SmartStand

Sebastian Garcia, Mary Rose Rubino, and Olivia Yang

Department of Electrical and Computer Engineering,
Carnegie Mellon University

***Abstract*—Laptops, while portable, are not ergonomic to use as the screen height is several inches lower than eye level when placed on a table. This less-than-ideal screen height can cause muscle strain and poor posture as users often hunch over for extended periods of time in order to better see the screen. We aim to fix this issue by creating a portable laptop stand that is able to calibrate and rise to the ergonomic height for any user. Additionally, we will provide a graphical user interface that will track a user's eye movement and posture over time and notify them when they are exhibiting poor posture and when they should take a break to prevent eye strain as well provide feedback data on how often they showed poor posture.**

***Index Terms*—Arduino, Computer Vision, Eye Tracking, Facial Landmark Detection, Neural Network**

I. Introduction

Laptops are incredibly convenient and portable; they're lightweight, compact, and can provide the ability for a user to work and communicate with others from practically any location. However, laptops are usually not ergonomic since the screen sits just around 10-15 inches above the surface that they are placed atop whereas the ergonomic height for a computer screen is slightly below eye level. For a user sitting straight with good posture, the screen provided by laptops are often several inches lower than the ideal ergonomic height. For some groups of individuals that are often required to spend long periods of time using their laptops such as students and professionals that work remotely, this screen height can cause several health issues. Since these groups may spend many hours at a time working on their laptop, they can experience neck pain and muscle strain since they have to crane their neck at an uncomfortable angle, and they may often find themselves hunching or slouching in order to better see the screen.

Our project aims to solve this issue for those who spend a lot of time looking at their computer screen by creating a portable laptop stand that will automate the height and angle adjustment to face parallel to a user at an ergonomic height for them. This will help their posture by allowing them to use their laptops without constantly looking down at the screen. We also aim to help improve a user's posture by using a trained neural network that will analyze users' posture in real time, and send them notifications through a GUI when they are slouching, hunching, or showing overall poor posture so that they can hopefully correct their posture to better reduce muscle pain. Additionally, to reduce eye strain that can arise from long periods of screen time, we will use computer vision to track the user's eye movement and send notifications for them to take breaks after 20 minutes of intense focus.

While portable laptop stands currently exist, most retail laptop stands have predetermined heights and are not adjustable to match a user's height. Additionally, our system will serve multiple purposes to help the wellbeing of the user rather than just being a stand on which to place a laptop. Our project aims to help reduce the strain and discomfort of using a laptop for long periods of time.

II. Use-Case Requirements

Based on our system's user base, we compiled a list of requirements that guided our design process:

- Our device is intended to be portable, so it will need to be light enough for the user to carry without struggling. We determined that 5 lbs would be a reasonable maximum weight, as this is on the lower end of average laptop weights.
- Our device is intended to be more convenient than existing manual laptop stands, which includes its calibration time. Because we use a feedback loop to gain the most accurate calibration results, it is difficult to estimate setup times. However, we determined that 3 minutes would be a reasonable total calibration length, including landmark measurements as well as lifting and tilting of the stand).
- Since our device is meant to be portable, we determined that we would need a power source so that the device could be used anywhere. Based on the average work/school day, we determined that an 8-hour battery life would be reasonable.
- One of the intended uses of our device is to monitor the user's line of sight to limit eye strain from prolonged staring at the laptop screen. After a user looks at the screen for 20 minutes, it is recommended that they look away from the screen for 20 seconds or longer. To implement this, we will sample at a rate of once every 3 seconds. This will eliminate quick glances away from the screen that do not contribute to reducing eye fatigue.
- One of the intended uses of our device is to notify users when they are slouching. To implement this, we will sample camera data to obtain the user's current position. We will then compare this to the landmarks gathered at the calibration stage. We will sample this data every 15 seconds, which will eliminate changes in position that are unrelated to posture, such as bending down.
- Another use of our device is to display the user's posture progress in the app so that the user can view the number of negative changes over time. We determined that information about the number of times the user changed posture per day should be recorded and displayed as a bar graph that includes

the past 30 days.

III. Architecture and/or Principle of Operation

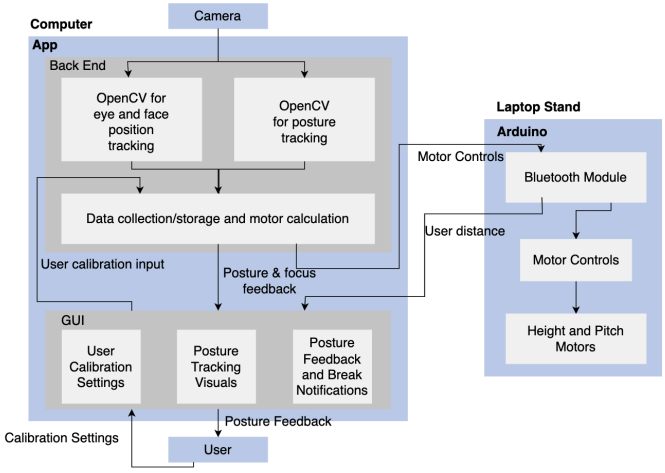

Fig. 1. Detailed overall block diagram of the device.

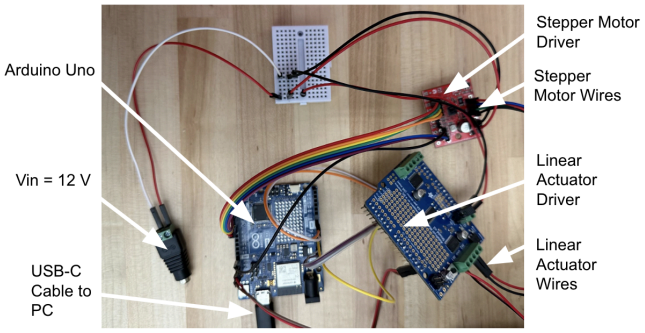

Fig. 2. Hardware components.

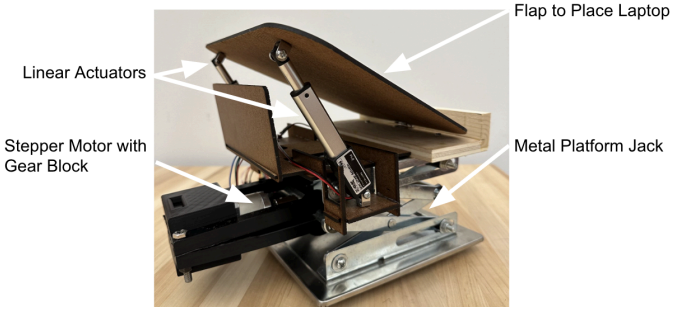

Fig. 3. Overall structure of SmartStand.

Our system's design and subsystem interactions are shown in Fig 1. Overall our system is composed of hardware, software, and mechanical components. On the software side, all components are included as part of the GUI. On the hardware side, all components, including the motors, linear actuators, and the Bluetooth module will be included on the stand. Our mechanical components refer to the stand itself.

Changes from the design report include: using a standard wall outlet for power (12 V, 3 A) instead of using a standalone battery. The motor we used ended up being much heavier than expected (because we ended up attaching a gearbox for added torque), so we didn't want to weight the stand down even fu

A. *Hardware*:

The Arduino will receive data from the laptop via Bluetooth. Specifically, the ESP32 module is included in the Arduino Rev4. This data will indicate how many rotations the motors need to turn to correctly adjust height, or how much the linear actuators will need to raise to adjust angle. This will then be passed through the motor shield to control the motors and linear actuators.

B. *Software*:

This section describes the GUI that the user will download to interact with the stand.

B.1 GUI: *Front End*

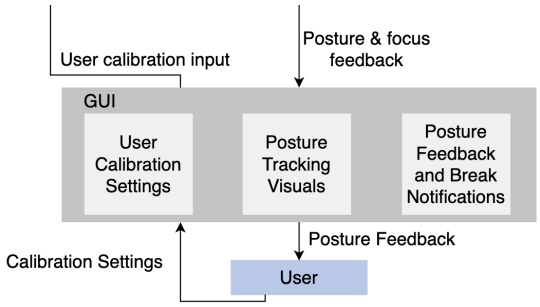

Fig. 4. Block diagram showing the flow of the frontend portion of the software subsystem in more detail. Note that "Calibration settings" is sent to the backend, and that "Posture & focus feedback" is sent from the backend.

A basic overview of the front end is demonstrated in Fig 2. As demonstrated, the user has access to stand controls, including recalibration, which requires the use of camera data to determine the user's posture landmarks and a manual shut-off option. The user also has the option to view their tracked posture data. This will also be how the user views notifications, including when it is time to look away from the screen, or when a change in posture was detected. A more detailed mock-up of the UI is shown in Fig 3.

Fig. 5. Brief example of how the home screen of our app will appear to the user. This is the simple base layer without andy notifications.

B.2 GUI: *Back End*

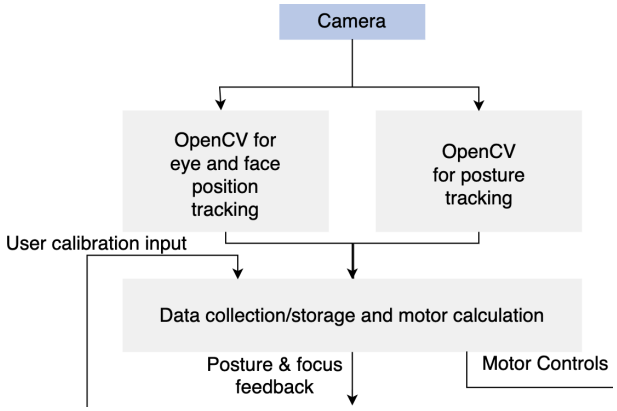

Fig. 6. Block diagram showing the flow of the backend portion of the software subsystem in more detail. Note that user calibration comes from the user, and that motor controls are forwarded to the hardware.

On the backend, as pictured in Fig X, OpenCV will be used to track posture and eye movement. Based on these measurements, notifications will be sent to the front end of the application, or motor adjustment information will be sent to the hardware subsystem. Here we will include the laptop's end of the Bluetooth communication to the Arduino.

IV. Design Requirements

Based upon our use case requirements, we were able to determine a set of design requirements that ensure our device will be effective.

- The stand must lift to 12 inches. This height was determined based upon the average desk height, and the upper average person's sitting height to determine a maximum height that would allow for most people to be able to use the product.
- The stand must angle up to 45 degrees. This angle was what we determined to be a reasonable degree of freedom to account for different sizes of laptops that open to different maximum angles.
- Slouch detection must be determined in less than one second. This means that when the camera is sampled The current data must be compared to the calibration data within this period. This was determined to be a reasonable amount of time so that the user would receive a notification quickly.
- Eye detection must be determined in less than one second. This ensures that notifications about how long the user looked at the screen are sent in a timely manner.
- The current ideal height must be reached within 5 seconds. This means that during one cycle of the height-raising feedback loop, the height sent to the Arduino from the laptop must be reached within 5 seconds. This was determined in case multiple cycles are necessary. Making this less than 5 seconds ensures that our calibration time remains low.
- The current ideal angle must be reached within 5 seconds. This means that during one cycle of the tilting feedback loop, the angle sent to the Arduino from the laptop must be reached within 5 seconds. This was determined in case multiple cycles are necessary. Making this less than 5 seconds ensures that our calibration time remains low.
- The margin of error for the laptop stand tilt must be less than 5 degrees. We determined this angle based on the tradeoffs between usability (whether the tilt is accurate) vs calibration time.
- The margin of error for the laptop stand height must be less than 3 inches. We determined this angle based on the tradeoffs between usability (whether the tilt is accurate) vs calibration time.

V. Design Trade Studies

Our design involved many tradeoffs mainly in the following areas: platform jack choice, battery pack, linear actuators, wired connection between arduino and PC, neural network for posture detection.

A. Trade Study: Platform Jack

One of our main design requirements was that SmartStand is compact. Initially, our plan was to have a 3D printer type construction where z-axis movement was accomplished by using screwed stepper motors similar to how a 3D printer moves the print nozzle in the z axis. However, we felt that this would be too cumbersome to carry around and would detract from the user being able to comfortably use his computer. So instead we opted for a platform jack which is extremely compact in its closed state (great for transportation), has incredibly simple vertical control (only requires motor rotation of a screw), and is capable of lifting heavy loads as demonstrated by the use of jacks in industrial settings.

B. Trade Study: Power Supply

Another trade study we conducted was in the power supply. We initially wanted to power the motors over the USB connection from the PC but we realized that this would exceed the current rating for USB. This would have been the best option because it would have made the whole design lighter and more compact which was a use case requirement (< 3 lb weight).

We also considered using a wired connection from an outlet but we felt that this would make SmartStand more

cumbersome to use as this extra step would be a large burden on the user.

In the end, we settled on using a AA battery pack because we knew that the batteries would supply enough voltage (12 V) to power the system. Also, they had such a surprisingly high capacity that they would last for 2 years (assuming the user uses SmartStand 3x per day). This longevity in the design made it clear to us that we would not even need to make the batteries rechargeable as originally planned. Changing the batteries in the system every 2 years is such a small burden on the user that we felt it was much lower than requiring the user to have to charge SmartStand regularly.

C. *Trade Study: Linear Actuators*

We conducted a lot of research to decide on how to change the angle of the computer so that eventually the screen is parallel to the plane of the user's face. We considered using stepper motors with screws similar to a 3D printed but again for the same reasons stated in trade study A we felt that this design would be too invasive for the user. Additionally, we were not confident that the stepper motors would be fast enough. Since we aimed for a 5 second setup phase for SmartStand, we knew that the linear actuators we chose that are capable of 2 inches per second would meet the requirement. Also, the linear actuator implementation is extremely simple. We will simply use a GPIO switch from the Arduino to toggle the actuators.

D. *Trade Study: USB Cable to Arduino*

We originally planned on SmartStand being a fully wireless system by using a bluetooth/wifi connection between the PC and the SmartStand Arduino but we felt that this was an unnecessary complexity. While it would be nice for SmartStand to be wireless, the cost to the user of connecting to bluetooth would be about the same as him connecting the USB cable to his computer. And the USB cable would be much more reliable. Since we have aggressive use-case requirements regarding the accuracy of the height/pitch adjustment (+/- 1 in of elevation error and +/- 5 deg of pitch error) we could not afford any packet loss over the bluetooth connection. Additionally, we have an aggressive timing requirement (5 seconds to set up the system) which would be jeopardized by using a potentially unreliable wireless communication protocol.

E. *Trade Study: Neural Network for Posture Detection*

The input data to our slouch classifier is simply the deviation of body landmarks from a known "good posture" state. From this statement you could argue that the classifier could simply be a step function. For example, if landmark A (e.g. the shoulder landmark) is 10 pixels away from the ideal position, then slouch is true, else slouch is false. While this scheme is simple, it fails to take into account the big picture, that is all the other landmark data (e.g. the other shoulder, and facial landmarks). Furthermore, the thresholder fails to assign weights to these other landmarks.

We chose a neural network design because it would assign weights automatically, would be able to input all the landmark data, and it would be straightforward to train because our team could generate the training data.

VI. System Implementation

The entire system will consist of a modified platform jack where the bottom platform of the jack contains the arduino, motor controller, and batteries to power the stand.

A. *Subsystem A: Platform Jack*

The platform jack will be responsible for lifting and lowering the computer along the z axis. This will be accomplished by having a motor rotate a screw inside the platform jack, converting rotational motion to linear motion. A diagram depicting the system is shown below.

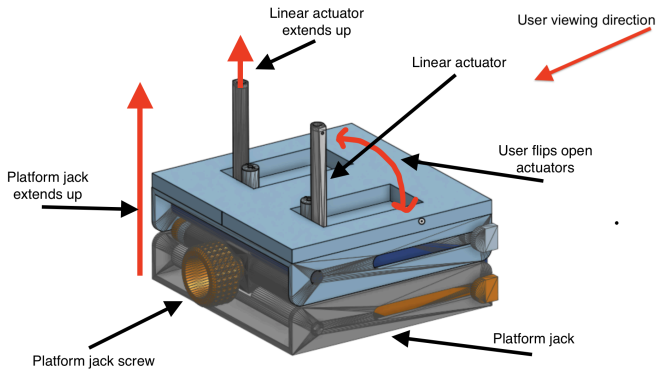

Figure 5: Platform Jack Design

The platform jack will not be 3D printed because we found this process to be expensive, time consuming, and resulting in a jack that is not sturdy enough to be used in a rugged way with a heavy object like your computer. So, instead we opted to buy an off-the-shelf metal platform jack which will be much more robust. An image of the jack is shown below.

Figure 6: Off-the-shelf Platform Jack

This off-the-shelf metal platform jack will be large enough to balance a laptop (~3 lbs) and will have enough vertical elevation change to meet our height requirement (our requirement was 12 inches of elevation to accommodate almost all users and the platform jack is capable of 13 inches).

This off-the-shelf jack does not suit all of our needs, however. We will still need to fabricate an enclosure

underneath the jack where we will house the arduino, motor controller, and batteries.

We will 3D print this enclosure because the geometry will be simple, not much infill will be required because the enclosure does not have to be very robust, and because the print time will not take long. Essentially, the enclosure will be a flat box with separate slots for the arduino, motor controller, and battery pack. Since the thickest item will be the arduino with the motor controller shield attached on top, this will force the thickness of the enclosure to be ~ 2 inches. Our battery pack will consist of a layer of 8 AA batteries which are each about 1 in thick.

B. Subsystem B: Linear Actuators

Figure 7: Linear Actuators Selected

As shown in Figure 5, the linear actuators in the design will be responsible for angling your computer so that the computer screen is parallel to the plane of your face. These linear actuators will flip out of the top platform of the jack so that when SmartStand is not being used, it is a compact box.

When the user would like to set up the stand, he will flip open the linear actuators as shown in Figure 5, and then simply place the computer on top of them. That way, when the linear actuators extend up or down, the computer screen angle will change. It is important to note that we will have to modify the top platform of the off-the-shelf jack so that the computer does not slip off the stand. To accomplish this, we will simply 3D print a platform with a lip on one side to keep the computer on the stand. To keep exposed wiring to a minimum, we will drill holes in the top platform of the jack so that the linear actuator power and ground wires can be threaded inside the jack.

Additionally, since extending the tips of the actuators will cause them to rub against the bottom of the computer, we will attach cotton swabs or potentially velvet patches to the tips of the actuators so they can easily slide across the bottom panel of the computer.

The linear actuators we selected are rated for 4.5 lbs of force and since most laptops weigh about 3.5 lbs and we will have 2 actuators for redundancy, this means each actuator will be responsible for about 1.75 lbs which is well below the max rating. So, we have a factor of safety of about 2.

The flipping mechanism will be accomplished by using 2 mechanisms. The first is the rotation brackets that come with the linear actuators and are shown below.

Figure 7: Bracket for Rotation

These brackets will allow for smooth rotation of the actuators coming out of the platform jack. The second mechanism we will use will be a simple rod which will lock the actuators in the upright position.

C. Subsystem C: Battery Pack

The linear actuators require 12 V and the motor controller requires a power supply of 5 - 12 V so we will opt for a 12 V power supply to power both units in SmartStand. Based on this assumption we opted for using 8 AA batteries as our battery pack but we had to perform power calculations to ensure that this battery pack would have sufficient energy to meet our use case requirements. These calculations are shown below.

8 AA batteries contain 4 * 8 = 32 watt-hours. We expect our 12 V motor to run at the max torque of 2 kg-cm where it is expected to draw 300 mA. This means that it will have a power consumption of 3.6 W.

In terms of the linear actuator, there is no data about its current draw so we will safely assume that in total, both linear actuators consume just as much current as the motor (they likely are an order of magnitude based on the fact that the linear actuators do not have the additional burden of applying force through a high-friction platform jack system). So, the linear actuator's power consumption will also be 3.6 W which means that the total power consumption will be about 7 W.

Since the batteries contain 32 watt-hours of energy, this implies that the user has about 5 hours of usage before replacing the batteries. This may seem long but the motors are only running for a small fraction of time when the user uses SmartStand.

For example, if we assume the user activates smartstand 3x per day and each activation takes 10 seconds (5 seconds for the stand to expand, and 5 seconds to contract), that equates to 30 seconds per day). This means that the user can use SmartStand for 5 hrs = 180,000 seconds / 30 seconds per day = 600 days. A 2 year lifetime before replacing batteries is competitive with other peripherals like keyboards

and mice. It is important to note however, that this is an ideal calculation and that batteries depletion rates vary greatly.

D. Subsystem D: Posture Detection

Figure 8: OpenCV for Posture Landmarks

To notify the user when their posture suffers, we will be running a Python program that the user will launch from their machine. This program will stream camera frames from the webcam and pass them through an off-the-shelf neural net that detects landmarks in the face and shoulders of the user.

To determine when the user is slouching, we will have a calibration stage after SmartStand has transformed the computer position and orientation to the most ergonomic state. In this stage, we will ask the user to sit with the most perfect posture they can, and the program running will take a "screenshot" of the posture, essentially recording the landmark data. Then the Python program will be taking the difference in positions between the ideal landmark coordinates and the current coordinates for every frame received by the camera. When there is sufficient deviation in the landmark coordinates, the user will receive a notification to fix their posture.

We plan on using a neural network for the slouch classification because it will handle the weighting of the various landmarks automatically and will be easy to train. All the members of team B2 will supply hundreds of "bad posture" and "good posture" photos with which to train the model.

E. Subsystem E: Motor Control Pipeline

The actual control of the motors is a complex and lengthy process. It will start with a calibration stage (this is the first calibration stage, the second is described in Subsystem D section). This stage will take place when the user places the computer on the stand in its fully closed position and opens the screen to an arbitrary angle. At this point our application (a Python program) will prompt the user to face the screen so the place of their face is parallel to the screen. This will allow the application to take note of the distance between two chosen landmarks on the user's face, shown below (landmarks 27 and 33).

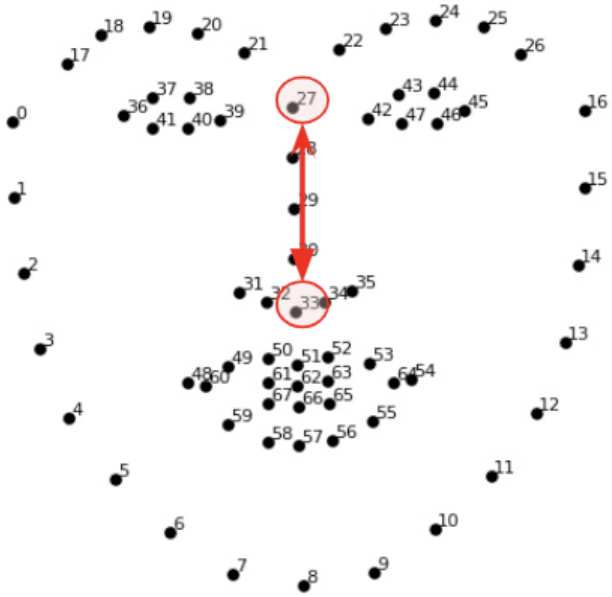

Figure 9: Facial Landmarks on Dlib 68

We chose Dlib 68 as our facial landmark neural net because of its high resolution. After this calibration stage, the user will be prompted to sit up straight with the best posture they can. After doing this, the distance between landmarks 27 and 33 will decrease because the plane of the user's face will be at an angle with respect to the plane of the computer screen. This difference in landmark distance is the basis for our SmartStand automatic adjustment.

To actually change the computer's physical position and orientation so the computer is in an ergonomic position, we will map the landmark distance error to motor and linear actuator control inputs. To accomplish this, we will have two separate PID functions running where the error term is the landmark distance difference. The proportional term in the PID controllers will ensure that SmartStand adjusts height and pitch so that both elements of the computer's state converge on the optimal values.

This Dlib 68 neural network will also be used for real-time eye tracking. This will be used to alert the user when he/she has been staring at the screen for too long. After 20 minutes of direct staring, the user will receive a notification to take his/her eyes off the screen to prevent eye fatigue.

F. Subsystem F: Motor Selection

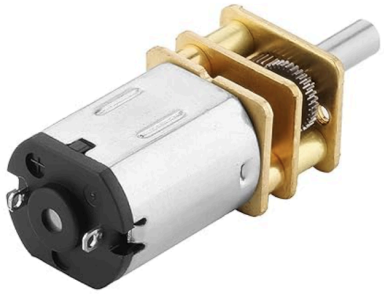

Figure 10: 12 V Motor Selected

To actually lift the computer we will need to rotate the screw in the platform jack. This will involve attaching a motor to the screw head. We have opted for the 12 V motor shown above as we believe it will provide sufficient torque (0.2 Nm). Since the weight of the computer is roughly 10 N we believe an order of magnitude difference is sufficient for elevation change. We have neglected formal calculations however because we are unsure of the force of friction associated with the internals of the metal platform jack. Once the platform jack arrives we will experiment with the motor to test its functionality.

Additionally, to actually attach the motor to the platform jack we will 3D print an attachment piece. We have found plans online which we plan to adapt for our own use.

VII. Test, Verification and Validation

To test for the height adjustment delay, we will place the stand at a low height and test the ability of the stand to both raise to the maximum height and complete its adjustment cycle and settle at the appropriate height within 5 seconds.

To test for the height adjustment accuracy, we will use the Python program to send a predetermined height to the Arduino in the stand and see if the motors raise the stand to the correct height by measuring its extension and ensuring that the margin of error is within 5 inches.

To test for the angle adjustment accuracy, we will similarly use the Python program to send a predetermined angle to the Arduino in the stand and see if the linear actuators angle the screen parallel to the user by measuring the angle and ensuring that the margin of error is within 5 seconds.

To test for the latency of the posture detection, we will calibrate the stand with a user exhibiting good posture first, then having the user slouch and checking that the Python program processed the image and categorized the posture type within one second.

To test the weight of the stand, we will weigh the stand when fully composed with the motor, linear actuators, battery, and Arduino with it's motor driver and make sure that this weight is below <> pounds.

To test for the battery life of the stand, we will measure the average current when the stand is going through a calibration period, when the stand is doing an automatic adjustment period, and when the stand is shutting down and closing. We will then calculate the power consumption of this one-time use and see if our battery is capable of providing power if the stand does one of these one-time uses per hour for 8 hours .

VIII. Project Management

A. Schedule

Our schedule is detailed in Figure 11. It was crafted with the intent that the three of us would work on different aspects of the product concurrently that eventually will need to be integrated. The weeks preceding Spring Break were focused on finishing the mechanical design of the stand and beginning the Arduino motor control code, eye line detection and height adjustment, and training the neural network. The next 2-3 weeks will be designated to finish these components, and the following weeks will begin the testing and integration phase of the project as well as creating the GUI.

B. Team Member Responsibilities

Sebastian's primary responsibility is to train the neural network with images of varying different types of posture and create the classification of "good" vs "bad" posture that will be used with real time images of the user as inputs. His secondary responsibility is to work on the software that will analyze the data produced by the network and determine when to notify a user of bad posture as well as display the user's progress over time.

Mary Rose's primary responsibility is to work on the firmware that will run on the Arduino to receive information and commands from the computer and translate those requests into motor movement and linear actuation. Her secondary responsibility is to create a PCB for the design if time permits and work on code to send the information over bluetooth.

Olivia's primary responsibility is to work on the height and angle adjustment software using OpenCV and Python for facial landmark detection and eye tracking. Since that task is more vital to the project, she will focus on that aspect for the next few weeks. Her secondary responsibility will be to work on the Python GUI that will receive input from the user and send it to the Python program or the Arduino as well as analyze posture data.

C. Bill of Materials and Budget

The bill of materials is detailed in Figure 12.

D. Risk Mitigation Plans

A critical risk factor in our design will lie in our ability to accurately determine the height to which the laptop stand should raise since our group has little to no experience using computer vision and it is difficult to tell the exact angle that a user's head may be directed when the camera is at different heights. To manage this risk, we will be adding a calibration stage in which the laptop stand will raise to a height such that the user's full face can be seen at a height more similar to that of the ideal ergonomic height. From this calibration stage, we can determine reference measurements of

the user's face that can then be used in the future to adjust the stand automatically.

Another critical risk factor in our design is the difficulty of determining "good" posture versus "bad" posture using the neural network. We foresee that it may be very difficult to get accurate categorizations of posture by training the neural network. So, we plan to manage this risk by using another calibration stage where a user will be instructed and guided to exhibit good posture. From this stage, the reference measurements and coordinates of shoulder landmarks on a user can be gathered and saved in order to do analysis on the user's posture in the future based on the changes.

IX. Related Work

Insignia Ergonomic Laptop Stand with Adjustable Height and Angle.

This is an adjustable laptop stand that is for sale that can be manually raised and angled. Even though there is no automatic component, this is most similar to our design, as it allows the user to adjust the stand more precisely than most traditional laptop stands.

X. Summary

SmartStand is a computer stand that automatically adjusts the elevation and angle of your computer so that it is always at the most ergonomic position for you. We believe that this is a high value benefit because many people, especially students, spend hours hunched over the PCs simply by working at a desk. With the advent of smaller and smaller USB keyboards, we believe that people would also be willing to carry around with them a compact box that will expand to make their PC as comfortable to use as a desktop. We believe that our users will be able to sit comfortably with their backs, necks, and shoulders straight so that their health is not a sacrifice to be made when using a PC.

References

[1] Thingiverse.com. "Platform Jack [Fully Assembled, No Supports] by Intentional3D." *Thingiverse*, www.thingiverse.com/thing:925556. Accessed 1 Mar. 2024.

[2] Thingiverse.com. "Motorized Platform Jack by Mitchamccaskill." *Thingiverse*, www.thingiverse.com/thing:6161381. Accessed 1 Mar. 2024.

[3] Singh, Riddhi Kumari. "Real-Time Pose Estimation from Video Using MediaPipe and Opencv in Python." *Medium*, Medium, 26 Apr. 2023, medium.com/@riddhisi238/real-time-pose-estimation-from-video-using-mediapipe-and-opencv-in-python-20f9f19c77a6.

[4] "InsigniaTM." *BestBuy.Com*, www.bestbuy.com/site/best-buy-brands/insignia/pcmcat159800050007.c?id=pcmcat159800050007. Accessed 1 Mar. 2024.

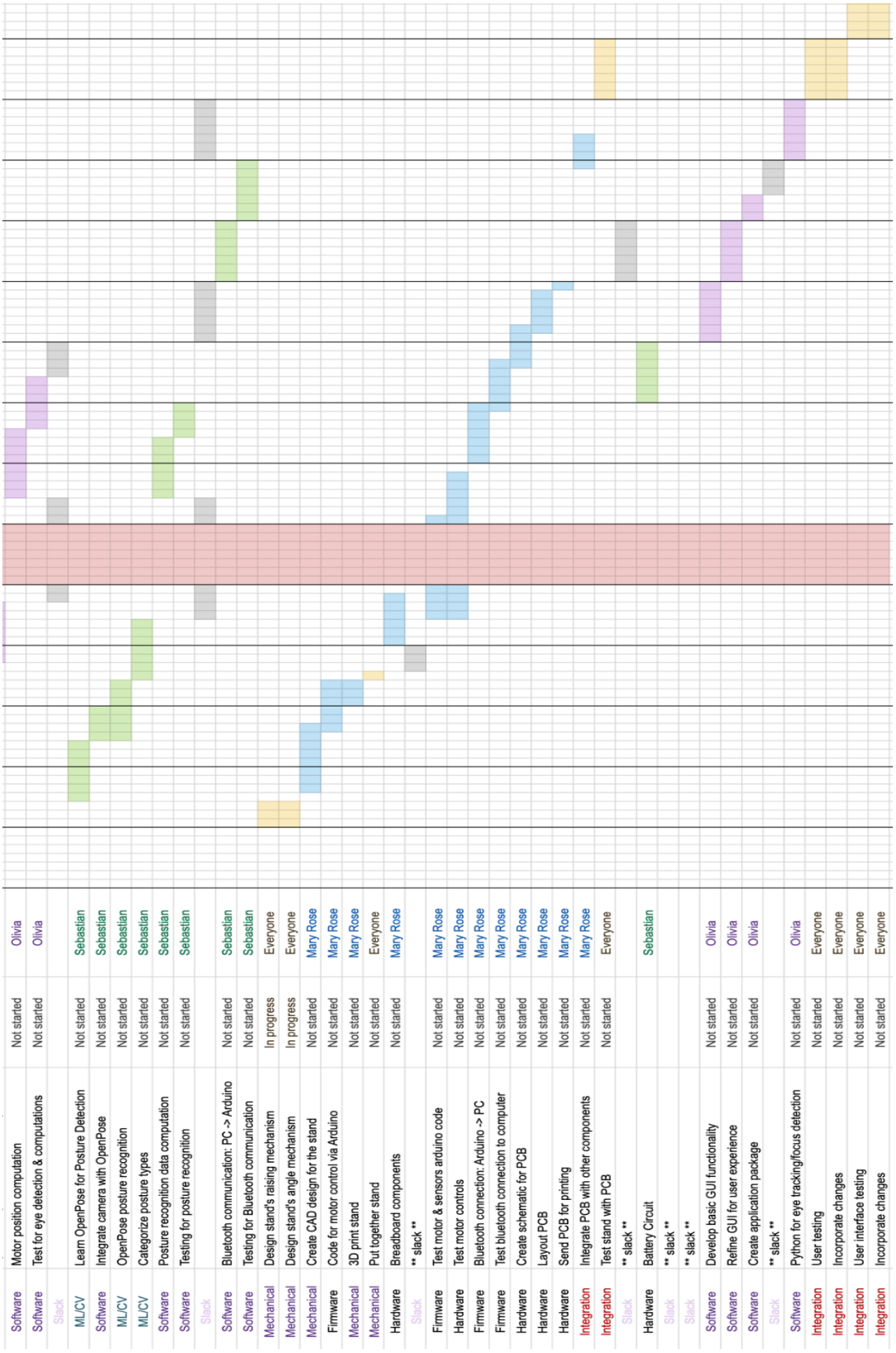

Figure 11: Gantt Chart Schedule

Item Description	Model	Manufacturer	Quantity	Cost
Platform jack	N/A	Amazing-us	1	27.99
L293D Motor Shield	03-01-0506	HiLetGo	1	8.99
Arduino Uno Rev4	ABX00087	Arduino	1	27.5
Linear Actuator	N/A	ECO LLC	2	$34.99
Motor	GA12-N20	Fielect	1	$9.99
Ultrasonic Distance Sensor	HC-SR04	Smraza	1	6.99
Total Cost				116.45

Figure 12: Bill of Materials