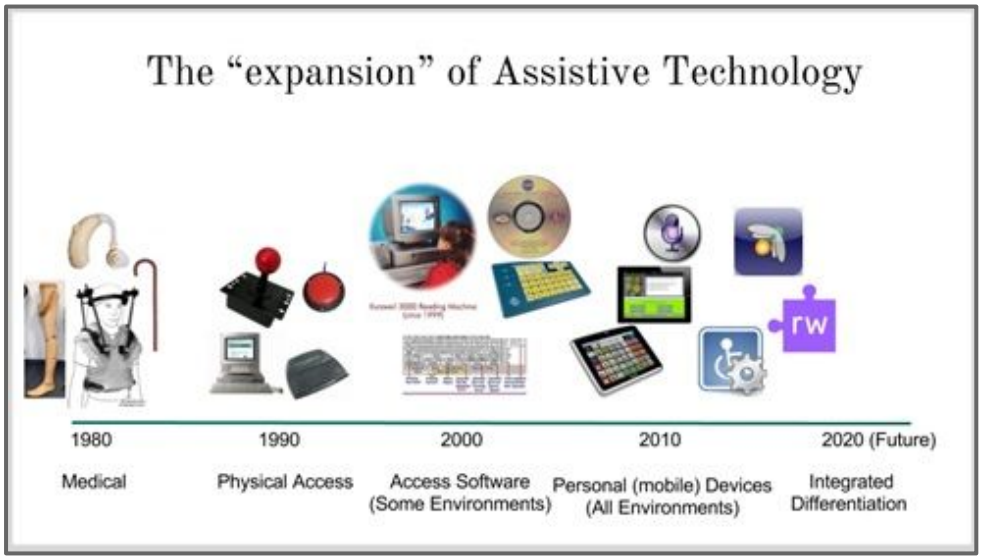
SignText

Kelvin Basdeo, Kat Chonka, Steni Stephan

CSCI 499: Capstone
February 26, 2020

Introduction

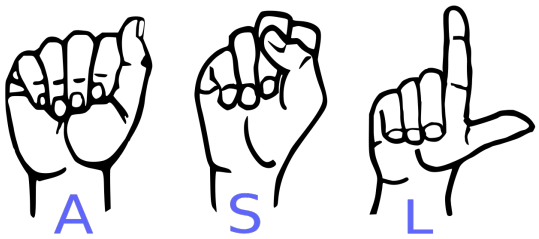
Assistive technology for the Deaf & Hard of Hearing has evolved dramatically over time:



- However, integrated technology is still in the process of development.
- Currently, most tools are:
  - Speech ➔ Text
  - Text ➔ Speech
  - Text ➔ Text
- **Deaf & Hard of Hearing people need to be able to interface with technology in a more *native language* (ASL).**

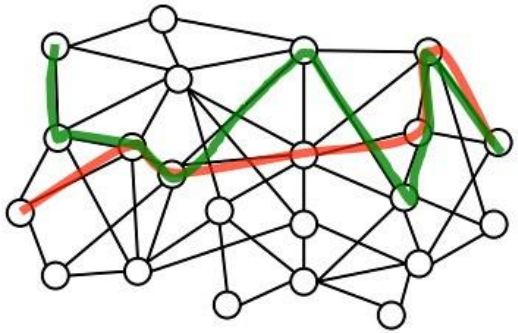
Our Vision:

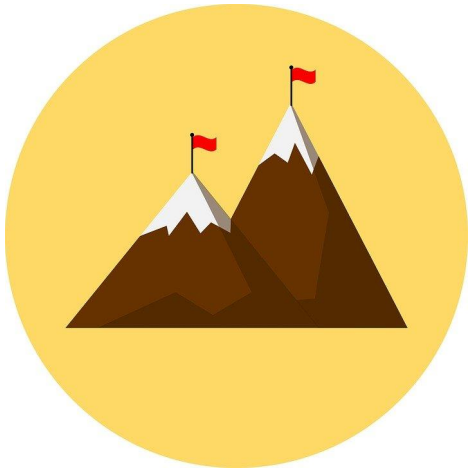
- A web application that translates American Sign Language (ASL) into the written English alphabet, built from a machine learning model of our creation.
- Our application would:
  - Give DDH people another means of communication.
  - Assist with education of the written English language.
  - Bridge the language gap between hearing and non-hearing peoples.
  - **Provide a cheaper & more <u>accessible</u> resource.**



Goals

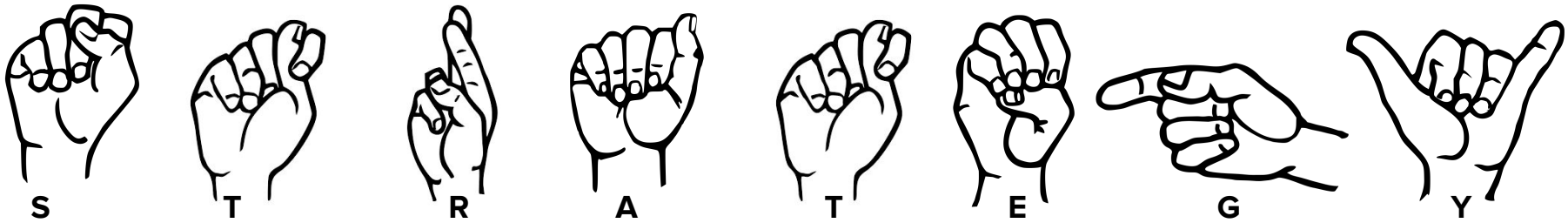
- Creating a fully-functional deliverable.
- Aiming for our recognition algorithm to successfully recognize signs.
- Success measured by:
  - Functionality
  - Accuracy
  - Speed of translation



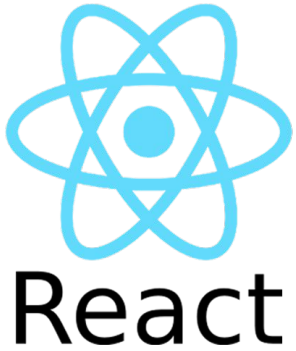


Strategy: How will we do this?

- **Within Scope:**
  - Developing a web app product.
  - Designing a clear, easy-to-use interface.
  - Utilize machine learning to implement an algorithm that will successfully recognize the ASL alphabet.
- **Outside of Scope:**
  - Expanding the algorithm to recognize words/symbols along with alphabet.
  - Creating portable versions of the product for hand-held devices.

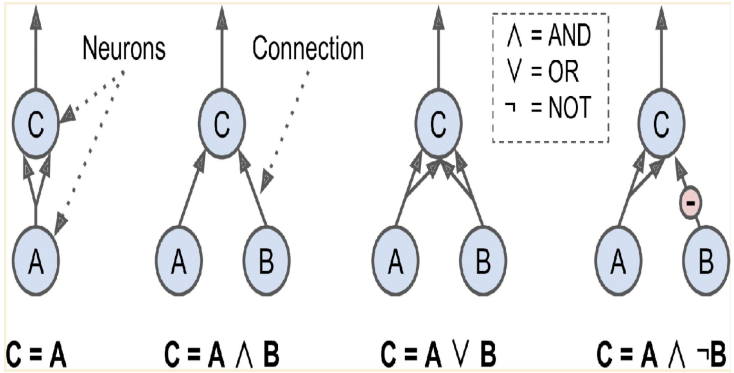


Technologies:



TensorFlow

- Open-source machine learning library using data flow graphs to build models.
- Allows developers to create large-scale neural networks
- Keras:
  - High-level neural network TensorFlow API used for building, training, evaluating, and running neural networks.
  - Will be used to develop a neural network that can accurately perform image classification.

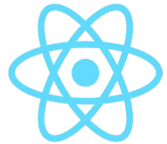


- Neural Networks:
  - Based off of biological neurons, they are composed of “cells” that can have multiple inputs with only one output. These simple neurons get combined to perform complex computations.

Front-End

Framework

- **React**: A JavaScript library for building User Interfaces
  - **Components** are the building blocks of React
- **Bootstrap**: Library for UI components and styling
  - Custom HTML and CSS components



Features


Video Capture


Recording Media

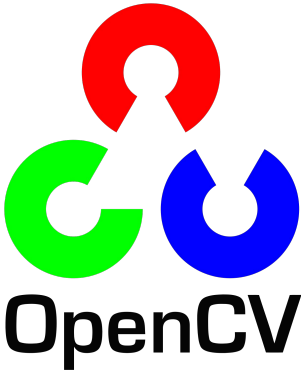
File Access


Text Automation

Translator

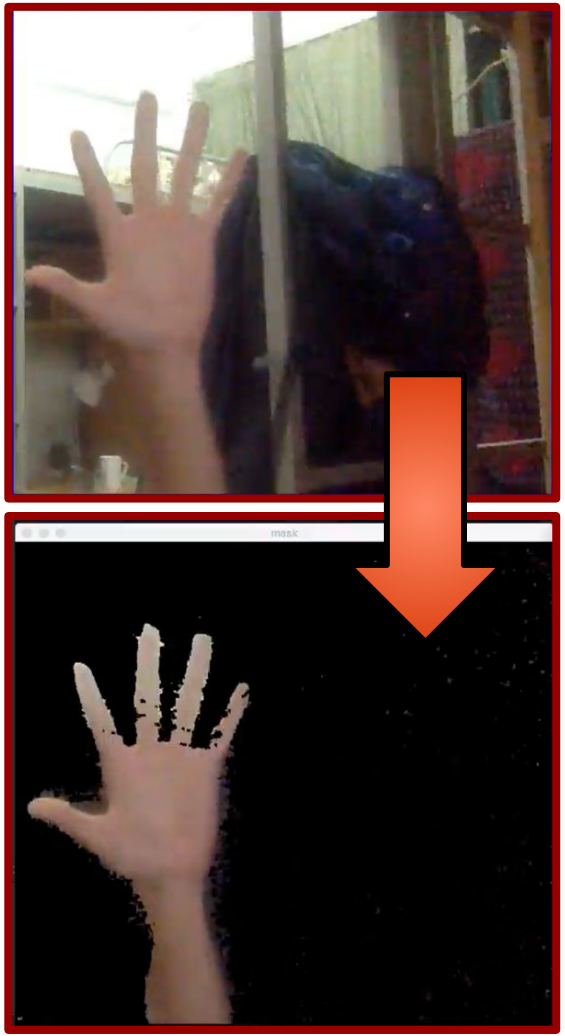
OpenCV

- “Open Source Computer Vision”
- Image Processing library
- Can be imported for Python
- Used to *capture & process* video data:
  - Read & write images
  - Finger detection & tracking
  - Differentiating a hand from the background
  - Frame-by-frame video extraction



Hand Isolation

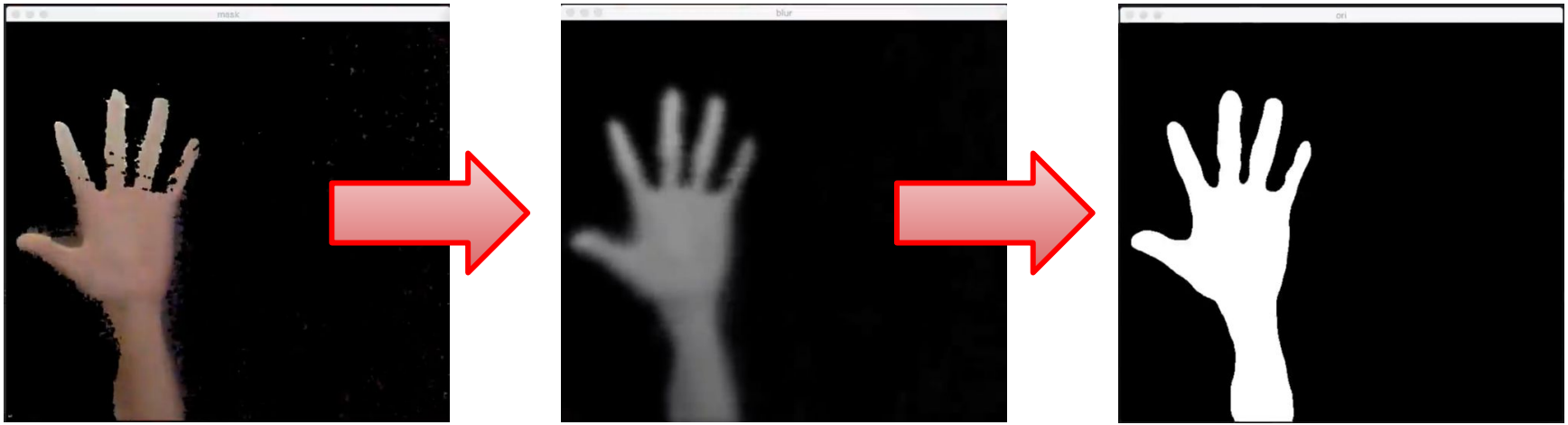
1. Extract a frame using OpenCV.
2. Use OpenCV's built-in **background subtraction** methods to subtract the background & create a model for doing so.
3. After applying the model to your frame, you will get the hand image with the background subtracted.



Hand Isolation

4. Convert the image to grayscale & apply blurring to reduce the edges.

5. Create binary images from grayscale images using thresholding.

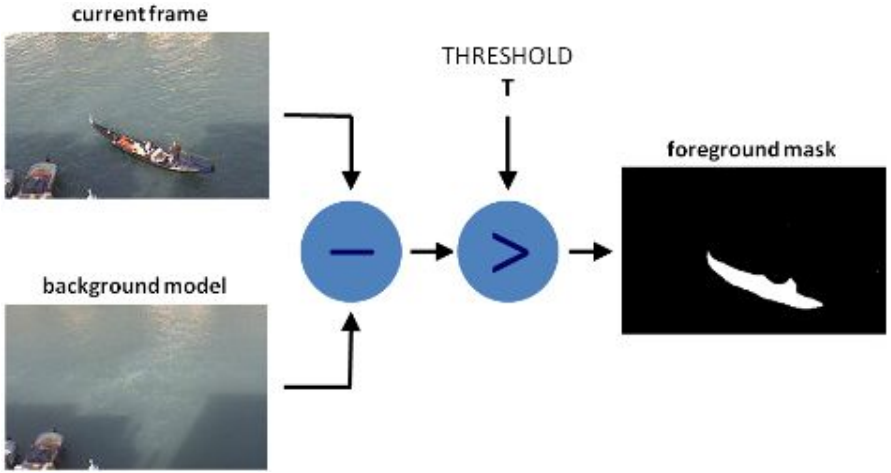


Step 3 **Step 4** **Step 5**

Hand Isolation

Background Subtraction:

- Performs a subtraction between the current frame and the background model
  - Background model : contains the “static” part
  - (relevant to the rest of the scene)

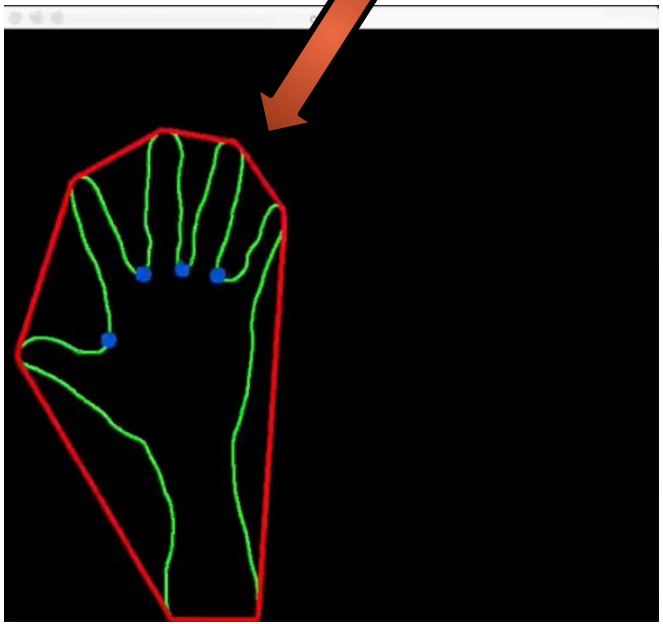


- Happens in 2 steps:
  - Background Initialization : An initial model of the background is computed
  - Background Update : Background is updated to adapt to possible changes in the scene

Detection

- **Contours**: curves joining all the continuous points along the boundary of an image that have the same intensity
- Used for shape analysis and object detection.
- OpenCV has built-in contour finding functions:
  - findContours
  - convexHull - finds the outline of the image
  - convexityDefects
- Accuracy is best with binary images.
- This is how we will grab user input and run it on our own model to *then recognize* the ASL alphabet.

Convex Hull: The enclosure of the shape [red line] - using the smallest geometric polygon



Mock GUI:

Isolation of the hands:
Will create a rectangle on the screen for users to place their hand

Webcam Scan

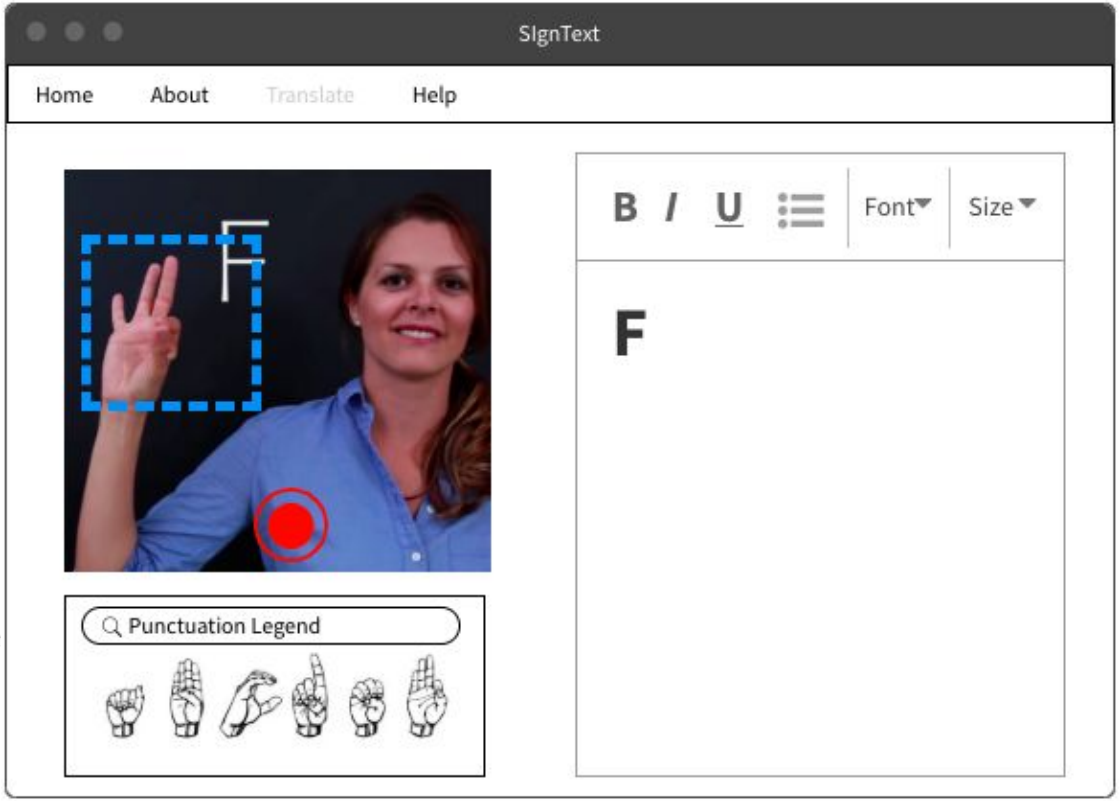
Punctuation Legend

Translation

Example: Our project pitch is an ASL to text converter.

Simple, easy to use UI

Mock GUI: Example



Conclusion

- Overall, we wish to build this ASL ➔ English translator to:
  - Make technology more accessible
  - Bridge communication barriers
  - Allow Deaf & Hard of Hearing people to communicate in a more native language
- As a team, we hope to learn more about the different technologies involved and create a fully-functional deliverable.

Thank you!

Questions?