Final Report for Project Lab 2 at Texas Tech University

Project Interactive Halloween Pumpkins

Stephanie Gregory

Partners: Damien Archer, Jacob Rothschild, Loica Carmelle Tatmba Ngatchoua

Project Lab 2 ECE 3332

Texas Tech University

Dec 2022

Abstract

This paper describes the implementation of designing 2 unique Halloween pumpkins using a Raspberry Pi that will be able to perform different functions based on a visitor’s gesture/movement. The pumpkins also need to be weather resistant, have auto turn off/on and have enough charge to last at least 4 hours. In order to fulfill these requirements in the small timeframe of six weeks; two already 3D printed pumpkins were used and modified to fit our design needs. Each pumpkin used a USB webcam camera and a Raspberry Pi 4 model B+. Each pumpkin also used a USB power bank to power the Raspberry Pi and a 7.4 Volt LIPO battery with a 5-volt regulator to power design features attached to the pumpkins. Design features will be discussed in more detail later, but consisted of 2 led rings, 3 – 8 x 8 led matrix panels, 2 TFT LCD displays with an Adafruit Snake Eyes Bonnet, 4 Hs- 422 servos, a speaker, and a 12-inch LED ring light with a 73inch tripod stand also powered by a USB power bank. In order to perform different functions based on a visitor’s movement; the use of OpenCV, TensorFlow, and DeepFace software were also implemented on the raspberry Pi’s to detect facial expressions. And all code that was used had to be done using the coding language python. This project also had to be demonstrated at the Lubbock Pumpkin Trail Halloween event for four nights. These constraints were all met and the project was extended to add onto these components with the idea of making the pumpkins better in mind. Elecrow 5-inch HDMI 800x480 touch screen LCD displays were added to each pumpkin for viewers to see themselves as well as decorations and code was modified for the Christmas Holiday season. Additions will be discussed after the conclusion section; see Table of Contents below for details.

Table of Contents

Abstract……………………………………………………………………………………………1

Table of Contents………………………………………………………………………………….2

List of Figures……………………………………………………………………………………..3

1. Introduction………………………………………………………………………………..6
2. Body……………………………………………………………………………………6-31

2.1 Design…………………………………………………………………………6

2.2 Hardware………………………………………………………………………9

2.3 Software……………………………………………………………………...17

3. Conclusion……………………………………………………………………………….31

4. Alterations………………………………………………………………………………..34

5. Appendix A- References...……………………………………………………….………40

6. Appendix B – Budget/Materials…………………………………………………….…...41

7. Appendix C-Ethics/Safety……………………………………………………………….43

8. Acknowledgments…………………………………………………………………….….44

List of Figures

Figure 1: Interactive Pumpkins Sketch……………………………………………………………..

Figure 2: W-King Speaker…………...……………………………………………………………..

Figure 3: Blackweb Portable Charger…………..…………………………………………………..

Figure 4: PowerXcel Portable Charger……………………………………………………………..

Figure 5: LED Ring Light…………………………………………………………………………..

Figure 6: Onn Power Bank…………………………………………………………………………

Figure 7: Raspberry Pi 4 and Components…………………………………………………………

Figure 8: USB Camera……………………………………………………………………………...

Figure 9: Pi 4 GPIO Pins…………………………………………………………………………...

Figure 10: LED Ring……………………………………………………………………………...

Figure 11: LED Matrix Panel……………………………………………………………………..

Figure 12: 7.4 Volt LIPO Battery…………………………………………………………………

Figure 13: Voltage Regulator Circuit……………………………………………………………..

Figure 14: CPU Fan……………………………………………………………………………….

Figure 15: HS-422 Servo…………………………………………………………………………...

Figure 16: Adafruit Eyes Kit………………………………………………………………………..

Figure 17: Face Detection Code 1………………………………………………………………….

Figure 18: Face Detection Code 2………………………………………………………………….

Figure 19: Expression Detection Example…………………………………………………………

Figure 20: Pumpkin 1 Flowchart…………………………………………………………………...

Figure 21: Pumpkin 2 Flowchart…………………………………………………………………...

Figure 22: Servo GPIO Setup………………………………………………………………………

Figure 23: Servo Duty Cycle Setup………………………………………………………………...

Figure 24: Servo Implementation……………...…………………………………………………...

Figure 25: LED Setup………………………………………………………………………………

Figure 26: LED Matrix Panel Setup………………………………………………………………..

Figure 27: LED Strip Setup………………………………………………………………………...

Figure 28: LED Text Scrolling……………………………………………………………………..

Figure 29: LED Ring Animation 1…………………………………………………………………

Figure 30: LED Ring Animation 2…………………………………………………………………

Figure 31: colorWipe Example……………………………………………………………………..

Figure 32: theaterChase Example…………………………………………………………………..

Figure 33: Using Pygame…………………………………………………………………………...

Figure 34: Final Pumpkin Products………………………………………………………………...

Figure 35: EEPROM Booting………………………………………………………………………

Figure 36: LCD Screen……………………………………………………………………………..

Figure 37: Adjusted Camera Code………………………………………………………………….

Figure 38: Altered Interactive Pumpkins…………………………………………………………...

Figure 39: Pumpkin 1 Mood Expressions/Colors…………………………………………………..

Figure 40: Pumpkin 1 Text Alterations…………………………………………………………….

Figure 41: Pumpkin 1 LED Ring Alterations………………………………………………………

Figure 42: Pumpkin 2 Sound Alterations…………………………………………………………...

Figure 43: Final Altered Pumpkins…………………………………………………………………

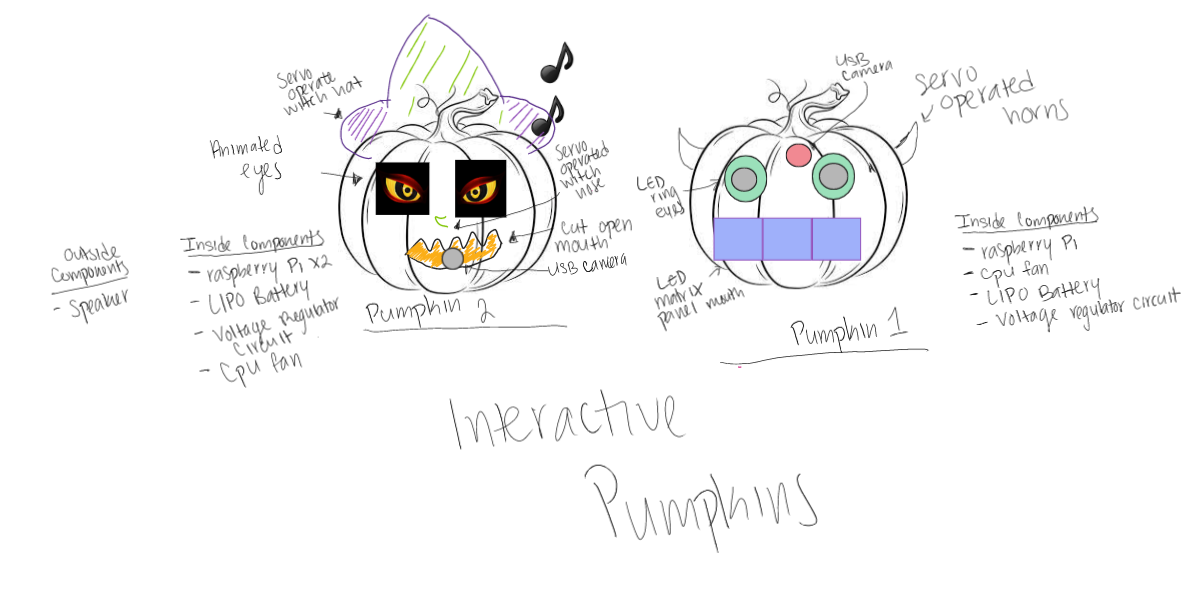
For an object to be interactive it must allow a user to perform tasks then based on those tasks, perform an action whether it’s a display on a screen or monitor or a physical action. In our case after the user performs a facial expression. As seen in below in the *Interactive Pumpkins Sketch Figure 1*, there are two pumpkins labeled ‘pumpkin 1’ and ‘pumpkin 2’. 

Figure Interactive Pumpkins Sketch

Pumpkin 1 consists of two led ring lights that perform as the pumpkin’s eyes, 3 – 8x8 led matrix panels that acts as the pumpkins mouth, and two horns atop of its head to be devil horns. Pumpkin 2 consists of two TFT lcd screens that are the pumpkins eyes, a carved-out mouth, a witch nose as well as a witch hat.

Figure W-King Speaker

Figure Blackweb Portable Charger



Figure PowerXcel Portable Charger

The second pumpkin also consists of a *W-King 80W* speaker that can go up to 105dB and can operate with the attachment of an auxiliary cord from the raspberry pi to the speaker. It charges with a 5V/2A power adapter that comes with the device and shouldn’t need to be charged again for this particle project. It has up 24hours of battery life, and our event accumulated to about half of that time. Both pumpkins also have a USB webcam; for pumpkin 1 it resides in the forehead of the pumpkin, and for pumpkin 2 it resides on the mouth of the pumpkin. All raspberry pi’s are powered by portable chargers. Pumpkin 1 used the *Blackweb 20100mAh Portable Battery* to power the two pi’s that were needed to operate it and Pumpkin 2 just used the *Powerxcel Portable Charger* to operate it’s one pi. Its capacity is 6600mAH and has an output of 5 Volts 1Amp or 5 Volts 2.1Amp depending on the usb outlet that is used. Now each pumpkin also has its own facial emotions that it will read and perform its actions. Pumpkin 1’s emotions are sad, happy and surprise while pumpkin 2’s emotions are happy, neutral, and fear. There are few other emotions that could have been used, but it was decided to keep each pumpkin at three to limit the variety of faces a user could make as well as some emotions are quite similar.

Figure LED Ring Light

Figure Onn Power bank

Lastly, due to the pumpkin trail commencing later in the evening a 12-inch dimmable ring light *LED Ring Light* was also implemented when it started to get dark so both cameras could easily see guests faces. It is powered by an *Onn portable Bank* which supplied 5V/2A.

For the hardware components it all depends on the design pieces of each pumpkin. For similarities both pumpkins are using the raspberry pi 4 model B+. *The Raspberry Pi 4 and its Components* are pictured below in figure 7.

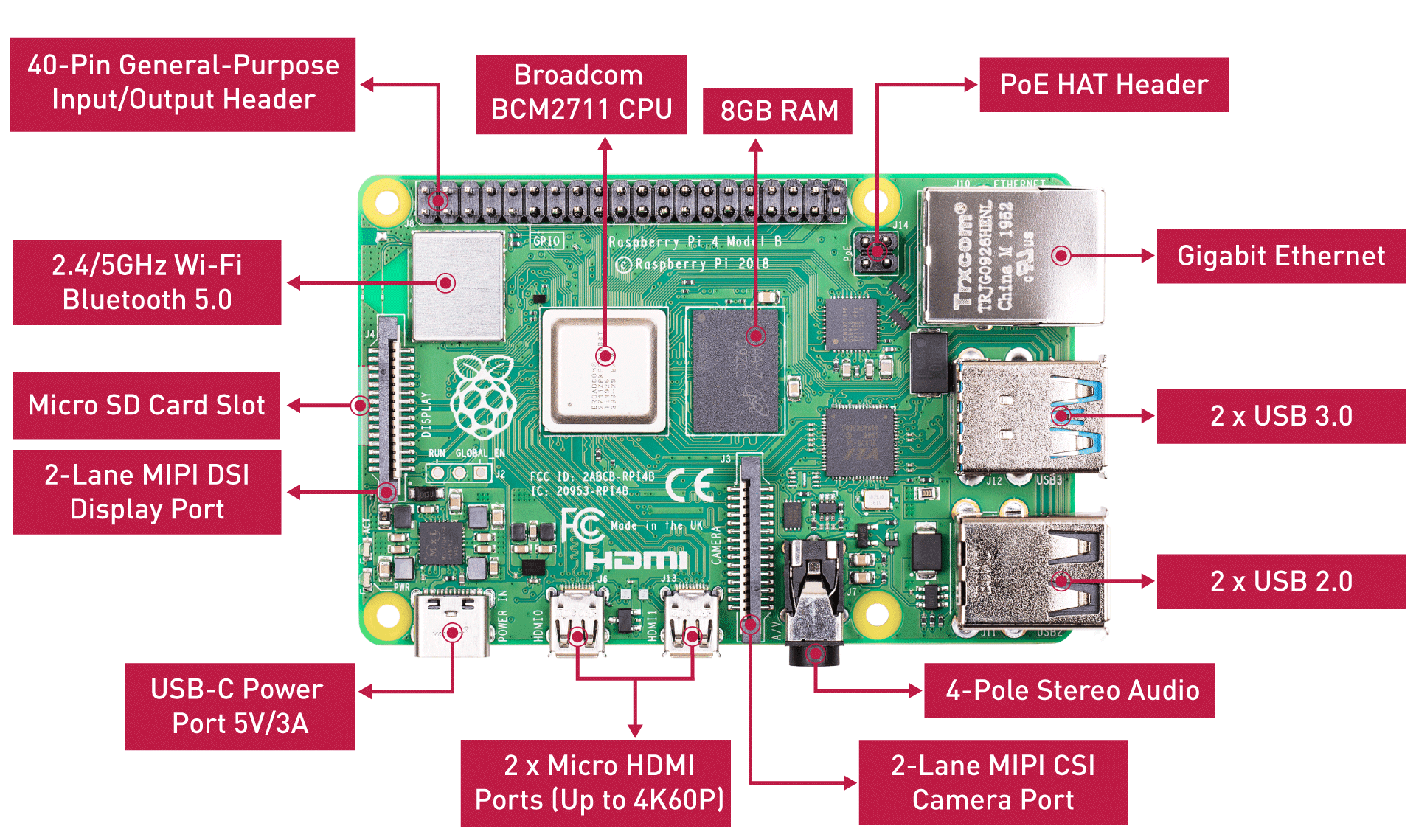


Figure Raspberry Pi 4 and Components



Figure USB Camera

The raspberry pi is a mini computer that can be used for numerous functions. It runs on any of the Raspberry Pi Operating Systems that are available to download from the Raspberry company app Pi Imager. After the operating system is downloaded onto a microSD card and implemented into the pi 4 board, the Pi is now able to operate like any computer by attaching a monitor, keyboard, and mouse into the USB and micro-HDMI ports on the Pi. Although from *The Raspberry Pi 4 and its Components* figure you see they are specific ports for a camera, it was decided that it was easier to obtain a better resolution USB camera that could be plugged into any of the USB ports on the Pi and be used. The camera that was used for each pumpkin was the EMEET 1080P Webcam with a 70-degree field of view and auto light correction; *USB Camera* shown above (Figure 8). It also requires less care as a ribbon cord cable for the camera port connection requires great care as it is fragile and can be damaged easily.

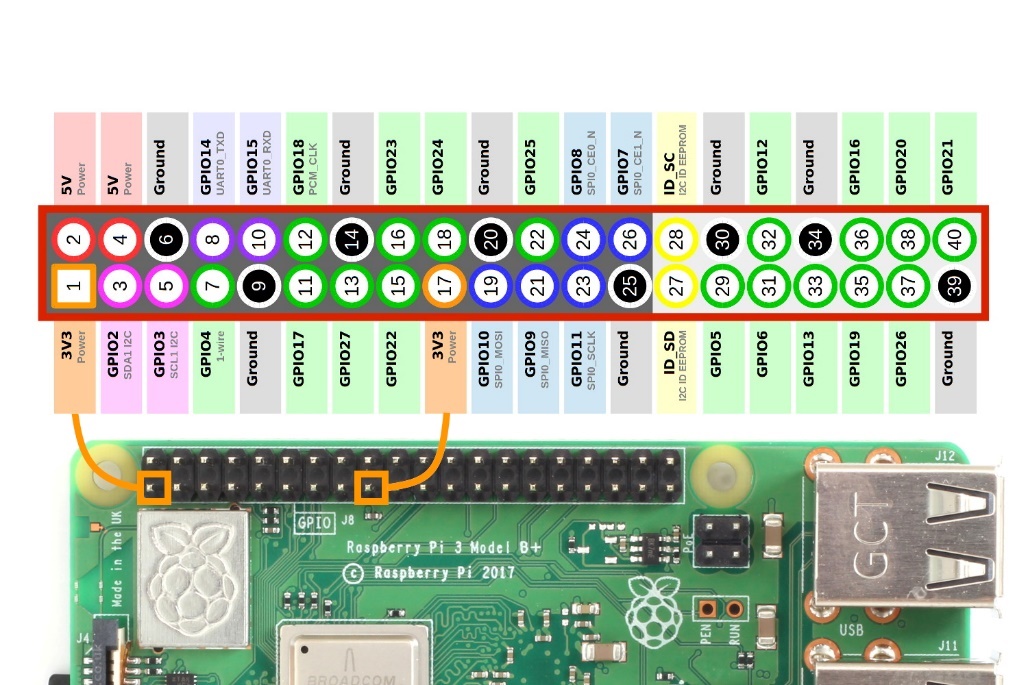


Figure Pi 4 GPIO Pins

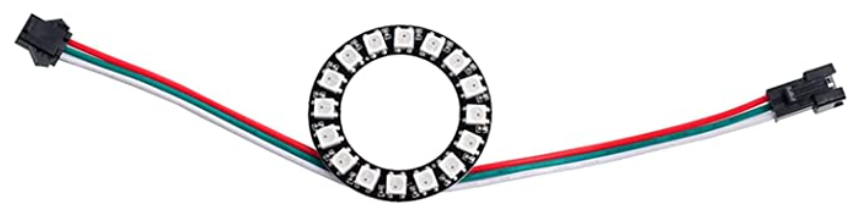
The Pi board also consists of input and output pins otherwise known as GPIO pins that are used to send or receive signals from our external devices *Pi 4 GPIO Pins* (figure 9) above shows the functions of each pin. They have two naming conventions; Broadband Chip-Specific Pin Numbers also known as BCM and physical pin numbers. The BCM numbering only numbers the pins you can use and not the ground and power pins. The physical pin numbering system numbers all 40 pins. For pumpkin 1 the main pieces to consider are the LEDs. 

Figure LED Ring

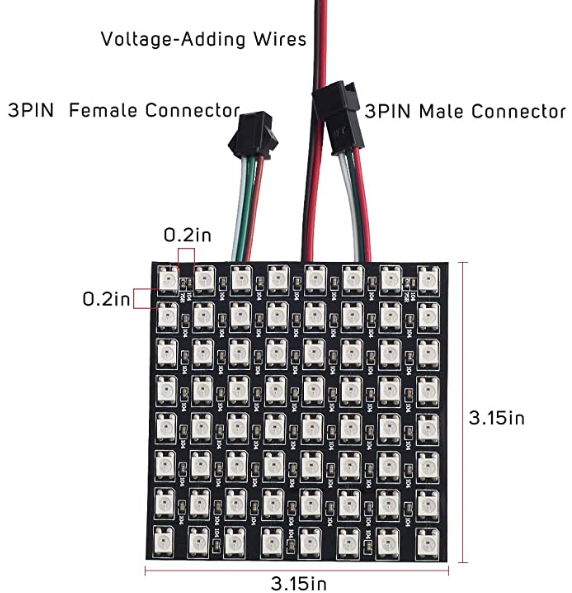


Figure LED Matrix Panel

The eyes consist of 2 led rings that consist of 16 WS2182b LEDs on each ring as shown in *LED Ring* (figure 10). Each LED on the ring has its own microcontroller which makes the ring individually addressable. This means each LED can be programmed individually and also be on at the same time. The mouth also consists of the same type of WS2182b LEDs with each panel consisting of 64 LEDs in the form of 8x8 depicted in *LED Matrix Panel*, These LEDs are also addressable. From reading the data sheet located in [Appendix B] both led apparatuses both need 5 volts to power on and for 1 panel if all the lights are set to full brightness can draw at most 16mA, but when tested it never drew that much current. The Pi does have two 5-volt power source pins that take the voltage directly from the rail, but these pins couldn’t be used for multiple reasons. When you use the 5-volt pin on the raspberry pi it limits the rest of the pins on the board in term of power and current. LEDs also draw more current than the board can provide, so connecting it could damage the board.



Figure 7.4 Volt LIPO Battery



Figure Voltage Regulator Circuit

This is why we decided to use a *7.4-volt LIPO Battery* typically used for toy race cars (figure 12) and step the voltage down with a 5-volt linear voltage regulator seen above in the *Voltage Regulator Circuit* (figure 13). Due to the time demand this circuit was implemented on a breadboard and powered the panels as well as the rings. The LED panel as well as the LED ring were both connected via daisy chaining which means the signal and power goes into one device and with connecting the input of the second ring to the output of the first ring, the signal is able to continue flowing and the program is able to run on both. This was convenient due to the limited GPIO pins that are able to run with the need of a clock. They are labeled on the pin pinout picture above. The panel used GPIO pin 18 while the ring used GPIO pin 21. It also helps add less stress to the raspberry Pi as it is prone to heating when operating, and when it overheats it either operates slower or turns off from the extreme temperature.



Figure CPU Fan



Figure HS-422 Servo

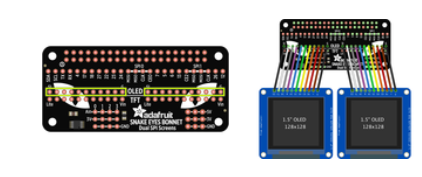
To help cool the Pi we implemented a computer fan to each pumpkin as seen in (figure 14). These fans also were able to operate using the circuit system that was created for the LEDs. These fans take the air from inside of the pumpkin and suck it outward to cool all the contents inside the pumpkin. For the case of this project, it helped with the cooling of the Raspberry Pi and the voltage regulator. A heatsink was also added to the voltage regulator to prevent it from overheating. Pumpkin 1 also had two horns that rotated by *HS-422 Servos* (figure 15). These servos received 3.3 volts each directly from the raspberry pi. Servos run off of pulse width modulation. This means the position the servo turns corresponds to specific duty cycle. A duty cycle is the percent of time in a period where the signal is on, so theoretically a 100 percent duty cycle means in the clock segment, you are sending a one to the data pin the entire time. The seros used for this project could only turn a maximun of 180 degrees. 

Figure Adafruit Eyes Kit

The second Pumpkin used the *Adafruit LCD eyes Kit* (figure 16). It came with 2 TFT LCD screens and Adafruit snake eyes bonnet to connect to the raspberry pi. Adafruit’s website listed in the resources provides thorough instructions to connect the color-coded wires from the bonnet to your screens for the proper configuration because it is also compatible with OLED screens. From there the bonnet connects to all 40 pins of the Pi. Pumpkin 2 also used two servos; one for the witch hat and one for the witch nose. As previously stated, the TFT screen occupy all of the GPIO pins on the board of the Pi, so to use the servos another Raspberry Pi 4 was implemented. A GPIO pin expansion board was initially supposed to be implemented, but it will be explained later when discussing software why that wasn’t a viable option. An expansion board is a board with more pins than a raspberry Pi. This board is able to connects to the GPIO pins on the PI and all the extra pins on the expansion board are addressable through SPI. This allows more external input and output connections to the Pi. Every component mentioned for pumpkin two can be powered via the PI, but the use of the voltage regulator circuit was also implemented for this pumpkin as well. In conclusion both pumpkins components received power from the 7.4 volt stepped down to 5 volts power circuit. All three raspberry Pi’s; one for pumpkin one and two for pumpkin two received power from two USB power banks. One for each pumpkin. For better lighting at night so the camera could see the guest, a ring light was also implemented for when the sun went down and that was also powered by a USB power bank.

The software for these interactive Halloween pumpkins was the biggest part of this project. It was required to only use a Raspberry Pi and python coding language to operate these pumpkins. Before you can use the Pi, you must first install the Operating System. Raspberry Pi has an app called Pi Imager where you can download any of the operating systems to a microSD card. From there you can put the microSD card in the slot designated for the Pi. Some operating systems are older than others and some are designated for a desktop. For this project Raspberry Pi 64-bit desktop version was used because the software the camera used was last updated to work with this OS. In python you can download outside libraries and software to use, but it may not be compatible with the OS that you choose. The Raspberry Pi company updates its Raspbian Operating Systems about once a year and the desired software may not do that. It was found that the Adafruit Eyes program can only be run on the 32-bit Operating System while the software that was used to detect facial expressions only uses the 64-bit Operating System. For that reason it was discovered there could be no implementation of the GPIO expansion board, and the only solution was to run the code for the eyes on a separate Raspberry Pi which is why there is an implementation of three Raspberry Pi’s. After the operating system was placed onto the microSD card and the card was placed onto the Pi, the Pi would boot up. From there you can connect to the internet and configure the Pi to your needs. The OS also comes with two Integrated Development Environments otherwise known as IDEs to write code on. All code in this project was written using the IDE Thonny. To run code there are two ways. The first way is clicking play in the Thonny IDE, the second way is to use the terminal and run the code in the command line. Many times, you may have to use the phrase ‘sudo’ before typing your program name and the type of file. All of our files our python files so the extension of the file type is “.py” for python. Sudo means you are running the program as the super user. You would only need sudo if you downloaded a library using sudo and implemented it in your program, then your program will need sudo to run.

The biggest software component to implement were the USB cameras. Both cameras would need to detect facial expressions, then the other components will perform different actions after. OpenCV stands for Open-Source Vision Library and is a computer vision and machine learning library and allows the user to use that framework for projects involving a camera. TensorFlow is a machine learning and artificial intelligence open-source library. Deepface is a facial recognition system created by a group at Facebook. When they created this, they implemented TensorFlow and OpenCV into their system. So, when we downloaded Deepface in the Pi terminal it automatically downloaded OpenCV and TensorFlow, but it is also okay to download them separately. From there we set our camera to refresh every 10 frames and look to see if someone is in the camera view. If so, using the Deepface software look at what facial expression they are making. We then narrowed down the facial expressions to look for and to also only read the closest face if there are multiple faces in the camera view. If there are many faces within the same distance then the camera will not read any of the faces.

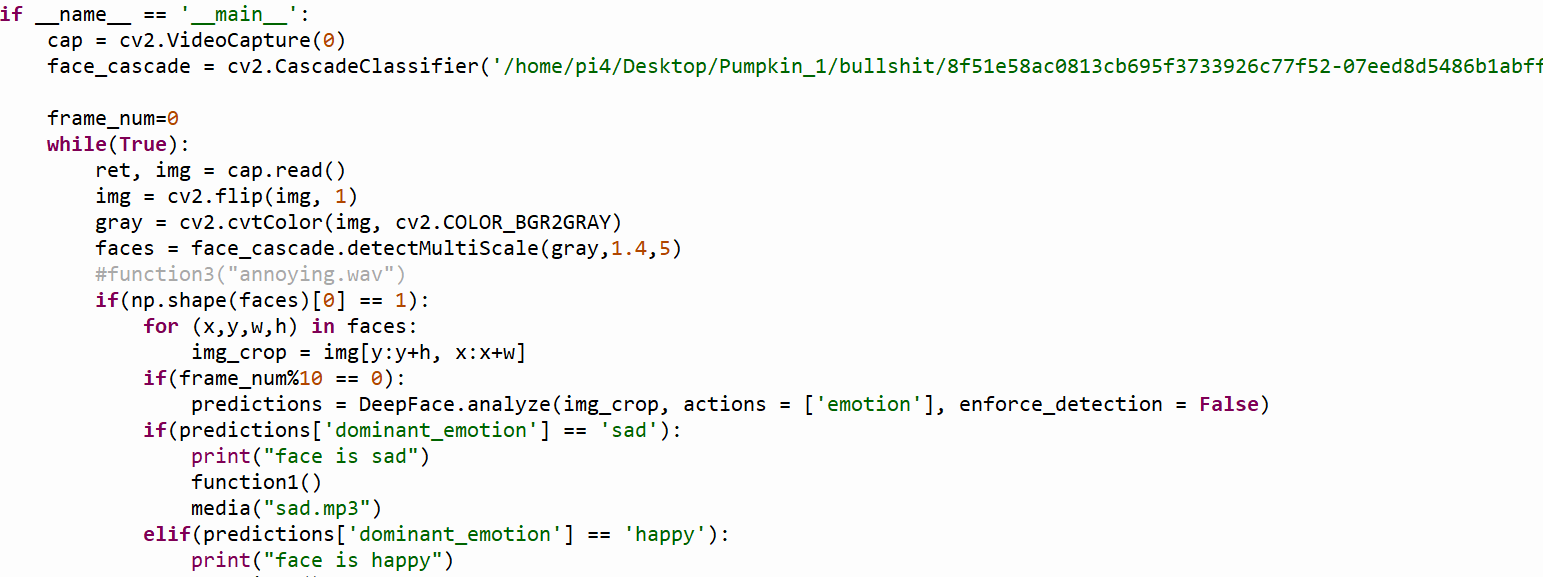


Figure Face Detection Code 1

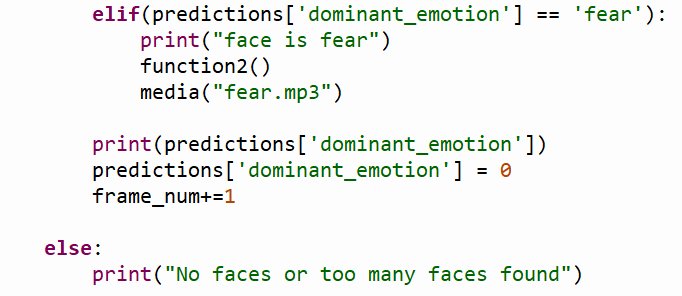


Figure Face Detection Code 2

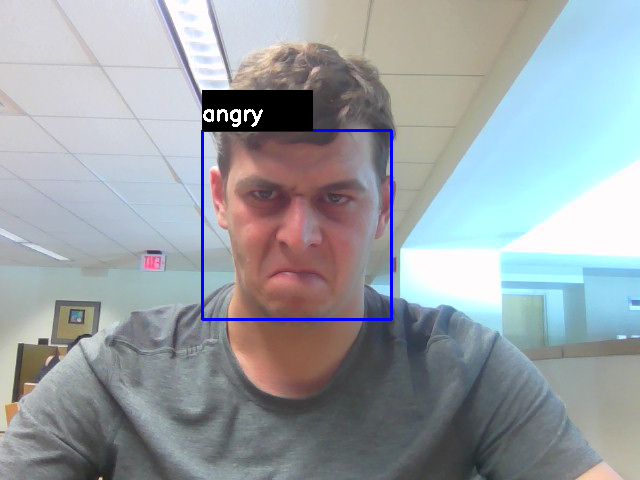


Figure Expression Detection Example

From looking at these code snippets from pumpkin two you can better see how the camera was implemented. First you see the calling of OpenCV and the use of it to start a video and get a still image from the video. From there you see the implantation of the DeepFace program to detect the facial expressions every 10 frames and depending on the expression the code moves on to perform specific function which will be discussed later in the document. And lastly you see if too many faces or no faces are detected then the program will do nothing. This is all in a loop so it is like the camera is waiting to detect a face. An example of the camera seeing a facial expression can be seen above in the *Expression Detection Example* (figure 19).

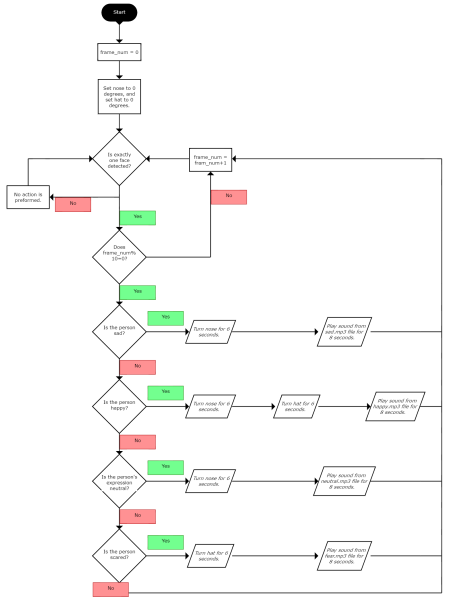


Figure Pumpkin 1 Flowchart

Here is the *Pumpkin 1 Flowchart* (figure 20) that better depicts how the components work together as a whole for pumpkin one. When the Pi is first powered on the frame number of the camera is set to zero and the horns attached to the servos are set to position 0 degrees. If the one face is detected it will then look for the facial expression sad, happy, and surprised. If the it detects a sad face, then the LED panels scroll the text “Happy Halloween” in the color green then the whole panel briefly turns colors blue, purple, then yellow and finally turn off. The LED ring eyes then proceed to turn on and perform a color wipe animation in the color white. The color wipe animation is one led turns the color at a time in a domino effect. If the camera detects a happy face, then both horns turn forward and backwards from 0 to 180 degrees for six seconds followed by a theatre chase animation of the LED ring of red then blue. The theatre chase animation performs a blinking of a few of the LEDs on the ring, and this animation happens to both rings at the same time. Lastly the panel scrolls through the text “Happy Halloween”. Lastly upon the detection of a surprised face the LED ring performs a color wipe of the colors green, blue and red, followed by the LED panel also displaying the text “Happy Halloween”. If no face or too many faces are detected then none of the actions will be performed.

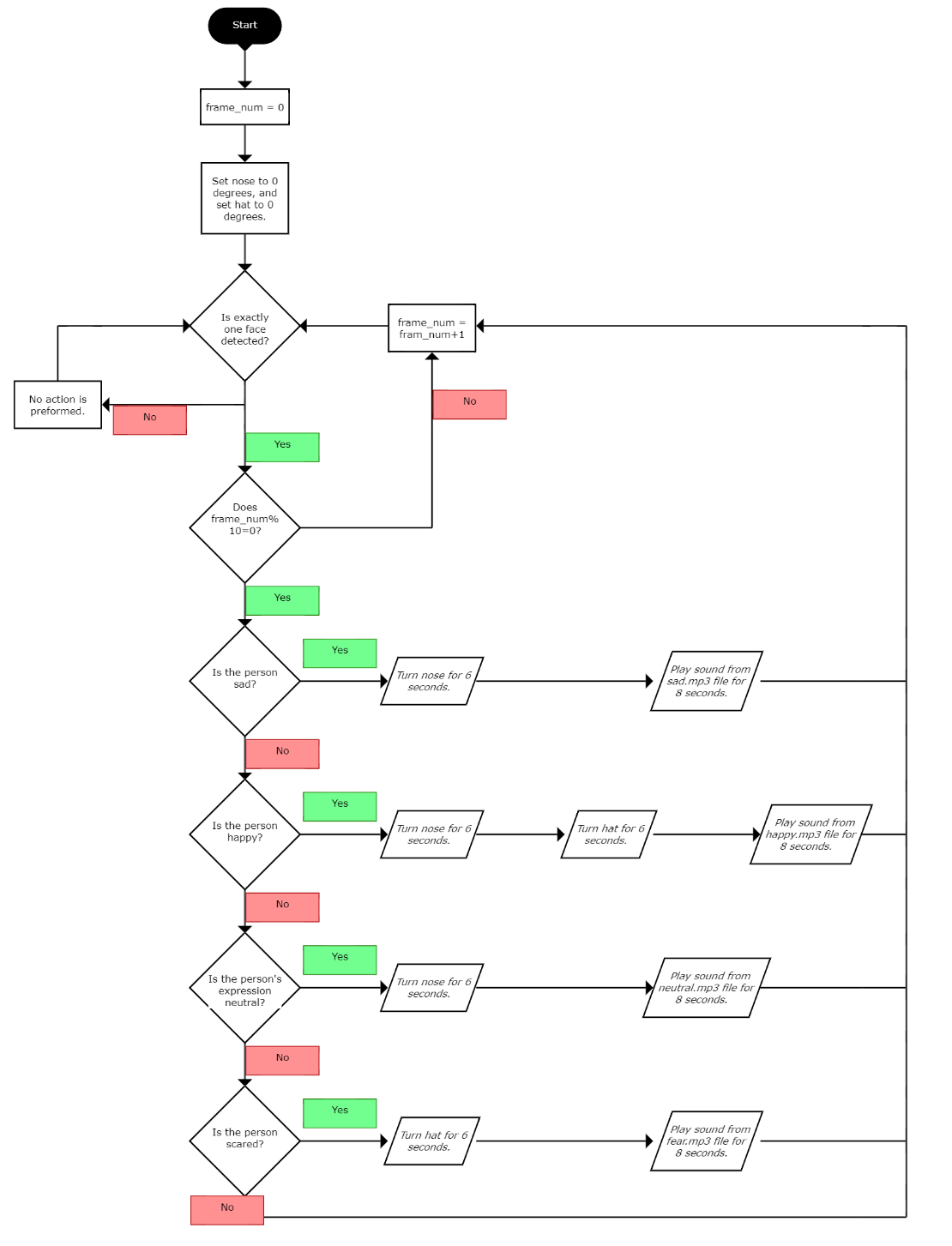


Figure Pumpkin 2 Flowchart

The *Pumpkin 2 Flowchart* (figure 21) for pumpkin 2 is similar to the flowchart of pumpkin 1 but the actions of the pumpkin are different and there is one different facial expression. Upon the Pi receiving power the nose and witch hat is set to 0 degrees and the frame number is also set to zero. If a face is detected it then proceeds to check if the expressions happy, sad, neutral, or fear are detected. If the camera detects a sad face, then the nose turns for about six seconds and plays a mp3 file sound we titled sad. The sound is just a trombone sound that is usually heard in many comedies or comedic relief moments. If the camera detects a happy face, then the nose turns for six seconds, followed by the hat turning for six seconds. Afterwards another mp3 file will play; it mimics a creepy witch’s laugh. If a neutral face is detected, that is a plain face expression, then the nose will turn for six seconds followed by another mp3 sound file that mimics a man’s laugh. Lastly if the camera detects a face that mimics a fearful look, the hat will turn for six seconds followed by a mp3 sound of a zombie throwing up. If too many or no faces are detected then nothing will happen.

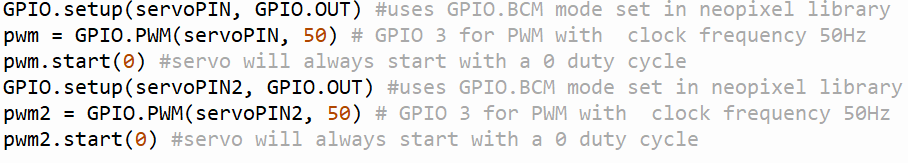


Figure Servo GPIO Setup

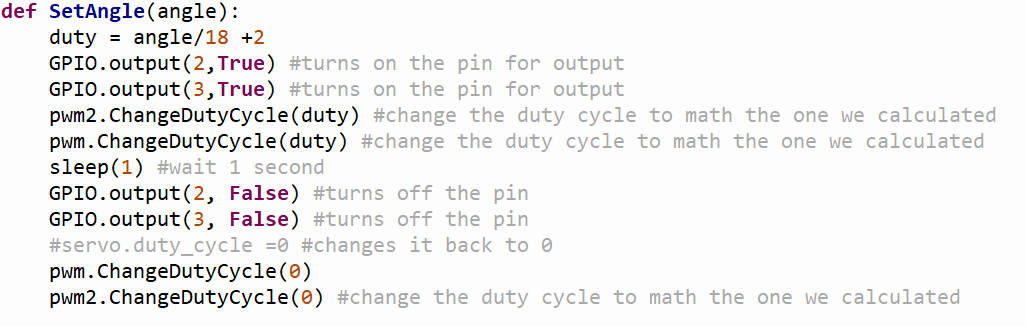


Figure Servo Duty Cycle Setup

As stated earlier each pumpkin have servos attached to different accessories. They both are implemented in python in a similar fashion. As seen in the *Servo GPIO Setup* (figure 22) they both need their allocated GPIO pin to be set up for output. The clock also needs to be initialized to use the pulse width modulation feature that is already given when you use the call “import board”. From there a function named “Set Angle”; *Servo Duty Cycle Setup* (figure 23) was declared and takes the parameters of the desired degree you would like to servo to turn. That degree was calculated by taking the desired degree and dividing it by eighteen and adding two. This shortcut was found online otherwise you would have to mathematically retrieve the desired angle by guess and check what percentage of the clock cycle to keep the output signal a one. For pumpkin one both horns move together so both servos are implemented in the same SetAngle function. For pumpkin two the hat and nose move separately depending on the facial expression case so both servos are implemented in their own functions named SetAngle and SetAngle2.

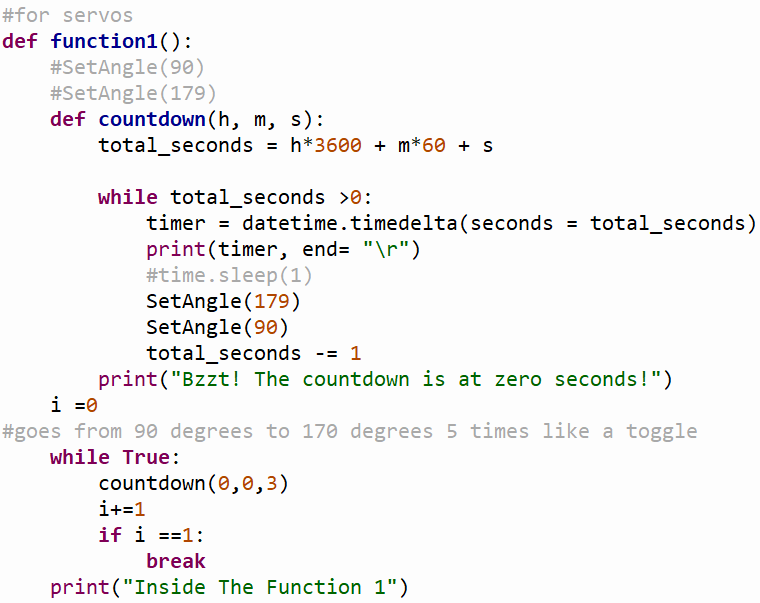


Figure Servo Implementation

Now to get the servos to run for a specific amount of time the SetAngle function was placed inside another function that is responsible for counting down to zero. That was finally placed inside another function just to easier be called for any desired face expression. The code snippet from above in servos (figure 24) shows the timer implementation for pumpkin one, but the code for pumpkin two is very similar.

Pumpkin one also used LEDs in the form of a panel and ring. They both used the “rpi\_ws2182” library to be able to address the LEDs, the neopixel library was also used to manipulate parameters such as brightness, color configuration, number of pixels used and the GPIO pin it is connected to.

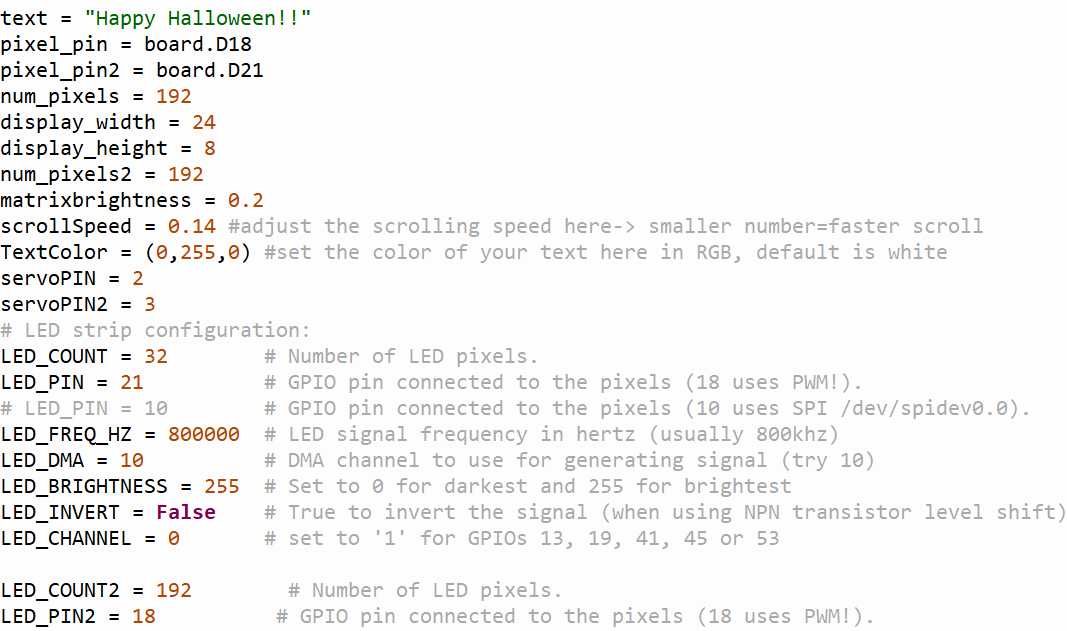


Figure LED Setup

As seen in the *LED Setup* (figure 25) there are many variables that need to be set before you can use the LEDs. The matrix daisy chained together has a total of 192 pixels; a width of 24, and a height of 8. To display text on the panels there also needs to be the declaration of the specific text, scroll speed, text color, as well as font which isn’t depicted in this snippet. The LED ring is set up as a strip since the shape of a circle is similar to a straight line just curved, and has a similar set up to the panel, besides the brightness. Having the panels set to its highest brightest would draw a lot of current and overheat/overwork the pi as well as be to bright for the individuals standing in front of the pumpkin.



Figure LED Matrix Panel Setup

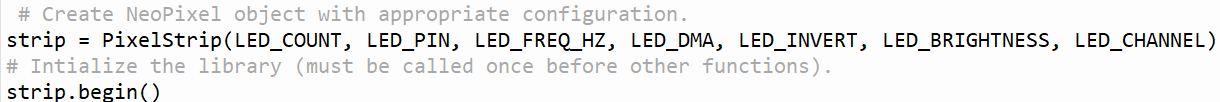


Figure LED Strip Setup

In the code snippets above *LED Matrix Panel Setup and LED Strip Setup* (figure 26 and 27), we see the final set up for the panel and the strip to be used with the Neopixel library. The library has built in function such as ‘fill’ which lets you turn one every LED in the given ring or panel with a desired color.

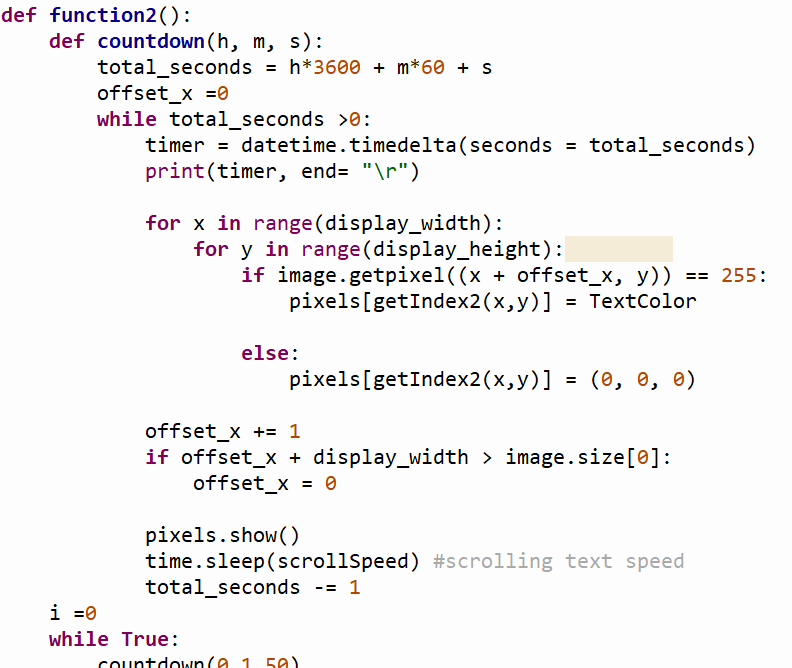


Figure LED Text Scrolling

Now the other animations require more work. Fortunately, many animations are on GitHub under neopixel library uses. The *LED Text Scrolling* snippet above is to scroll text across the panel (figure 28). Before this works you need to get the index of the matrix grid, the size of the font you selected, as well as change your text to an image, then the image scrolls across the panel with the code above. It was put inside of the countdown function otherwise the text would scroll on the panels indefinitely and for the purpose of the project it only needs to scroll once. The other animations such as ‘theaterChase’ and ‘colorWipe’ were also in the GitHub site as shown below in the (figures 29 and 30).

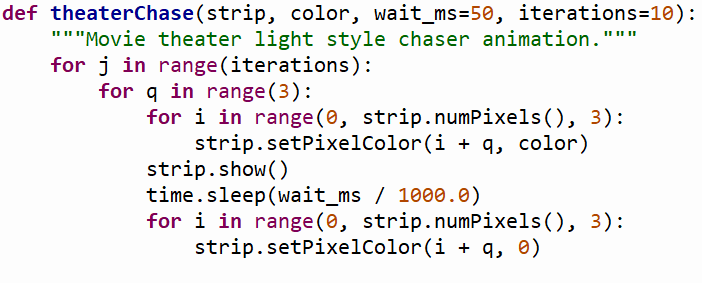


Figure LED Ring Animation 1

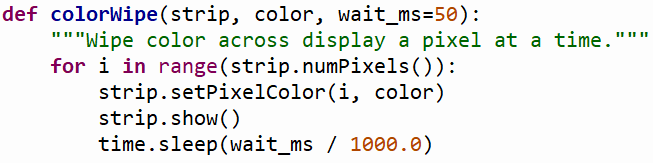


Figure LED Ring Animation 2

The first function ‘theaterChase’ evenly spreads three LEDs from the entire strip and will only light up those three when the function is called. The function ‘colorWipe’, indexes through the strip one at a time and makes the LED it is currently on the desired colored you set it to in a function call. Both of these functions were placed inside another function to be called numerous times for different facial expressions. Examples of these function calls can be seen below in the *colorWipe* and *theaterChase* examples (figures 31 and 32) where the led ring performs a green color wipe and a red theater chase. The color orientation is red, green, and blue with each color having different hues ranging from 0 to 255.



Figure colorWipe Example



Figure theaterChase Example

As stated earlier, sound was also included for pumpkin 2. It was first implemented by importing ‘Pygame’. Pygame is already in the python library so there is no need to download it, just import it to be used in the program.

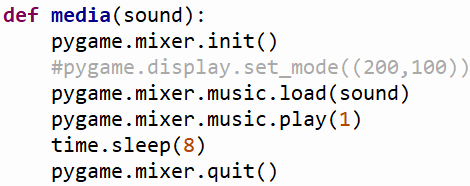


Figure Using Pygame

In order to use Pygame for sound you first need to initialize a pre-built function within Pygame called mixer, as well as load and play the sound. Once the sound finishes playing you add a sleep timer to make sure Pygame is caught up with the file because it sometimes has timing issues then you ‘quit’ to declare you are no longer using sound. In this function named media as seen in the *Using Pygame* (figure 28), all you do is put the name of the file you are using. It works best with mp3 files that are at least 8 seconds long. As this is an easy implementation it would have been easy to implement onto both pumpkins, but it was later discovered after trying to implement sound onto pumpkin one that the Pygame and RPI\_Ws2182b libraries have conflicts with a microphone. It is believed that both can be used with a microphone for different purposes and although the microphone was not used for this project, both of these libraries still wouldn’t run when implemented in the same program.

The last software piece to implement to both pumpkins was a run-on boot command, this means when the Pi gets power it immediately runs a desired code. There are many ways to implement this but it was found the AutoStart method worked the best for the code. AutoStart is a feature that is already on the Pi. To use it, all you need to do is navigate to the configuration file from the terminal then put the name of the file you wish to run and save. If you do not wish for the program to run once the Pi is initialized all you have to do is remove the file name from the file, the same way it was added.



Figure Final Pumpkin Products

This concludes all of the software and hardware implementations that came together to create the interactive pumpkins labeled as *Final Pumpkin Products* (figure 34), displays what they final outcome of the pumpkins came to be with pumpkin 1 on the left and pumpkin two on the right. These pumpkins were featured at the Lubbock Pumpkin Trail in Lubbock, Texas for four consecutive nights ranging from 3- 4 hours to celebrate Halloween. Both pumpkins functioned well and were able to perform the whole duration of the event. The children and adults had fun making faces at the pumpkins, there are some photos and videos from the event below in Appendix B 4-6. To better provide instructions poster boards with written instructions for each pumpkin were placed in front of the pumpkins, and after that didn’t need any verbal instruction for people to use. Most people liked the Adafruit eyes the most and the sound effects that would play after each facial expression for pumpkin 2. Placement of the pumpkins are also important so they could each get a wide range of varying heights. The camera for pumpkin 1 is placed in the forehead and the camera for pumpkin 2 is places in its mouth. We also noticed in the daylight if the pumpkins are facing the sun, it is harder for the cameras to read anyone’s face due to a glare the sun creates. The pumpkins were also required to be weather proof. There was a potential threat of rain on the first night of the pumpkin trail, but only a few drops of rain were present. To combat the weather the components of the pumpkins that were susceptible to damage were covered in plastic wrap. Those points were openings where the cameras were present and the section where the panels were used since it was on the outside of the pumpkin. There was also a large umbrella to capture most of the rain if it began to rain hard. During the creation of this project there were a few changes and interruptions that occurred, such as the eyes needing a different operating system then the cameras, and sound not being able to be implemented on pumpkin one. Another close call that was not mentioned above was the Pi’s configuring boot system.



Figure EEPROM Booting

The EEPROM chip seen above (figure 35), controls your Pi reading the microSD card and starting up. When the Pi gets a dip in its 5-volt voltage supply or any form of insufficient power, you have to factory reset this chip become it has no become corrupted. Fortunately, it is an easy fix that it is pretty common. On the Pi Imager where the operating system was downloaded there is also an option to factory reset this chip. Just upload the factory reset code onto another microSD card and place it on the Pi and wait a few minutes while it resets the chip. From here the Pi is back to normal and you can then proceed to place the microSD card containing your operating system and code.

Video demonstration of the pumpkins working at the Lubbock Pumpkin Trail can be found in Appendix B-4-6. This project was completed by its deadline October 26th and overall project progression can be found in Appendix B3. This project also has an overall estimated budget of $11,880, which can be found in the Appendix section B-1 and ended up costing $16,383. Labor hours did heavily increase from the steady rate of hours as the due date of the project’s completion drew near, as well as the initial estimate of hours to complete this project was underestimated which explains why the project was about $4,500 over budget, but having the project run smoothly makes up for the going over budget. For extending the project for December 3rd, the budget did increase to $20,836 and a new estimate of $23,067 was calculated and is the final budget that resides in the Appendix below.

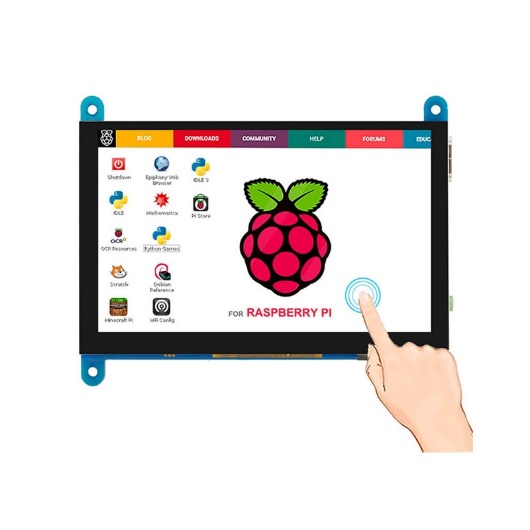


Figure LCD Screen

After this project’s completion there was time to make some changes to the interactive pumpkins. After the Halloween Pumpkin Trail event many visitors mentioned how they wished they could see what the camera saw, so we decided to add an *LCD Screen* to each pumpkin (figure 36). These screens connect to the raspberry pi via HDMI and can be powered from the pi’s micro-HDMI port via the help of a micro-HDMI to usb-type c converter, supplying 5V/3A. The screen shows an image viewer of what the camera is seeing and when the camera detects a person’s face and facial expression, it takes an image of the detected person and places a blue bounding box around their face with the expression labeled at the top of the box. The picture will remain displayed on the screen until the code finishes running the actions associated with that facial expression. If not, the screen still shows what the camera is seeing.

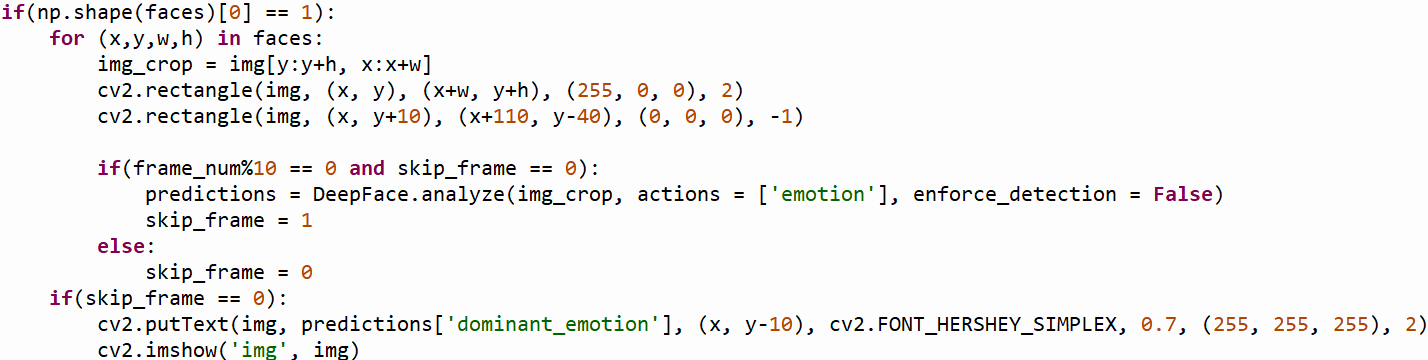


Figure Adjusted Camera Code

The camera code had to be altered slightly to perform these tasks, first with opening an image viewer and also with an image capture for when an expression is detected, and place a bounding box around the users face with the text of the caught emotion. These implementations are shown in the *Adjusted Camera Code* (figure 37) and did take some time to get right because initially it was happening so fast that when the camera would take the picture of the person but the image capture was delayed giving a misrepresentation of what the actions the pumpkin was doing. To solve this, we added a skip frame. As soon as the camera detects a facial expression it skips the frame after it, so when there is a delay, the delayed image; the image that holds our facial expression that started the pumpkins actions is displayed.

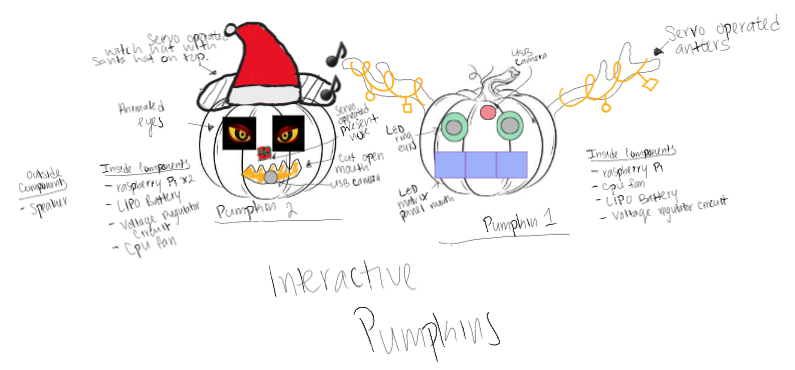


Figure Altered Interactive Pumpkins

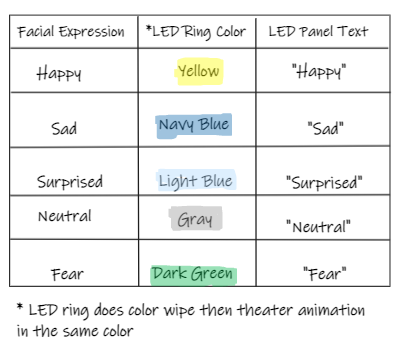


Figure Pumpkin 1 Mood Expressions/Colors

A sketch of the new pumpkins design without the LCD screen as it sits off to the side of each pumpkin can be seen in the *Altered Interactive Pumpkins Sketch* (figure 38). As for pumpkin one’s individual changes, it now has reindeer antlers wrapped in tinsel for the Christmas Holiday, and the panel and led ring actions now match color with expressions. In psychology they associate color with mood, for example angry is associated with red. Since many people unconsciously do this as well, it was thought that users could also easily see what emotion the camera detected when they see specific colors and text from the pumpkin. The pumpkin now detects the expressions, fear, surprised, neutral, angry, happy and a table listing the color associated with each expression can be seen above in *Pumpkin 1 Mood Expression/Colors* (figure 39) With each expression detected the led ring does a color wheel animation along with the theater chase animation showing the color associated with that emotion, then the text scrolls stating the expression that was detected.

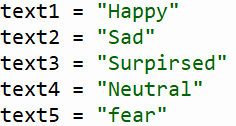
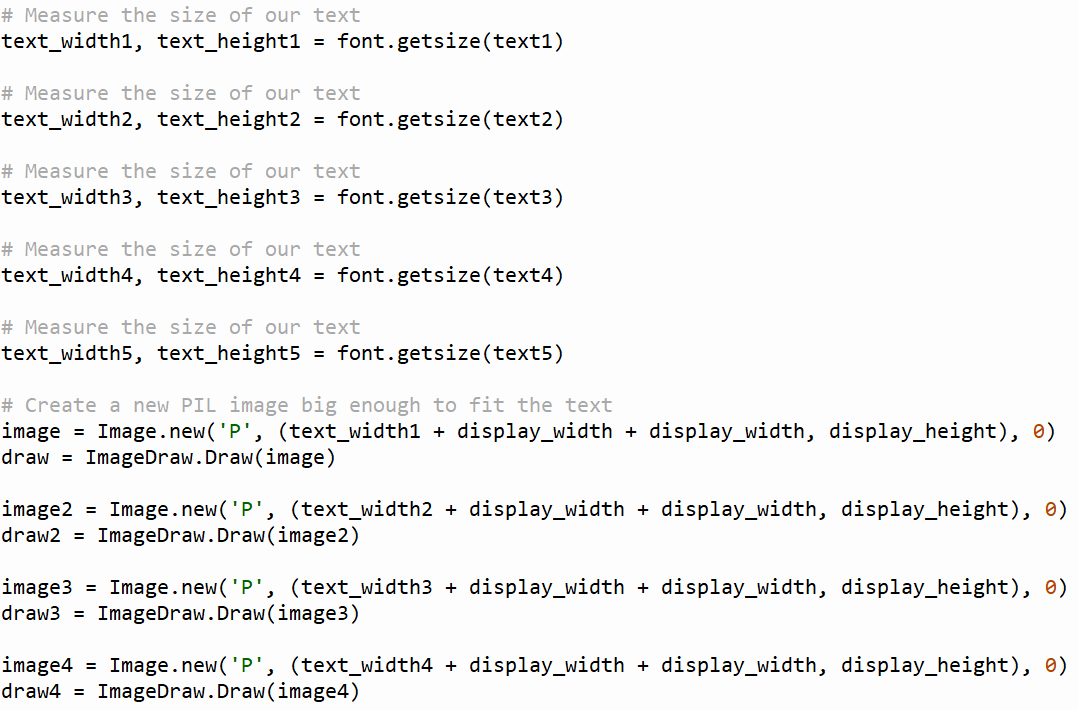
 

Figure Pumpkin 1 Text Alterations



Figure Pumpkin 1 LED Ring Alterations

The code implementations of this were adding the text for each expression to be stored as an image, and that image scrolled through the three panels that make the mouth of the pumpkin; *Pumpkin 1 Text Alterations* (figure 40). The LED ring change was setting the particular color to each expression and can be seen in *Pumpkin 1 LED Alterations* (figure 41).

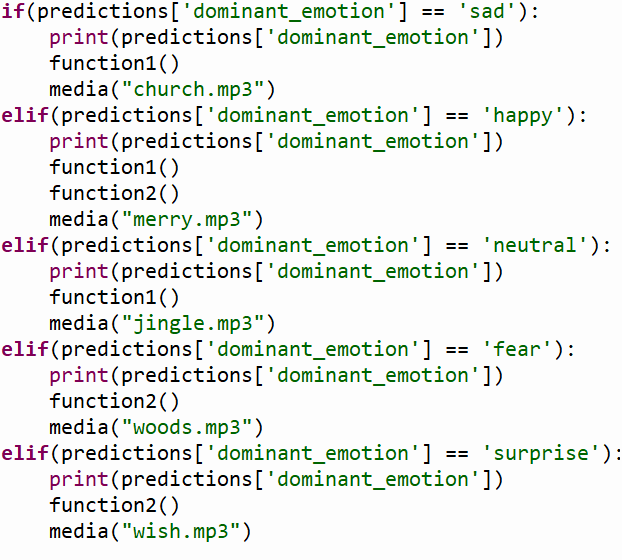


Figure Pumpkin 2 Sound Alterations

For pumpkin two, as seen in the *Altered Interactive Pumpkins*(figure 38), the witch’s nose was replaced with a present and a Santa hat was applied on top of the witch’s hat to show the pumpkin witch is now in a Christmas spirit. This pumpkin was altered to be more festive so each sound effect that corresponds to an expression is no longer scary and is more festive for the Holidays. Many kids and adults did like the scarier sound effects for Halloween, but it did scare the younger children. The code changes were minor and can be seen above in *Pumpkin 2 Sound Alterations* (figure 42), just had to place the name of the new downloaded sound files. The final product of both can be seen below in the *Final Altered Pumpkins* (figure 43).



Figure Final Altered Pumpkins

In conclusion, the new adjustments were implemented successfully. The addition of the screen was a useful addition in which it helped users better understand where the camera was as well as increased interaction with the pumpkins itself. If the screen was used for the Halloween event, I believe it would have helped the users and allowed the pumpkins to be more independent and reduce instructions.

Appendix A – References

Fromaget, P. (n.d.). *How to auto start a program on Raspberry Pi? (4 ways)*. Raspberry tips. Retrieved December 5, 2022, from https://raspberrytips.com/autostart-a-program-on-boot/

*ΜA7800 series positive-voltage regulators - SparkFun Electronics*. sparkfun.com. (2003, May). Retrieved November 1, 2022, from https://www.sparkfun.com/datasheets/Components/LM7805.pdf

Nelson, C. (2019). *Raspberry pi care and troubleshooting*. Adafruit Learning System. Retrieved December 5, 2022, from https://learn.adafruit.com/raspberry-pi-care-and-troubleshooting/rpi4-recovery

unknown. (2018, November). *Driving servo motor with PWM Signal*. Electrical Engineering Stack Exchange. Retrieved March 25, 2022, from

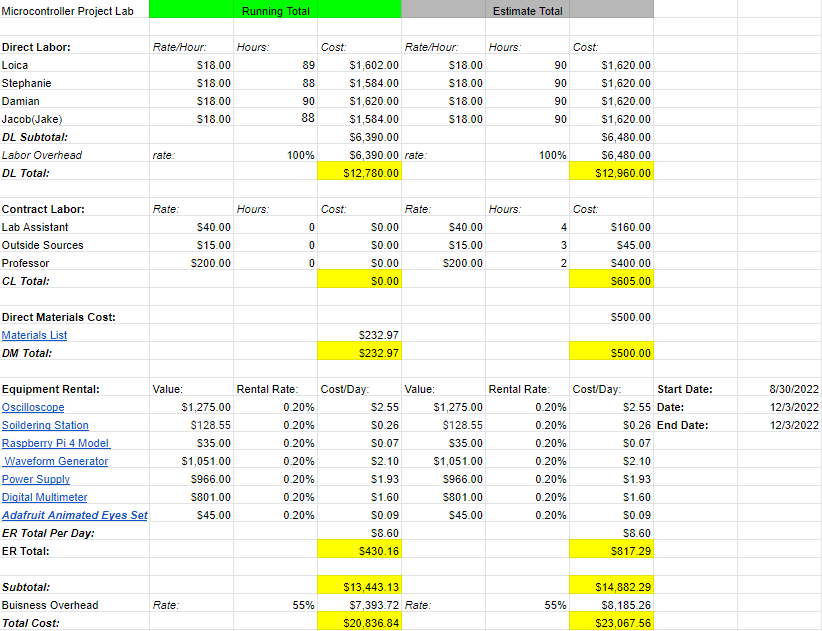
https://electronics.stackexchange.com/questions/346603/driving-servo-motor-with-pwmsignal

Worldsemi. (n.d.). WS2182B.

Zeller, H. (n.d.). *Hzeller/RPI-RGB-led-matrix: Controlling up to three chains of 64x64, 32x32, 16x32 or similar RGB LED displays using Raspberry Pi Gpio*. GitHub. Retrieved December 5, 2022, from https://github.com/hzeller/rpi-rgb-led-matrix

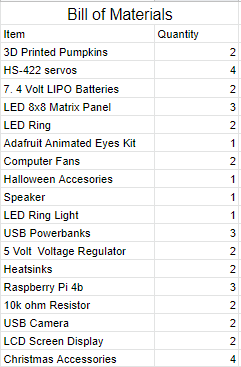
Appendix B – Budget/Gantt Chart

1. Update Budget



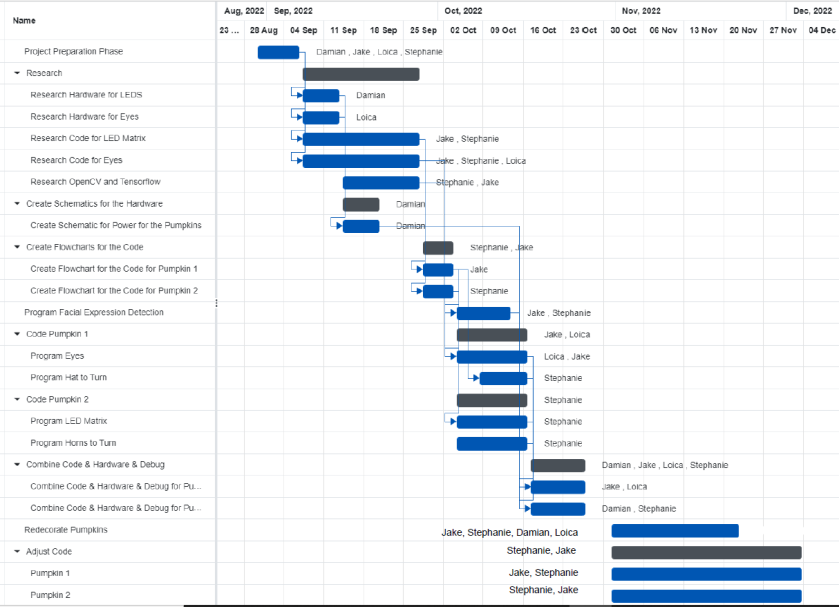
*Above is the estimated and final budget for the project. The budget is the total which consists of labored and contracted hours, material cost, and equipment rental from the day the project began until the day the project ends. With the completion of the project with total cost at $20,836.84. The previous budget for the initial project was over budget due to the miscalculation of hours estimated to complete this project. It resulted in $3,000 over budget. This budget for the new deadline of December 3rd had more a more realistic estimate of labor hours and resulted in the project being completed while being underbudget.*

1. Update Bill of Materials



*Here lists all the materials that was used in completing the Interactive Pumpkins.*

1. Update Gantt Chart



*This chart above shows the tasks that were completed in relation to a scheduled deadline of October 26th. And the project was extended to December 3rd. From analyzing the Gantt Chart, the project was completed on time. As you can see there was a lot of research involved early with this project and coding/debugging were the main components for this project. There was about a month dedicated to code the different elements, camera then debug issues present. The remaining 4- 5weeks were lighter in workload, so there was a dip in time working on adjustments which you can see by the extended time on the back end of the project. Overall everything was completed in time, but the camera did pose a major issue and could have greatly affected the timeline of the finished product.*

1. [Lubbock Pumpkin Trail Video 1](file:///C:\Users\steph\OneDrive\Documents\Project_Labs\Project%20Lab%202\trim.18E2A6AA-3680-4CE0-80AA-9BB20AE54461.mov)
2. [Lubbock Pumpkin Trail Video 2](file:///C:\Users\steph\OneDrive\Documents\Project_Labs\Project%20Lab%202\trim.ACABA35A-36EB-45FF-8612-435432F24887.mov)
3. [Making a Face](file:///C:\Users\steph\OneDrive\Documents\Project_Labs\Project%20Lab%202\865A81E7-36A9-4D25-83AD-FA99E1AF7967.jpg)

Ethics/Safety

It is important to always remain safe when proceeding with electrical equipment, devices, sharp objects, and other hazards when working and developing projects. When working on this project, it was done in a lab provided by Texas Tech University. Proper clothing such as closed toe shoes, pants, and short sleeves were required at all times. And eating and drinking was prohibited. In the event there was a minor incident where in testing the voltage regulator circuit, it started to smoke due to possibly receiving too much voltage. The group member responsible immediately shut off the power to the circuit and inspected it to make sure a flame wouldn’t occur. The incident was also discussed with the professor the following day and no student was harmed. Other safety measures that were taken during this project was spray painting outside with more ventilation rather than inside to not hinder the breathing and respiratory issues with ourselves and others. As well as all of the night tests of the camera as well as Lubbock Pumpkin Trail events more than one group member was always present to ensure the safety of the group and proper setup/ implementation of the interactive pumpkins.

Acknowledgments

In completion of this project, I would like to thank Dr. Haustein for giving me the opportunity of doing a creative project that cheered up the Lubbock community. I would like to also thank him for guidance to make sure my group was properly prepared for the event. I secondly would like to thank my groupmates Damien Archer, Jacob Rothschild, and Loica Carmelle Tatmba Ngatchoua for putting in a lot of hours in the short six-week time frame to complete this project and as some materials were purchased out of pocket to meet the time demands given. And for putting forth effort for the continuation of this project after its initial completion. I would also like to thank Texas Tech University for providing Lab materials/ supplies as well as let us represent the Computer and Electrical Department of the College of Engineering at the Annual Lubbock Pumpkin Trail