Gesture Guided Computer Navigation

06.02.2023

Report Author : Aparna Ramachandran
Group Members : Aparna Ramachandran, Shiker Srivastava

Introduction and Personal Views

Computer vision is an exciting, dynamic field of computer science with very interesting applications. Computer Vision focuses on enabling computers to interpret and understand visual information from the world around us. At a high level, it is replicating parts of the complexity of the human vision system and enabling computers to identify and process objects in images and videos in the same way that humans do. This way, computer vision helps machines to perceive and interpret visual data, which also leads to improved decision-making, automation, and human-computer interaction. Computer vision techniques are now widely being used to transform many industries including transportation, healthcare, robotics and entertainment, through object detection and recognition to image and video analysis, augmented reality, autonomous driving, and gesture recognition. So I believe the field of computer vision holds immense promise for enhancing our daily lives and driving innovation across various domains.

Currently, most of the computer vision applications use Convolutional Neural Networks, where deep neural networks automatically learn hierarchical representations from raw image data. This shift from traditional vision has led to remarkable advancements in various vision tasks, including image classification, object detection, and semantic segmentation, and enabled the development of highly sophisticated computer vision systems with unprecedented performance.

However, deep learning based computer vision is possible because of the abundance of image data available today and the computing power available that is required to process the datasets. I think it was crucial in the development of deep learning models and helped them surpass the hard work of traditional machine learning models for manual feature detectors. As an Artificial Intelligence/ Computer Science student, I believe it is important to

know and understand how traditional computer vision system work,i.e, how you can program the computer to look for “what is there in the image” as opposed to deep learning methods which analyzes and learns what’s there in the image by itself.

So I found it to be really powerful and exciting that there was an extensive range of computer vision techniques explored in this course. It was invaluable to learn and implement concepts such as connected component labeling, edge detection, segmentation, line detection etc which are crucial computer vision techniques. As these are available as packages in OpenCV and other libraries, the implementation is often overlooked. Personally, I found the assignments very helpful in strengthening the understanding of the concepts learned in class. There were a lot of learnings from this class, including image analysis- identifying connected components, which are key for object recognition, tracking segmentation etc, boundary detection using morphological operators, smoothing techniques , edge detection, normalization, object tracking and object recognition, and image stitching. The course was very well structured by finishing assignments early, which made doing the project highly enjoyable. I was able to leverage the topics covered in the course and implement an end-to-end gesture-guided computer navigation system. As the traditional vision topics were covered in detail and I am equipped with the skills, I would now like to learn more about how computer vision has progressed over the years and the deep learning techniques that were developed in the recent years.

Project Description

Our project **‘Gesture-Guided Computer Navigation’** is a seamless and intuitive computer interaction system that utilizes face tracking for navigation and gesture recognition techniques for control, using computer vision techniques. It also has a Face Detection and

Face Recognizer for hands-free login and speech-to-text for hands-free input. The system will eliminate the reliance on traditional input devices and instead leverage the power of computer vision to track facial movements and interpret intuitive gestures, providing an alternative and accessible means of interacting with computers.

The primary motivation for our project is to enhance the accessibility challenges faced by individuals with disabilities. We are living in a digital era where computers are an essential part of day-to-day life. But many individuals with disabilities may have limited mobility, making it difficult to interact with computers. Others may find it difficult to use traditional input methods such as keyboards and mouse pads for longer periods of time. In some cases where people suffer from cognitive impairments, they may find it difficult to use complex interfaces such as that of a computer. In these cases, a gesture-guided computer navigation system can significantly enhance accessibility and empower individuals to navigate and control computers effortlessly.

Additionally, individuals without disabilities can also benefit from gesture-guided computer navigation. In virtual reality and gaming environments, where traditional input methods may disrupt the immersive experience, gesture recognition can offer a more natural and seamless interaction paradigm. It is also beneficial in public displays or interactive kiosks to make interactions more user-friendly and engaging while minimizing contact. Moreover, in medical facilities or laboratories, where hygiene and contamination control are important, gesture-guided navigation can provide a touchless and hygienic solution for interacting with computer systems.

By leveraging the power of computer vision techniques, we aspire to create a more inclusive and intuitive computing experience for all users.

Personal Contribution

In this project, my main contributions are developing and implementing the **Face Detection**, **Face Training**, and **Face Recognition** components. Additionally, I have also worked on the **Speech-To-Text** component as it was a shared task.

Firstly I decided to work on these components for the project as I have experience in working with the other component, which is Mediapipe. I have worked with Mediapipe for landmark generation and pose estimation as part of my practicum, so I wanted to learn and get hands-on experience on something new, which in this case was Face Detection, Training and Recognizer.

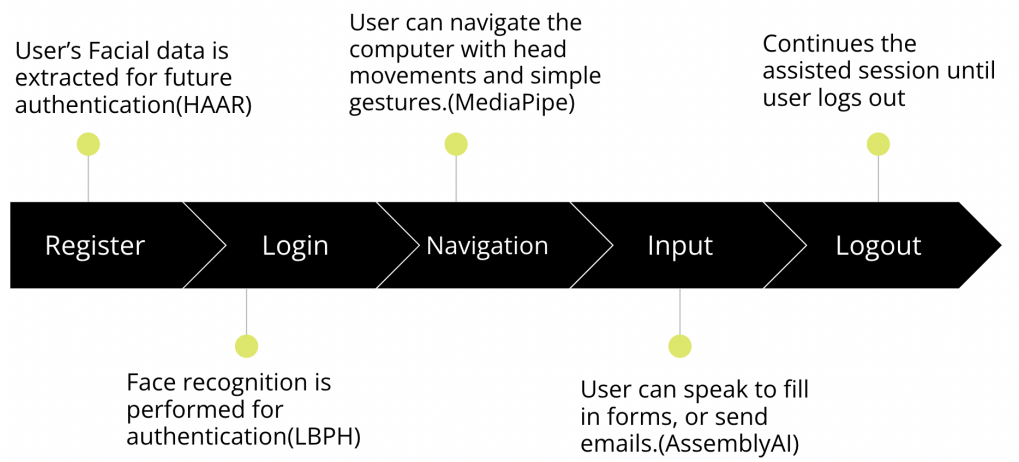
For the Face Detection component, I used the Haar Cascade classifier as I had a theoretical understanding of how it works from class so it was exciting to implement it. After implementation, I fine-tuned the parameters of the classifier to improve its performance and get reliable face detection in different environments and angles. For Face Recognizer, I implemented LBPH classifier, as it has a good tradeoff between accuracy and computational efficiency.

For Speech-To-Text, both of us contributed to its successful integration into the project. I set up the Assembly AI platform and wrote an initial code to send requests to the endpoint and get responses. But there was more work to be done to integrate it with our platform and get accurate speech-to-text conversion.

These contributions were crucial for the project's success in accurately identifying and recognizing faces. The full detail of what we have done in each of these tasks is given in the following **Design** section.

Design

High-Level architecture



At a high level, Gesture-Guided Navigation works as an assist system, which the user can enable and disable by log in and log out. First, the user must register using a simple webpage, when the user's face data will be registered for future authentication. From then on the user is able to login using face recognition, and the system remains active throughout the time period when the user is logged in whilst the user can navigate the computer via head movements and simple hand gestures.

Firstly during registration, the user's facial data is extracted using the HAAR cascade classifier for future authentication. At the time of login, face recognition is performed using the LBPH recognizer for authentication. The user can then navigate the computer using head movements and simple gestures, which are tracked and recognized using MediaPipe. To make it more interesting and fully functional, our system incorporates AssemblyAI for speech input, allowing users to fill in forms or send emails just by speaking. The assisted session continues until the user logs out, providing continuous support and guidance throughout the interaction.

Technical Stack

In this project we are using a variety of technologies to create this end-to-end system

1. OpenCV: We used the OpenCV library extensively for face detection and recognition processes. It's also used for other image processing tasks like reading camera input streams

2. MediaPipe: We use this open-source library for facial tracking (FaceMesh) and gesture detection (MediaPipe Hands). The real-time facial and hand landmarks generated by mediapipe helped for user interaction.

3. PyAudio: This Python library was used to record the user's voice, capturing an audio stream that could be converted to text.

4. AssemblyAI: We use real time speech-to-text endpoint provided by AssemblyAI for converting the recorded audio stream into written text. It is now a paid service, we wish to explore more open source libraries in the future

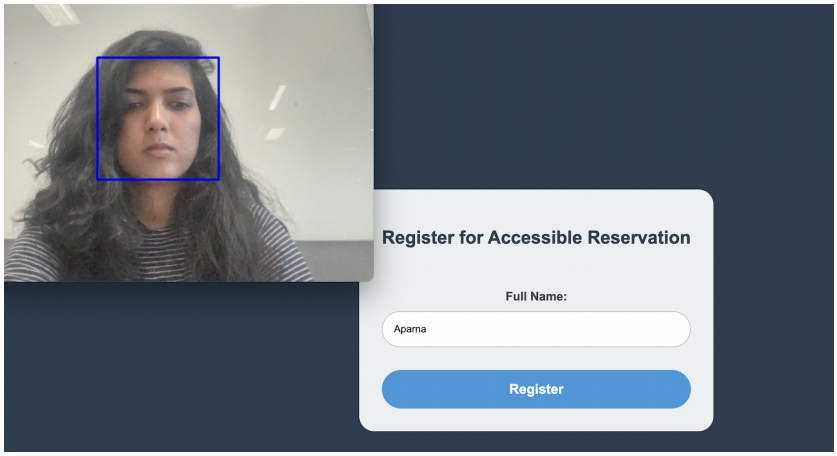
5. PyInput: This Python library is useful for controlling and monitoring input devices. Specifically, it was utilized to move the mouse pointer in accordance with the user's head movements.

Register and Login Interface

Register and Login interfaces are hosted as a **Web Application** using **Flask.** Register interface allows us to take the user's ***name*** and store the user's face data samples with a ***unique ID*** corresponding to the name. This is done so that the user can later on login hands-free, using face ID. Although this is not a required feature for personal devices, we thought this might be useful in shared settings like medical facilities or public places where

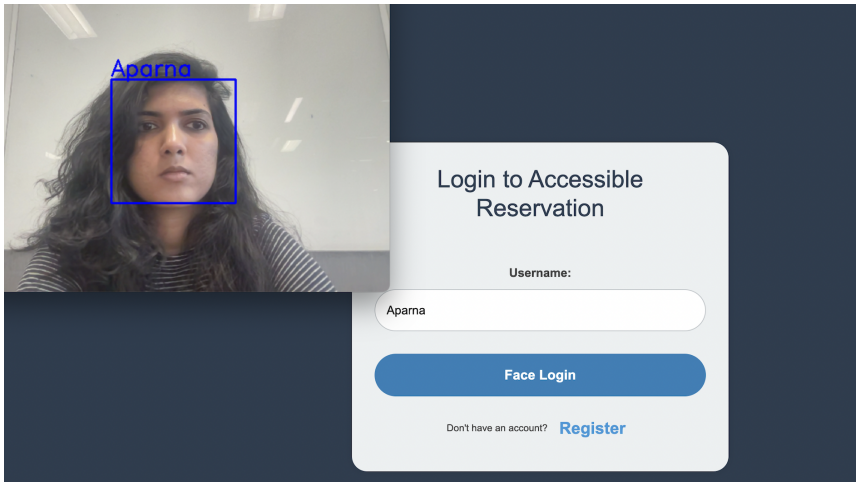
many people are sharing one system and everyone might not require gesture-guided navigation assistance. On a personal computer, which is used by a trusted user, we will not require the registration process and it can be an optional feature.

Register screen:



As you can see, the register screen prompts the user for a name, and when the name is provided and the ‘Register’ button is clicked it opens up the camera for a few seconds (30 frames) for detecting the user’s face and collecting face data samples.

Login screen:



Once registration is done, the natural next step is login. Login screen as shown here gives the user an interface to login. When the user clicks the 'Face Login' button, the camera opens up, activating the face recognizer to authenticate the Gesture-Guided session. If the user is recognized within the first n (configurable, default = 10) seconds, access is granted and the session is activated. If the user is not recognized, then the session will not be activated but the user is able to go to the Register interface from this page as shown at the bottom right corner of the login window.

Face Detection

The most basic task of Face Recognizer is "Face Detection". We should be able to detect and capture a "face" in the image inorder to compare it to a new face for authentication in the future. There are many techniques available, but I decided to go with the **HAAR Classifier** for face detection as it was covered in class and I wanted to explore and get hands-on experience on it.

Face Detection using **Haar feature-based cascade classifiers** is an effective object detection method proposed by Paul Viola and Michael Jones in their paper, "*Rapid Object Detection using a Boosted Cascade of Simple Features*". Initially, the classifier is trained with a lot of positive images (images of faces) and negative images (images without faces) to train the classifier. This is done so because the classifier should be able to distinguish between "face" and "non-faces". During training, Haar features are applied on all the images and the best features that are able to discriminate well between face and non-face are selected. During Face Detection(inference), these selected features are applied in a cascading fashion to the image to detect a face.

For this project, I am using a pre-trained **Haar classifier** available in **OpenCV.** The function takes video frames as input and converts it to grayscale to pass it to the Haar classifier. If a face is detected a **blue rectangle** is drawn around it. And the image is saved in a dataset under a unique ID. This is repeated until enough samples are obtained (30 frames with face).

Dataset showing how user face data sample is stored:

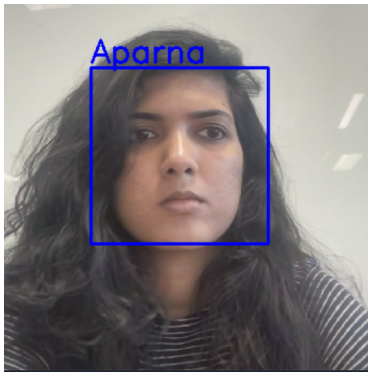
Face Recognition

When a user logs in, we should be able to correctly match the face to any of the users we have in the database to recognize and authenticate the user. With the facial images already extracted, cropped, resized and converted to grayscale, the face recognition algorithm is responsible for finding characteristics which best describe the image to recognize an image. For this, I have decided to use **LBPH recognizer (LOCAL BINARY PATTERNS HISTOGRAMS)**. LBPH is robust to variations in illumination and can handle grayscale images effectively so it will work well in our application where the users can be in different lighting conditions while interacting with the computer system.

LBPH operates by dividing an image into small regions and extracting binary patterns from each region. The binary patterns are generated by comparing the intensity of each pixel with its neighboring pixels and encoding the comparison results as binary codes. LBPH then constructs a histogram of the patterns within each region, representing the distribution of different pattern types. The final feature representation is obtained by concatenating the histograms from all regions, creating a high-dimensional feature vector.

For this project, I am using the LBPH **recognizer** available in **OpenCV.** As soon as a new user is registered (i.e, we now have their face data sample), we train the recognizer with the data samples and unique user ID. When the same user logs in, Recognizer compares the feature vectors and returns the **closest match** with a **confidence score.**

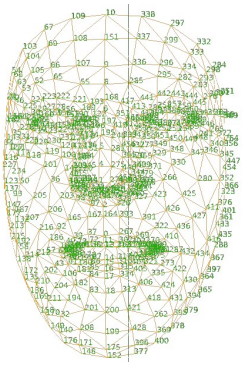
Image annotated with user's name when Face Recognizer finds a match:



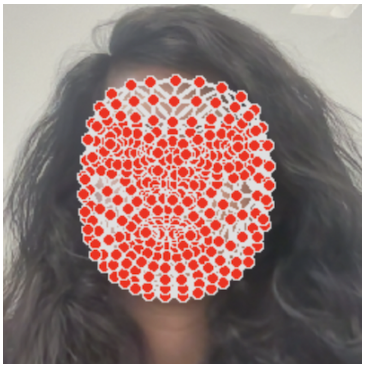
Navigation With Face Tracking

For navigation of mouse pointers on the screen, we are using **MediaPipe Facemesh**. Facemesh detects 468 3D facial landmarks in real-time, providing a rich understanding of facial features and movements.

FaceMesh landmarks:



FaceMesh Landmarks on real-time video:

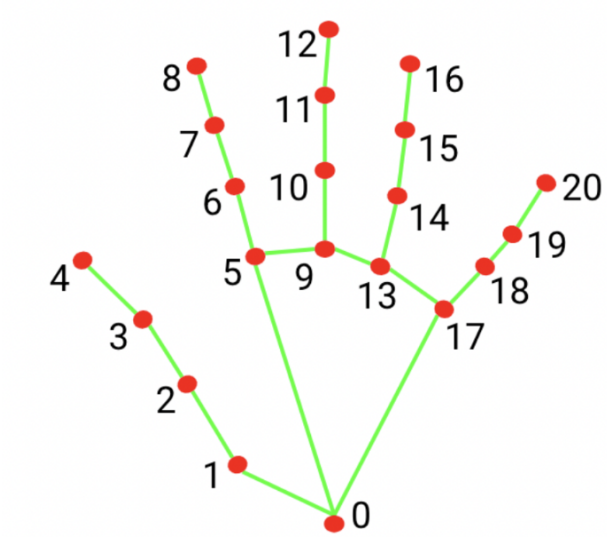


We are focusing on the landmark of the **Nose** for real time navigation. By tracking the nose landmark, across consecutive video frames, we compute the head movement and adjust the cursor position on the computer screen accordingly. We then use **Pynput** to move the mouse pointer to the new coordinates. This feature allows users to move mouse pointers anywhere on the screen by just pointing their head slightly to that area, thereby allowing hand-free navigation.

Actions with Hand Gestures

For performing actions on the screen, like clicking and scrolling, we are using **MediaPipe Hands**. MediaPipe Hand detects and infers 21 3D landmarks of a hand from a single frame, providing detailed understanding of hand positions and movements.

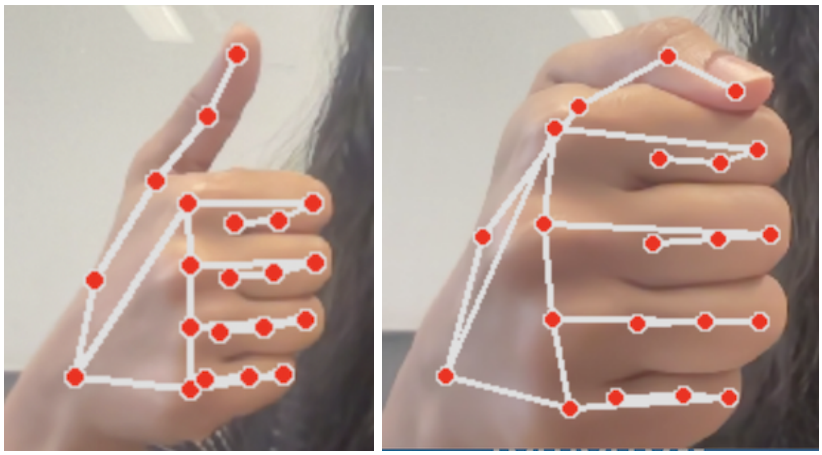
MediaPipe Hand Landmarks:



In our system, we leverage 21 landmarks, marking key points like finger-tips, joints, and the palm's base, are used to detect actions like **mouse clicks and scrolling**. This gives the user's ability to perform actions with very simple hand-gestures.

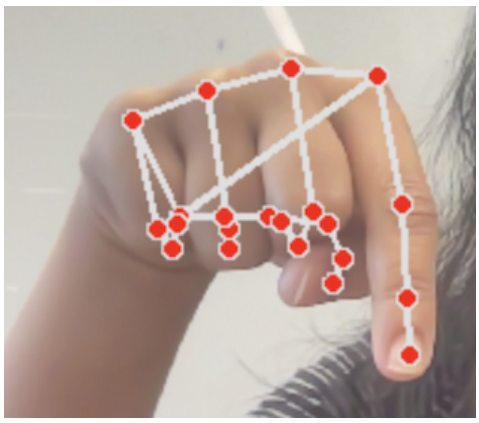
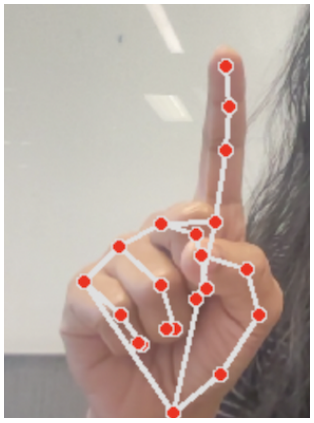
Click Action: We calculate the distance between the thumb and other fingers to detect a 'click' action.

Images showing click action:



Scrolling: We observe the orientation of the index finger, intuitively controlling scrollable elements on the screen. Scroll down and Scroll up is detected by the position of the index finger TIP, index finger DIP, index finger PIP, and index finger MCP landmarks.

Scroll Down: Scroll Up:

Speech-To-Text for Hands-free Input

Typing anything requires a lot of coordination and can be difficult for people with disabilities. But it is an essential part of day-to-day form filling, sending emails, and other text-based interactions. So we decided to add a Speech-To-Text feature in our application for enhancing accessibility and usability of our Gesture-guided Computer Navigation system, although it's not related to computer vision.

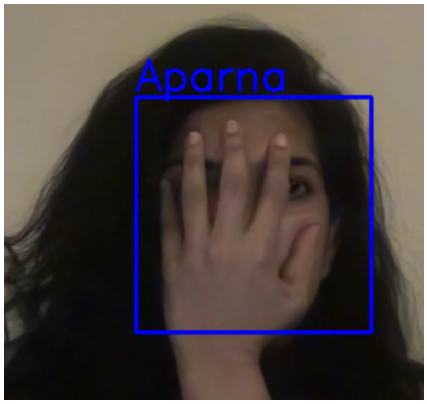
I've used **PyAudio** to record audio streams from the user, **AssemblyAI** real time endpoint to convert the recorded audio stream into written text in real-time, facilitating easy data entry without the need for typing.

Results and Analysis

Overall, our results were quite impressive. We were able to combine all these different components such as **Web Application, Face Detector, Face Recognizer, Face Tracking, Hand Gestures, and Speech-To-Text** and build a fully functional end-to-end application.

1. FaceRecognition:The Face Recognition module was able to detect and recognize registered user faces in varying positions across the frame. The face recognizer is able to recognize a person even when the face is partially obscured, which I found is remarkable.

Partially Obscured Face Recognized:



2. Navigation with gestures: The Gesture-controlled navigation was able to accurately detect and interpret gestures with a very low latency. Navigating the mouse pointer on the screen is accurate. We initially faced issues in this part where the mouse movements were abrupt, but we were able to resolve this and make the movements slower by adding a **smooth factor** in the mouse coordinates. Hand gestures such as click and scroll are detected well and the action is performed immediately without any delay. However, the accuracy of gesture detection has a lot of scope of improvement, as we faced some false

positives and false negatives during our testing phase.

3.Speech-to-Text: The Speech-to-Text conversion successfully transcribed spoken words to written text with an average success rate of 92%. This was measured by comparing the system's transcription against the actual speech of individuals during the testing phase.

To summarize, we got expected results out of the project. Deeper analysis of the system under different environments could be performed to test how well face detection and recognition performs under different angles and lighting conditions, how face tracking(navigation) and gesture detection(actions) performs with different angles, skin colors etc. Due to the time limitation, we are not able to perform these stress tests and will be a task for the future. Overall, I am very satisfied with our results. The application performs accurately, changing controls to gesture-guided system and back, and the gesture guide is **active only when a hand is in the frame of the video.** This really helps to avoid accidental clicks and mouse pointer movements. I think this system will help to make computers more accessible to people with disabilities.

Remarks and Future work

The potential areas of improvements of our application are the following:

- Accessibility:
    - To make the system even more accessible, adding ASL gesture recognition capability as an alternate option to Speech-To-Text will be a great next step for this application. This will help people with speech difficulties.
- Integration with Other Accessibility Tools:

    - The system could potentially integrate with other accessibility tools, like screen readers, to provide a holistic accessibility solution.

- Robustness in Varied Conditions:
    - As noted earlier, we can improve the system's ability to function reliably in various environmental conditions like differing light levels, background noise, and physical orientation.
- Additional gestures:
    - Adding additional gestures for Zoom In/Out, Scroll Right/Left, Back/Forward Navigation, Drag/Drop, Pause/Play Media, Volume Control,Application Switching will be necessary next steps.

Course feedback and Suggestions

This course and its respective content was very thought-provoking and enjoyable. As computer vision has advanced over the years and deep neural networks are used for everything these days, we generally don't get to learn(or sometimes we overlook) the basic concepts of how traditional computer vision techniques can be implemented. So for me, it was very valuable to learn how to implement important image processing and analysis techniques. As for suggestions, I think it would be helpful to have some more information in the slides as the lectures are not recorded. In some lectures, I felt a lot of content was covered in class which were not available in slides, having this would be helpful to revisit the content and resolve any confusion about concepts taught in class.

References

1. Ahonen, Timo, Abdenour Hadid, and Matti Pietikainen. *"Face description with local binary patterns: Application to face recognition."* IEEE transactions on pattern analysis and machine intelligence 28.12 (2006): 2037–2041
2. Ahonen, Timo, Abdenour Hadid, and Matti Pietikäinen. *"Face recognition with local binary patterns."* Computer vision-eccv 2004 (2004): 469–481
3. Paul Viola, and Michael Jones. *"Rapid Object Detection using a Boosted Cascade of Simple Features"*
4. https://docs.opencv.org/3.3.0/d7/d8b/tutorial_py_face_detection.html
5. https://docs.opencv.org/4.x/df/d25/classcv_1_1face_1_1LBPHFaceRecognizer.html