Project Title: **Vision++**

Team: *Enrique Flores Medina, Tianyi Liu*

EECS 149/249A Milestone 2, Fall, 2022

Progress:

- Enrique: PCB design has been manufactured, received, and mostly completely assembled. Only capacitors are missing.

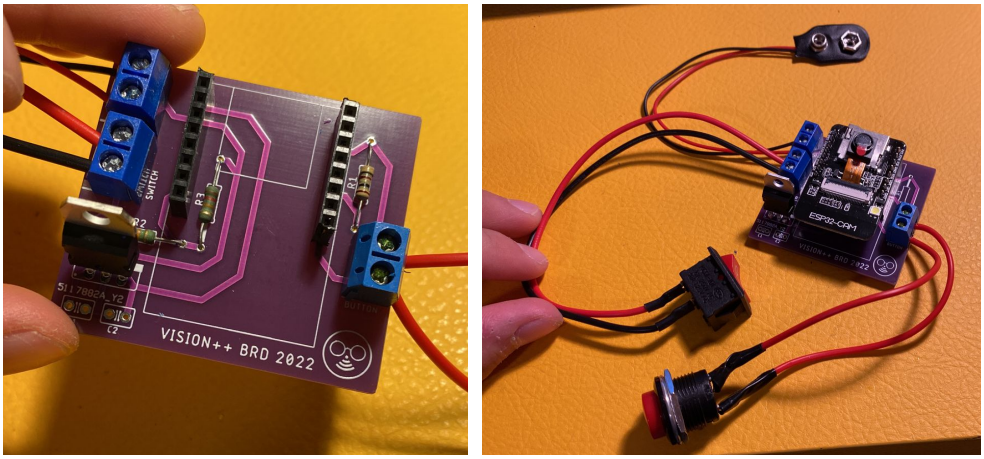


Figure 1. Assembled PCB, with and without ESP32-CAM board

- Enrique and Tianyi: Finalized code to upload images taken by the ESP32-CAM into a webserver through wifi (see resources for source code)

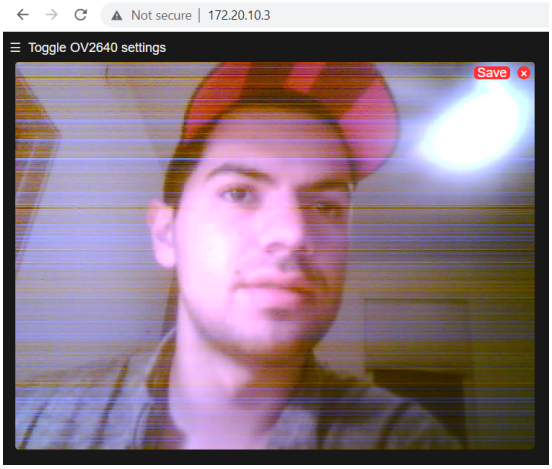
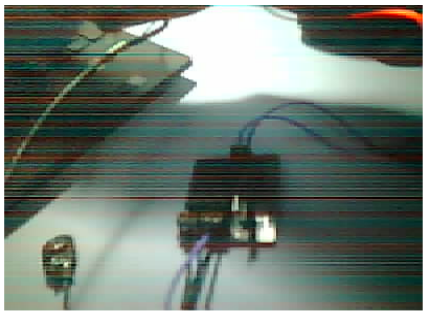


Figure 2. Web Server to upload images taken from ESP32-CAM with wifi

- Tianyi and Enrique: Finalized MATLAB script to extract the text from the image using the OCR function. A picture from the ESP32-CAM is taken every time a “Y” or “y” is pressed on the keyboard. (Eventually, a button input will replace the keyboard input).


```
Read picture? (Y/N): y
Recognized text:


UAW STRIKE;
```



```
Read picture? (Y/N): y
Recognized text:
No text
```

Figure 3. MATLAB's OCR function results for pictures taken with the ESP32-CAM

Goals:

The goals for the following weeks, as detailed in the schedule, are (1) to modify the system input from being the computer keyboard to a physical button connected to the glove, (2) replacing the embedded camera to increase the quality, since the one that was included in the ESP32-CAM board was accidentally damaged during the testing process, (3) uploading the images to a cloud-based storage platform, such as Google Drive, to avoid having to connect the computer to the smartphone's hotspot, which significantly reduces the portability of the device, and, finally, (4) transferring the recognized text from MATLAB into a smartphone application that will, in turn, convert it into speech.

Class Topics:

State machines, since the behavior of our system is governed by a state machine shown in the GitHub repository. Wireless networks, since we utilize wifi to send images from a development board to a cloud server that is accessible by third party entities, which extract and speak out the text from said images. Sensors, since we have to communicate through I2C to the embedded camera, and we have to read a GPIO button input to trigger this camera.

Resources:

We have used the following libraries for our ESP32 development board, using an arduino IDE:

1. To build a web server and upload the images:
    a. https://github.com/me-no-dev/ESPAsyncWebServer
    b. https://github.com/me-no-dev/AsyncTCP

2. Object Oriented API to use OV2640 camera on ESP32 microcontroller:
    a. https://github.com/yoursunny/esp32cam
3. To take, process, and send the images through wifi
    a. WiFi.h, Esp_camera.h, Esp_timer.h, and Img_converters.h

MATLAB's OCR function to extract text: https://www.mathworks.com/help/vision/ref/ocr.html

The pinout reference for the ESP32 chip, which the ESP32-CAM development board uses: https://randomnerdtutorials.com/esp32-pinout-reference-gpios/

Smartphone app will be developed using App Inventor: https://appinventor.mit.edu/

Our parts and mechanical assembly tools:

1. 9V battery, and clip
2. L7805CV Voltage regulator 5V
3. ESP32-CAM development board with FTDI programmer
4. Momentary push button
5. Male and female pin headers
6. Three 1kΩ resistors
7. Three 5mm 2 pin screw terminals
8. Switch to turn power on and off
9. One 0.1 uF and one 0.33 uF capacitors for supply voltage filtering
10. Polymer glue (to attach the PCB to the enclosure)
11. Velcro tape (to attach enclosure to glove)
12. Gym or exercise glove

Updated schedule: .

- November 23: MATLAB script that detects any text on a picture uploaded to Google Drive (see GitHub repository). Assemble the majority of the components on the PCB.
- November 25: Source code that uploads images from ESP32 - CAM to Google Drive. Finalize assembly of PCB with remaining components.
- November 28 - December 2: Milestone 2 Meetings. Finish and 3D print enclosure. Enable button from enclosure to take pictures only when desired by the user.
- December 7: Smartphone app to read the text files that the MATLAB script outputs and turn it into speech. Finalize details on the rest of the project.
- December 15: Project Poster/Demo Expo
- December 16: Peer Evaluations, Project report

Risks:

There are many unknowns. First, by using a low cost embedded camera, we are losing potential image quality and, therefore, reducing the effectiveness of our system to detect text the user finds in the real world. Since there is a strike going on, obtaining the hardware necessary to finalize the project is practically impossible, which means we may have to buy the missing components with our personal

money. Finally, there is no available assistance and guidance from our mentor GSI, which also results in a slower pace for the fulfillment of the project goals.

GitHub repository:

https://github.com/tianyi-liu-fr/project_ee149_ucBERKELEY