Facial Recognition System

Shiv Radhakrishnan & Benjamin Hughes

Northeastern EECE 2140

Introduction

Objectives and Goals:

- Implement OpenCVs LBPH Algorithm
- Create a program that can distinct between multiple faces
- Integrate this program to create a 2-step security system

Personal goals

- Gain more comfortability in python
- Understand facial recognition algorithms

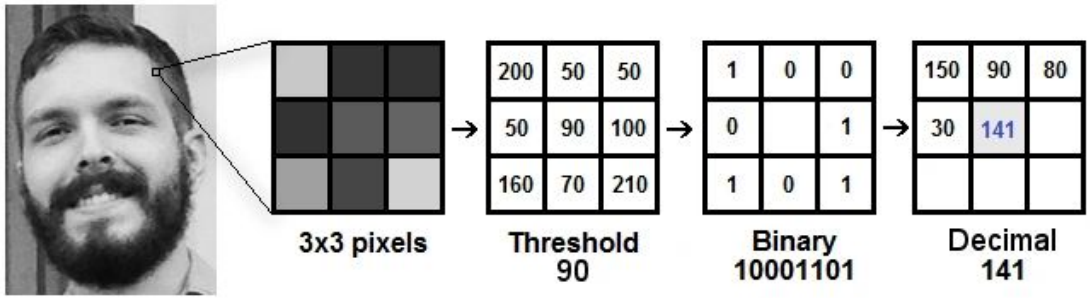
Project Scope

- Deliverables
  - Facial Recognition Program
  - 2-Step Security System
- Constraints
  - Time
  - OpenCV Models

Literature Review

Local Binary Pattern Histogram Algorithm

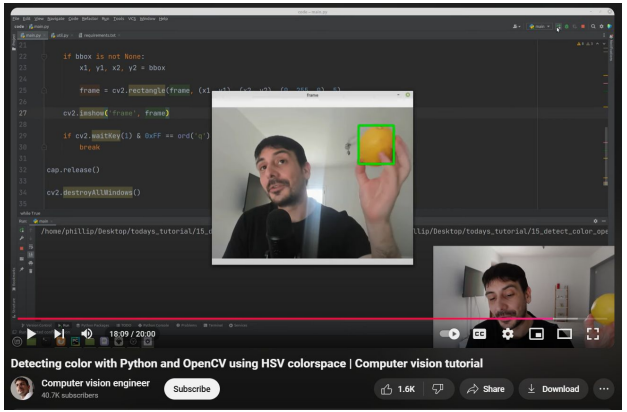
- How OpenCV Facial Recognition library works
- Divides the image into regions
- Looks at each pixel in region and compares it to the center pixel and then assigns a binary number
- This makes an image that can be divided into smaller regions, and for each region, a histogram is calculated that represents texture features
- For our case, the histograms from given image are compared to the webcam and measures similarity and recognizes the face from there.



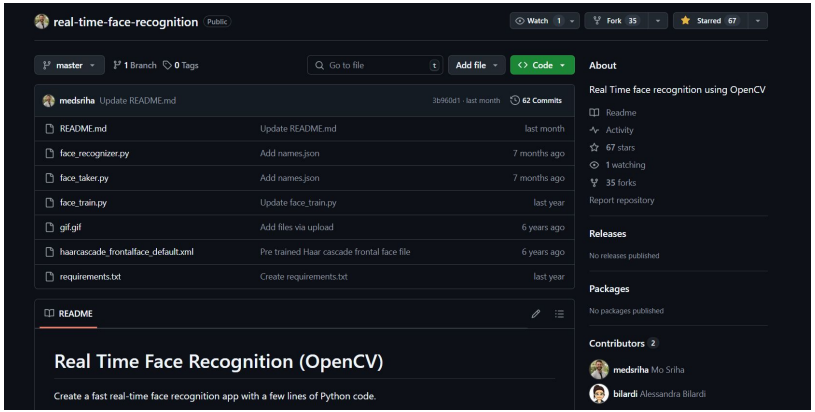
Parth. “Understanding Face Recognition Using LBPH Algorithm.” *Analytics Vidhya*, 21 Oct. 2024, www.analyticsvidhya.com/blog/2021/07/understanding-face-recognition-using-lbph-algorithm/.

Literature Review/Background

References for project



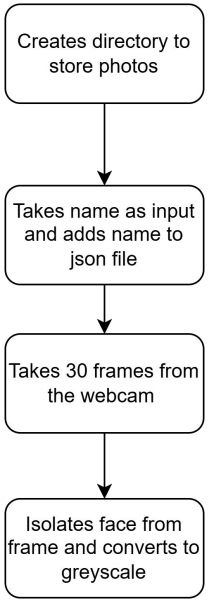
Computer Vision Engineer. “Detecting Color with Python and OpenCV Using HSV Colorspace | Computer Vision Tutorial.” *YouTube*, YouTube, 27 Oct. 2022, www.youtube.com/watch?v=aFNDh5k3SjU.

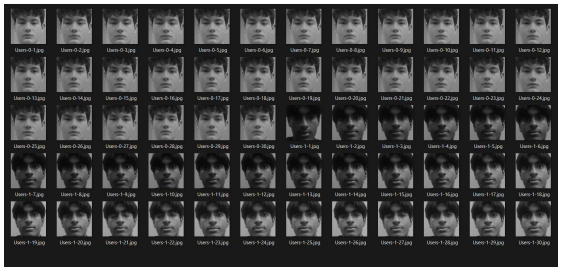


Medsriha. “Medsriha/Real-Time-Face-Recognition: Real Time Face Recognition Using Opencv.” *GitHub*, github.com/medsriha/real-time-face-recognition. Accessed 17 Nov. 2024.

Methodology

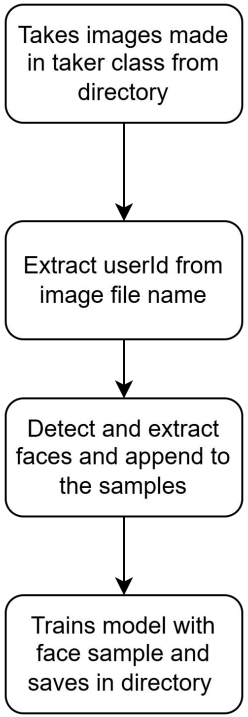
Taker Class



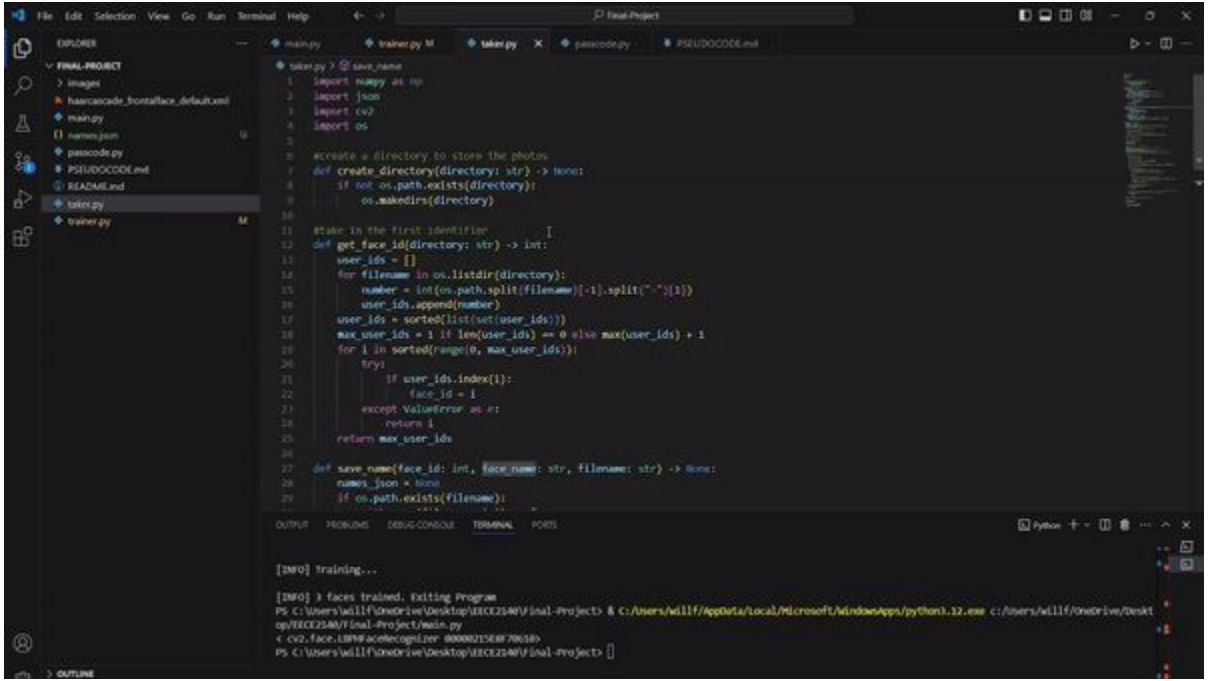


Taker class output

Trainer Class



Taker and Trainer class examples



Methodology

Main Class

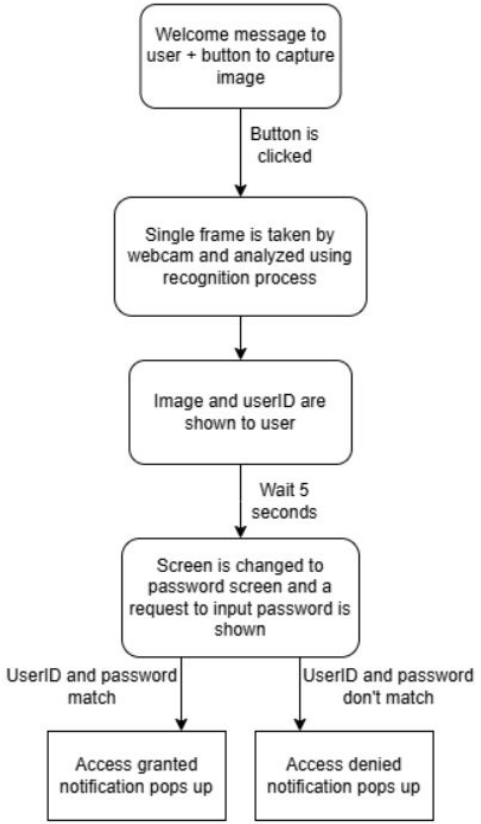
UI

- Tkinter (Python GUI toolkit)

Facial recognition

- Takes 1 frame from webcam
- Perforates image for use in our system
- Compares frame using LBPH recognizer
- Returns user ID with highest confidence value

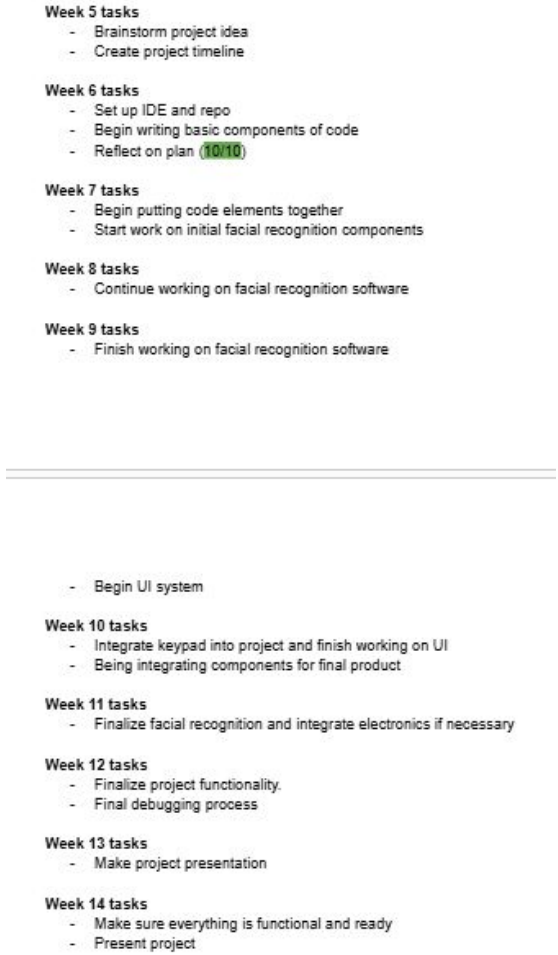
```
# Predict the user ID and confidence
user_id, confidence = recognizer.predict(gray_image[y:y+h, x:x+w])
```



UI Process flowchart

Results

- Generally, followed this outline for our project
- Had to redefine our goals during the project
  - Project would need work to implement into real life, but is fundamentally there



Week 5 tasks
- Brainstorm project idea
- Create project timeline

Week 6 tasks
- Set up IDE and repo
- Begin writing basic components of code
- Reflect on plan (10/10)

Week 7 tasks
- Begin putting code elements together
- Start work on initial facial recognition components

Week 8 tasks
- Continue working on facial recognition software

Week 9 tasks
- Finish working on facial recognition software

- Begin UI system

Week 10 tasks
- Integrate keypad into project and finish working on UI
- Being integrating components for final product

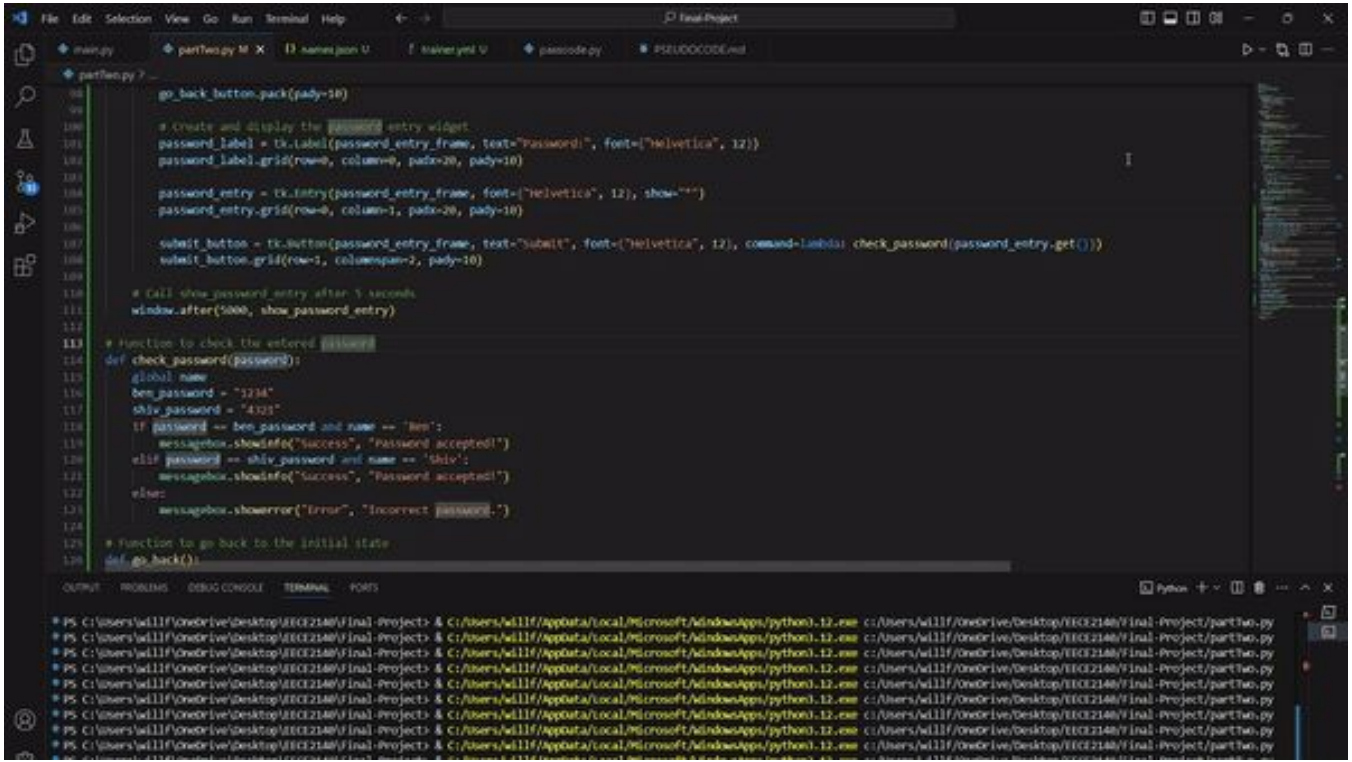
Week 11 tasks
- Finalize facial recognition and integrate electronics if necessary

Week 12 tasks
- Finalize project functionality.
- Final debugging process

Week 13 tasks
- Make project presentation

Week 14 tasks
- Make sure everything is functional and ready
- Present project

Analysis



- Program runs how we wanted it to
- Because it calls to take in a single frame from webcam, it takes a while to connect and “capture” the frame
- GUI is bare, could use a different library

Discussion

Implications of Findings

- Adds multiple layers to security
- Applicable to many real life scenarios (e.g FaceID)
- Can be applied in a much larger scale

Limitations

- The algorithm isn’t super accurate when not given enough samples
- Not having enough processing power to take more samples to attain an more accurate program
- Not having enough time to flesh out the 2-step security; looks simple.

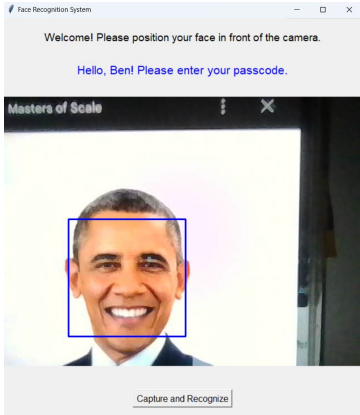
Conclusion

Conclusion

- Able to develop a Facial Recognition program using OpenCV
- Implemented it into create a 2-step security system
- Limited by technology and time

Recommendations for Future Work

- still use OpenCV and their library as LBPH is quite easy to understand and work with
- Find a way to make algorithm more accurate
  - Take more samples?
- Use different GUI library



References

Computer Vision Engineer. “Detecting Color with Python and OpenCV Using HSV Colorspace | Computer Vision Tutorial.” *YouTube*, YouTube, 27 Oct. 2022, www.youtube.com/watch?v=aFNDh5k3SjU.

Medsriha. “Medsriha/Real-Time-Face-Recognition: Real Time Face Recognition Using Opencv.” *GitHub*, github.com/medsriha/real-time-face-recognition. Accessed 17 Nov. 2024.

Parth. “Understanding Face Recognition Using LBPH Algorithm.” Analytics Vidhya, 21 Oct. 2024, www.analyticsvidhya.com/blog/2021/07/understanding-face-recognition-using-lbph-algorithm/.

“Python Tkinter Tutorial.” *GeeksforGeeks*, GeeksforGeeks, 7 Aug. 2024, www.geeksforgeeks.org/python-tkinter-tutorial/.