Omar El-Azab
ID: 9710

Principal Component Analysis-Based Face Recognition

Department of Computer and Communication Engineering
Faculty of Engineering, Alexandria University, Egypt

جامعة الإسكندرية
ALEXANDRIA UNIVERSITY
كلية الهندسة
Faculty of Engineering

Omar El-Azab
ID: 9710

Principal Component Analysis-Based Face Recognition

Department of Computer and Communication Engineering
Faculty of Engineering, Alexandria University, Egypt

Educational Project
2024-2025

Educational project aims to design, simulate and test

a face recognition system with eigenface-based feature extraction,

Contributing to the advancement of knowledge.

Marwan A. Eid
ID:9532
Student
Alexandria University

Nour E. Sakr
ID:9271
Student
Alexandria University

Omar M. El-Azab
ID:9710
Student
Alexandria University

Saeed M. El-Saghir
ID:9698
Student
Alexandria University

Seif Zaki
ID:9353
Student
Alexandria University

Youssef E. Deyab
ID: 9798
Student
Alexandria University

Omar El-Azab
ID: 9710

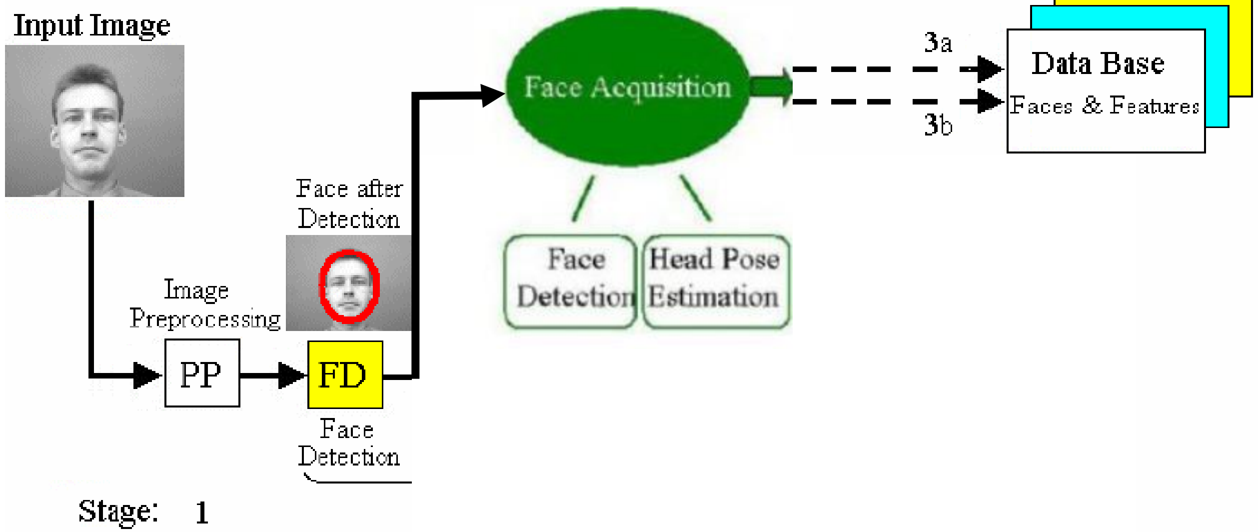
Contents

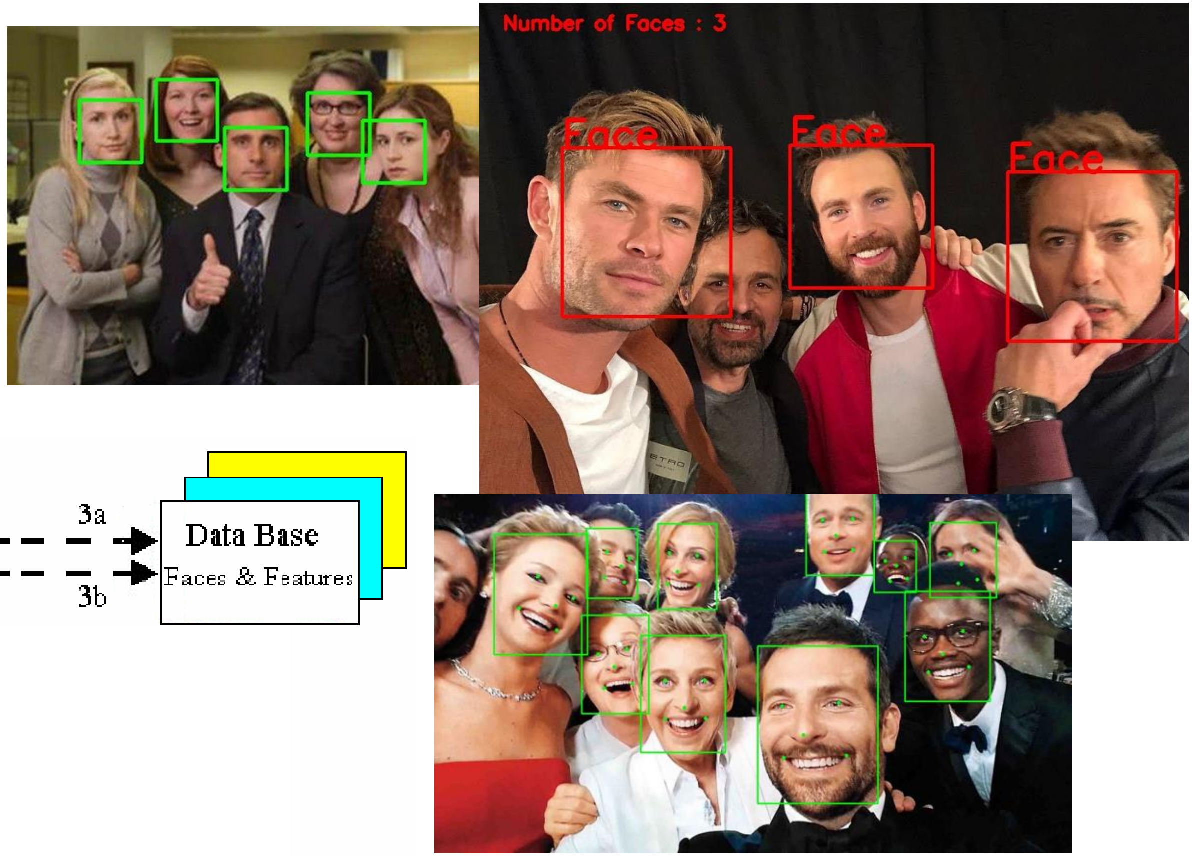
Seif Zaki
ID: 9353

Introduction

Face Detection

- Definition:
    - Face detection is a computer vision technique that identifies and locates human faces in digital images or video.
- Key Features:
    - Detects the presence of faces in a scene.



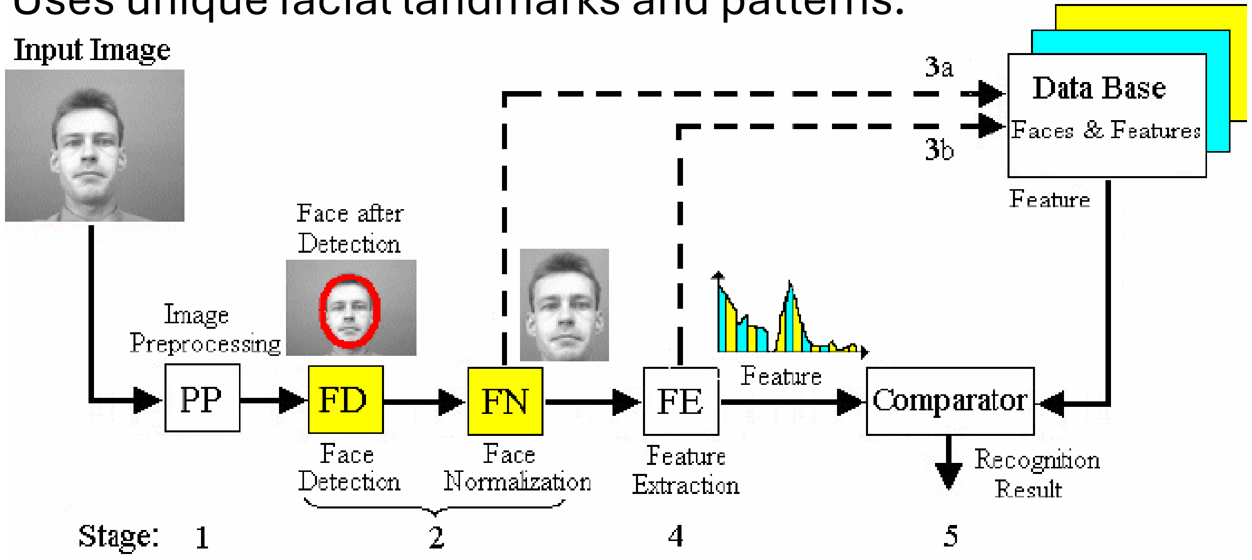


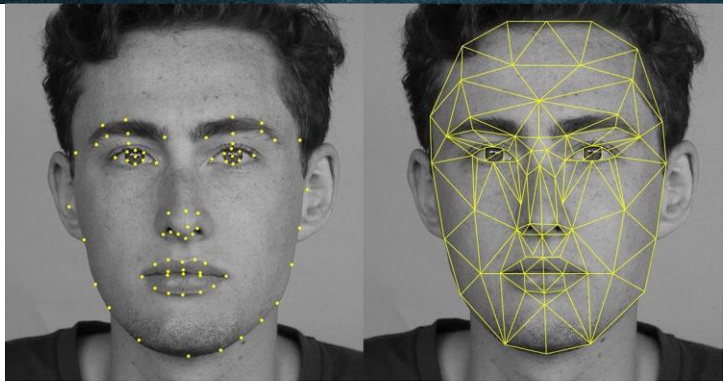
Seif Zaki
ID: 9353

Introduction

Face Recognition

- Definition:
  - Face recognition goes beyond detection to identify or verify individuals based on their facial features.
- Key Features:
  - Compares a detected face to stored one.
  - Uses unique facial landmarks and patterns.





Seif Zaki
_ID: 9353_

Introduction

Face Detection vs. Face Recognition

Feature	Face Detection	Face Recognition
Purpose	Locate faces in an image or video.	Identify or verify individuals.
Output	Bounding boxes around faces.	Identity match or verification result.
Complexity	Less complex, faster processing.	More complex, requires comparisons.
Applications	Photography, AR/VR, surveillance.	Security, authentication, marketing.

Nour Sakr
ID: 9271

جامعة الإسكندرية
ALEXANDRIA
UNIVERSITY

Mathematical Model

1. **Mean of a vector**

The mean of a dataset is the average value of each feature. In PCA, the mean is used to center the data, before applying PCA, it is common to subtract the mean of each vector from the dataset. This centers the data around zero for each feature, ensuring that PCA focuses on variance rather than the location of the data in the feature space.

2. **Standard Deviation**

The standard deviation measures the spread or dispersion of a dataset.

In PCA, it is used to scale the data and their average distance from the mean.

. **Formula**: $\sigma = \sqrt{(\Sigma (Xi - \mu)^2 / N)}$

Nour Sakr
ID: 9271

Mathematical Model

3.Covariance-Matrix

Covariance is a measure of how two vectors vary together. In PCA the covariance matrix is a key component used to determine the relationships between features and to identify the directions of maximum variance in the data.

$$
\begin{array}{cc}
 & \begin{array}{cc} \text{x} & \text{y} \end{array} \\
\begin{array}{c} \text{x} \\ \text{y} \end{array} &
\begin{bmatrix} var(x) & cov(x, y) \\ cov(x, y) & var(y) \end{bmatrix}
\end{array}
\qquad
\begin{array}{cc}
 & \begin{array}{ccc} \text{x} & \text{y} & \text{z} \end{array} \\
\begin{array}{c} \text{x} \\ \text{y} \\ \text{z} \end{array} &
\begin{bmatrix} var(x) & cov(x, y) & cov(x, z) \\ cov(x, y) & var(y) & cov(y, z) \\ cov(x, z) & cov(y, z) & var(z) \end{bmatrix}
\end{array}
$$

Marwan Eid
ID: 9532

Mathematical Model

Introduction

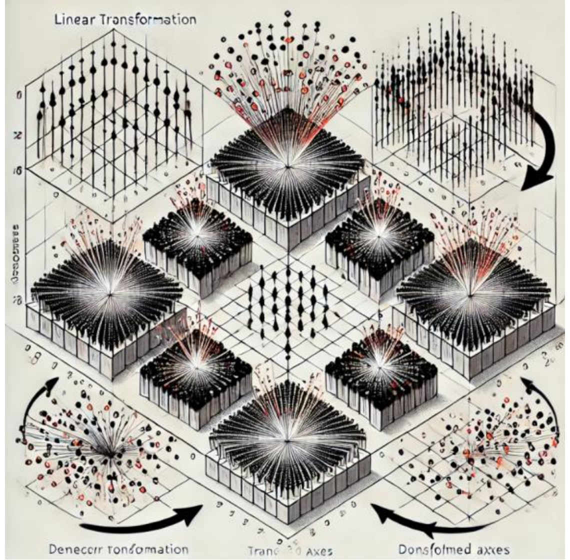
- Face recognition systems rely on advanced mathematical techniques to process and analyze facial data.
- A key method used is Principal Component Analysis (PCA), which simplifies the data by reducing its dimensions while retaining the most critical information.
- PCA is particularly useful in handling the high dimensionality of facial images, making it easier for systems to recognize patterns in facial features.
- Central to PCA are concepts like **Linear Transformation**, **Eigenvectors**, and **Eigenvalues**, which enable efficient recognition and feature extraction.

Marwan Eid
ID: 9532

Mathematical Model

Linear Transformation

- A linear transformation is a mathematical operation that maps data from one space to another while preserving its linear properties.



- In the context of PCA:
  - It rearranges the data into new axes that are better aligned with its variance, emphasizing the most important patterns.
  - Operations such as **rotation**, **scaling**, or **projection** are applied to highlight specific features of the data.

- This transformation is essential for uncovering the underlying structure of the data, ensuring that key patterns are preserved and enabling dimensionality reduction while retaining important information.

Marwan Eid
ID: 9532

Mathematical Model

- **Eigenvectors and Eigenvalues**

Once the data has been linearly transformed, **Eigenvectors** and **Eigenvalues** play a critical role in identifying the most informative directions in the data.

1. **Eigenvectors**
    - Eigenvectors are special vectors that remain unchanged in direction after a linear transformation is applied.
    - They define the axes or directions along which the data exhibits the greatest variation. In simpler terms, they point to where the data "spreads out" the most in the transformed space.
    - These directions are crucial because they allow us to focus on the most significant features of the data, such as key facial attributes (e.g., the shape of eyes, nose, and mouth) in the context of face recognition.

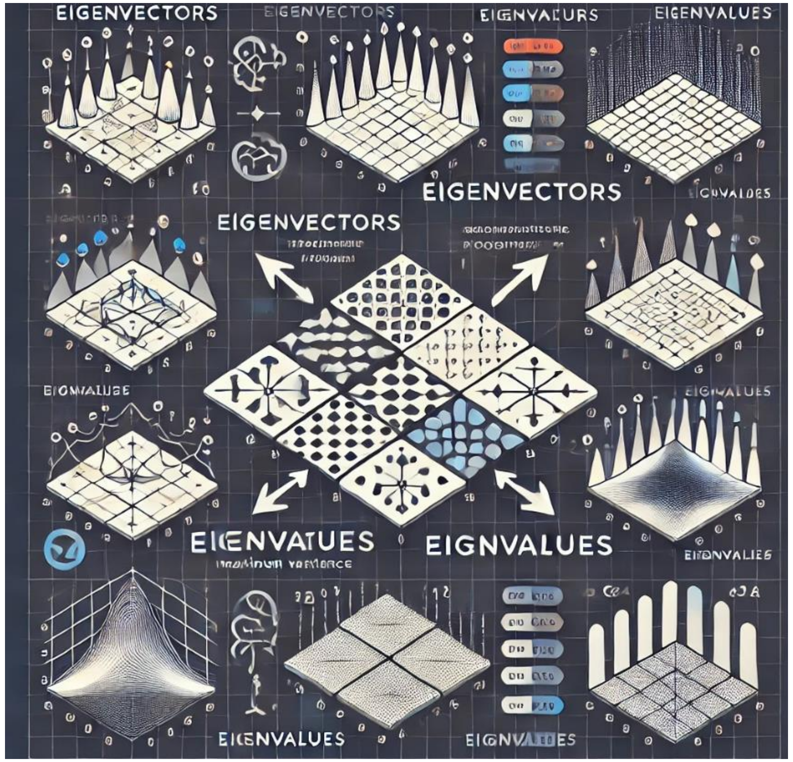
Marwan Eid
ID: 9532

جامعة الإسكندرية
ALEXANDRIA
UNIVERSITY

Mathematical Model

2. Eigenvalues

- Each eigenvector has a corresponding eigenvalue, which quantifies the importance of that eigenvector.
- An eigenvalue represents the magnitude of the variance along its eigenvector's direction. Larger eigenvalues indicate that the direction captures more critical information about the data.



- By ranking the eigenvalues, we can identify the most influential eigenvectors to use in the analysis, prioritizing the components that capture the most distinguishing features of the data

Marwan Eid
ID: 9532

Mathematical Model

Summary

In PCA, **Eigenvectors**, **Eigenvalues**, and **Linear Transformation** are essential for simplifying data.

- **Eigenvectors** define the new axes or principal components that capture the most important patterns in the data.
- **Eigenvalues** indicate the significance of each eigenvector, helping prioritize the most important components.
- **Linear transformation** projects the data onto the new axes, reducing dimensionality while preserving key features.

• Together, these elements enable efficient data representation and simplification, which is crucial for applications like face recognition.
By focusing on the most important features and reducing the complexity of the data, PCA makes it easier for face recognition systems to detect and identify faces even in large and complex datasets.

Saeed El-Saghir
ID: 9698

Image representation

- A square, N by N image is converted into an N^2-dimensional vector.

$$X = (x_1 \ x_2 \ x_3 \ x_4 \ x_5 \cdots x_n), where\ X\ is\ the\ Image\ Vector$$

- Rows of pixels are concatenated into a single column vector. (each element representing the Pixel intensity values (e.g., grayscale, from 0 to 255)).

Images Vector

Say we have 20 images. Each image is N pixels high by N pixels wide. For each image we can create an image vector as described. We can then put all the images together in one big image-matrix like this:

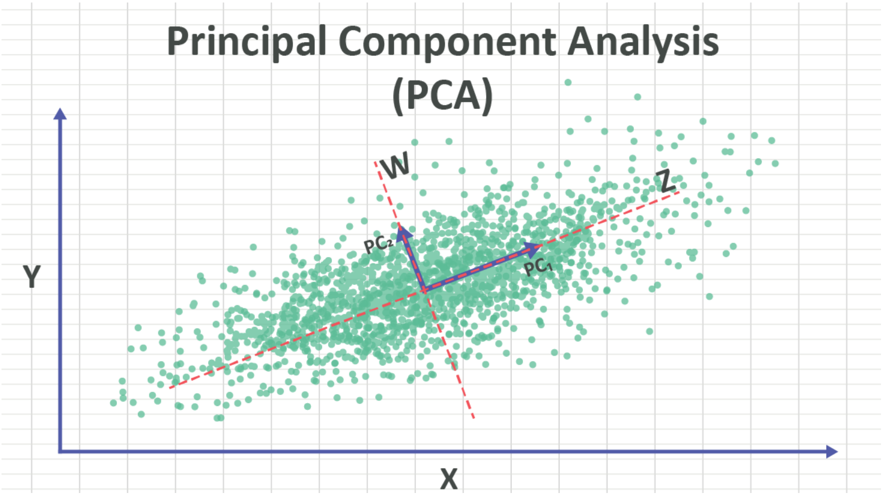
$$ImagesMatrix = \begin{matrix} Imagevec1 \\ Imagevec2 \\ \vdots \\ Imagevec20 \end{matrix}$$

- And this is our starting point to perform PCA

Saeed El-Saghir
ID: 9698

Principal Component Analysis (PCA)

- Principal component analysis (PCA) is a linear dimensionality reduction technique with applications in exploratory data analysis, visualization and data preprocessing.
- It is defined as an orthogonal linear transformation on a real inner product space that transforms the data to a new coordinate system such that the greatest variance by some scalar projection of the data comes to lie on the first coordinate (called the first principal component), the second greatest variance on the second coordinate, and so on.

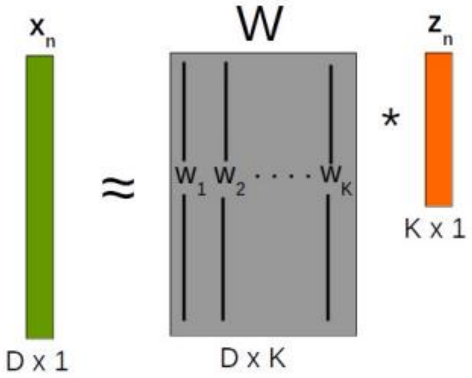


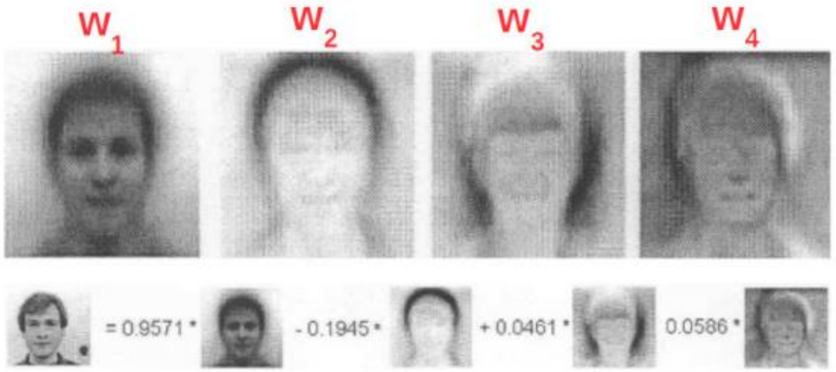
- **Purpose in Biometrics**:
  - Compress high-dimensional face data.
  - Retain the most important features for recognition.
  - Reduce computational complexity.

Saeed El-Saghir
ID: 9698

Dimensionality Reduction

- **Dim-red for face images**





- In this example, $\boldsymbol{z}_n \in \mathbb{R}^K$ $(K = 4)$ is a low-dim feature rep. for $\boldsymbol{x}_n \in \mathbb{R}^D$
- Essentially, each face image in the dataset now represented by just 4 real numbers
- Different dim-red algos differ in terms of how the basis vectors are defined/learned
  - .. And in general, how the function f in the mapping $\boldsymbol{x}_n = f(\boldsymbol{z}_n)$ is defined

Saeed El-Saghir
ID: 9698

PCA Model

- Center the data (subtract the mean $\boldsymbol{\mu} = \frac{1}{N}\sum_{n=1}^{N} \boldsymbol{x}_n$ from each data point)
- Compute the $D \times D$ covariance matrix $\mathbf{S}$ using the centered data matrix $\mathbf{X}$ as
- Do an eigen decomposition of the covariance matrix $\boldsymbol{S}$
- Take top $K < D$ leading eigvectors $\{\boldsymbol{w}_1, \boldsymbol{w}_2, \dots, \boldsymbol{w}_K\}$ with eigvalues $\{\lambda_1, \lambda_2, \dots, \lambda_K\}$
- The K-dimensional projection/embedding of each input is
- Features Matrix (Eigenfaces) E = Top leading eigenvectors multiplied by its transpose
- $E = W_{1*k} \cdot W_{K*1}^{T}$

Youssef Deyab
ID: 9798

Eigen Faces

1. Introduction :

- Eigenfaces are a facial recognition technique based on Principal Component Analysis (PCA).
- They reduce the dimensionality of face data, capturing key features for identification.

Youssef Deyab
ID: 9798

Eigen Faces

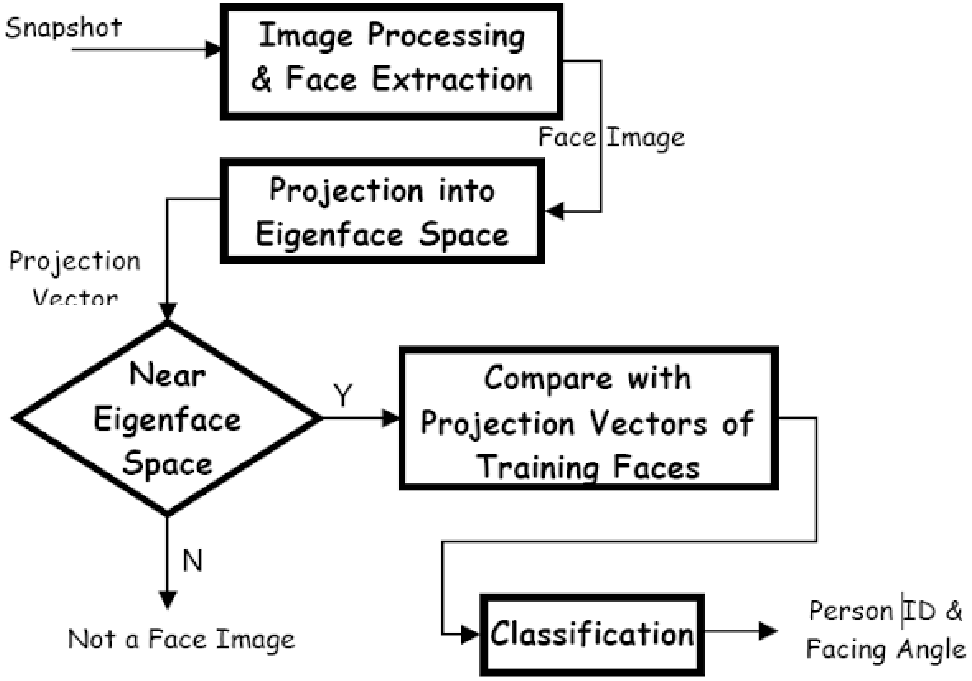
2. How Eigenfaces Work :

• First, a dataset of face images is collected and converted into vectors.

• The mean face is computed and subtracted from each image to center data.

• Eigenfaces are then calculated as the eigenvectors of the covariance matrix, representing the most significant facial features.

• Each face is expressed as a linear combination of these eigenfaces, simplifying storage and comparison .



Algorithm of Face Recognition.

Youssef Deyab
ID: 9798

Eigen Faces

3. Projection in Eigenfaces :

- A new face is projected onto the eigenspace by computing its weights(coefficients) for each eigenface.
- Formula:

$$w_k = u_k^T \Phi$$

where u_k is the eigenface and Φ is the input face after mean subtraction.

- These weights form a feature vector that uniquely represents the face in a lower-dimensional space.
- Recognition is done by comparing these weights with stored faces using distance measures like Euclidean distance.

Youssef Deyab
ID: 9798

Eigen Faces

4. Applications and Advantages :

- Used in face recognition, expression analysis, and image compression .
- Advantages: Fast, memory-efficient, and effective under small variations.
- Limitations: Sensitive to lighting and pose changes.

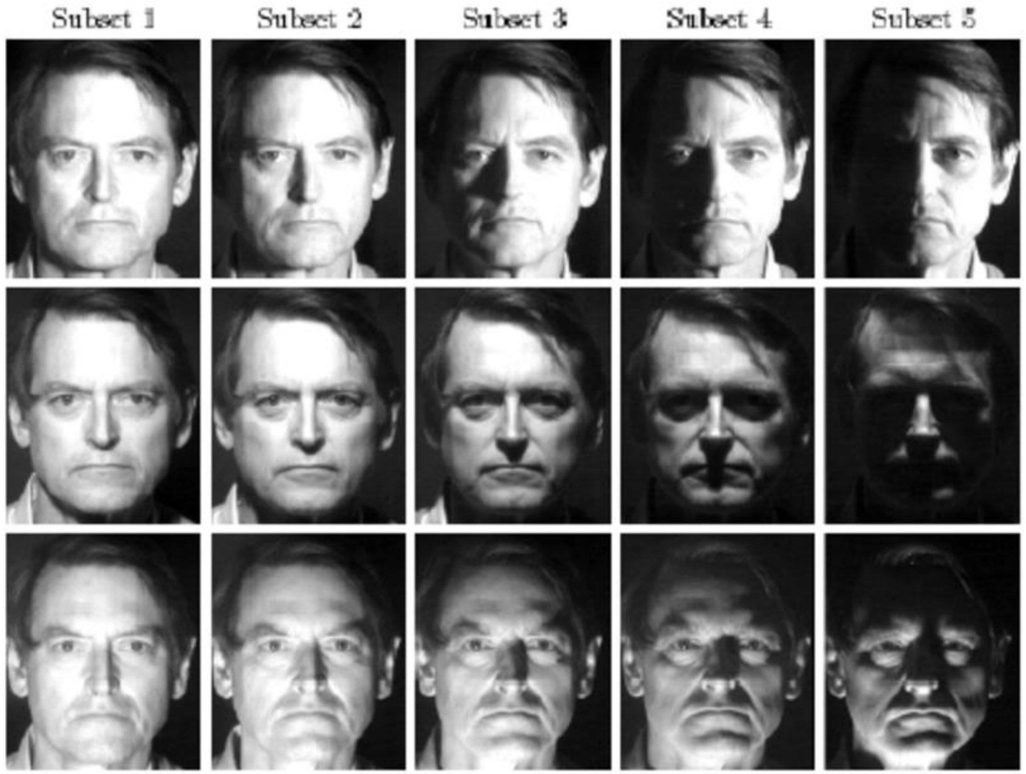
5. Closing Statement :

- Eigenfaces revolutionized facial recognition by focusing on statistical features rather than geometric details.
- They laid the foundation for modern machine learning techniques used today

Youssef Deyab
ID: 9798

جامعة الإسكندرية
ALEXANDRIA
UNIVERSITY

Eigen Faces



(a) This sample of eigenfaces shows the tendency of the principal components to capture major variations in the training set such as lighting direction ; (©1996 IEEE)

Omar El-Azab
ID: 9710

جامعة الإسكندرية
ALEXANDRIA
UNIVERSITY

Model Simulation

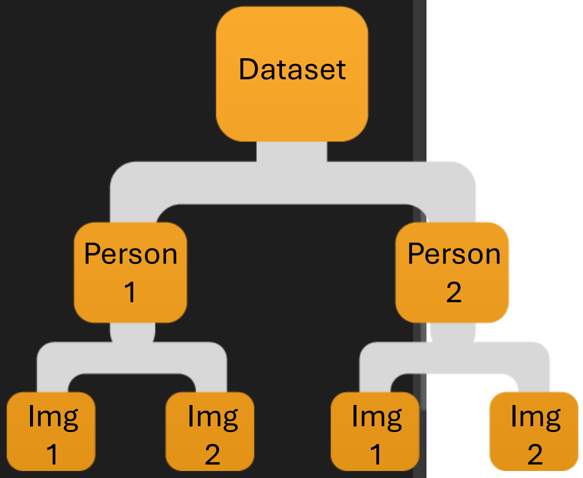
Libraries

```python
import os                          # Library to work with the file system (folders and files)
import cv2                         # OpenCV library for image processing
import numpy as np                 # Library for numerical operations
import matplotlib.pyplot as plt    # Library for plotting images and visualizations
```

Omar El-Azab
ID: 9710

Model Simulation

```python
# Load the dataset with multiple images per person (labeled images)
def load_dataset(folder, target_size):
    images = []        # To store all image data
    labels = []        # To store numeric labels
    label_names = []  # To store names of individuals in the dataset

    for label_id, subfolder in enumerate(os.listdir(folder)):
        subfolder_path = os.path.join(folder, subfolder)
        if os.path.isdir(subfolder_path):
            label_names.append(subfolder)  # Person's name
            for filename in os.listdir(subfolder_path):
                img_path = os.path.join(subfolder_path, filename)
                img = cv2.imread(img_path, cv2.IMREAD_GRAYSCALE)
                if img is not None:
                    img_resized = cv2.resize(img, target_size)
                    images.append(img_resized / 255.0)  # Normalize to [0, 1]
                    labels.append(label_id)
    return np.array(images), np.array(labels), label_names
```



Omar El-Azab
ID: 9710

ALEXANDRIA UNIVERSITY

Model Simulation

```
[ ]  # PCA implementation
     def perform_pca(data, n_components):
         mean_face = np.mean(data, axis=0)                          # Compute the average face (mean vector)
         data_centered = data - mean_face                           # Center data by subtracting the mean
         cov_matrix = np.cov(data_centered.T)                       # Compute the covariance matrix
         eigenvalues, eigenvectors = np.linalg.eig(cov_matrix) # Eigen decomposition
         sorted_indices = np.argsort(-eigenvalues)                  # Sort eigenvalues in descending order
         top_eigenvectors = eigenvectors[:, sorted_indices[:n_components]]# Top eigenvectors
         eigenfaces = top_eigenvectors.T                            # Transpose for easier dot-product operations
         return mean_face, eigenfaces
```

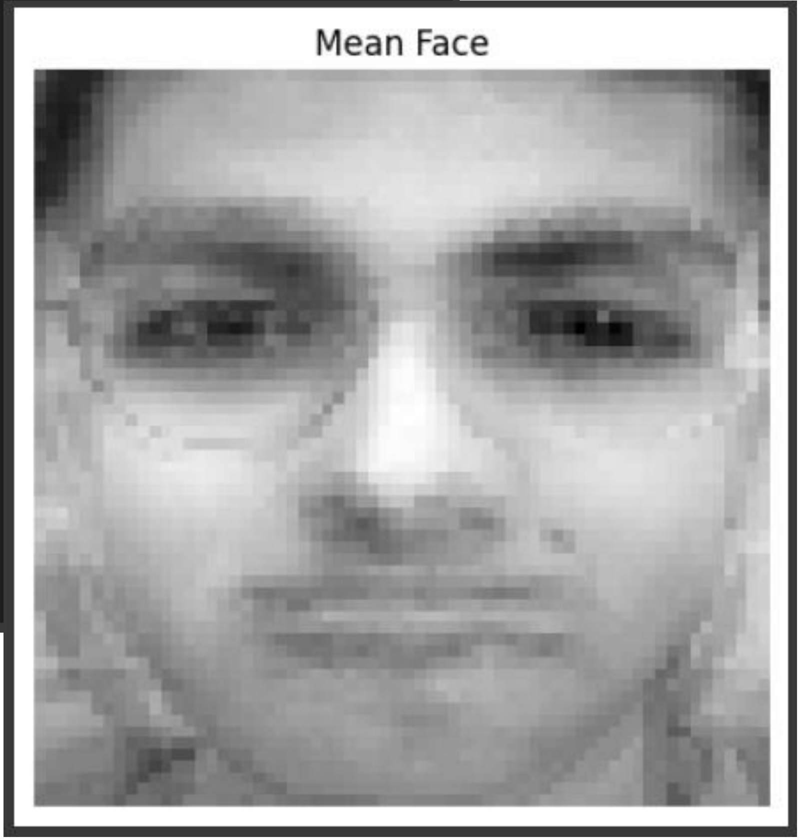
```
[ ]  # Project data into PCA space
     def project_to_pca_space(data, mean_face, eigenfaces):
         data_centered = data - mean_face                       # Subtract the mean to center the data
         projections = np.dot(data_centered, eigenfaces.T) # Project onto PCA components
         return projections
```

Omar El-Azab
ID: 9710

ALEXANDRIA UNIVERSITY

Model Simulation

```
[ ]  # Main program
     dataset_path = "/content/drive/MyDrive/Projects/Principal Component Analysis-Based Face Recognition/model/dataset"
     target_size = (64, 64)
     n_components = 50  # Number of principal components

     # Load dataset
     images, labels, label_names = load_dataset(dataset_path, target_size)
     data = images.reshape(len(images), -1) # Flatten each image into a single vector

     # Perform PCA
     mean_face, eigenfaces = perform_pca(data, n_components)

     # Project data to PCA space
     projections = project_to_pca_space(data, mean_face, eigenfaces)

     # Visualize the mean face
     plt.imshow(mean_face.reshape(target_size), cmap='gray')
     plt.title("Mean Face")
     plt.axis('off')
     plt.show()
```



Omar El-Azab
ID: 9710

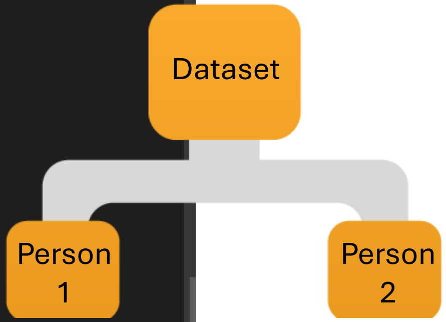
Model Simulation

```
[ ] # Recognize a person
    def recognize_person(test_projection, projections, labels, label_names, threshold=10):
        distances = np.linalg.norm(projections - test_projection, axis=1)
        min_distance = np.min(distances)
        if min_distance > threshold:
            return "Unknown", min_distance
        recognized_index = np.argmin(distances)
        return label_names[labels[recognized_index]], min_distance
```

Omar El-Azab
ID: 9710

Model Simulation

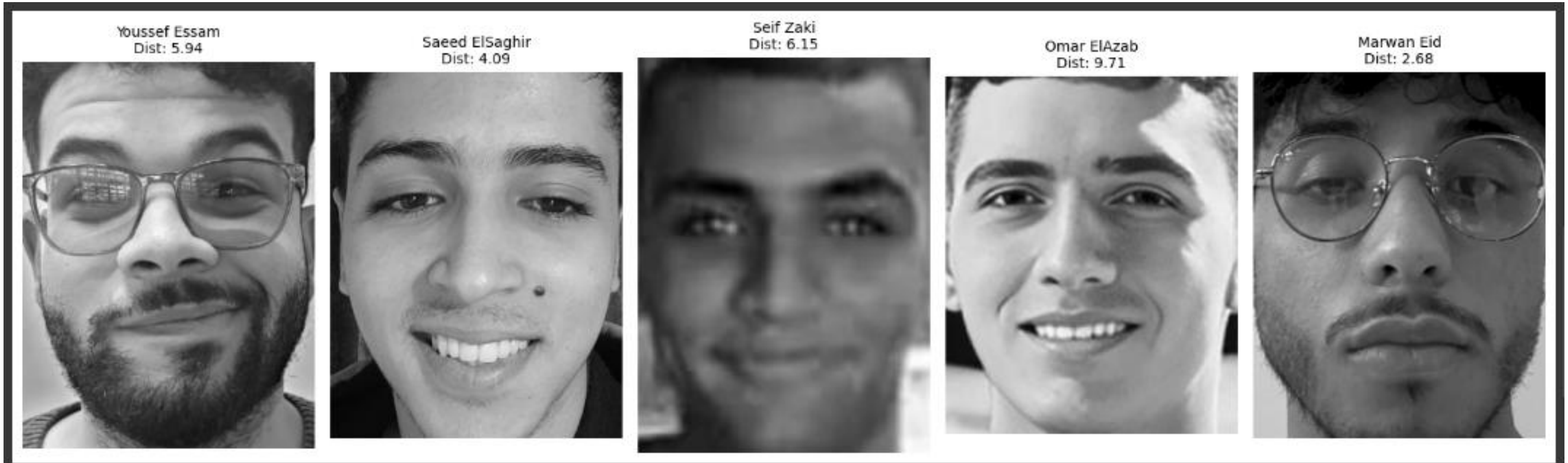
```
[ ] # Function to test all images in a folder
    def test_all_images_in_folder(folder_path, target_size, mean_face, eigenfaces, projections, labels, label_names, threshol
        test_images = []
        results = []

        # Load all images from the test folder
        for filename in os.listdir(folder_path):
            img_path = os.path.join(folder_path, filename)
            img = cv2.imread(img_path, cv2.IMREAD_GRAYSCALE)
            if img is not None:
                img_resized = cv2.resize(img, target_size) / 255.0
                test_images.append((img, img_resized.flatten()))

        # Recognize each image
        for img, img_flat in test_images:
            test_projection = np.dot(img_flat - mean_face, eigenfaces.T)
            recognized_name, distance = recognize_person(test_projection, projections, labels, label_names, threshold)
            results.append((img, recognized_name, distance))

        # Plot results
        fig, axes = plt.subplots(1, len(results), figsize=(15, 5))
        for ax, (img, name, dist) in zip(axes, results):
            ax.imshow(img, cmap='gray')
            ax.set_title(f"{name}\nDist: {dist:.2f}", fontsize=10)
            ax.axis('off')
        plt.tight_layout()
        plt.show()
```



Omar El-Azab
ID: 9710

Model Simulation

```
# Test with a new image
test_image_path = "/content/drive/MyDrive/Projects/Principal Component Analysis-Based Face Recognition/model/test"
test_folder_path = test_image_path
test_all_images_in_folder(test_folder_path, target_size, mean_face, eigenfaces, projections, labels, label_names)
```

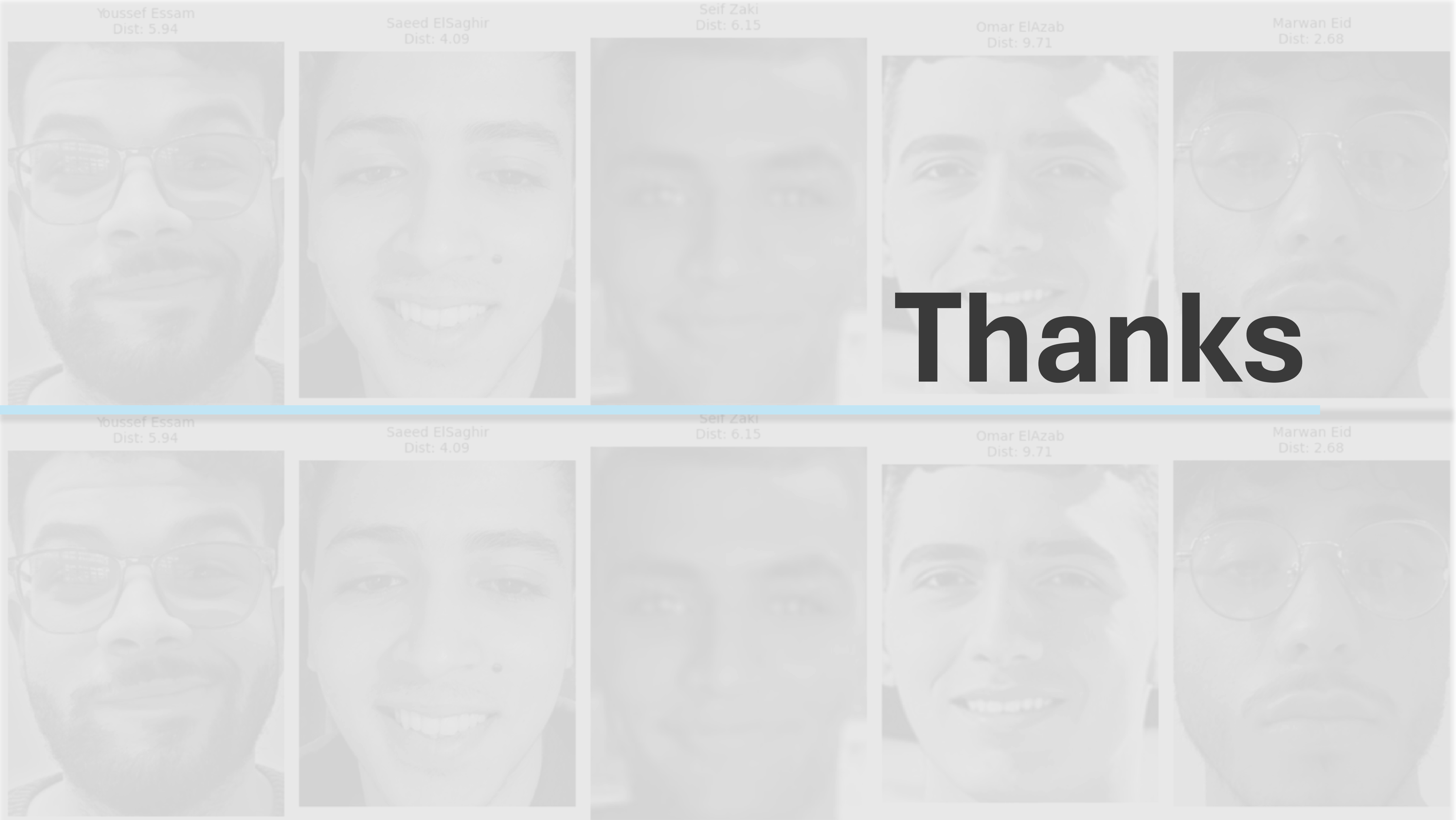


Omar El-Azab
ID: 9710

جامعة الإسكندرية
ALEXANDRIA
UNIVERSITY

GitHub Repository

Scan QR Code


Youssef Essam
Dist: 5.94
Saeed ElSaghir
Dist: 4.09
Seif Zaki
Dist: 6.15
Omar ElAzab
Dist: 9.71
Marwan Eid
Dist: 2.68

Thanks

CCE
Class ‘28

جامعة الإسكندرية
ALEXANDRIA
UNIVERSITY
كلية الهندسة
Faculty of Engineering