Holiday Reverse Programmable Light Strings

PROJECT PLAN

SDDEC18-10
Adviser: Tom Daniels
Client: Thomas Daniels
Aaron Hudson — Web Server/IPC Communication
Robert Tyynismaa — Android Developer
Rajiv Bhoopala — Web Application/Server
Michael Scholl — Android Developer
Mir Ahbab — Electrical Engineer/Microcontroller
Justin Falat — Android Developer

Email: sddec18-10@iastate.edu
Website: http://sddec18-10.sd.ece.iastate.edu/

Revised: 4-18-18-V3

Table of Contents

List of Figures

List of Tables

List of Symbols

List of Definitions

PWM - Pulse width modulation

RGB - Red Green Blue

LED - Light emitting diode

1 Introductory Material

1.1 Acknowledgement

We would like to thank Dr. Tom Daniels for the assistance in researching and developing the plan for the project thus far.

1.2 Problem Statement

Christmas is a time for celebration and with celebration comes the arrival of holiday displays. Many people decorate their homes and other objects like trees with sets of lights. However many products on the market are limited to individual design and creativity by not being customizable. For instance, in order to decorate an arboreal display in a pattern you would have first visualize that display and then lay the lights accordingly. The complexity changes when the person wants the patterns displayed to change. Thus, our team has decided to tackle this problem of being able to create complicated displays in a simple manner.

In order to provide more customization, our team wants to utilize technology in the form of smart phones and web apps. Our idea is for users to set up their lights on a tree and then upload patterns to that string of lights. Smartphone cameras will be used to record the position of LED's within the display. The data will be sent to our web app/android app where a 3d model of the display with LED's will be created. Users are then able to decorate their display through the web app/android app. Thus, the ability for customization is far greater than your average string of lights.

1.3 Operating Environment

The operating environment for our final product is focused around use on a standard pagan holiday arboreal display. This display can either be inside or outside. The RGB lights are waterproof and the controller box will also be weather-resistant.

1.4 Intended Users and Intended Uses

The intended users for our holiday lights are people who are interested in programmable LED lights, but are also interested in arts and gadgets. Our end goal is to create a product that can be used by anyone, even those who are not familiar with technology.

Although our project is mainly focused on creating programmable LED for standard pagan holiday arboreal displays, our final product can be used with anything in mind. Some other uses include displaying the programmable LED lights on stairs, desks, bed frames, hangers, etc.

1.5 Assumptions and Limitations

Assumptions:

- User will only have one controller per house
- The user has a working WiFi connection
- User owns a smartphone with video functionality

Limitations:

- Keep cost at a minimum for components
- Have to use specific string of lights that are either individually addressable or use PWM

1.6 Expected End Product and Other Deliverables

Mobile Application - An Android Application for the user to perform set up. The user takes video from different points around the display, and the app will process the video to determine the location of each LED on the string. This will create a 3d representation of the display for the user, and they will be able to select a pattern to displayed. This information will be sent to a web server on the Raspberry Pi.

LED lights w/ Raspberry Pi Controller -The Raspberry Pi will have two modes: set-up and display. For set-up, the Raspberry Pi will send PWM signals to the string of lights so as to only have a certain amount lit up for a set of frames in the video that the Android app is capturing. After the location of each LED has been found and the user has selected a pattern, the Raspberry Pi will send the PWM signal for that pattern.

2 Proposed Approach and Statement of Work

2.1 Objective of the Task

The objective of this project is to engineer a product that will provide our intended audience the ability to program their own holiday arboreal light display. This will include components such as led light strands, a raspberry pi controller and custom driver, as well as a mobile android application. The mobile application will be used to calibrate the light display through the use of computer vision by use of the phone camera. Then the user can create a custom pattern to be displayed, as well as send their pattern through a web server to the hardware components which in turn, will display the users desired pattern.

2.2 Functional Requirements

- A PWM controller will load a byte-stream into the lights (shift-registers) that will illuminate each individual RGB light.
- The Raspberry Pi will host the web server that will receive the buffer of bytes from the mobile application, as well as drive the PWM controller

- The mobile application will use smartphones camera and 24-bit color sensor to detect the location of the individual lights on the display.
- The mobile app will handle all of the creation of the byte-stream for illuminating the lights.
- The mobile app will send the byte stream via wifi to the web server.
- The mobile app will construct a graphical image of the tree in a custom coordinate plane to map the placement of each light.

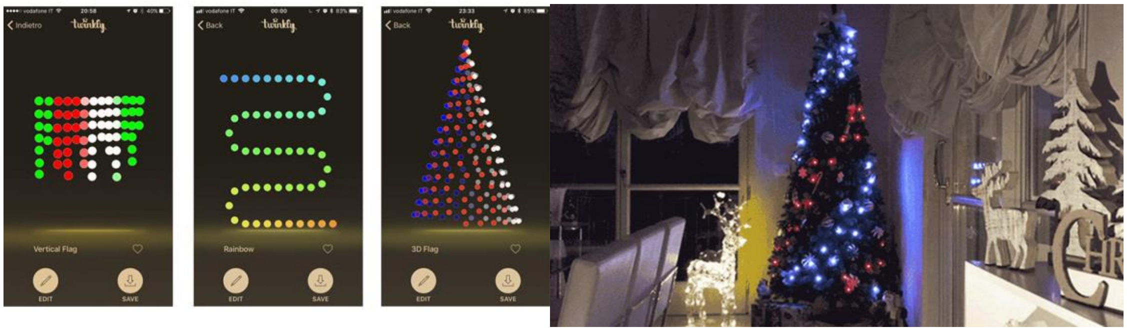
2.3 Constraints Considerations

The byte stream should refresh fast enough such that it should not be detectable by the human eye.

Code should be commented and easily readable: e.g., variable names should not be too complex and should correlate with the functions they are being used for. Comments above a function should summarize what it is and how it is to be used while comments within should go into more specifics.

2.4 Previous Work And Literature

There is one product out on the market that is similar to our projects goal and it is called Twinkly. After some research we assume the LED strand and controller communicate with an android/iphone app through wifi. The application takes into account that the LED's were placed in specific locations on the tree. Calibration is decided by how the strand of lights is laid onto the tree. So for the calibration of the Twinkly tree you have the LED's light up in a pattern and place them correctly onto the tree. Then the application assumes the locations of the LED's based on that process to then display patterns. But the Twinkly product is actually very similar to our goal in terms of design and feel of a product. We would like for the user to use the application and change the design of the tree then, possibly draw/upload new designs within the application. The difference in our project from the Twinkly product are the ability to analyze the lights locations so that the lights theoretically could be placed not just in a tree shape.



Figures 1 & 2

All photos were taken from the twinkly website. https://www.twinkly.com/

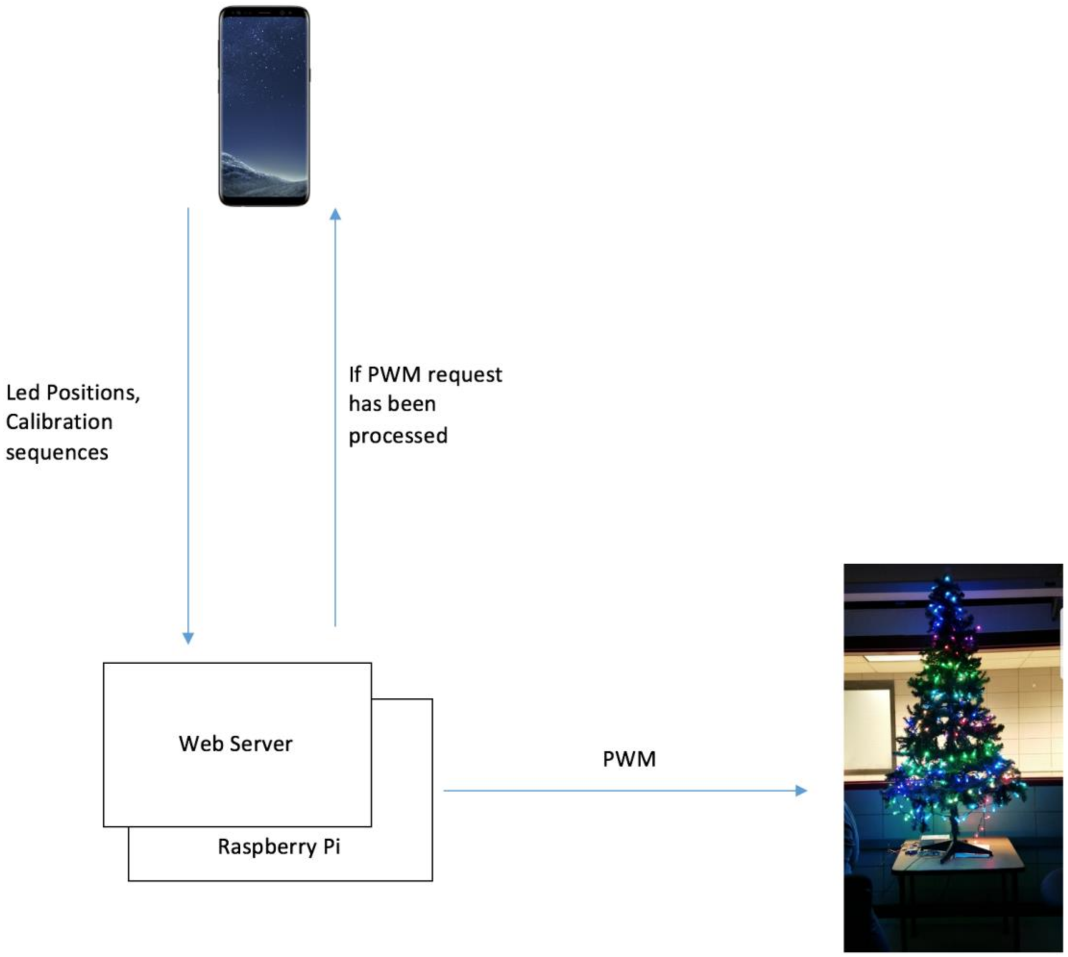
After further research it seems there really isn't another product similar to our goals. There are tons of different LED strips and customizable LED strips that you can purchase, but to reach our goal it would require more work. You have the ability to purchase a LED tree that is pre-built, then use a remote to change the tree to have certain patterns. This can be found by googling pre-lit christmas tree with remote control, or something along those lines. This is kinda relative to our project but not really, because they are using a pre-built version of a tree it doesn't not require another device to interact with besides the remote. Our project involves actually placing the LED's on the tree so that we can place them anywhere we want on the tree.

2.5 Proposed Design

The lights will be controlled via a Raspberry Pi through its PWM output. The user will send a signal with what they intend to display to the Raspberry Pi through either a mobile application or a web application.

The main purpose of the mobile application will be calibration: finding where each of the LEDs are in 3D space. The user sets up the phone such that it will not move while it takes a video of the tree. Once the user presses start, the phone will take a video of the tree, keeping fixed the values for the camera (e.g. the focus of the camera), while sending signals to the Pi about what LED to light up. There are 3 types of pictures: an unlit tree, the fully lit tree, and a picture for each individual it LED. Either the web or mobile app will process the images; it will subtract an unlit and fully lit image to determine where the locations of the LEDs are in x,y coordinates. Then it will go through the individual LED pictures and see which x,y coordinates matches that LED. This process will repeat two more times (e.g. each picture captures 120°) and the cylindrical coordinates of each LED can be calculated.

The user will then be able to specify what pattern they would like to display using the web app, either a pre-defined pattern or their own. The web app will calculate what color each LED needs to be in the case that the user uploads their own pattern. The user will be able to specify the type of rotation and the speed of the rotation using the web app.



2.6 Assessments of proposed solution

The initial setup process where the mobile app is iterating through each LED may take a considerable chunk of time, upwards of a couple of minutes, which introduces more room for error; the lighting in the room could change, the location of the mobile device could change, internet connection could be lost. If the lighting changes, or if the mobile device location changes that will influence the LED detection algorithm; if the internet connection is lost, the mobile app will no longer be able to send info to the microcontroller, which would cause the calibration system to hang. Potentially causing fatal errors at the worst and having to redo the process at the least.

As it is, the mobile application solution will be designed for Android, which is a weakness and trade-off that we have to make. We decided to allow a web application to communicate with the controller and control the lights as that eliminates the previous problem, and gives us more leeway in how we implement the application.

2.7 Safety Considerations

- There is a risk of electric shock via the outlet
- Long strands of lights may present a risk for strangulation
- Electrical components may be a fire hazard through incorrect use
- Long cords might present a tripping hazard if set up improperly

2.8 Task Approach

Use knowledge of Android Studio to design an Android application to facilitate the setup of the lights and to send control signals to the Raspberry Pi. Code will need to be written for the Raspberry Pi to control the light string using PWM, after receiving a signal from the Android Application via WiFi.

2.9 Possible Risks And Risk Management

-Feature creep

-Unrealistic goals

-Lack of communication and inability to meet regularly

2.10 Project Proposed Milestones and Evaluation Criteria

1. Sending a signal from Raspberry Pi to Lights
2. Scanning tree LEDs with mobile application
3. Changing LED color using web application

2.11 Project Tracking Procedures

Our team will utilize the following applications to track project progress over the next two semesters:

-Github: We will use Github to save and share software that is written over the course of the project. This will give use a form of version control in order to revert to an older version if necessary.

-Trello: We will use Trello boards to efficiently plan both short and long term goals as well as track the progress of teams and individuals over the course of the project.

-Slack: We will use Slack as our primary form of communication with team members for the course of the project.

2.12 Expected Results and Validation

The desired final outcome of our project is to be able to control programmable RGB LEDs through our application. This includes changing the color of the lights, determining which lights light up, as well as loading a preset pattern onto the lights to verify that they are in the correct location.

Requirement and Outcome	Validation Test(s)
A PWM controller will send a byte-stream through the lights via shift-registers that will illuminate each individual RGB light.	Load a pre-existing library to send the byte-stream to the lights. Since we know the pattern that should be displayed, we should be able to see where the errors occur.
The Raspberry Pi will host the web server that will receive the buffer of bytes from the mobile application, as well as drive the PWM controller.	We can test that the server is working by accessing it via a web browser to make sure it is connected. We can SSH into the Pi and run the PHP scripts to ensure that those are working properly to receive and store the information.
The mobile application will use smartphones camera and 24-bit color sensor to detect the location of the individual lights on the display.	We have an algorithm that performs image subtraction of an unlit tree from a lit tree, which then displays the location of each individual LED.
The mobile app will handle all of the creation of the byte-stream for illuminating the lights.	We can test the creation of the byte stream by configuring it to be the same as an LED pattern that can be pre-loaded from a library. If the pattern displayed on the tree via the byte stream sent is the same as the pre-loaded, we know that the byte stream sent is correct.
The mobile app will send the byte stream via wifi to the web server.	We can test sending requests from the mobile app to the server via GET/POST requests and track the output in a callback.
The mobile app will construct a graphical image of the tree in a custom coordinate plane to map the placement of each light.	We can start with a small or simple pattern and transfer the 3D image into the rotating coordinate plane that we plan to have displayed in the app. To make sure it is accurate, we can take the position of those and compare it to the position from the image subtraction step.

2.13 Process Details

	Present Project Version	Future Project Versions
Android Application	Uses Camera2 API to passthrough feed from camera to display inside application. This API allows us to obtain camera settings and lock them into place to ensure minimal light disturbance.	Successfully utilizes camera to calibrate LEDs through image subtraction of unlit tree from lit tree. Connects to web server/Pi to send bitstream to generate PWM wave to drive lights, receives confirmation that PWM wave was successfully generated.
Raspberry Pi	Web server successfully setup on Pi, static IP obtained and able to accept post requests to php scripts.	Successfully receives PWM bitstream for light pattern, generates PWM wave to drive lights and provides confirmation to application.
Level Shifter Circuit	We have acquired the level shifter chip and designed a circuit to step 3.3V to 5V.	Successfully steps 3.3V output from Pi to 5V input for LEDs.
LEDs	Successfully displays pre-set patterns from Arduino FastLED library.	Successfully displays proper patterns on tree.

2.14 Standards

[1] IEEE standard for LED lighting flicker and potential health concerns

This IEEE standard gives the recommended practices for modulating current in High Brightness LEDs for mitigating health risks to viewers. This applies to the project because of the way our LEDs will need to calibrate. We assume the user will use the lights safely and not adjust them to a higher brightness than advised. But during the calibration period, our lights and camera application we will be flashing and adjusting the brightness of our LEDs. This is something of concern that we are taking into account and making sure we are operating in a safe environment.

[2] IEEE Standard for Camera Phone Image Quality

This IEEE standard quantifies the performance of mobile devices equipped with camera and photographic capabilities. It discusses the types of sensors and lenses on such devices to give them a set metric standard. These metric standards are aimed at targeting the ability of the camera so that you can assume you are getting a certain quality image. “Metrics include spatial frequency response, color uniformity, chroma level, lateral chromatic displacement, local geometric distortion, texture blur, and visual noise.” [2] These metric play a huge role as we will be looking at the images taken from the smart devices and analyzing them to find set data. Without a basic quality of image to analyze, creating a application to get information from that image would be challenging. This standard allows the application to get a consist image that we can assume has certain values. It also allows us to interact with the camera to change certain settings allowing for an increase in picture quality depending on the setting.

2.15 Test Plan

The first major step in testing our project is to make sure that the lights light up appropriately with pre-determined patterns. We have used an Arduino Esplora and the FastLED library to ensure that each light is of the appropriate brightness and that the power supply we are using supplies enough power so that there is no drop off of current to lights at the end of the strand.

In addition to testing the that the LEDs light properly, it is important that we ensure the safety of the consumer. We will do this by ensuring that the lights cannot flash at certain frequencies that have been found to be problematic in the potential cause of seizures. We will utilize IEEE standard [1], referenced above in section 2.13, to ensure that we avoid such frequencies and protect our consumers from such harm.

After acquiring the Raspberry Pi 3 we needed to create a voltage level shifter circuit in order to step the 3.3V output from the Pi to a 5V input signal for the LEDs. In order to test this circuit, we are using a white box test method. This is because we are aware of all the components being used in this circuit and want to ensure that the level shifter circuit is working and behaving correctly before connecting it to our hardware (Pi and LEDs).

For testing the detection of the lights via the Android app, we have a program that allows us to subtract an image of the unlit tree from the lit tree, and then searches through that grayscale image for pixels with a brightness above a certain threshold, usually 300. Once it finds them, it highlights those pixels with a red point.

In regards to further testing of our Android app we are implementing regression testing to ensure that the app remains functional as we add different features. This is an appropriate testing method for us because frequent testing at each update will help to limit potential issues that may arise as well as ensure that previously working components of the app will remain functional.

3 Project Timeline, Estimated Resources, and Challenges

3.1 Project Timeline

Link to Gantt Chart

The current iteration of the Gantt chart displays what has been accomplished for the semester, and what needs to be done for the next.

The hardware for basic implementation and testing has been acquired, with a good portion being set up (e.g. the light strings, the controller).

Basics of an Android application have been developed, as well as the development of an algorithm for the app for the detection of the lights in a 3D space.

A web server has been set up on the Raspberry Pi and has been designated a static IP address to better facilitate communication between it and the android application, as the app will not have to search for it every time for future calibrations.

For next semester: the light detection algorithm will need to be implemented in tandem with Inter Process Communication (IPC) between the Android application and the web server on the Pi. The web server will need to be able to communicate with a web application to receive user requests for patterns, as well as changes to current patterns.

3.2 Feasibility Assessment

The project will consist of an Android application and a Raspberry Pi controller connected to a string of PWM LEDs. A challenge we may have is ensuring that there is no current drop along the string of LEDs, such that each light is powered the same. Another issue may be identifying the lights through the android application; the camera must have a high enough resolution to be able to differentiate between multiple lights. The other challenge is putting the found lights within a coordinate system (either 3 dimensional or cylindrical coordinates).

3.3 Personnel Effort Requirements

As things currently stand, all group members will be required to attend all weekly meetings with the client and advisor. In addition, members of the app development team can begin work on the proposed mobile android application.

3.4 Other Resource Requirements

- Raspberry Pi
- RGB Lights
- Op Amp

- 12V power source
- Smartphone
- PWM controller
- Android Studio/XCode
- Linux web server

3.5 Financial Requirements

We will require funding from our client and the department to purchase the required resources referenced above in section 4.3 (Sans smartphone, web server and software development tools such as Android Studio/XCode).

4 Closure Materials

4.1 Conclusion

Our goal is to provide a customizable LED holiday display for a user to interact with. This will be accomplished using an Android application to set-up the lights and allow the user to select patterns, as well as a Raspberry Pi receiving information from the app and controlling the string of lights with PWM. The Raspberry Pi will be running a web server to receive this information.

4.2 References

[1]A. Wilkins, J. Veitch and B. Lehman, "LED lighting flicker and potential health concerns: IEEE standard PAR1789 update," *2010 IEEE Energy Conversion Congress and Exposition*, Atlanta, GA, 2010, pp. 171-178.

[2]IEEE Standard for Camera Phone Image Quality," in *IEEE Std 1858-2016 (Incorporating IEEE Std 1858-2016/Cor 1-2017)* , vol., no., pp.1-146, May 5 2017

[3]LEDWORKS S.R.L. (2016). Twinkly Smart Decoration. Via Arcivescovo Calabiana, Italy. Company.

4.3 Appendices

None as of yet.