SMART BACKPACK / BOOKSHELF

Team 15 - Anusha Kandula (kandula2), Tae Kyung Lee (tlee82), Gautam Putcha (gputcha2)

ECE 445 Final Report – Spring 2020

TA - Jonathan Hoff

Table of Contents

Abstract

A problem that students have been facing for a long time has been overstuffed backpacks that are too heavy. Carrying heavy backpacks can not only cause temporary issues, but also chronic back and neck problems later on in a student's life [1]. Students face this problem on a daily basis and it can lead to severe problems. The original solution approaches the problem by simply warning the user of their backpack being overweight instead of actually solving the issue at hand. This is done by weighing the bag before use and getting feedback on the weight limits on an IOS application that is bluetooth enabled. Our product solves the customer's problem with a simple app and bookshelf system that can be used to tell the student what books he/she needs according to their schedule everyday. The key difference between the projects is the fact that the device in the original project is screwed into the bag, defeating the original purpose of reducing weight. Our design proposes a solution to leave the backpack intact but simply minimize the content based on a student's schedule. We use schedule data provided by the user to indicate directly to the user which books to pack on a daily basis

1. Second Project Motivation

1.1 Updated Problem Statement

We are building a smart bookshelf that serves all types of students. Our customer has a problem, and it is carrying a heavy and overstuffed bag on a daily basis because of the weight of several heavy books and in many cases, even the books not required for a particular day. Our product solves the customer's problem with a simple app and bookshelf system that can be used to tell the student what books he/she needs according to their schedule everyday.

1.2 Updated Solution

Our design proposes a solution to have a lighter backpack based on a student's schedule. Repacking a backpack on a daily basis is a task that most students avoid and prefer to suffer the effects of a heavy backpack. Our solution uses the schedule data provided by the user to indicate directly to the user which books to pack on a daily basis to avoid packing any unnecessary books. This is done in the form of a shelf design. We propose a system that uses

schedule data to light up LEDs on a shelf either red or green based on whether the book is required for the following day. The user picks up the books with a green LED and leaves the red LED books on the shelf. This not only solves the problem of a very heavy backpack but also allows the user to accurately pack his/her bag.

1.3 Updated High-Level Requirements

1) The user needs to enter their schedule and a picture of their textbook on the application and only then the “pack” button is functional and lets the vision system and the LEDs work to match the schedule.
2) The device must light up all 16 LEDs either red or green within 30 seconds after the pack button has been pressed and the LEDs will light up with an accuracy of at least 95% according to the subject data it gets.
3) The application will play an alarm-like sound within 5 seconds if the user takes out the “wrong” book (a book which has a red LED under it).

1.4 Updated Visual Aid

Figure 1: Visual Aid of how the system will look

1.5 Updated Block Diagram

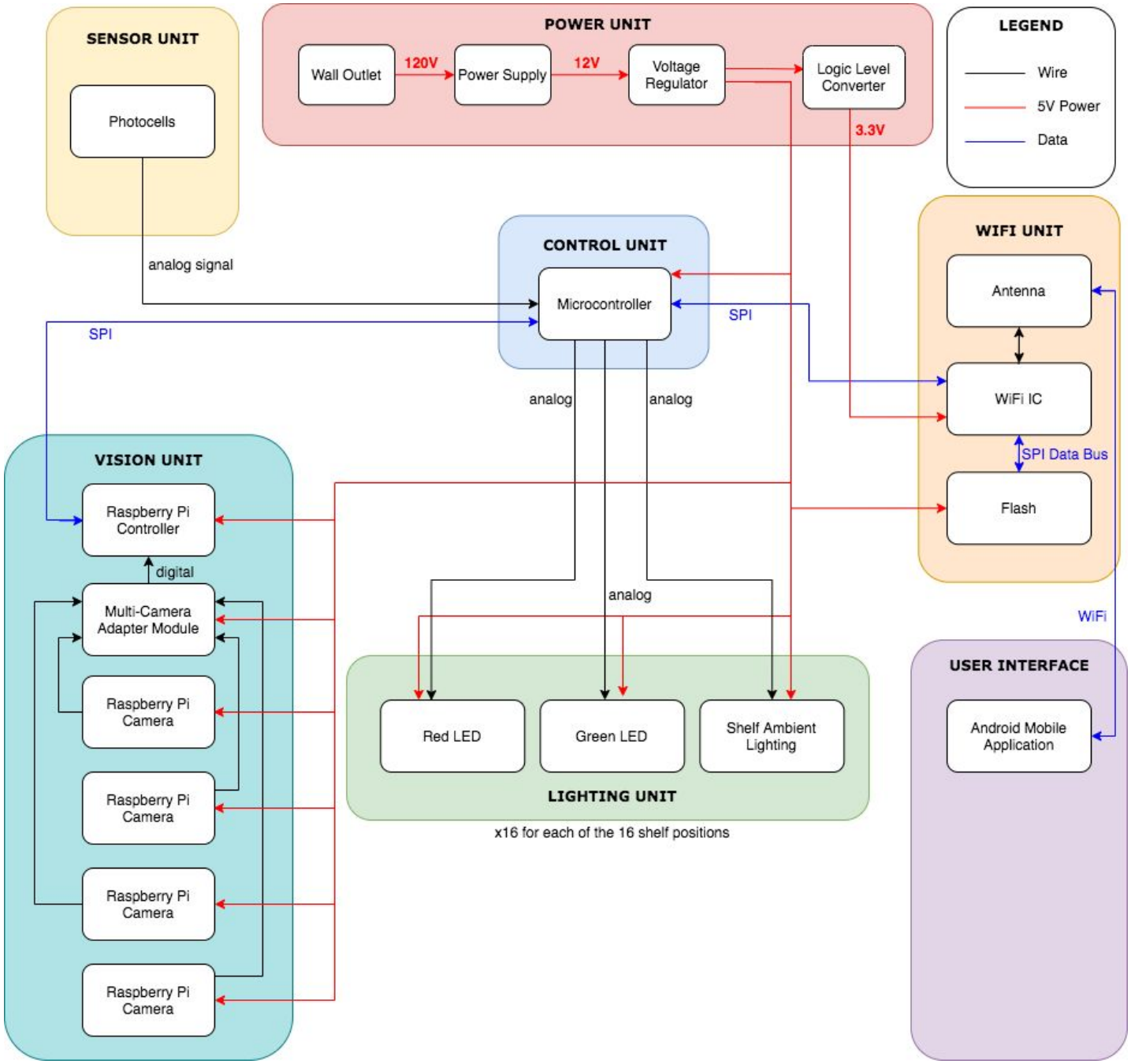


Figure 2: Block Diagram for our proposed design of the smart bookshelf

2. Second Project Implementation

One of the most critical features of our project is to build a vision system that can correctly compare the image of the textbook taken from raspberry pi and match it among the images provided by the user. From the high-level requirement, we have to know which book is stored at the exact position of the bookshelf and in the data set. To do this, we are going to use computer vision powers of the OpenCV library in python. First we convert our original image from the BGR color space to grayscale and then compare the Luma (brightness) of the picture to differentiate between images. There are 2 mathematical models that we are going to use to distinguish images: Mean Squared Error and Structural Similarity Index Measure.

Another feature we deemed necessary was the accurate working of the scheduling system. Without the scheduling system working properly, it is very difficult for the remainder of the design to work as expected since it is dependent on lighting up the positional LEDs based on the necessary books for the next day of classes. For these reasons, we have carried out our implementation on the following project features:

1. Computer Vision and it’s analysis
2. Scheduling system allowing the data from the vision unit to be used in a meaningful way

2.1 Implementation Details and Analysis

1. Implementation & Analysis of Computer Vision

In this section we will discuss the implementation of the computer vision component of our system. This will be essential to our project and connects directly to our high-level requirements. The ability for the correct LEDs to light up green or red accurately is dependent on the computer vision performing as expected. We have compared and contrasted different models of image comparison. This is done on the basis of the angle of rotation. We analyzed the accuracy of the image comparison between the two models for rotated images before coming to a conclusion on which model to use.

Of the two equations used to process images, we will first look at the Mean Squared Error equation.

$$MSE = \frac{1}{m\,n}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[I(i,j) - K(i,j)]^2$$

Equation 1: Mean Squared Error

Using the values of grayscale in every pixel of the pictures, we are going to add all differences in brightness of pixels throughout the image. In Mean Squared Error, 0 means that the following images are exactly the same and 1 being totally different. Using the numbers, we are going to associate the image of the lowest MSE score to be the most similar one. [2]

Structural Similarity Index Measure is calculated as follows.

$$\mathrm{SSIM}(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}$$

Equation 2: Structural Similarity Index

Structural Similarity Index Measure does not compare the difference in error but it attempts to model the perceived change in the structural information of the image. Instead of comparing the grayscale values of the pixels, the SSIM takes into account of the (x,y) location of the N x N window in each image, the mean of the pixel intensities in the x and y direction, the variance of intensities in the x and y direction and covariance [2].

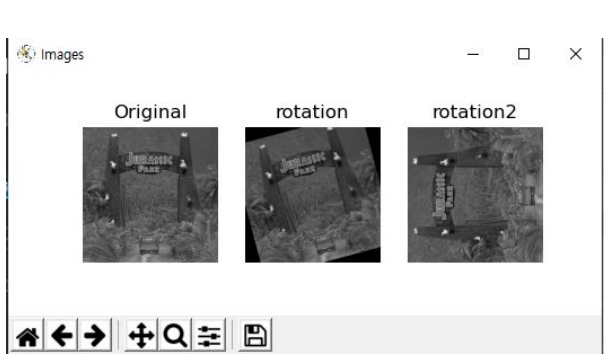
Using the structural base of the code from the computer vision tutorial [2], we are going to implement the effect of rotation when using the two comparison methods. This is done because the angle of book at which the raspberry pi camera will take will not always be the same as the one that the user has taken. Thus it is critically important to understand the limits of the comparison methods and limit or use one that is least affected by the angle of rotation of pictures. The code that is used to compare images are as follows.

```python
# import the necessary packages
from skimage.measure import compare_ssim as ssim
import matplotlib.pyplot as plt
import numpy as np
import cv2

def mse(imageA, imageB):
    # the 'Mean Squared Error' between the two images is the
    # sum of the squared difference between the two images;
    # NOTE: the two images must have the same dimension
    err = np.sum((imageA.astype("float") - imageB.astype("float")) ** 2)
    err /= float(imageA.shape[0] * imageA.shape[1])

    # return the MSE, the lower the error, the more "similar"
    # the two images are
    return err

def compare_images(imageA, imageB, title):
    # compute the mean squared error and structural similarity
    # index for the images
    m = mse(imageA, imageB)
    s = ssim(imageA, imageB)

    # setup the figure
    fig = plt.figure(title)
    plt.suptitle("MSE: %.2f, SSIM: %.2f" % (m, s))

    # show first image
    ax = fig.add_subplot(1, 2, 1)
    plt.imshow(imageA, cmap = plt.cm.gray)
    plt.axis("off")

    # show the second image
    ax = fig.add_subplot(1, 2, 2)
    plt.imshow(imageB, cmap = plt.cm.gray)
    plt.axis("off")
```

```python
    # show the images
    plt.show()

# load the images -- the original, the original + rotation1,
# and the original + rotation2
original = cv2.imread("images/book_original.png")
rotation1 = cv2.imread("images/book_rotaion15.png")
rotation2 = cv2.imread("images/book_rotation90.png")

# convert the images to grayscale
original = cv2.cvtColor(original, cv2.COLOR_BGR2GRAY)
rotation1 = cv2.cvtColor(rotation1, cv2.COLOR_BGR2GRAY)
rotation2 = cv2.cvtColor(rotation2, cv2.COLOR_BGR2GRAY)

# initialize the figure
fig = plt.figure("Images")
images = ("Original", original), ("rotation", rotation1), ("rotation2", rotation2)

# loop over the images
for (i, (name, image)) in enumerate(images):
    # show the image
    ax = fig.add_subplot(1, 3, i + 1)
    ax.set_title(name)
    plt.imshow(image, cmap = plt.cm.gray)
    plt.axis("off")

# show the figure
plt.show()

# compare the images
compare_images(original, original, "Original vs. Original")
compare_images(original, rotation1, "Original vs. rotation")
compare_images(original, rotation2, "Original vs. rotation2")
```

Figure 3: Python Code for the image comparison algorithm described

Upon running the code, we will get the following results. We first show which 3 images are going to be compared so that we do not make a mistake of comparing wrong images. Then we compare two images with Mean Squared Error and Structural Similarity Index Model and show the results.



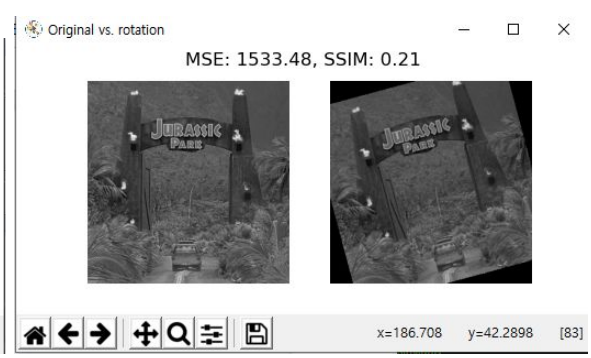


Figure 4: Results of the implementation

The results of the implementations with different angles are shown in the following graph.

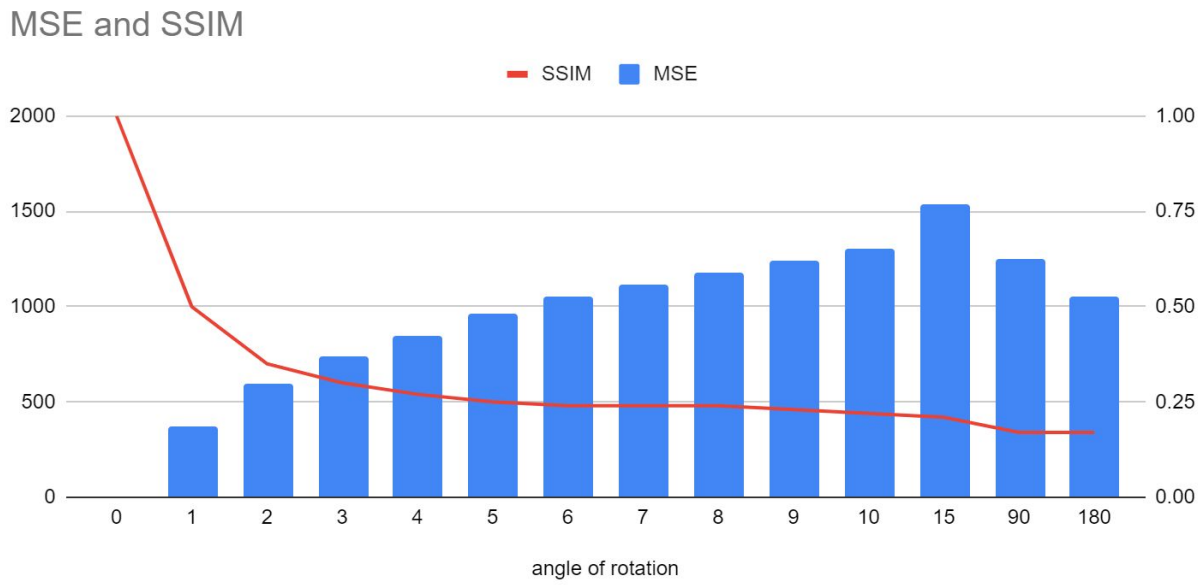


Figure 5: Graph of the implementation

We have inserted the data for 90 and 180 degrees rotation as an indicator/anomaly of what the results of MSE and SSIM would be if the pictures are unarguably different. The graph shows that any number of MSE greater than 1051 and any number of SSIM lower than 0.17 would indicate that the pictures are different. Then comparing them to the smaller angles of rotation, it seems that MSE is trustworthy below 5 degrees of rotation and SSIM can be trusted up to 15 degrees of rotation. Reflecting on such values, We have determined to use SSIM to differentiate the images of textbooks and limit the rotation of the book to be 5 degrees to the horizon.

2. Scheduling System Implementation and in-depth look

In this section we will discuss the implementation of the scheduling system that will be used in the mobile application that we will build. This implementation is essential to uphold multiple high-level requirements that we have put forth. In order for the right LEDs to light up, it would be absolutely necessary for the system to correctly collect, process and make use of the schedule

data entered by the user. The following is the python script we have written to implement the scheduling functionality of the application.

```python
import datetime
import numpy as np


def schedule(books):
    full_schedule = []
    print("Enter your schedule:")
    print("Day 1: Monday    Day 2: Tuesday   Day 3: Wednesday")
    print("Day 4: Thursday  Day 5: Friday")
    for i in range(5):
        print("Enter the schedule for Day " + str(
            i + 1) + "  with 3 subjects (IN CAPS) separated by commas and spaces:")
        day = input()
        day_list = day.split(", ")
        full_schedule.append(day_list)
    action = input("Type p to pack, q to quit : ")
    if action == "q":
        exit()
    print("Detecting books to be packed...")
    today = datetime.datetime.today().weekday()
    tomorrow = today + 1
    if tomorrow > 4:
        tomorrow = 0
    tomorrow_schedule = full_schedule[tomorrow]
    LED = np.zeros(16)
    for i in range(len(books)):
        b = books[i]
        if b in tomorrow_schedule:
            LED[i] = 1
    return LED
```

Figure 6: Python code for scheduling implementation

In the above code, we ask the user to enter his/her schedule with the assumption that there are 3 subjects per day. This may be further changed to accomodate a higher number of subjects. After the schedule is entered by the user, the user may choose the “Pack” option to begin the packing process to see which books would be required to class the following day. The above function “schedule” takes in the list variable “books” as input. This list “books” comes from the computer vision system. The system outputs the subjects of the books placed in the various positions of the shelf. The list coming from the computer vision system as an input to the scheduling system may look something like the following.

```
["PHY", "CHEM", "MATH", "BIO",
"CHEM", "GEO", "HIST", "MUS",
"BIO", "ECE", "CS", "CS",
"MATH", "ECE", "PHY", "CHEM"]
```

Figure 7: Output of Vision System, Input into scheduling system
List in which each position represents a spot on the shelf

Each subject above would represent the subject of the books placed in positions 1-16 on the shelf respectively.

This script automatically determines the next “working” day of the week and uses that schedule data of that day. Based on the subjects required for the next day of classes, the script assigns either a 1, or 0 to each book position on the shelf where 0 represents red and 1 represents green. This data can then be relayed to the LED system that will light up the LEDs either red or green as necessary. If we run the schedule function with the necessary input data as follows:

```
x = schedule(["PHY", "CHEM", "MATH", "BIO",
              "CHEM", "GEO", "HIST", "MUS",
              "BIO", "ECE", "CS", "CS",
              "MATH", "ECE", "PHY", "CHEM"])

print("SHELF LEDS \n")
for i in range(4):
    for j in range(4):
        print(str(x[(4*i+j)]) + "\t", end =" ")
    print("\n")
```

Figure 8: Running the schedule function

We would follow the following user process to pack:

```
Enter your schedule:
Day 1: Monday    Day 2: Tuesday   Day 3: Wednesday
Day 4: Thursday  Day 5: Friday
Enter the schedule for Day 1  with 3 subjects (IN CAPS) separated by commas and spaces:
CS, ECE, BIO
Enter the schedule for Day 2  with 3 subjects (IN CAPS) separated by commas and spaces:
PHY, CHEM, BIO
Enter the schedule for Day 3  with 3 subjects (IN CAPS) separated by commas and spaces:
MATH, ECE, MUS
Enter the schedule for Day 4  with 3 subjects (IN CAPS) separated by commas and spaces:
PHY, CHEM, GEO
Enter the schedule for Day 5  with 3 subjects (IN CAPS) separated by commas and spaces:
CS, GEO, HIST
Type p to pack, q to quit : p
Detecting books to be packed...
SHELF LEDS

0.0      0.0     0.0     1.0

0.0      0.0     0.0     0.0

1.0      1.0     1.0     1.0

0.0      1.0     0.0     0.0
```

Figure 9: Entering the schedule and receiving the output for LED lighting

As seen above, since the day was Friday, the books for Monday were the ones required; namely, CS, ECE and BIO. Based on the shelf positions of the books we have seen in Figure 7, our scheduling system has correctly identified the books necessary for the next day of classes and labelled them as such with a “1”.

3. Second Project Conclusions

3.1 Implementation Summary

In chapter 2, we implemented some of the key components of our system. These included the computer vision system and the scheduling system. For the computer vision system, we analyzed two models of image comparison and deduced which model would best suit our needs and maintain the accuracy necessary for our design. We also went on to design a scheduling system for our design. This scheduler by Gautam takes input from the computer vision system on the subjects placed in the 16 different positions on the shelf. The system then uses the schedule data from the user to determine LEDs of the 16 must be lit green and which ones must be lit red.

We strove to eliminate as much inaccuracy as possible in the vision system. In our implementations of image comparison model by TK and Anusha, we found out which mathematical model of image comparison is more lenient when used with rotation. We found that angle is a critical part of image comparison and a quantitative angle should be set as the maximum allowance of inaccuracy to correctly compare the user's picture with the bookshelf's photo.

Using our knowledge of image comparison, we want to make the user take a photo of the book just at the same angle as the bookshelf's camera is going to take. To accomplish this, we will have to be meticulous when we install our camera according to the angle of the book when it is rested on the bookshelf. The picture taken by the raspberry pi camera will have to be perfectly horizontal for easier comparison with the user's input. We will also have a **dotted line** across the application's camera function so that the user can align the book horizontally before they take a photo of it. The overall image comparison is made more accurate when the angle difference between the images is minimized to the greatest extent possible.

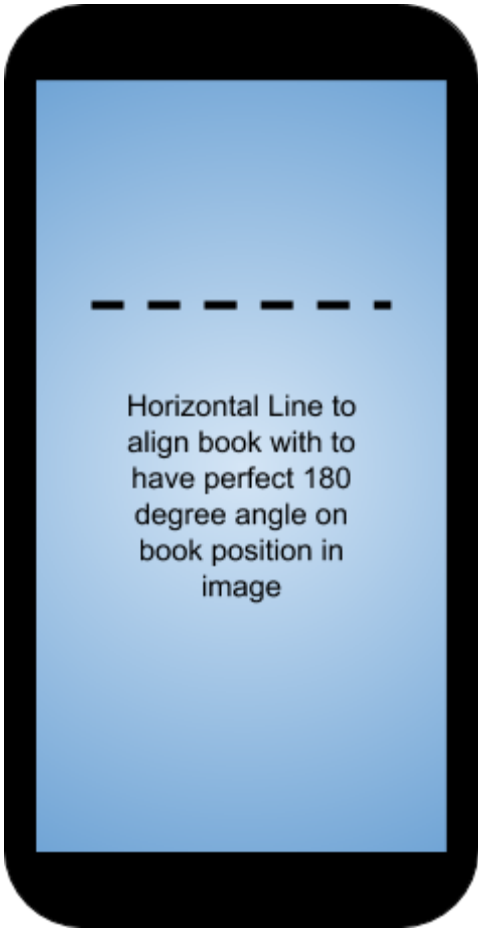


Figure 10: Providing the user with a perfectly horizontal dotted line to align the top of the book with

3.2 Unknowns, Uncertainties, Testing Needed

In our smart bookshelf, 4 books will have to fit in a single picture. Since we are using only 4 cameras, a camera will have to share 4 book positions. Then it is up to us to differentiate where the book is located within the picture using previous knowledge of the bookshelf. Due to the COVID-19 pandemic, we cannot physically build this bookshelf due to lack of parts, limited time and void of help from the machine shop. We will have to install our cameras, hard code where the books would be in the photos, and calculate the inaccuracy according to their difference in angle and distance.

We were also slightly uncertain about the extent to which our current image comparison algorithm would hold good under slightly modified circumstances. If the cameras are not angled perfectly, our algorithm could potentially lose a significant amount of accuracy. For this, we would consider using an alternative algorithm that uses feature descriptors. We would also consider using a filter on the image to be able to detect the book more accurately irrespective of the surroundings.

One aspect of uncertainty in our design was the angle difference in the positioning of the books in our images. An assumption that we make in our design is that the books in all of the images of the cameras are positioned perfectly horizontally. While we have chosen a position of the camera that brings this to be close to true, it is not possible for this to be perfect. This is one of the reasons why we chose to perform a tolerance analysis for the impact of the angle on the accuracy of the comparison algorithm. Since each camera is responsible for 4 book positions, the perfect accordance of the top of every book with a 180 degree angle cannot be guaranteed. Due to this, there may be some error to a certain extent, which was analyzed in our tolerance analysis. Overall, we were slightly uncertain about the positioning of the cameras, and this is something we would have had to work on side-by-side with the machine shop.

3.3 Ethics and Safety

3.3.1 Ethics

We would like to build a system that is accurate in finding the assignment of books to LED. Even though we would like the vision system to get the exact book assignments perfectly, it is one of our high-level requirements that we aim for an accuracy of ninety nine percent or above. We did this to be realistic and not lie about the efficiency of our product. We are abiding by the [3] IEEE Code of Ethics #3 by doing this. To alarm the users of which books to use, we use colored LED indicators: Green for correct behavior and red for incorrect user behavior. According to IEC 60601-1-8 standards [3], in the collateral standards, test and guidance for alarm systems in equipment is necessary using colors of indicator lights. Red indicates that immediate user intervention is required or used in dangerous situations and green indicates normal situations and equipment is ready to be used. We will be following the IEC 60601-1-8 collateral alarm standard and implementing the alarm system into the smart bookshelf.

3.3.2 Safety

We plan to address the safety concerns with a few precautions so that our users are not afraid to use our product. We would want the alarm system on the phone to not be louder than 70 decibels as that is the limit to safe listening according to the World Health Organization [4]. Another safety issue is that in domestic situations that this bookshelf could be damaged by the spilling of water or other liquids and that could cause a short circuit and ruin the components [5]. This is a risk and adding a rubber casing for some of the electrical components of the device such as the pcb, wifi, and the raspberry pi will make it safer to use for our scenario. These are the only important components that need to be covered because they are the only ones which are in the open.

Additionally, we will be using a converter to change 120 V to 5V for some of our devices which would avoid any potential electrical hazards. Since we will be working with wall power, we have to be extra careful with high voltage outlets. We will first test our project in a safe lab environment where we have a guaranteed 5V source, then we will test our voltage converter to see if it does indeed provide a 5V source while using the one hand rule. This way, we will prevent potential damage to the sensors and microprocessors as well as ourselves. All these precautions comply with the lab safety guidelines.

3.4 Project Improvements

If we were given some more time to finish this project, we would first like to add another filter such as an object detection in OpenCv, into the image comparison program. In the current system, if the book does not completely fill the whole image, then the accuracy of the measurement drops tremendously due to any input from the background. If we could have some more time to implement some sort of a filter that can distinguish the book and crop it, then we can greatly decrease the inaccuracy created by the distance and angle of the photo taken. This is because it will be a simple matter of taking the cropped image of the book from the photo, then resizing and rotating it for better comparison. This is why we have done the tolerance analysis on the effect of the angle of rotation when comparing images. We want to have enough

distance between the camera and the bookshelf so that the books look very horizontal regardless of their position.

We would also want to include feature descriptors as a part of our image comparison system. Due to the limited time we had on the project, we could only implement algorithms that we were familiar with. However during the design review, one of the main points of feedback was to use feature descriptors that identify interesting parts of the image as features and use them as a fingerprint to compare images. This method of comparison would allow us more leniency on the angle and size of the image. If we were given enough time, we would like to research and implement such models to further improve on our accuracy.

If we were given more time to get the parts, we would also like to test our image comparison with the materials that are going to be used in the bookshelf. Currently we do not have the raspberry pi and camera so we do not know the specific size, definition and the angle at which the photo of the books will be taken. Because of that, we have only tested with well defined images from the internet which would drastically increase in performance. If we were given enough time, we would like to test our product with less perfect images.

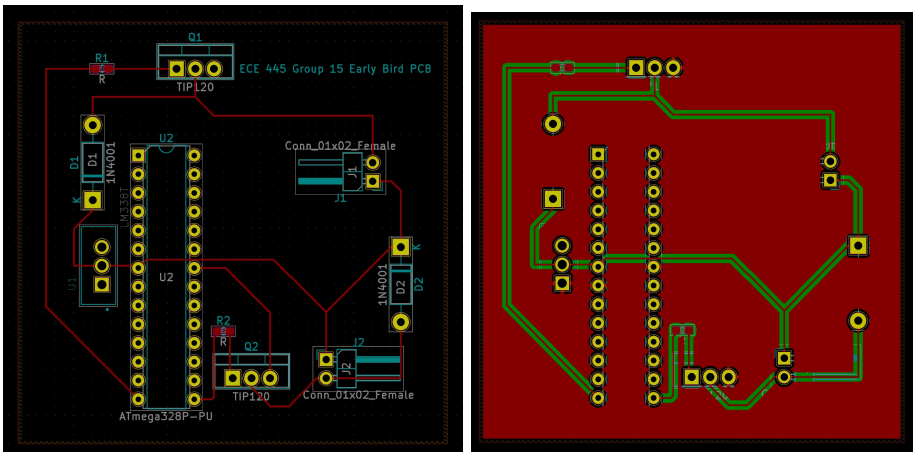
Another improvement we would make is positioning the cameras at the perfectly optimum positions. Since we have tasked each camera with 4 book positions, it will be very important for the cameras to have an accurate (w.r.t angle and position) view of the books. As stated, we made an assumption that the images from the camera always had the books perfectly aligned to a 180° line. For this assumption to hold perfectly true, the camera positions would have to be adjusted accordingly, which unfortunately couldn’t be done due to the current circumstances.

4. Progress Made on First Project

After the first design review meeting, we continued to work on our first project until Spring Break began. We managed to complete the following components of the first project:

1. **First PCB Design ordered for our first project**

   We submitted an early-bird PCB order on PCBway for our initial design. We were able to pass the audit with PCBway and able to place an order for our initial design.



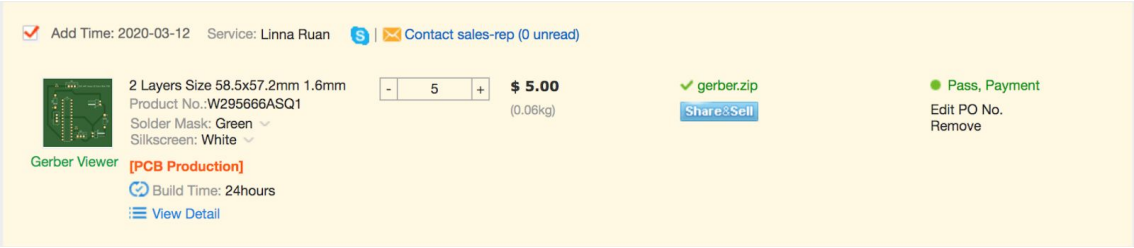
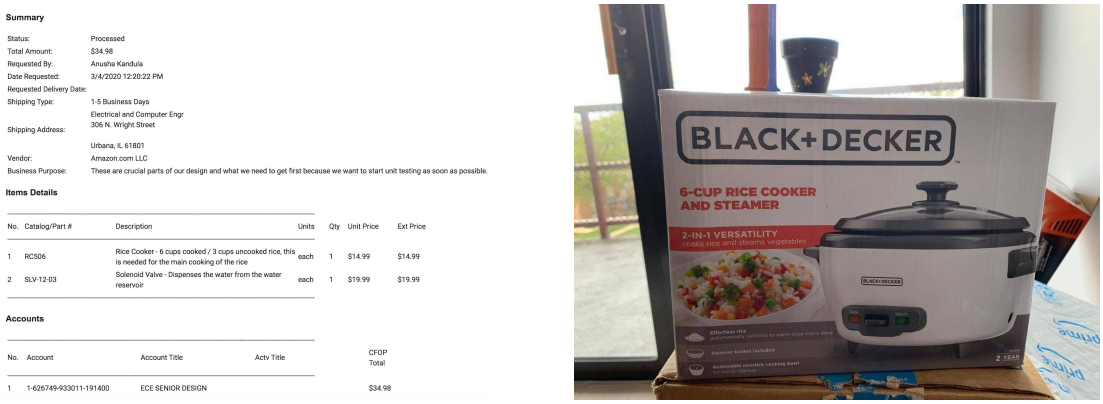


Figure 11: PCB design for first project and approved PCBway audit

2. **Ordering of some of the necessary parts to begin testing and building our project**



Summary

Status:	Processed
Total Amount:	$34.98
Requested By:	Anusha Kandula
Date Requested:	3/4/2020 12:20:22 PM
Requested Delivery Date:	
Shipping Type:	1-5 Business Days
Shipping Address:	Electrical and Computer Engr 306 N. Wright Street Urbana, IL 61801
Vendor:	Amazon.com LLC
Business Purpose:	These are crucial parts of our design and what we need to get first because we want to start unit testing as soon as possible.

Items Details

No.	Catalog/Part #	Description	Units	Qty	Unit Price	Ext Price
1	RC506	Rice Cooker - 6 cups cooked / 3 cups uncooked rice, this is needed for the main cooking of the rice	each	1	$14.99	$14.99
2	SLV-12-03	Solenoid Valve - Dispenses the water from the water reservoir	each	1	$19.99	$19.99

Accounts

No.	Account	Account Title	Actv Title	CFOP Total
1	1-626749-933011-191400	ECE SENIOR DESIGN		$34.98

Figure 12: Invoice of parts ordered and rice-cooker arrived

5. References

[1] Iosrjournals.org, ‘Back Problems Due To Heavy Backpacks in School Children’, 2013. [Online]. Available: https://www.iosrjournals.org/iosr-jhss/papers/Vol10-issue6/D1062226.pdf?id=6253. [Accessed: 30 - Mar - 2020].
[2] pyimagesearch.com, “How-To: Python Compare Two Images”, September 15, 2014 [Online]. Available: https://www.pyimagesearch.com/2014/09/15/python-compare-two-images/ [Accessed: 31- Mar - 2020].
[3] Ieee.org, “IEEE Code of Ethics”, 2020. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed: 30 - Mar - 2020].
[4] World Health Organization, ‘Make Listening Safe’, 2020. [Online]. Available: ’https://www.who.int/pbd/deafness/activities/MLS_Brochure_English_lowres_for_web.pdf. [Accessed: 30 - Mar - 2020].
[5] Acm.org, “ACM Code of Ethics and Professional Conduct”, 2020. [Online]. Available: https://www.acm.org/code-of-ethics. [Accessed: 30 - Mar - 2020].