Ethan Owen
2/25/2024
EE113DW Status Report 6
Gesture Control for Home Devices Project - (solo project)
https://github.com/Ehanlion/HGR_project

Main Intention:
Week 7 was a critical development week, and one that paves the way to a project completed on time. With research completed in prior weeks, it was time this week then to perform actual code development and make some concrete progress. I focused on learning about and utilizing a new library named 'huesdk' which is the final piece of the puzzle for this project. This library allows my gesture processing code to remote-control devices in my home. I will use Phillips' Hue Smart Lighting system for this project since it is ubiquitous, easy to use, and has good python support via the huesdk package. Integrating the functionality of huesdk with the gesture processing I already had functioning is a critical step towards development. If I can integrate it properly, it means my project is all but completed.

Conceptual Development:
A big step in conceptual development this week was understanding how the huesdk package works and how I can integrate it with my project. Initially, I looked at other, more generic packages that could control other home devices, but I had spare Phillip Hue lights and since the package is quite simple to use, it felt like a good marriage of convenience. Not to say that huesdk isn't an optimal choice, far from it in fact.

The package has in-built functionality to search for and connect to a Hues Bridge. The Bridge is how the huesdk package communicates to all connected devices; the bridge essentially pairs to the devices, and then it's a simple matter of asking the bridge to perform an action with a connected device. The package makes addressing, listing, and controlling devices extremely convenient and efficient which is why it's such an excellent choice for this project. In code, you can simply search for bridges and connect to a Hue Bridge via its static IP address. Then you can list all connected devices on the bridge and address them by their port on the bridge.

A future consideration could be using a more wide-sweeping library. Huesdk only allows for connection to a *hue* bridge and not a generic one. Controlling just a Hue Bridge is a good proof-of-concept step although for a more complete project, it would be a reasonable step to find a way to connect to any bridge instead.

Implementation and Development:
This week, I created a test program using the huesdk package. It searches for a bridge and then connects to it via its IP address. Then I can access a device connected to it as if I were accessing an element of an array. At the current time, I do not know if the program actually works yet since I am waiting on parts to ship. Within the next few days I will know my answer however and then I can make adjustments to the program to get the code working.

Resources:
I continued to use online resources from both OpenCV and MediaPipe to advise me on the project. Now though, I have added resources directly from RaspberryPi's website to help me with the RPI design, interaction, and configuration. Furthermore, I have added Huesdk to the list of packages being used by the project. The documentation for huesdk has great tutorials on setting up test programs and code to begin integrating the packages with a project so I have been using those as guidance.

Satisfaction:
By the end of week 7, my original project guidelines had established that I would train a new AI model using a website called Kaggle to perform the gesture recognition. However this step was originally added assuming that my first implementation of the project *would not use an AI model.* Under actual development though, I *did* use an AI model so this step is a little bit nonsensical. Also by this point my project would have been integrated and tested on the RPI5 that I planned on using. Since it is still shipping, I have not completed this step, but I have performed testing with an older and less powerful board to a successful degree. The gesture recognizing function already works great and now that I have a library to control device connections (huesdk), I am very confident my project is on track to finish on time or even ahead of time.

Individual Contribution:
This project is an individual project and as such, all development was undertaken by me.

References:

1. https://github.com/googlesamples/mediapipe/blob/main/examples/gesture_recognizer/python/gesture_recognizer.ipynb - Website for MediaPipe with an installation guide as well as a python example.
2. https://docs.python.org/3/library/asyncio.html - Website for Asyncio library
3. https://developers.google.com/mediapipe/solutions/vision/gesture_recognizer/python#live-stream - Website for MediaPipe with a python implementation of a hand tracking overlay.
4. https://docs.opencv.org/3.4/dd/d43/tutorial_py_video_display.html - Website for OpenCV's example on getting as video feed functioning
5. https://store-usa.arduino.cc/collections/iot-cloud-compatible?selectedStore=us - Website for Arduino's cloud computing boards
6. https://www.raspberrypi.com/news/introducing-raspberry-pi-5/ - Website for Raspberry Pi's introduction of the RPI5
7. https://forums.raspberrypi.com/viewtopic.php?t=229888 - Forums posting on Lite and non-gui variants of the RPI OS called Raspbian Lite.
8. https://www.raspberrypi.com/documentation/computers/remote-access.html - Website detailing how to setup and RPI over a network
9. https://pypi.org/project/huesdk/ - Website for Huesdk with tutorials