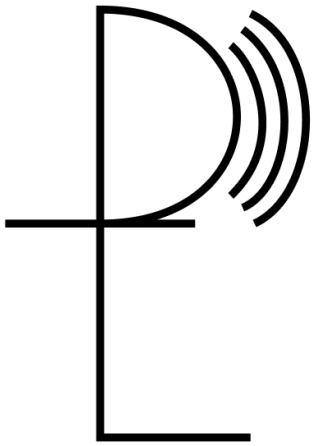
Boston University
Electrical & Computer Engineering

EC 463 Senior Design Project

First Prototype Testing Plan

Portable Language Translator



By

Team 23

Portable Language Translator

Team Members

Yohan Kim yohank@bu.edu
Ryan Liao ryanliao@bu.edu
Cristian Palencia cris0211@bu.edu
Andrew Nguyen aynguyen@bu.edu

Required Materials:

Software:

- Language Translation
    - Google Cloud Platform project with text-to-speech, speech-to-text, translate, and storage API's
    - Python3 Packages:
        - Numpy
        - Sounddevice
        - Google-cloud-speech
        - Google-cloud-texttospeech
        - Google-cloud-translate
        - Webrtcvad
        - Pydub
        - Pynput
        - Flask
        - Flask-cors
- ASL Detection Model
    - Python3 Packages
        - OpenCV
        - Tensorflow
        - Mediapipe
        - Google-generativeai
- User Interface
    - Python3 Packages
        - PyQt5

- Firmware
    - Python3 Packages
        - Adafruit_ssd1306
        - PIL
        - Pygame
        - Sounddevice
        - Numpy
        - scipy.io.wavfile

Hardware:

- Raspberry Pi 3
- I2S MEMS Adafruit Microphone
- USB 2.0 Speaker

- OLED Display (ssd1309 driver)
- Microusb Power Supply

Set Up:

For the language translation, a react web app is used to control the base language and voice settings of the program, which it sends using a flask api. One device is used for the conversation. The users must wait for the translation and playback to fully finish before beginning the next phrase. The program captures audio, splits it into chunks and uses voice activity detection software to determine if someone is speaking. Once it detects no speech, it sends the audio file to the google cloud for the transcription, translation, and finally text-to-speech. The audio file is sent back to the program which it plays from the system's speaker.

For the ASL detection, the external camera (USB 2.0 Camera) will record a live video stream and the script will initialize the video feed. Mediapipe Hands will detect the hand in the video feed and locate 21 key landmarks along the user's hand which are then normalized and prepared for classification. The keypoints are passed into the KeyPointClassifier, which is a pre-trained ML model that matches the keypoints to an ASL alphabet gesture. Each frame is checked to identify the letter being signed and the predicted letter is displayed on the laptop screen. If the signed label is held stable for .75 seconds then the detected letter will be put into the detected string. The user will consecutively sign letters until their word is complete. Once 2 seconds have passed, the detected string will be updated into the word list and the user may continue signing their next word if needed. Once the user has stopped signing their sentence for 3 seconds, the finalized word list/sentence is moved into a txt file.

For the hardware setup we have a Raspberry Pi that interfaces with all the relevant components. For one, we use the GPIO pins to connect to the SPI interface for the OLED display. The OLED display interfaces with the adafruit_ssd1309 library to send over the appropriate signals over the GPIOs and uses PIL to allow for relevant drawing capabilities to the OLED. Additionally, we have our I2S MEMS microphone also connected to the Pi's GPIO pins. The microphone directly interfaces with the sounddevice library which handles reading from a connected I2S audio device such that we can generate a .wav file recording. Finally, we have a USB 2.0 Speaker that, using the pygame library, can output the audio recorded by the microphone, or any provided .wav file.

Pre-Testing Setup Procedure:

Initial Setup:

1. Install dependencies using: "pip install -r requirements.txt"

For Language Translation:

1. Run the react app using: "npm run dev"
2. Run the script using: "python \main.py"

For ASL Translation:

1. Run ASL detect script using "python detect.py"

For Raspberry Pi Hardware Test:

1. First enable the following interfaces on the Raspberry Pi
    a. SSH: allows remote connection to the Pi such that we can execute commands from an alternative terminal
    b. SPI: allows communication to the OLED display
    c. I2S: allows the use of the microphone
2. With SSH enabled, we can connect to the Raspberry Pi using plt@<ip_addr>
3. Once we are connected we can navigate to the directory with our fw code.
4. Once we are here we can activate our virtual environment (contains all of our dependencies) using source ~/Documents/ec463/sd/fw/venv/bin/source
5. Run python3 hardware_test.py to run the test that integrates all relevant hardware

Testing Procedure:

Language Translation:

1. Run the scripts
2. Set the base language, voice gender and type on web app and press confirm to send settings. This will change what language the program translates to and what voice it uses for the translation.
3. Have a conversation switching between english, spanish, and/or korean
    a. English - Spanish script: (male)
        i. ¿Hola, como estas? (Hi, how are you?)
        ii. I am very good thank you! How was your day?
        iii. ¡Mi dia a sido bueno! ¿Que comiste hoy? (My day has been good! What did you eat today?)
        iv. I ate a hamburger and drank water
    b. English - Korean script: (male)
        i. 병원 어디 있어? (Where is the hospital?)
        ii. It is just down the street
        iii. Do you need help getting there?
        iv. 병원 얼마나 멀어요? (How far is your hospital?)
        v. It should take you 15 minutes to walk there
    c. Spanish - Korean script: (female)
        i. ¿Hola, como estas? (Hi, how are you?)
        ii. 나는 오늘 좋았어 고마워 너는 어땠어 (I am good, thank you! How was your day?)
        iii. ¡Mi dia a sido bueno! (My day has been good!)
        iv. ¿Que comiste hoy? (What did you eat today?)
        v. 나는 햄버거를 먹고 물을 마셨어 (I ate a hamburger and drank water)

ASL Translation:

1. Run the script
2. See if the model is capable of accurately translating ASL letters into phrases (ASL to English).
3. Test the following phrases:
    a. Hello my name is Yohan
    b. The car is red

4. Translated sentences will be stored in a txt file that will be used for audio output.

Hardware Test:

1. Run the script off of the Raspberry Pi
2. Start recording audio from the microphone
    a. OLED should display that the test has started and that the recording has started
    b. OLED will also notify us once the recording has stopped (5 seconds)
3. Save to audio file (output.wav)
    a. Once again OLED display will notify us of this
4. Finally, output .wav file to the usb speaker
    a. Final OLED display notification

Measurable Criteria:

Language Translation:

- React app should successfully run and have options for base settings
- Settings should accurately change the voice and base language of the program
- User should be able to have a bilingual conversation between two parties relatively seamlessly without manual input
- Speech to Speech latency is under 5 seconds

ASL Translation:

- ML model should be able to detect ASL letters using the camera module, not laptop camera.
- Model should be able to detect individual letters and create a sentence in English.
- Model should be able to translate ASL with at least 90% accuracy.

Hardware Test:

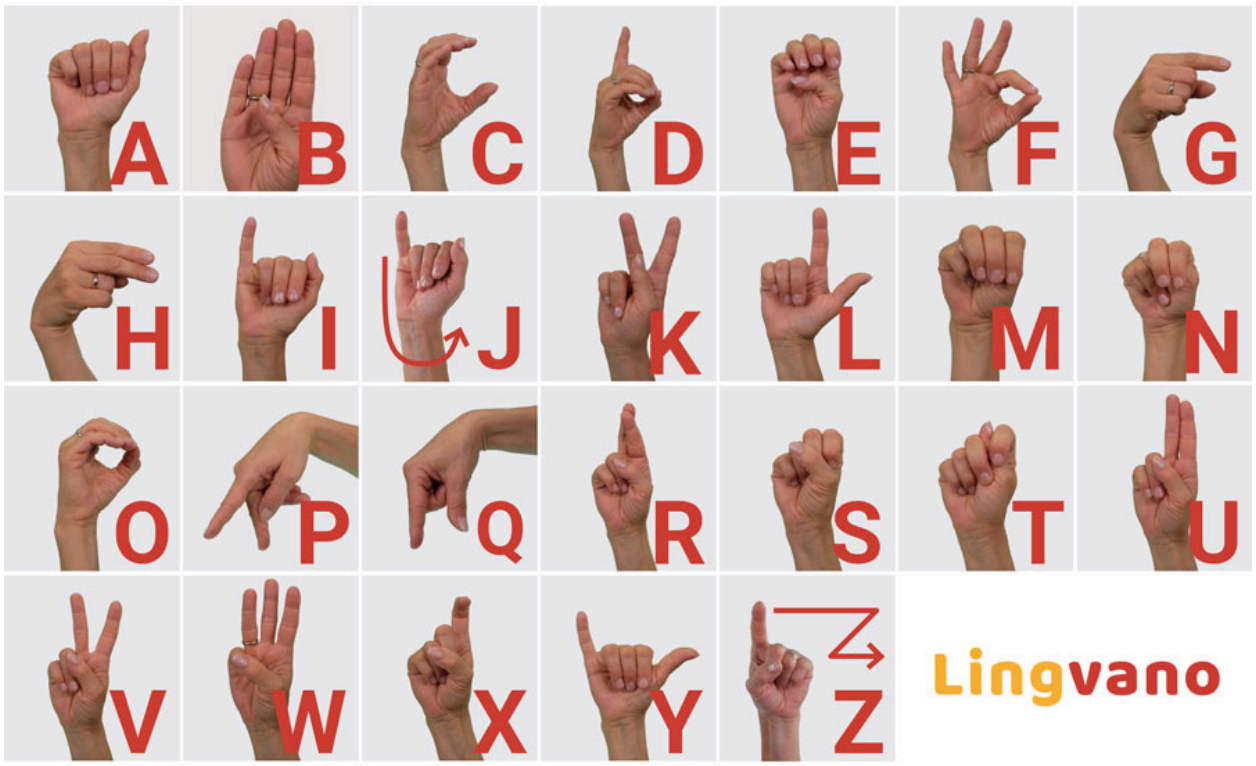
- Microphone
    - We should be able to see that the microphone is able to record quality audio.
    - Less dependent on capable distance for prototype since we can amplify the signal as needed later on
- Speaker
    - Verify microphone capabilities with a USB 2.0 Speaker. Additionally, reference point for using other USB speakers in the future parts of the project
- OLED
    - Should be able to be written to at a fast speed
    - Readable text

Score Sheet:

Test	Correct?
English - Spanish conversation	
Transitions to next conversation w/o manual input	
English - Korean conversation	
Able to translate same language in succession	
Spanish - Korean conversation	
ASL - Hello my name is Yohan	
ASL - The car is red	

Hardware Pinout

Pi3 Pin #	Usage / Description
1	3.3V Power to Microphone
2	5V Power to OLED
6/9	GND
12	GPIO1 → BCLK for Microphone
18	GPIO24 → DC for OLED
19	GPIO12 → MOSI for OLED
22	GPIO25 → RST for OLED
23	GPIO14 → CLK for OLED
24	GPIO8 → CS for OLED
35	GPIO24 → LRCL for Microphone
38	GPIO28 → DOUT for Microphone


A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
Q
R
S
T
U
V
W
X
Y
Z
Lingvano