SmartStand

B2: Sebastian Garcia, Mary-Rose Rubino, Olivia Yang
18-500 Capstone Design, Spring 2024
Electrical and Computer Engineering Department
Carnegie Mellon University

Product Pitch

We are in a posture crisis. PCs are not kind to our shoulders and necks. There are computer stands on the market but adjusting them is slow and cumbersome. So why not use the intelligence of a computer to do the stand adjustment for us?

SmartStand uses your computer's webcam and facial tracking to intelligently set your computer to the right height and angle, not matter where you are. Let's use technology to improve our health.

System Architecture

The key to SmartStand is the webcam that tells the rest of the system how to behave. We pass webcam data into OpenCV which produces facial landmarks which feed into our stand adjustment algorithm. Webcam data is also used to track eye fatigue. The user can then select different stand adjustment options (automatic and manual) and receive health notifications through the GUI. On the hardware side, we have an Arduino that receives packets via a serial connection to the PC that contain motor inputs. The Arduino connects to two motor drivers to control the linear actuators and main motor.

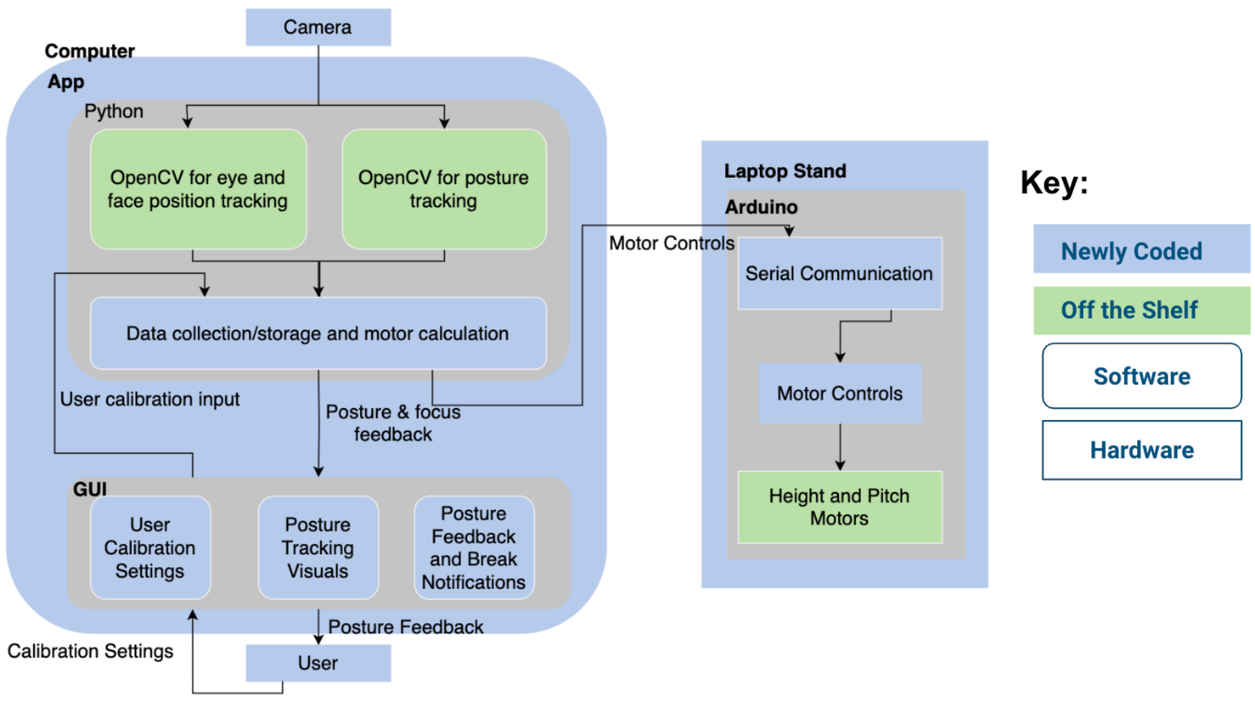

Figure 1. System Block Diagram

Conclusions & Additional Information

http://www.ece.cmu.edu/~ece500/projects/S24-teamxx

Overall, we are happy with the product we created. But there is still plenty to improve on. Design-wise, SmartStand could be lighter, smaller, more elegant.

On the application-side, we tried to make a posture detection model, but it was inaccurate because of a lack of data available to train it.

We ended up learning a great amount of mechanical design and engineering concepts when modifying the platform jack, and learned new fabrication processes like laser cutting.

System Description

Our system is comprised of software **(Figure 2),** hardware **(Figure 4),** and mechanical **(Figure 3)** components. The user will interface with the product via our GUI. On the backend, our algorithm will use OpenCV calculate the ideal stand position. This will be sent to the Arduino serially. On the python backend, the eye-tracking, and stretch reminders will be calculated and then displayed to the user as notifications.

Figure 2. GUI Homepage

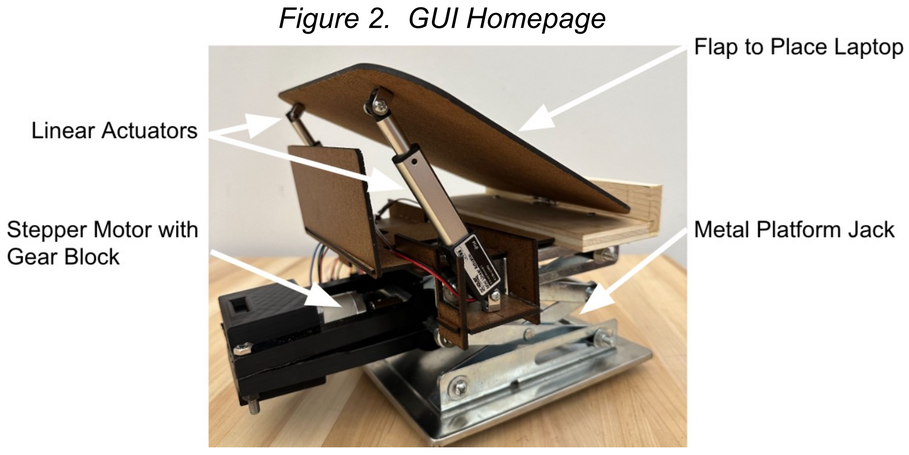

Figure 3. Mechanical View

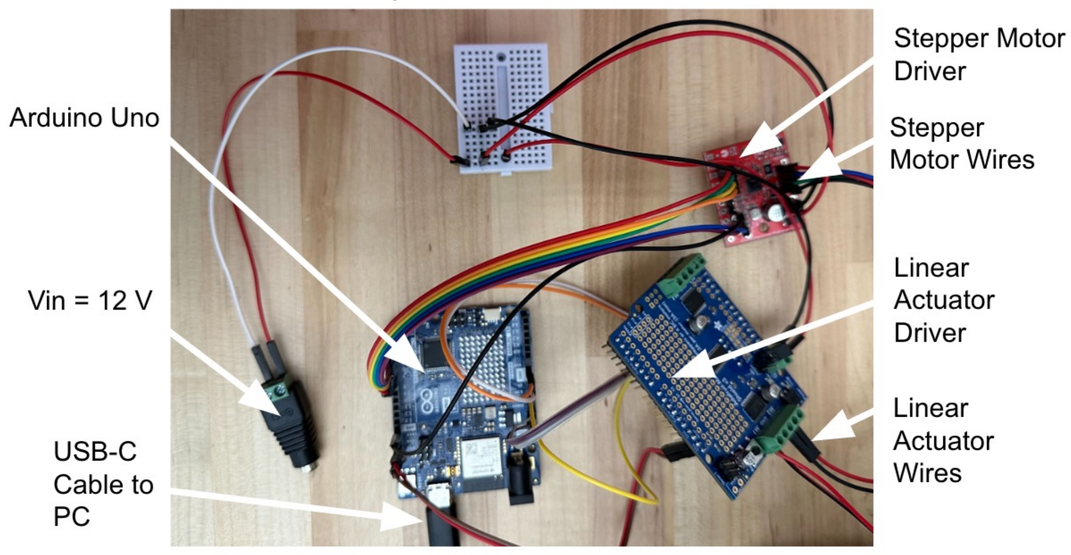

Figure 4. Circuit View

System Evaluation

We tested the system by measuring performance metrics related to the whole system which are most important to the user. The table to the right shows how we met these holistic quantitative use-case requirements. Some tradeoffs are shown in this data. We opted for a high torque motor with a gearbox for reliable lifting at the cost of speed. On the algorithm side, we aggressively sent motor commands to adjust quickly at the cost of accuracy.

Design Requirement	Target	Result
Total Adjustment Time	< 5 seconds	92.57 seconds
Height Error	< 3 inches	0.267 inches
Angle Error	< 5 degrees	4.33 degrees
Maximum Height Extension	>= 12 inches	12 inches
Maximum Angle Extension	>= 45 degrees	55 degrees

Table 1. Test Data Comparison to Requirements

	Benefits	Drawbacks
High Torque Gearboxes 51:1 VS. 14:1	Stand requires 4.4 Nm of torque to lift. 14:1 gearbox only provides 3.0 Nm. Higher torque = higher range of motion.	Using the gearbox means that the motor will achieve less rotations per second, slowing down calibration.
Linear Actuators placed below the top plate	Can use linear actuators with greater change in distance without increasing the starting angle.	More complex mechanical design that must be able to support the weight of a laptop.

Table 2. Design Tradeoffs