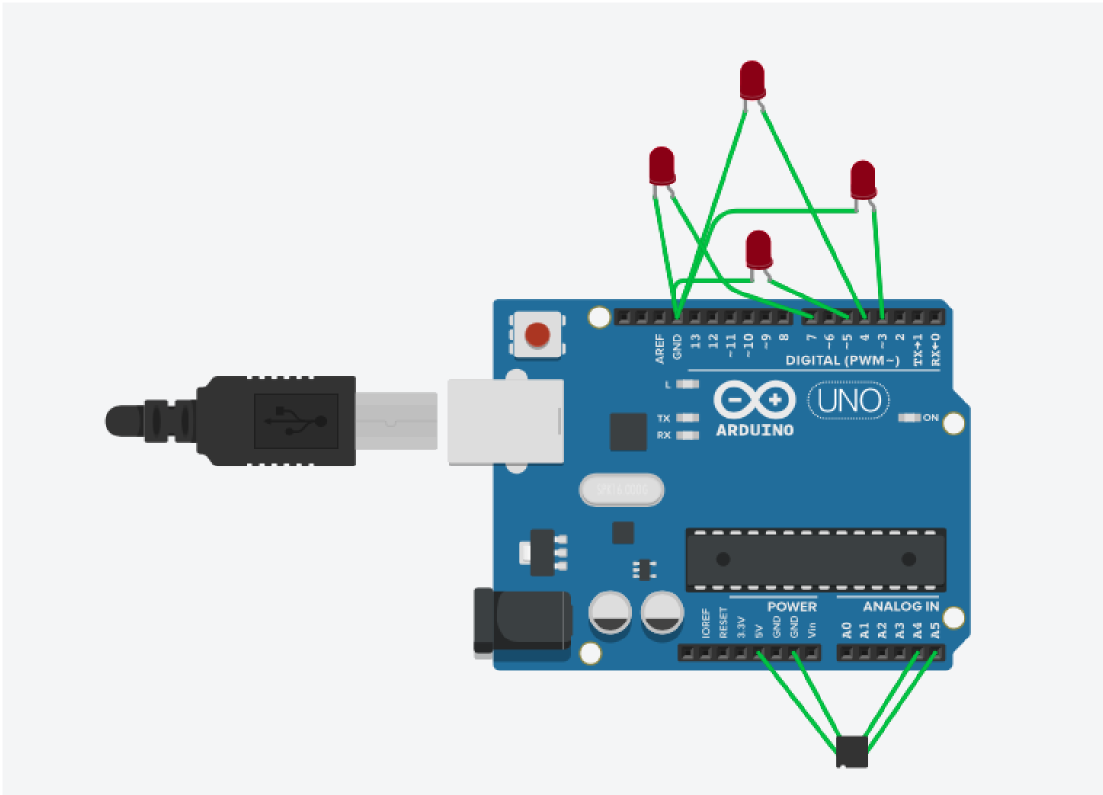


Figure 15: Gesture Sensor Test Setup

Here is the circuit configuration used to test the gesture sensor setup. First the gesture sensor was connected to the microcontroller using the two I^2C pins which on the Arduino Uno are analog Pin 5 and analog pin 4. Next the sensor needs power, so it's wired to the 5 volts output and ground on the Arduino board. Testing is difficult without any output to see so I wired up 4 LEDs to 4 digital pins on the Uno and used those to signal the output of the gesture sensor. When a left swipe is detected, the left LED lights up, right swipe lights up the right LED, etc. This made it very easy to automatically see if the circuit was working without the Serial monitor. The sensor worked up until about 5 inches and then wouldn't sense any gestures as far as I could see during testing.

Microcontroller-Raspberry Pi Communication Testing

The next piece of hardware that needs testing is communication between the Raspberry Pi and Microcontroller. This is essential because all the input methods will need to be communicated to the Pi so that the user interface can be updated accordingly. Doing this required me to write an application for the Raspberry Pi so I could receive and send Serial messages through the Pi itself. I decided to write the applications in Java because it can run on multiple operating systems.

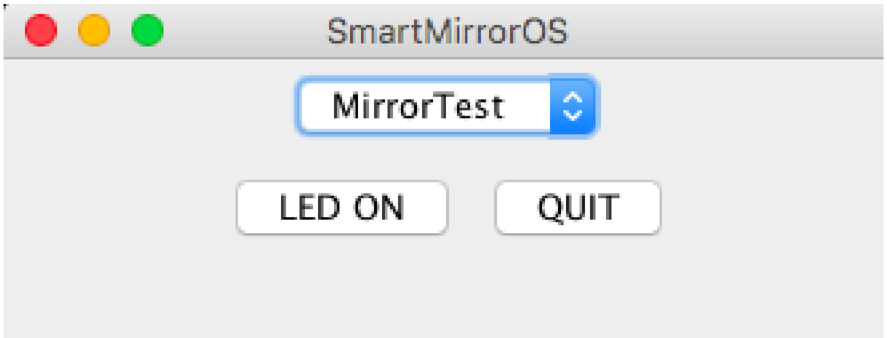


Figure 16: Java Application for Raspberry Pi

Using the WiredPi library, I was able to control a GPIO pin of the Raspberry Pi to turn on and off an LED. Now that I've figured out how to write java apps and get them working on the Pi, it's time to test the Serial Communication capabilities.

Our Microcontroller cannot simply be connected to the Raspberry Pi though, there's an issue with doing that. The Pi operates on 3.3-volt logic while our Microcontroller uses 5-volt logic. This imposes an issue when you want to communicate between the two. There is two options to make them compatible though and they're both fairly simple:

1. Buy a level shifter
2. Build a level shifter

Both options are simple enough but what will be the main difference? The only sacrifice will only be size. There's logic level shifter integrated circuits in the market and they're quite small but if we have the space for it, designing our own isn't too hard. It only requires one transistor and a few resistors to build. Here is the circuit and simulation:

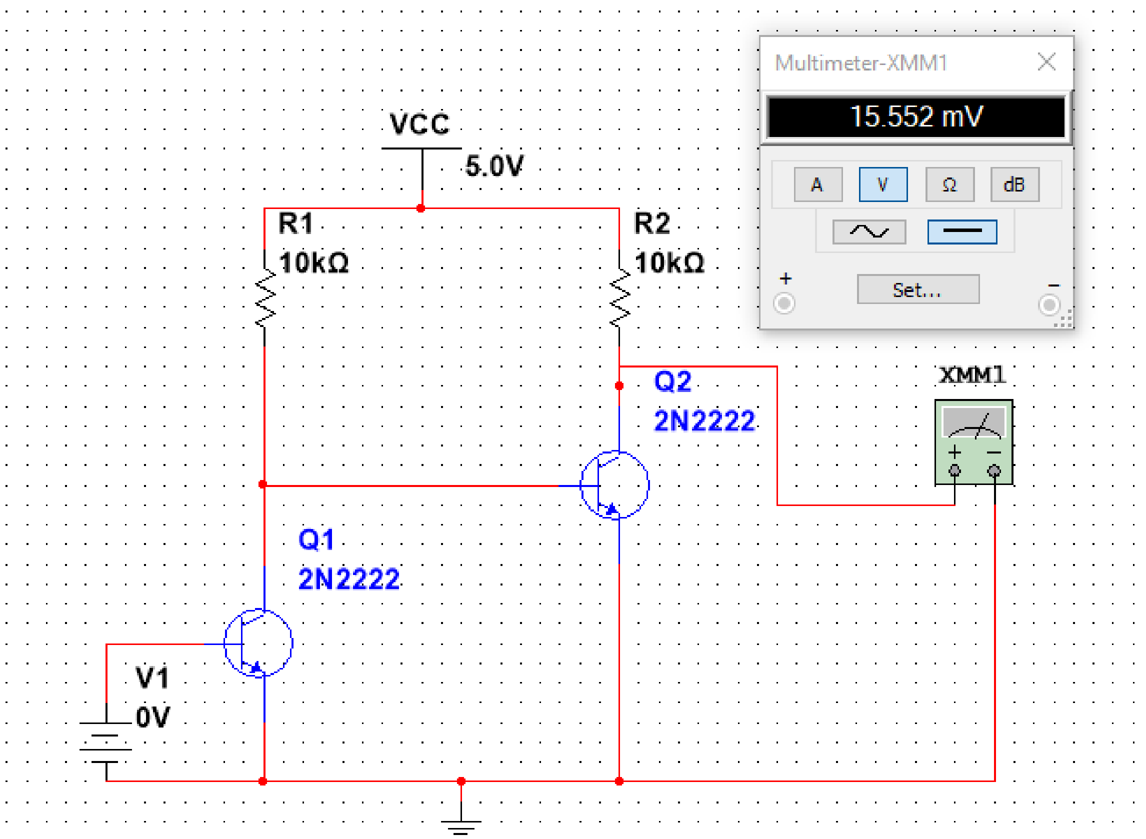


Figure 17: Level Shifter Simulation with 0-volt input

This circuit is quite simple in its construction and uses parts we already have so it won't use any of our budget and that is one of the reasons we will be building our own logic level shifter. The operation is as follows: The input is on the base of the first transistor, Q1, and when that is zero volts, so the transistor is turned off. Since no current is then flowing through the collector resistor, the voltage at the collector of Q1 is 3.3 volts and this will bias Q2 to be in saturation. Since Q2 is in saturation, the voltage across the transistor will be about 0.2 volts and so the collector voltage for Q2 will be about VCC or five volts. Using two of these will allow us to send data between our Microcontroller and Raspberry Pi without damaging anything. The above figure shows the simulation result for an input of zero volts, following is the simulation for a 3.3-volt input:

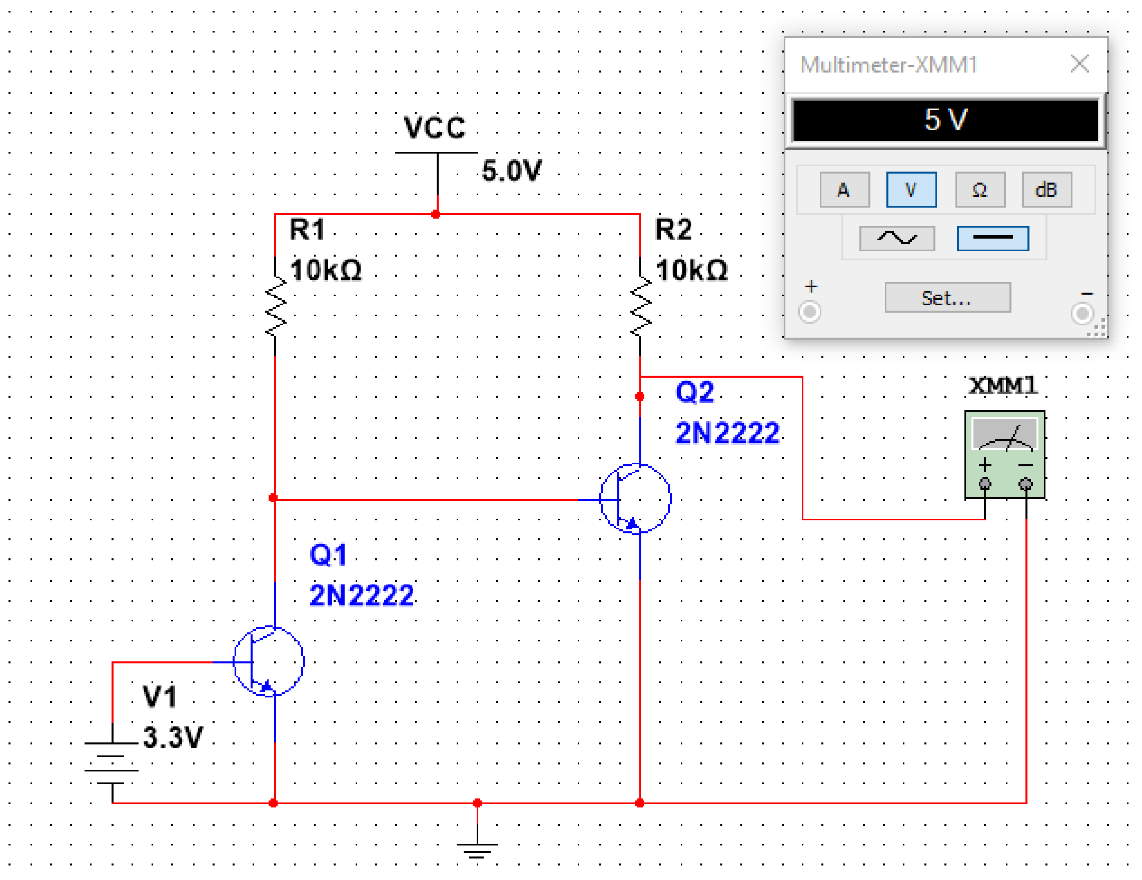


Figure 18: Level Shifter Simulation with 3.3-volt input

As you can see, the level shifter works as expected with just two transistors and a few resistors. Unfortunately, this circuit won't work both ways, but the same circuit can be made into a 5-volt to 3.3-volt level shifter by changing VCC to 3.3-volts.

Now that the level shifting is handled, it's time to setup a test that demonstrates the ability for the Raspberry Pi and Microcontroller to communicate to each other. The first step is connecting the Pi and the microcontroller and that will only require two wires for both a Tx and Rx line. The Tx will be the transmit of the Microcontroller and that feeds into the Rx, or receive, of the Raspberry Pi. Then the Pi Tx will be connected to the Rx of the microcontroller.

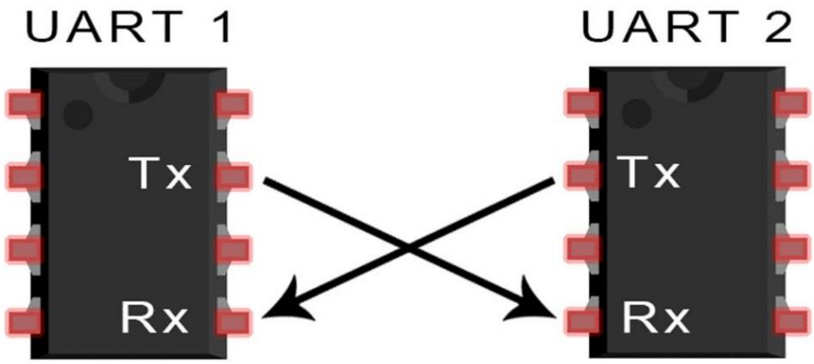


Figure 19: Serial UART

I will be writing a program that allows a button to be pressed in a raspberry Pi java program and the microcontroller will that light an LED depending on the button pressed. This will test if communication works and give a visual notice that the software and hardware is working. Eventually all of this will lead to the test of the whole system, but it is important to start small and work up from there. Following is the software diagram:

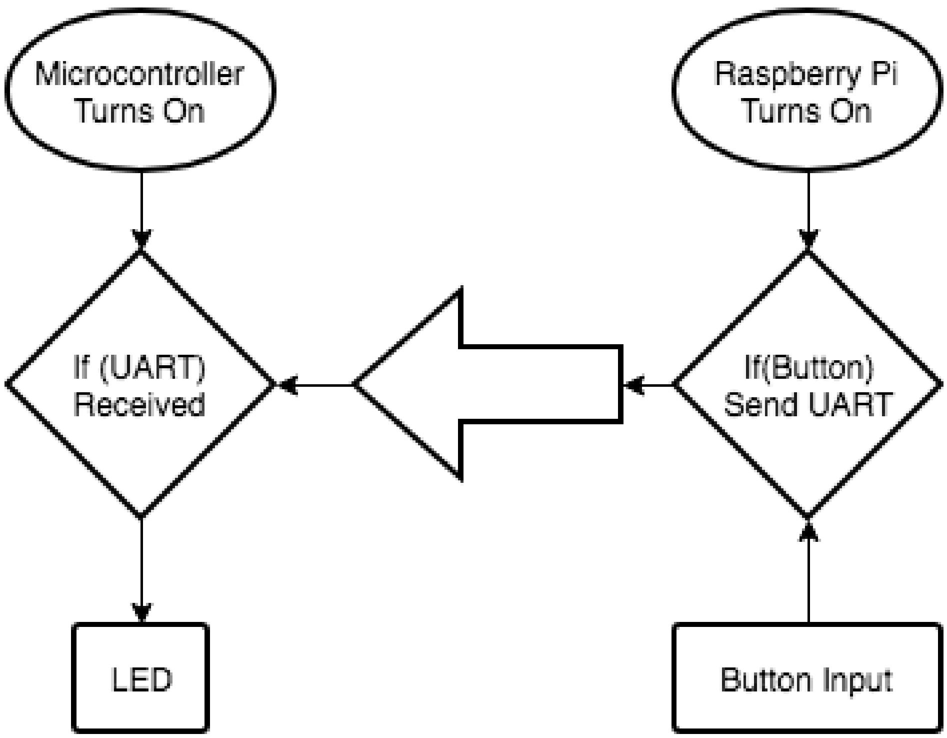


Figure 20: Communication Software Diagram

As detailed in the diagram, the program starts with both the microcontroller and Raspberry Pi starting up. The Microcontroller will be in a loop just waiting for a Serial input to happen and when it does it'll activate the digital pin that outputs to an LED. The Pi does the same in that it is waiting for a software button to be clicked and once that happens, it'll send a Serial Command to the microcontroller to be acted upon. This will very much show proof of concept for what the whole system will be like when complete because this communication is essential in making the project work correctly. Especially for relaying information such as the Gesture controls.

The tested circuit and program worked as expected and now can be used in the final project as a reliable way to communicate between the Microcontroller and Raspberry Pi. The pins used on the Raspberry Pi and Microcontroller are as follows:

Microcontroller

Pin	Purpose	Connection
Analog Pin 4	I^2C Bus	Gesture Sensor
Analog Pin 5	I^2C Bus	Gesture Sensor
Digital Pin 1	UART	Raspberry Pi
Digital Pin 2	UART	Raspberry Pi
Digital Pin 3	Digital Output	LED
Digital Pin 4	Digital Output	LED
Digital Pin 5	Digital Output	LED
Digital Pin 6	Digital Output	LED
Ground	Common GND	Connect all references
5-Volts	Sensor Power	Power Gesture Sensor

Figure: Microcontroller Pins Used

Raspberry Pi

Pin	Purpose	Connection
14	UART	Microcontroller
15	UART	Microcontroller
2	5-volts	Logic Level Shifter
1	3.3-volts	Logic Level Shifter
40	Digital Output	LED

Figure: Raspberry Pi Pins Used

Much of these pins will also be used in the final design and even more as the full set of Photodiodes are connected to the microcontroller. A small issue encountered during testing is with the UART module on the Raspberry Pi 3. Since Bluetooth was added to the Pi 3, it uses the UART module that was previously free for use with any programs written on the device. A small software dependent UART was installed for user programs and it can be less reliable and cause problems because it uses the CPU clock which is variable. The solution to this was to lock the CPU clock to one value so that it doesn't change and the UART is stable. There is a small decrease in performance, but it is barely noticeable and must be done if we want stable communication. There is a second fix as well and that is by disabling the Bluetooth module but since we plan to integrate Bluetooth functionality, that isn't an option for our project.

The microcontroller has been prototyped using an Arduino Uno board because it uses the same microcontroller as we plan on using.

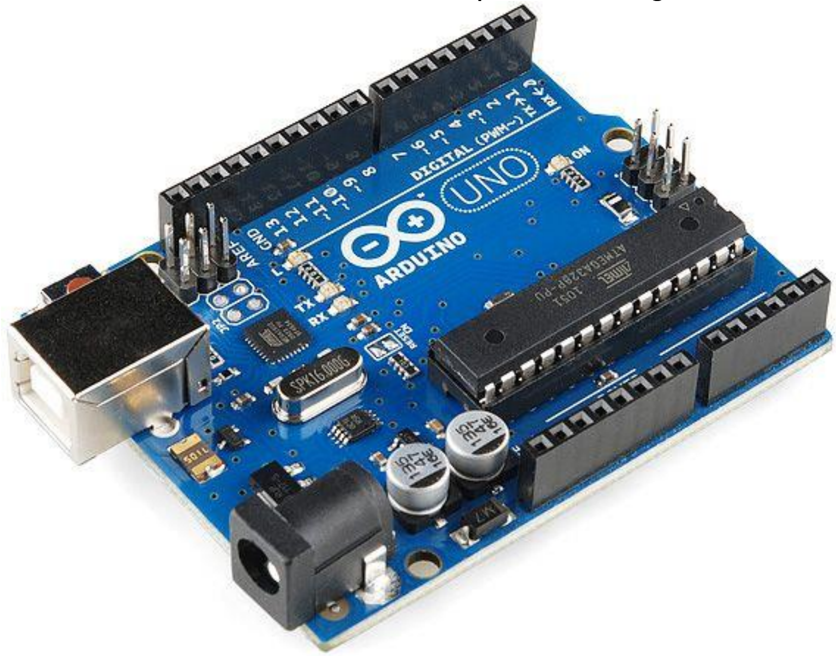


Figure 21: Arduino Uno Board

It's a very robust board with a lot of digital and analog pins available for use on the microcontroller. It's also easily programmable through USB and the Arduino IDE. The Arduino IDE will allow us to use a plethora of libraries made just for the Arduino platform and integrate them into our project, making it much easier. Using the Arduino Uno also provides us another benefit for our project. Once the PCB's are made and we have our own board with an integrated microcontroller, the Uno can be used as a programmer for the chip. A programming application can be uploaded to the Uno and once that's running on it, we can upload a bootloader to our own chip using it. It's a straightforward process and just means we will have to take care to break out the serial pins on our custom PCB. The chip will be programmed over serial using the SPI pins of the microcontroller.

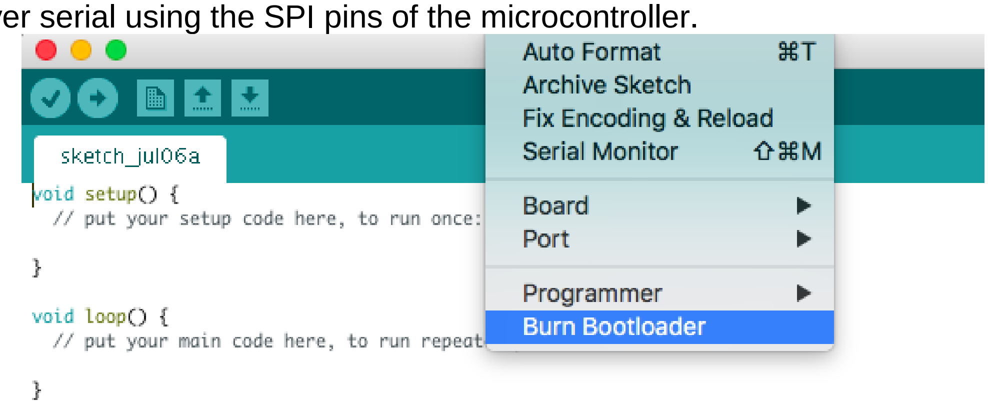


Figure 22: Arduino IDE with programming options

Once the Uno is setup as an ISP(in circuit programmer), it's as simple as pressing a button. Once the bootloader is on our own ATMega 328P chip, we can simply upload our code to the board. During current testing we've been programming the Arduino board through USB because it has a specialized chip for converting USB signals to Serial SPI that the microcontroller uses. Our final circuit won't use this chip so programming through USB won't be an option and so a program will be used that turns our testing device, the Uno, into an ISP (in circuit programmer). So, in the final circuit a 6-pin SPI cable can be connected between the two devices and our microcontroller will only be programmed through the Uno device.

Pin	Name	Purpose
Digital Pin 13	SCK	Serial Clock
Digital Pin 12	MISO	Master In-Slave Out
Digital Pin 11	MOSI	Master Out-Slave In
Digital Pin 10	SS	Chip Select

Figure: Programming pins and their corresponding digital pins

Although these programming pins use digital outputs, we can still use them in our own code. Once the circuit is programmed these pins are free to use for whatever purpose we require. The other two pins on the 6-pin connector will be VCC and Ground. VCC is needed to power the microcontroller while it's being programmed, and ground is also for that and to have a common reference for both boards.

Optical Components Testing

The optical components will be tested by creating a simple electrical circuit and testing the functionality of each component. Since the IR light sources and photodetectors will not be mounted onto a PCB we are able to buy through-hole components for both testing and use in the final product. This is possible due to the components being embedded in the frame of the mirror. We will have wire running from the optical components to connectors on the board for two reasons. The first being that, by having a connector between the optical components and PCB we are able to easily switch out the PCB should we discover a hardware bug or accidentally destroy it. This will save us time if such an event occurs since it will not require the disassembly of the entire smart mirror. Additionally, since the optical components will be tested before they are mounted in the mirror, we can use sturdier and more permanent mounting solutions since they will not need to be removed should the PCB fail. For the light source it will be done by creating the electrical circuit shown in the following figure.

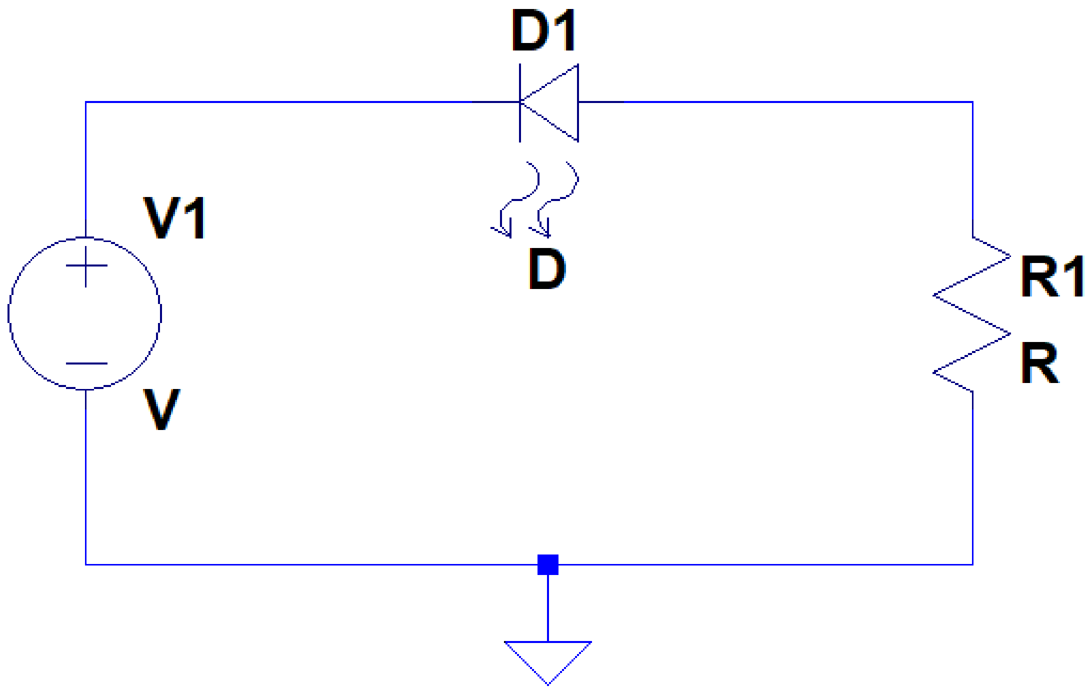


Figure 23: Test circuit for IR LED and photodiode Note that the IR LED cathode and anode will be connected in reverse to what is shown in the figure.

Once the circuit is built onto a breadboard and enough voltage is provided to push the semiconductor source above the threshold voltage and turn on, we will use an IR card to ensure that the source is emitting light as expected. If we are unable to see any light appear on the IR card after power is applied and we have verified the electrical connections with a multimeter, we can be certain that the IR light source is defective.

Once the functionality of the IR light source has been confirmed, we will create an identical circuit on another breadboard, however the IR light source will be replaced with the photodetector. With the IR light source positioned to emit light onto the surface of the photodetector and power provided to both circuits a multimeter will be used to measure the current flow through the detector circuit. If current is flowing, then the photodetector is operational. However, if that is not the case and we have confirmed that surface of the photodetector has been exposed to IR light by using the IR card we can be certain that the photodetector is defective.

<u>Voltage Regulator Testing</u>

Other than the ATmega328 microcontroller and Raspberry Pi, there are several circuit components that need testing, such as the LM7805 voltage regulators. In order to test the LM7805 voltage regulators we connected the input pin to the positive terminal of a DC power supply and the ground pin to the negative terminal of a DC power supply. With the power supply turned on, we increased the voltage above the regulated output levels, which for the LM7805 can range from 4.8V to 5.2V, but within the appropriate voltage levels for the device. Using a multimeter, we measured the voltage between the input pin and ground pin to make sure an adequate electrical connection was made, followed by measuring the voltage between the output pin and ground pin. Once we saw that the output is providing a voltage between 4.8V and 5.2V, we were able to confirm that the LM7805 voltage regulators we ordered were properly working.

CD4051 Multiplexer Testing

When testing the CD4051 multiplexer we connected +5V to the inputs of the multiplexer and made sure that the appropriate voltages were being applied to power the device. Once this was done, we applied voltage to the select lines in order to select the input pin which had voltage applied to it. Using a multimeter, we measured the output relative to ground to make sure that a voltage was being outputted. The following figure shows the schematic for testing the CD4051 multiplexers.

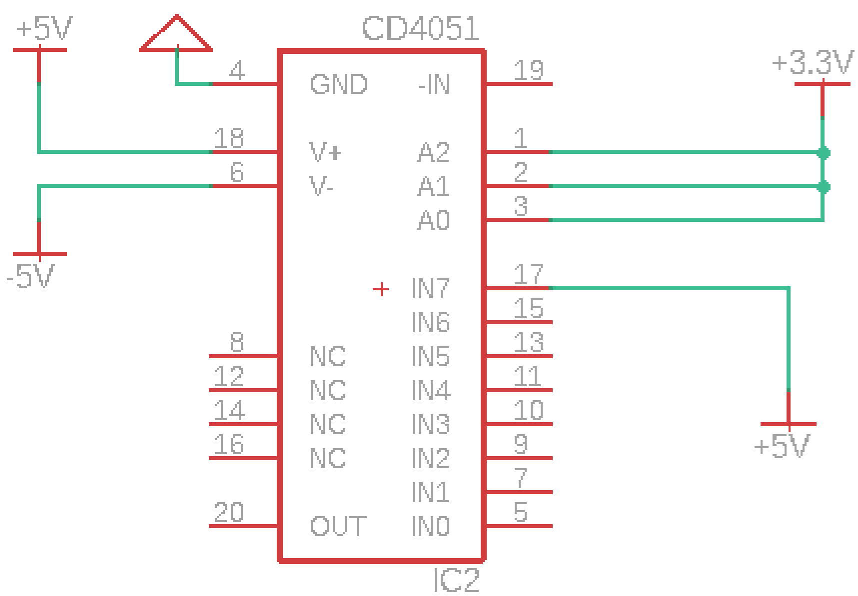


Figure 24: Circuit used to test CD4051 multiplexers

6.9 Frame Design

Unlike most smart mirror designs, this project does not have a real touch screen. Instead of a touch screen our smart mirror will take input through an optical grid which will be a close approximation of a touch screen. Due to this design choice, the frame of the mirror will have added complexity compared to other smart mirrors. This results in extra care being necessary when designing the frame of the smart mirror, because this particular mirror can have potential misalignment issues in addition to an increase in required internal space to route the extra wires.

In order to prevent misalignment of the LEDs and photodiodes when assembling the smart mirror we have decided to use wooden cutouts made from a laser cutter to ensure accurate and consistent spacing.

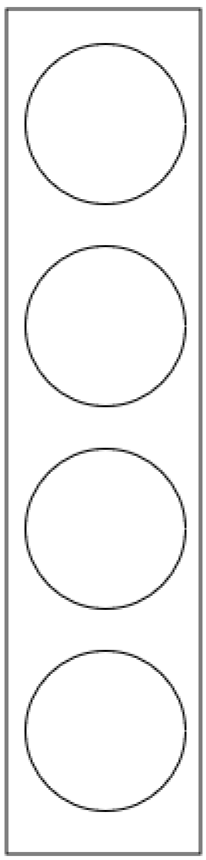


Figure 25: Cutout for LEDs and photodiodes

The figure to the left shows an example of the cutouts which would be used. The diameters of the holes will be just large enough to fit the LEDs and photodiodes inside of them. Once the components are placed in and the electrical connections have been confirmed, the components will be glued to the wooden cutout to ensure stability throughout the remainder of the frame construction.

These cutouts will be placed between the surface of the glass and the outer frame of the mirror with the cutout being perpendicular to both so that the light from the LEDs flow across the surface of the glass toward the corresponding photodiode. This methodology for installing the LEDs and photodiodes serves two purposes.

The first purpose is to make alignment of the LEDs and photodiodes as simple and automated as possible. By using a laser cutter to create the outer dimensions as well as the spacing and diameter of the circular cutouts we will prevent potential issues with alignment.

The second purpose is to provide adequate space within the frame of the mirror to route the wires from the LEDs, photodiodes and other electronic components.

In addition to the LEDs and photodiodes being embedded within the frame, the camera will also be embedded within the frame. A hole will be cutout from the surface of the frame to allow the camera through. However, depending on where the user decides to place the mirror, a camera may not be ideal or even desired. To combat this, a sliding camera lens cover will be installed within the frame to prevent potentially unwanted features from being abused. We decided to use a simple, mechanical method to shut the camera off from outside view to prevent speculation on the user's part. Every individual, regardless of their field of work or level of education understands that when you place a non-transparent object in front of a camera, the camera cannot see past that object. However, not every person understands how electronics work. Had we decided to turn off the camera electrically without being able to cover the camera, it is very likely that some users would begin to assume that the mirror will be used to spy on them. By making a simple cover for the camera this suspicion has no credence and user trust can be more easily retained and obtained.

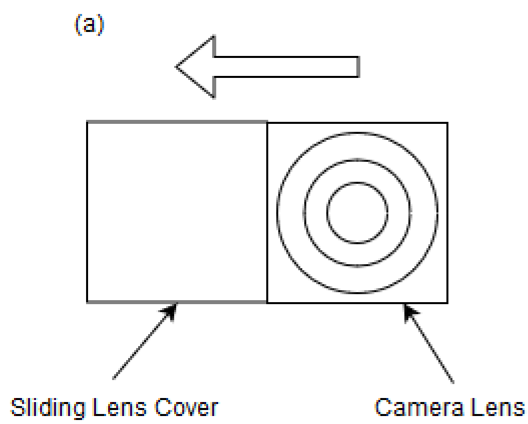


Figure 26: Sliding lens cover. Part (a) shows the camera fully visible with the lens cover open. Part (b) shoes the lens partially covered with the cover closing.

The material used for this frame should ideally be waterproof. Since we are going for a bathroom mirror design, this mirror is most likely going to come into contact with water or steam pretty often. The best material to use would be some sort of plastic material because its 100% waterproof and it comes in varying colors, so we would not have to paint the frame afterwards. The second-best choice would be some sort of hardwood, which we would still need to apply some sort of waterproof coating on and possibly paint/stain it as well. If we were to use some kind of softwood, water would easily penetrate if we do not do any waterproof coating and it would rot the wood and disrupt the integrity of the frame strength.

7 Project Prototype Testing Plan

The following section will contain the steps on which we plan to test our prototype to make sure that the outcome of the project will function as planned. Some parts will be tested individually, perhaps using different components for the purposes of testing the concepts. It will also include not just the hardware parts testing steps but also the software testing methods.

7.1 Test Methods

We decided between three development method that we like to implement for our project. In Software Development Life Cycle or SDLC is an outline that is utilized to structure and plan the process of developing a software system. or SDLC is a sequence of stages that specify common modalities or best practices of the software development process. The goal of the SDLC process is to provide a detailed definition of what the system requirements are. This is needed to make sure that everyone involved in the process has a clear understanding of the tasks assigned to them and how every task should be implemented. There has been a significant expansion in regard to the variety of models. This system diversity can be attributed to the wide variety of product types developers can encounter. starting with a simple website to a complex medical software. An important thing to note is that regardless of the model that is chosen, it should be adjusted to fit the scope of the product, project, and company.

Waterfall

The Waterfall software development model is a sequential approach in which development process is structured as progressively flowing downwards system through the different stages of t analysis, design, implementation. The development process is usually segmented into sequential stages, that will have some overlap acceptable between the stages. The emphasis is placed on planning, schedules, and the overall implementation of the entire simultaneously. Written documentation is a vital part of the development model.

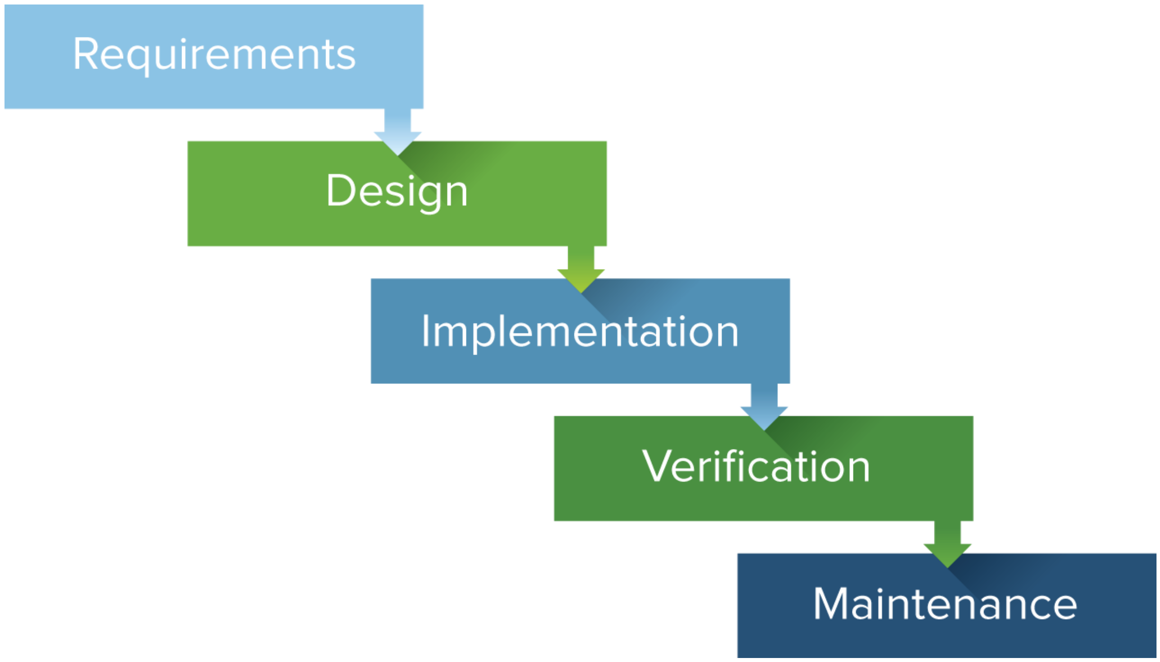


Figure : Waterfall Development Method

Agile

Agile software development is a development methodology that is based on an iterative/incremental approach to software development. In this particular model the requirements for the project and the manner in which they are solve will evolve over the course of the development process. By collaboration between development teams this model promotes an adaptive approach to planning, with regularly evolving development and delivery schedules, and an iterative approach, this encourages a rapid and flexible response to development modifications.

Figure 27: Agile Development method

Spiral

Spiral software development combines elements of both the design and prototyping stages of SDLC. This is done in an effort to take advantage of both the benefits of top-down and bottom-up approach. This model of development combines the features of the prototyping model and the waterfall model. The spiral model is the preferred method of development for large and complex projects. This model uses a lot of the same stages as the waterfall model, in the same order, separated by planning, risk assessment, and the building of prototypes and simulations.

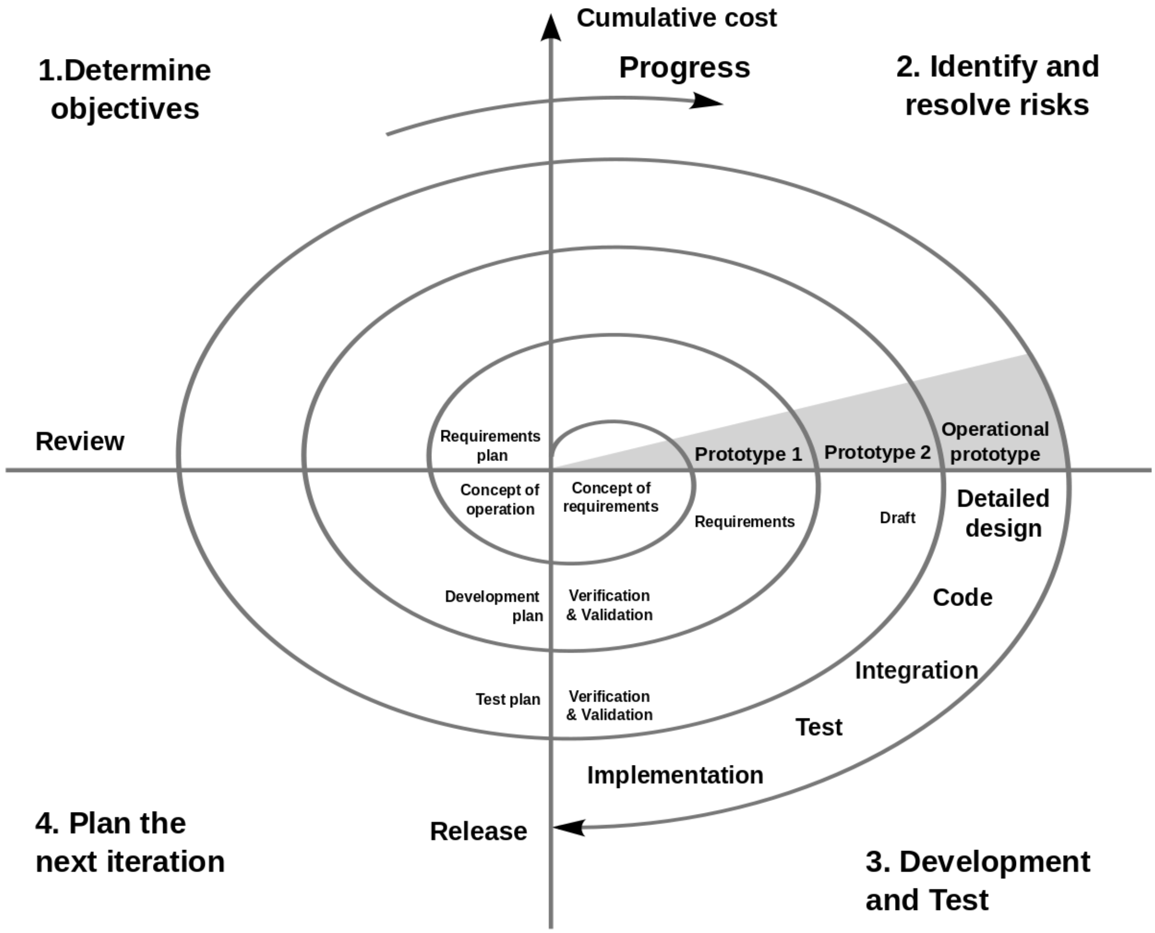


Figure 28: Spiral Development Method

7.2 Software Design

The Raspberry Pi 3 Model B is a single board computer with both wireless LAN and Bluetooth connectivity. The face recognition software will be called on the backend of the system by the OS to enable or disable user functions the mirror.

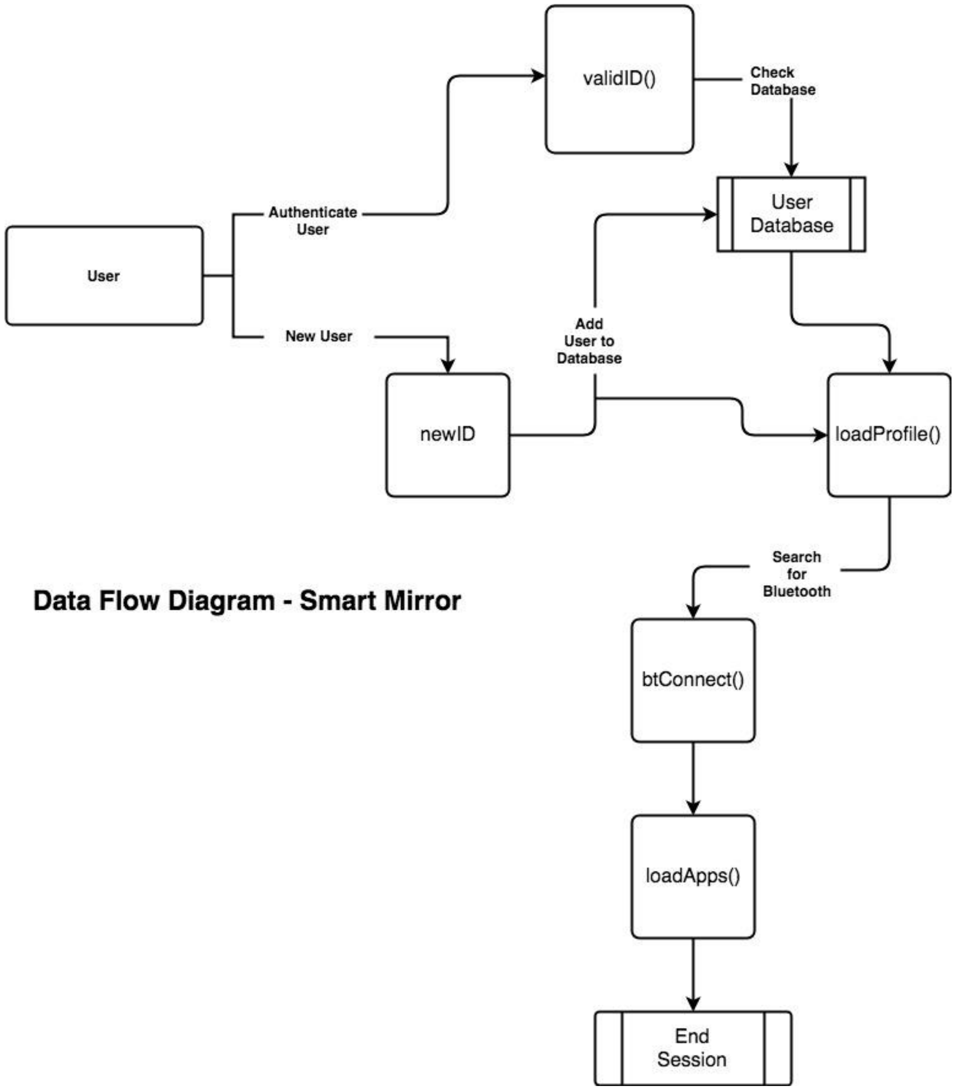


Figure 29: Data Flow Diagram For Operating System

7.3 Installation

The operating system we'll be using for our project is Raspbian. It's an extremely flexible open-source system and has serious community support. Once we downloaded the OS image we had to write the image to an SD card using the r-disk method as described in the documentation for our Raspberry pi.

We booted up the Pi launchpad and start configuring the module using the configuration wizard by entering command line prompt sudo raspi-config in the terminal. We need to make sure that the system had the ability to boot to desktop and not just the command line. We configured the time to so that we could use that process for our application instead of having to write one from scratch.

7.4 Screen rotation

We are designing our system to be used in used in both landscape and portrait mode. Therefore, we will have a script that will rotate the screen 90 degrees based on the orientation of the mirror. We were perplexed about how we would accomplish this at first, but we found out that it wasn't as hard as we thought it would be.

In the 'BIOS' settings of the Pi 3 Launchpad there is a stored /boot partition of the system. This partition holds a configuration file that has all of the settings for the module. In order for us to rotate the monitor we just have to manipulate this setting.

7.5 UI System Design

Due to the fact that we will be using a semi-reflective material for our smart mirror. Our mirror itself will only physical work as a mirror as long as there isn't a light behind it. It is extremely important to make that background for the system is black or dark. We get the best contrast when the information display is white against a black background.
We would like to add some colors to differentiate applications of course but for the main system itself we wanted to create as clean UI as possible. We don't want the colors to overpower the mirror.

7.6 Backend API

API stands for Application Programming Interface. API's are a piece of software that allows multiple applications to communicate with one another. This means that an API is the system that delivers service requests to a client from a provider that the information is being requested from. The system then returns a deliverable back to the client.

The API will define the functionalities that operate independent of their respective executions, which allows those functionalities and classifications to differ without conflicting with each other. In this regard, an API makes it easier to develop programs by providing building blocks that streamline the development process.

When we're creating our code for our backend API, we didn't have to begin the process from scratch. A fundamental part of improving our productivity was how the API made complex processes very reusable with simple lines of code. The speed of the API allowed us to build our application to keep up with pace of our application development.

We found that we were much more productive using an API instead of writing most of code from scratch. With the API we didn't have to reinvent the wheel every time we created a process for an application. We only needed to focus on the individual components of the application itself such as scheme design, while leaving all of the all of the communication functionality to API itself.

One of the biggest advantages of using APIs is that they allow for a certain independence of functionality between multiple systems within a program. The end-to-end of the API separates the receiving application from the framework that provides the service. As long as the deliverables from the service provider makes its way to the client and remains unchanged, any changes to the framework on the backend should not be noticed by the applications that rely on that API. This gives the service provider a lot of flexibility in regard to how particular services are offered. For our smart mirror, if the framework for the API involves server communication, the content provider can simply switch from physical servers to virtual ones.

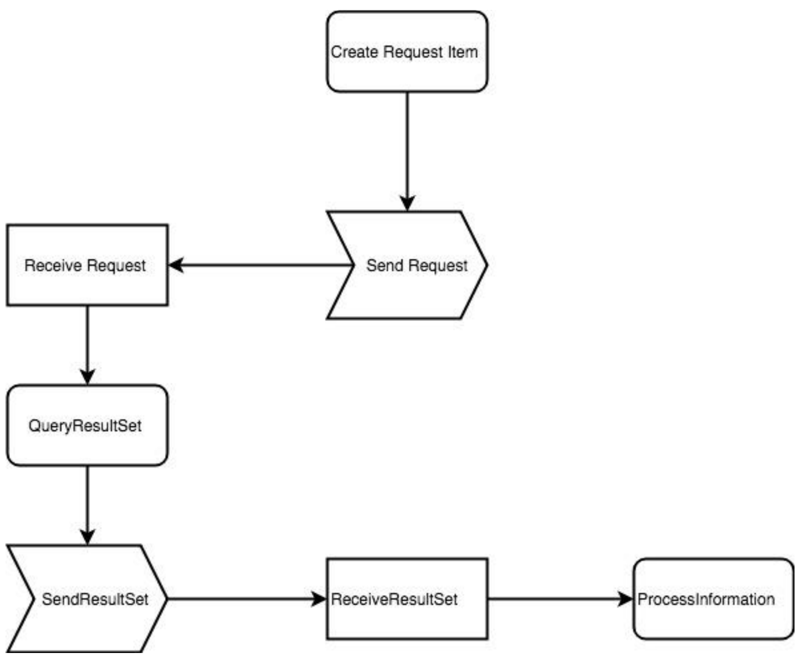


Figure 30: Data Flow Diagram for Back End API

7.7 Features

When we first started the project, We decide that the interface would allow for limited user interaction with the mirror itself. This is because we want the mirror interactive and still function as one and no one wants an excess of smudges on their mirror. Most importantly we found that we want our smart mirror to have as passive interaction as possible.

Making sure that the mirror could be used as a mirror without having the view impeded by excessive information. We wanted to make use of the outer sections of the mirror so that content can be viewed while at the same time leaving enough room for people to see themselves.

There was certain information we felt we needed to display on the main screen for the user to see:

- **Start up message**
  Once the user is recognized we need to have a personalized message for that particular user.

- **Weather**
  We will be showing them the forecast based on their location

- **Clock/Calendar**
  We will be integrating the internal clock and a calendar system.

- **News feed**
  We thought it was important display daily news for the user

We also entertained ideas for other possible home screen applications

7.8 Auto refresh

Early on in the development of our project we discussed nontraditional input methods. Being that we will not be have a keyboard and mouse connected to our Smart mirror whenever our system is updated, there would be no easy way for us to refresh the applications on the smart mirror. One solution we found and the most intrusive was to restart the entire module and have it update on start, but would be highly inefficient and take an extremely long time to check for updates

Since we have the ability to update our files using the GIT version control system, we determined that every update would have its own commit hash, which we would be able to read using PHP. This gave us the ability to add a piece of script that would compare the commit on the local host and against the one that is being displayed on the screen. If the two hashes don’t match, then the script should reload the page and display the most updated version.

This comparison will complete by the main js file. But this will only happen if the main application file has the most current version of the commit hash available.

The comparison itself will be straight forward as it will be checked every 5 seconds. This will allow us to update the smart mirror interface, by logging into our module using ssh, push our changes to the appropriate folder, and completing a pull.

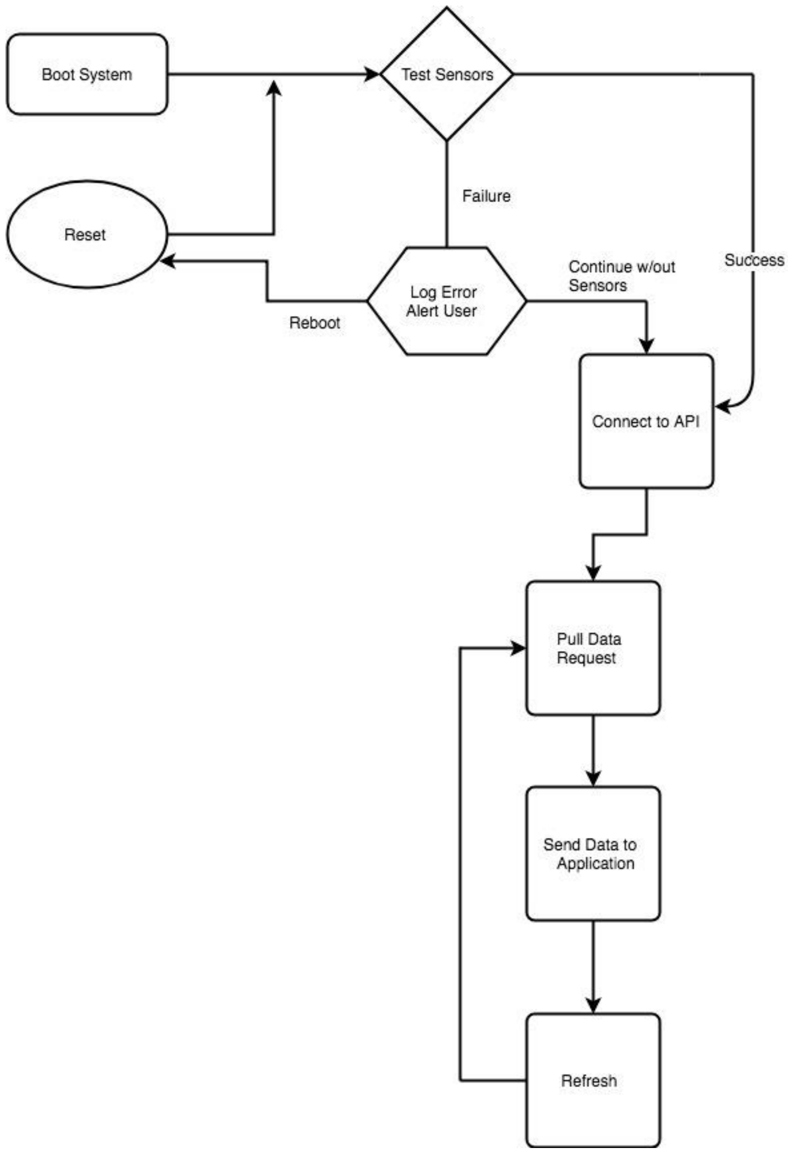


Figure 31: Data Flow Diagram Post-Boot Data Pull

7.9 Kiosk mode

What we want is for our system is to make sure that the Raspberry module actually shows the application in kiosk mode on launch. What this does is that it completely disables all screen saving features and makes sure our application starts after the module boots and points to our application in full screen mode.

7.10 Languages

Python:

Python has a comprehensive system library, that is also very well integrated. The libraries are usually very simple to implement, and you can begin using them very basic understanding of the language. Python is an object-oriented language, but most errors are detected at run time because it’s interpreted language rather than compiled.

Python allows programmers to be highly productive working individually due to its vast yet simply implemented features. Larger scale python development has issues. In some senses Python can be considered strongly typed, but not in the same way c++/java is expected. An example of this is function signatures. They never specify a type for return or input, so it is considered best practice to use distinguishable variable names. The types that are passed to a function are not resolved until they are called. It is entirely possible and perfectly acceptable for a function to return multiple types from different locations within function.
Python also comes with thread support but due the Global Interpreter Lock (GIL) it's not normally worth it to use.

Python manages memory automatically using a garbage collector. For speed Python compiles a script into bytecodes resulting in a PYC file that loads faster into the interpreter.

C++

C++ gives programmers access to low level hardware and direct access to memory. This puts C++ in a unique class in comparison to the other two high-level languages we mention. C++ is an Object-Oriented Language. In comparison to other Object-Oriented Languages C++ has always bogged from a somewhat labored syntax largely due to the fact that it supports a lot of legacy C code and still adds enough new arrangements for all the OOP concepts that were integrated as well.

C++ is a good choice in regard to ensuring performance at the sake of accepting some complexity to achieve that goal. In the past c++ had a pretty limited system

library although it was a language developed for creating system libraries, but the language itself did not provide a lot of support for it. This was eventually modified by improved libraries, and now most of these libraries are incorporated into the new versions of the language
With its latest enhancements c++ has made a resurgence and has a lot of support. it is still more challenging to get used to the syntax than Python.

Java

Java is an object-oriented language with a syntax that is similar to c++. Java has an extensive amount of libraries and frameworks. The productivity of a programmer tends to be higher due to its flexibility, but the learning curve for Java is relatively the same as c++.

The difference between Java and c++ is that it runs in a virtual machine called a JVM. The JVM manages memory through a process called garbage collection. Because it runs byte codes it is entirely possible to assemble other languages into the same code.

	C++	**Java**
1.	C++ was developed by **Bjarne Stroustrup**. Development began in 1979.	Java was developed by **James Gosling** and his team. Development began in 1991.
2.	C++ is a **compiled language**.	Java is both compiled and interpreted.
3.	C++ supports **conditional compilation** and inclusion.	Java does not support conditional compilation.
4.	C++ programs are **platform dependent**. They need to be compiled for a particular platform.	Java programs are platform independent. Java programs are written for Java Virtual Machine (JVM) and wherever a JVM is installed, Java program will run without needing recompilation.
5.	C++ does support **operator overloading**. Function overloading is also available.	Java does not support operator overloading. However, function overloading is possible.
6.	C++ fully support **pointers**.	Java has restricted support for pointers. Pointers are supported internally you can not writer pointer programs.

7.	C++ supports **structures**.	Java does not support structures.
8.	C++ supports **unions**.	Java does not support unions.
9.	C++ does not have built-in support for **threads**.	Java fully supports threads.
10.	C++ supports manual **object management** through *new*and *delete* keywords**.**	Java relies on automatic garbage collection. It does not support destructors the way C++ does.
11.	C++ supports ***goto*** statement (however the use of goto is discouraged as not considered a good practice)	Java does not support *goto* statement (although *goto*is a reserved keyword in Java)
12.	C++ supports **multiple inheritance**.	Java does not really support multiple inheritance. But similar results can be achieved through the use of interfaces.
13.	C++ provides support both for **call by value** and **call by reference**.	Java supports only *call by value*.
14.	C++ does not support **comments within source code**.	In Java programs, you can write comments using */** … */*
15.	C++ has no support for the **unsigned right shift** operator (>>>).	Java supports the unsigned right shift >>> operator.
16.	C++ provides ***virtual*** keyword to support function overriding.	Java does not support *virtual* keyword. All the non-static Java functions are by default virtual in nature, and therefore, can be overridden.

8 Integration & Improving the Design

Once the PCB design has been sent out to be manufactured the team intends to begin the final construction of the smart mirror. Connectors will be used to connect the wiring from the LEDs and photodiodes embedded in the frame of the smart mirror to the custom PCB in order to allow for easy removal of the board if tweaking or a rework of the circuit design is needed.

When all the components have been assembled into the initial smart mirror design, testing will begin. This will be done by first confirming that the hardware works both as intended and reliably to allow for a seamless user experience. To confirm that the hardware is working properly the team will repeatedly provide inputs through the various methods available, such as facial recognition, using the gesture sensor, and using the optical touch grid. Using an oscilloscope with probes connected to the various parts of the circuit board we can view the signals as we provide inputs in order to ensure that there is no excess noise somewhere in the signal path.

Once the hardware bugs have been ironed out, the team will begin testing the software. Software testing will be done by first checking the connectivity of the smart mirror by connecting it to various smartphones of various brands and operating systems via Bluetooth. This will allow us to confirm that we are able to obtain a consistent and reliable connection. Additionally, the Wi-Fi will be tested by making sure that the smart mirror is pulling relevant and accurate information for viewing, such as the local weather.

The inputs to the smart mirror must also be checked. Although by this point we will have already confirmed that the signals are propagating as intended, we must still make sure that the user interface responds appropriately and that there are no bugs in the software which would cause the input to be misread by the smart mirror.

The final stage of testing will consist of either a team member or a close friend of one of the team members using the smart mirror in their daily lives. This will provide us with feedback and constructive criticism regarding the features and user interface implemented.

9 Administrative Content

This section will be where the management portion of the project can be found. This section will discuss the current project schedule as well as the desired milestones and budgeting. It is important to note that this portion, like the rest of this document, is subject to change as the project evolves.

a. Budget & Funding

Our project will be self-funded and based on the research we have done, and will continue to do, as we progress with this project and improve the design. Currently we believe that it is possible to keep the total cost of the components under $550.

Table 8: Smart Mirror Budget

Device	**Quantity**	**Cost estimate**
Two-Way Mirror	1	$50 - $100
Monitor/Tv	1	$150 - $200
Processor/micro-controller	1	<= $10
Camera	1	<= $20
Wifi chip	1	<= $10
Bluetooth Chip	1	<= $10
Optical Touch Grid	N/A (multiple light sources)	$75 - $100
Gesture Sensor	1	<= 20
Custom PCB	3	$30 - $50
Power Supply	1	$15 - $30
Total Cost	**N/A**	**$390 - $550**

b. Project Milestones

These project milestones are tentative as we continue to improve our design and update our estimates for both cost and time to completion.

Table 9: Project Milestones

Milestone Name	Duration	Week
Senior Design 1		
Brainstorm for project ideas	3 weeks	5/14 - 6/4
Divide and conquer project report	5 days	6/4 - 6/9
60 page documentation report	~ 4 weeks	6/9 - 7/6
Design PCB & order all necessary items for project	~ 2 weeks	7/6 - 7/30
100 page report	~ 2 weeks	7/6 - 7/20
120 page final report	~ 1.5 weeks	7/20 – 7/30
Senior Design 2		
Building project prototype	~ 4 weeks	8/20 - 9/20
Test and adjust prototype	TBA	TBA
Finalize prototype	TBA	TBA
Peer presentation	TBA	TBA
Final report	TBA	TBA
Final representation	TBA	TBA

c. Division of Labor

The figure below shows the current division of labor. Tasks have been divided based on hardware and software tasks and everyone's respective strengths. This is subject to change depending on how the project changes as we get further into the building process. The division of labor shows what each person will be held responsible for. Since many of these components are dependent on each other there will be collaboration between the team members, even on components where there is a single individual responsible for its completion.

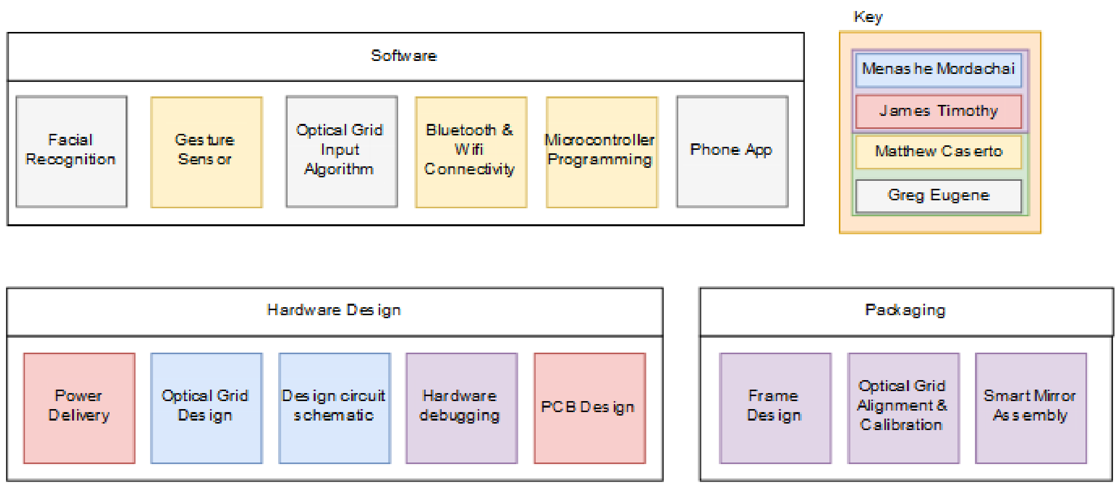


Figure 32: Current division of labor