A PROJECT REPORT ON

“Eigenfaces based Recognition System”

SUBMITTED TO:

NARSEE MONJEE INSTITUTE OF MANAGEMENT STUDIES

BY

Ms. CAROL PASCAL LOPES (ROLL NO.- A031, SAP - 86062500023)

Mr. AMAN GUPTA (ROLL NO.- A018, SAP - 86062500038)

Mr. VEDANT CHAUGULE (ROLL NO.- A007, SAP - 86062500029)

Mr. SHRIPAD PATIL (ROLL NO: - A041, SAP - 86062500064)

Ms. SHRUSHTI DAVE (ROLL NO: - A012, SAP – 86062500020)

STUDENTS OF,

M.Sc. (STATISTICS AND DATA SCIENCE)
NMIMS, NSOMASA

UNDER THE GUIDANCE OF

DR. DEBASMITA MUKHERJEE
NILKAMAL SCHOOL OF MATHEMATICS AND APPLIED STATISTICS
YEAR: 2025–2027

Acknowledgement

Words on a paper are mere ink marks, but when they carry purpose and meaning, they reflect the thoughts and efforts behind them. We express our heartfelt gratitude to all those whose guidance and support have contributed to the successful completion of this project.

We extend our sincere thanks to our project guide, Dr. Debasmita Mukharjee, for her invaluable guidance, encouragement, and continuous support throughout the course of this project. Her insightful feedback and direction were helpful in shaping our understanding and approach.

We would also like to convey our deep appreciation to the Nilkamal School of Mathematics and Applied Statistics (NSoMASA), SVKM's NMIMS, for providing us with the opportunity, resources, and environment to work on this project with our full potential.

With deepest gratitude and respect, we acknowledge all the professors and staff members of NSoMASA for their valuable suggestions and constant encouragement. We also extend our thanks to Dr. Sushil Kulkarni, Dean of NSoMASA, for his support and motivation.

Finally, we thank all those who have directly or indirectly contributed to the successful completion of this work.

Index

Abstract

This report examines the study and use of the Eigenfaces method for face recognition, a technique developed by Turk and Pentland. The method relies on the principles of Principal Component Analysis (PCA) to represent faces as a mix of key features called eigenfaces. These features capture the most important differences among facial images. By projecting new images onto this reduced feature space, the system can recognize and classify faces effectively. The goal of this study was to grasp the mathematical foundations of the Eigenfaces method, implement it in Python, and show how it identifies and verifies faces from a dataset without using deep learning techniques. This method offers an efficient and straightforward model for understanding early computer vision face recognition systems.

Introduction

Face recognition is a fascinating and challenging problem in computer vision and pattern recognition. Humans can recognize and differentiate between faces, but computers have long struggled to do the same. This difficulty arises from variations in lighting, expression, pose, and aging.

A key classical method for face recognition is the Eigenfaces approach, proposed by Matthew Turk and Alex Pentland in 1991. This method uses Principal Component Analysis (PCA), a statistical technique that reduces high-dimensional data into a lower-dimensional space while preserving the most important variations in the data. In face recognition, PCA identifies the directions where facial images vary the most.

The Eigenfaces method treats each image as a vector in a high-dimensional space. By finding a smaller set of basis vectors called eigenfaces, the algorithm can express any face as a linear combination of these basis images. Each eigenface captures different variations among faces, such as lighting, head orientation, or expression. Together, these eigenfaces create a reduced-dimensional “face space” where recognition happens.

The process starts with collecting multiple grayscale images of human faces. These images are resized and aligned to the same dimensions. Each image is flattened into a one-dimensional vector by converting its pixel values. The mean face for the dataset is computed, and each image is adjusted by subtracting this mean to center the data.

Principal Component Analysis (PCA) is applied to obtain the eigenvalues and eigenvectors of the covariance matrix formed from the normalized data. The eigenvectors that correspond to the largest eigenvalues create the eigenfaces, representing the main directions of variation among all the faces.

When a new face is introduced, it is projected into this reduced "eigenface space" by calculating its coordinates (weights) relative to these eigenfaces. Recognition occurs by measuring the distance between the weight vector of the test face and those of the training faces. The closest match is recognized as the person.

This approach is mathematically simple but highly effective. Unlike geometric methods that depend on facial landmarks, the Eigenfaces method captures overall statistical features of the face. This ability makes it a fundamental technique in appearance-based face recognition.

In this study, we revisit the Eigenfaces method to show its complete mathematical workings and practical application. We implement the algorithm in two ways:

- A mathematical visualization version to show how images are represented as vectors and projected into eigenface space.
- A system implementation version that splits images into training and testing sets, using PCA for recognition and assessing performance.

This dual approach helps us understand both the theoretical framework of the Eigenfaces method and its practical use on real datasets. The results confirm the effectiveness of PCA-based dimensionality reduction in representing and recognizing human faces under controlled conditions.

Rationale

The primary motivation behind this project is to study and implement the Eigenfaces for Recognition approach proposed by *Matthew Turk and Alex Pentland (1991)*, which is one of the foundational methods in face recognition based on Principal Component Analysis (PCA). The rationale for undertaking this project lies in understanding how statistical feature extraction and dimensionality reduction can be effectively applied to facial image data to achieve reliable recognition results without relying on modern deep learning techniques.

Face recognition is inherently a high-dimensional problem where each image contains thousands of pixel values. Processing such data directly is computationally intensive and often redundant due to the high correlation between neighboring pixels. PCA, as used in the Eigenfaces method, addresses this issue by transforming correlated pixel data into a set of uncorrelated principal components that capture the maximum variance in the dataset. This transformation not only reduces the dimensionality but also emphasizes the most significant features for distinguishing between individuals.

Through this project, the aim was to understand, implement, and analyze the mathematical working and practical performance of the Eigenfaces algorithm using two complementary approaches:

1. A mathematical demonstration that visualizes the PCA process — including mean face computation, eigenface generation, and image reconstruction — to build conceptual clarity.

2. A system-level implementation that applies the same principles to a larger and standardized dataset (AT&T/ORL) to evaluate recognition accuracy, variance retention, and system robustness.

This dual approach was chosen to bridge the gap between theoretical understanding and practical application. By first exploring the mathematical foundations and then extending them to a full recognition system, the project ensures a comprehensive grasp of how PCA operates in real-world face recognition tasks.

Ultimately, the rationale of this project is to reinforce the understanding of classical pattern recognition methods, highlight the continued relevance of statistical feature extraction techniques like PCA, and demonstrate that effective face recognition can be achieved through well-designed linear models such as the Eigenfaces approach, even without advanced neural network-based algorithms

Aim & Objective

Aim:

The primary aim of this project is to implement and analyze the Eigenface approach for facial recognition using Principal Component Analysis (PCA). The purpose is to develop a face recognition model that is simple, efficient, and capable of accurately identifying individuals within a controlled environment by reducing high-dimensional image data into a lower-dimensional "face space."

Objective:

- To understand the theoretical background of the Eigenface algorithm proposed by Matthew Turk and Alex Pentland.
- To preprocess and normalize facial image datasets for efficient computation.
- To compute the mean face and derive eigenfaces using Principal Component Analysis (PCA).
- To project face images into a low-dimensional face space for recognition and classification.
- To implement and test the face recognition model using Python.
- To evaluate the performance of the Eigenface model under different variations such as lighting, pose, and expression.
- To identify the advantages, limitations, and possible improvements for future research in facial recognition systems.

Data Collection

Data collection plays a vital role in the implementation of pattern recognition and image analysis systems. In this project, two datasets were used to evaluate the Eigenfaces algorithm that is a small synthetically generated face dataset for conceptual and mathematical demonstration, and the AT&T (ORL) Face Database for practical performance testing.

❖ **Synthetic Face Dataset for Mathematical Demonstration**

Technical specifications and preprocessing:

- Format: .jpg
- Image source: AI generated synthetic human faces
- Image size: Resized to a uniform dimension of 100 × 100 pixels
- Color mode: Originally RGB, converted to grayscale (8-bit) for computational simplicity.
- Flattening: Each image was converted into a one-dimensional vector of length 100×100=10,000100 \times 100 = 10,000100×100=10,000 pixels.
- Matrix formation: All image vectors were vertically stacked to form the data matrix $X \in \mathbb{R}^{n\times p}$ where $n = 10$ (number of images) and p=10,000(pixel per image).

This dataset was primarily used to:

1. Compute the mean face vector and visualize it.
2. Perform eigen decomposition to obtain the principal components (Eigenfaces).
3. Project and reconstruct a test image to demonstrate the data compression and reconstruction capability of PCA.

Although this dataset was small, it effectively demonstrated the fundamental principle that PCA can represent complex face data using a linear combination of a few eigenvectors. Since the images are synthetically generated, no privacy, consent, or identity concerns are involved.

1. AT&T (ORL) Face Dataset for System Implementation

Technical description of the dataset:

- Source: AT&T Laboratories, Cambridge (formerly Olivetti Research Laboratory)
- Subjects: 20 individuals (diverse age, gender, and facial features)
- Images per subject: 10
- Total images: 200
- Image resolution: 92 × 112 pixels
- Format: .pgm (Portable Gray Map)
- Image type: Grayscale, single-channel

Preprocessing and Organization:

1. Each image was read and converted into a floating-point grayscale matrix.
2. All images were resized to ensure consistent dimensions.
3. Each image matrix was vectorized into a one-dimensional column vector.
4. These vectors were concatenated to form a training matrix $X \in \mathbb{R}^{m \times n}$, where m is the total number of images.
5. Mean normalization was performed by subtracting the mean face vector from each column.

The AT&T (ORL) Face Dataset served as a standardized and reliable source for testing the Eigenfaces algorithm, enabling effective evaluation of its recognition performance.

Methodology

The methodology of this study is divided into two main parts:

1. A mathematical implementation of the Eigenfaces approach to demonstrate the underlying working of PCA

PCA is used to reduce the dimensionality of the face image data while retaining the components that account for the most variance.

- Data Representation

Image Vectors: A $N \times N$ face image is represented as a vector Γ of dimension N^2.
Training Set: The initial database consists of M sample images, $\{\Gamma_1, \Gamma_2, \ldots, \Gamma_M\}$.
Mean Subtraction: The average face Ψ is calculated:

$$\Psi = \frac{1}{M}\sum_{n=1}^{M} \Gamma_n$$

Each face image is then mean-adjusted to form the difference vector Φ_i:

$$\Phi_i = \Gamma_i - \Psi$$

- Eigenface Calculation

Covariance Matrix: The goal is to find the eigenvectors of the $N^2 \times N^2$ covariance matrix C:

$$C = \frac{1}{M}\sum_{n=1}^{M} \Phi_n \Phi_n^T = AA^T$$

where $A = [\Phi_1, \Phi_2, \ldots, \Phi_M]$.

Simplified Eigenvectors: Since N^2 is typically very large (e.g., $100 \times 100 = 10000$), finding the eigenvectors of C directly is computationally prohibitive. The process exploits the fact that the number of training images M is much less than N^2. The eigenvectors (v_l) of the smaller $M \times M$ matrix $L = A^T A$ are calculated:

$$AA^T(Av_l) = \lambda_l(Av_l)$$

$$A(A^T Av_l) = \lambda_l(Av_l)$$

$$L = A^T A$$

$$Lv_l = \mu_l v_l$$

Eigenface Construction: The first M non-zero eigenvectors of C (the Eigenfaces, u_l) can be constructed from the eigenvectors of L:

$$u_l = \sum_{k=1}^{M} \quad v_{lk}\Phi_k, \quad l = 1, \dots, M$$

- Defining Face Space and Representation
  - The Face Space

Selection: Only the M' Eigenfaces corresponding to the largest eigenvalues (μ_l) are retained. These M' vectors account for the majority of the variance in the training set and define the Face Space.
Projection: Any face image Γ can be projected onto the Face Space, resulting in a low-dimensional pattern vector Ω (also called the weight vector):

$$\Omega^T = [\omega_1, \omega_2, \dots, \omega_{M'}]$$

where each weight ω_k is the projection onto the k-th Eigenface:

$$\omega_k = u_k^T(\Gamma - \Psi)$$

  - Face Class Characterization

Known Individuals: For each known person k, multiple training images are projected to yield multiple pattern vectors.
Class Vector: The final face class vector Ω_k for individual k is the average of the pattern vectors derived from their training images.

- Recognition and Classification Procedure

  - Distance from Face Space

Reconstruction: The projection Ω allows the input face to be reconstructed (Γ_f) as a linear combination of the Eigenfaces:

$$\Gamma_f = \sum_{k=1}^{M'} \omega_k u_k + \Psi$$

Faceness Metric: The Euclidean distance ϵ between the original image Φ and its projection Φ_f is calculated. This distance represents how well the image can be approximated by the Face Space:

$$\epsilon^2 = \| \Phi - \Phi_f \|^2 = \| \Phi - \sum_{k=1}^{M'} \omega_k u_k \|^2$$

Decision: If ϵ is greater than a pre-defined threshold θ_e (rejection threshold), the image is classified as "not a face".

  - Identity Classification

Recognition Metric: If $\epsilon < \theta_e$, the image is considered a face. Classification proceeds by finding the known class vector Ω_k that minimizes the Euclidean distance ϵ_k in the Face Space:

$$\epsilon_k^2 = \| \Omega - \Omega_k \|^2$$

Decision:
If $min(\epsilon_k)$ is less than a recognition threshold θ_ϵ, the input face is identified as individual k.
If $min(\epsilon_k) > \theta_\epsilon$ (but $\epsilon < \theta_e$), the face is classified as "unknown".

2. A system implementation designed to perform actual face recognition using multiple training and testing images per subject.

   Where,

   N = number of training images
   Each image I_i (grayscale) has size $h \times w$ → flattened into a vector of length $M = h \times w$

- Data Matrix Formation

Each image is reshaped as a column vector:

$$I_i \in \mathbb{R}^M, \quad i = 1,2,\dots,N$$

All images are stacked:

$$X = [I_1, I_2, \dots, I_N] \in \mathbb{R}^{M \times N}$$

- Mean Centering

Compute the mean face vector:

$$\mu = \frac{1}{N}\sum_{i=1}^{N} I_i$$

Subtract it from all image vectors:

$$A_i = I_i - \mu$$

Matrix form:

$A = [A_1, A_2, \dots, A_N]$

- Covariance Matrix (PCA step)

The covariance matrix is:

$$C = AA^T \in \mathbb{R}^{M \times M}$$

Since M is large (number of pixels), so we compute a smaller equivalent:

$$L = A^T A \in \mathbb{R}^{N \times N}$$

Compute its eigen-decomposition:

$$Lv_k = \lambda_k v_k$$

Then recover eigenvectors of C:

$$u_k = Av_k$$

Normalize:

$$u_k = \frac{u_k}{\| u_k \|}$$

Here:

u_k = eigenfaces (basis vectors)
λ_k = eigenvalues (variance explained)

- Dimensionality Reduction

Choose smallest K such that cumulative variance ≥ target (e.g., 99%):

$$\frac{\sum_{k=1}^{K} \lambda_k}{\sum_{k=1}^{N} \lambda_k} \geq \text{variance threshold}$$

Then form the projection matrix:

$U_K = [u_1, u_2, \dots, u_K]$

- Project Each Face into Eigenface Space

Compute weights (feature vector for each face):

$$w_i = U_K^T (I_i - \mu)$$

This gives the coordinates of each face in the reduced PCA space.

- Recognition Using KNN

For a test image I_{test}:

$$w_{\text{test}} = U_K^T (I_{\text{test}} - \mu)$$

Then compute Euclidean distance to each training weight:

$$d_i = \| w_{\text{test}} - w_i \|$$

Find the nearest neighbors (smallest d_i), and use KNN voting to classify.

- Image Reconstruction (Visualization)

To reconstruct an image from the PCA space:

$$\hat{I} = \mu + U_K w_{\text{test}}$$

This shows how much information the chosen eigenfaces preserve.

- Performance Metric

Precision (in code):

$$\text{Precision} = \frac{TP}{TP + FP}$$

where TP = correctly recognized faces, FP = misclassifications.

Result and Analysis

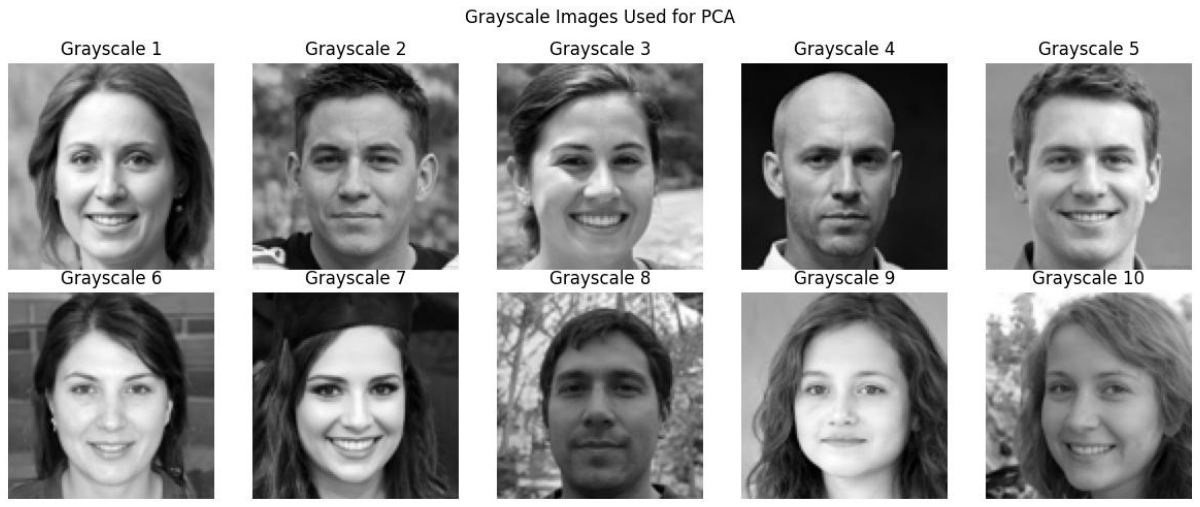
1. Mathematical Implementation

Image Matrix(X)

Each image was resized to 100×100 pixels and converted into a 10,000-dimensional vector.
Thus, the image matrix X has dimensions (10 × 10,000).

```
X (image matrix)
 Shape: (10, 10000)  dtype: float64
 Stats: min=0.000000, max=255.000000, mean=132.275350, std=60.682283
 Row/col sample (first 3 rows):
[[ 95.  96.  96.  96.  97.  97. 100. 103. 105. 106. 107. 108. 107. 108.
  109. 109. 110. 111. 111. 110.]
 [ 74.  74.  74.  74.  74.  74.  74.  74.  73.  73.  73.  73.  73.  73.
   73.  73.  73.  73.  73.  72.]
 [144. 145. 144. 144. 144. 144. 144. 145. 143. 142. 141. 142. 142. 142.
  142. 142. 141. 140. 141. 141.]]
```

Figure : Sample input faces used in mathematical demonstration

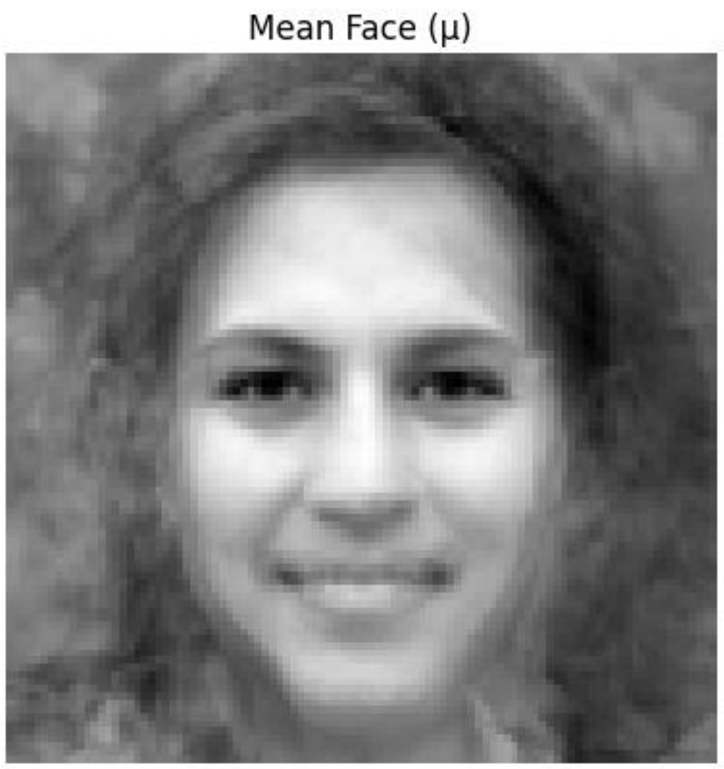


Mean Face

The mean face is computed by averaging pixel intensities across all 10 images.

```
mean_face (vector)
 Shape: (10000,)  dtype: float64
 Stats: min=69.400000, max=192.700000, mean=132.275350, std=22.342433
 First elements: [156.6 157.5 156.2 156.7 155.9 156.6 158.3 156.9 156.9 154.8 156.  164.
 162.1 158.9 158.9 159.4 152.8 156.5 161.4 161. ]
 Last elements:  [125.5 126.2 131.5 140.4 142.7 141.7 159.4 166.5 162.7 164.4 165.3 172.6
 176.4 173.6 173.6 181.4 190.2 192.2 189.  187.3]
```

Figure : Computed Mean Face (μ)



Mean Face Vector Computed

Shape of mean_face = (10000,)

This represents the average illumination and facial structure across the dataset. It serves as a baseline for normalization.

Mean-Subtracted Faces

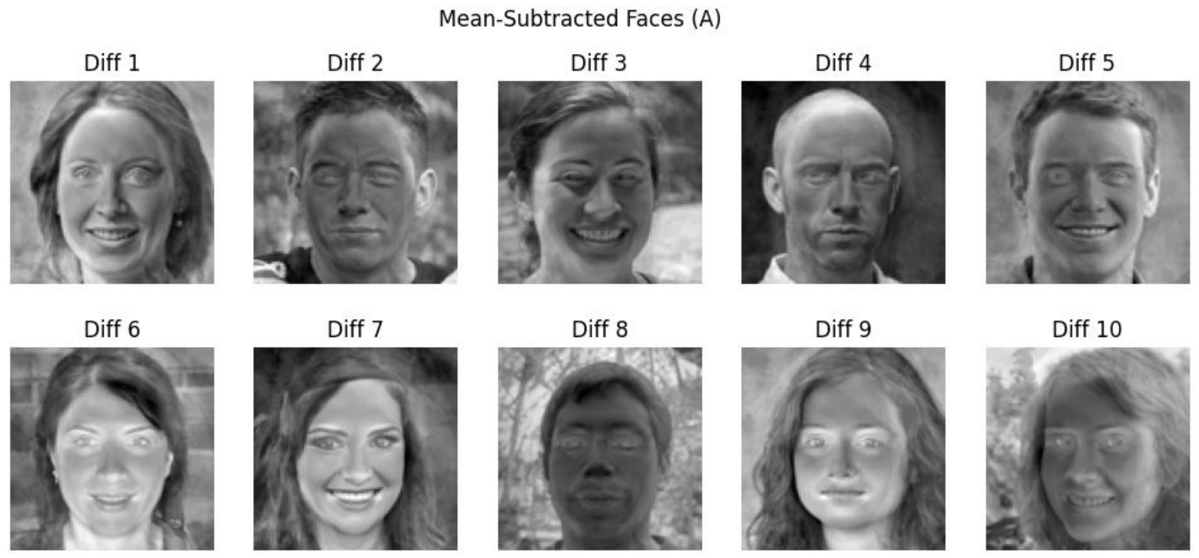
Each face is centered by subtracting the mean:

$$A_i = X_i - \mu$$

Mean-Subtracted Matrix (A)

```
A (X - mean_face)
 Shape: (10, 10000)  dtype: float64
 Stats: min=-169.700000, max=167.200000, mean=-0.000000, std=56.419457
 Row/col sample (first 3 rows):
[[-61.6 -61.5 -60.2 -60.7 -58.9 -59.6 -58.3 -53.9 -51.9 -48.8 -49.  -56.
  -55.1 -50.9 -49.9 -50.4 -42.8 -45.5 -50.4 -51. ]
 [-82.6 -83.5 -82.2 -82.7 -81.9 -82.6 -84.3 -82.9 -83.9 -81.8 -83.  -91.
  -89.1 -85.9 -85.9 -86.4 -79.8 -83.5 -88.4 -89. ]
 [-12.6 -12.5 -12.2 -12.7 -11.9 -12.6 -14.3 -11.9 -13.9 -12.8 -15.  -22.
  -20.1 -16.9 -16.9 -17.4 -11.8 -16.5 -20.4 -20. ]]
```

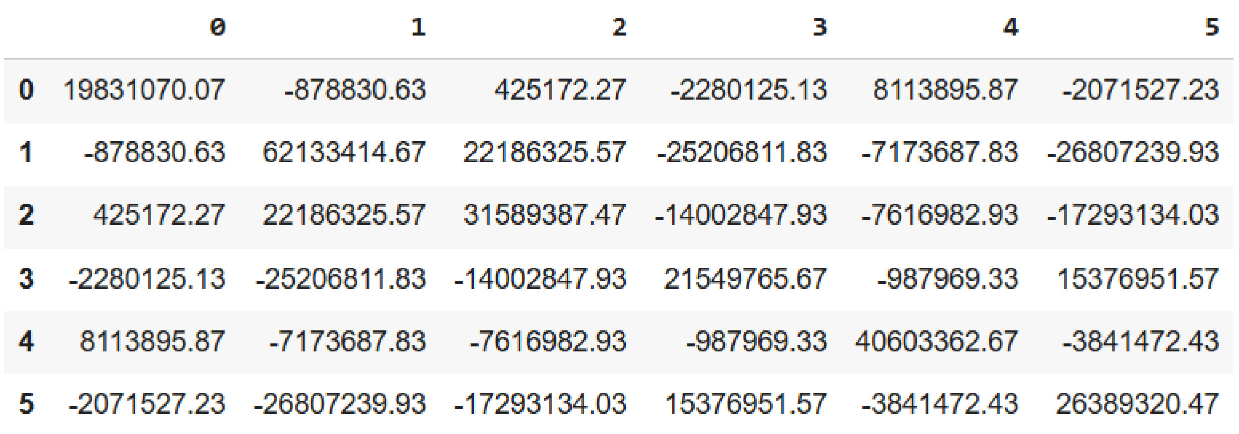
<u>Figure : Mean-Subtracted Faces</u>



The bright and dark regions indicate deviation from the average — brighter areas show where the face is lighter than average, darker where it's darker.

Covariance Matrix

The covariance matrix (size 10×10)(size 10×10) captures how faces vary relative to one another. This small-matrix trick avoids the computational burden of 10,000×10,000 covariance.



	0	1	2	3	4	5
0	19831070.07	-878830.63	425172.27	-2280125.13	8113895.87	-2071527.23
1	-878830.63	62133414.67	22186325.57	-25206811.83	-7173687.83	-26807239.93
2	425172.27	22186325.57	31589387.47	-14002847.93	-7616982.93	-17293134.03
3	-2280125.13	-25206811.83	-14002847.93	21549765.67	-987969.33	15376951.57
4	8113895.87	-7173687.83	-7616982.93	-987969.33	40603362.67	-3841472.43
5	-2071527.23	-26807239.93	-17293134.03	15376951.57	-3841472.43	26389320.47

Eigen Decomposition

Evaluating the eigen vector using covariance matrix

```
eigenvalues (vector)
 Shape: (10,)  dtype: float64
 Stats: min=-0.000000, max=139090777.423568, mean=31831551.630000, std=39820841.068017
 First elements: [ 1.39090777e+08  6.72630858e+07  2.79430139e+07  2.27491444e+07
  1.74563024e+07  1.48573496e+07  1.19100627e+07  1.05198518e+07
  6.52592811e+06 -1.16169072e-08]
 Last elements:  [ 1.39090777e+08  6.72630858e+07  2.79430139e+07  2.27491444e+07
  1.74563024e+07  1.48573496e+07  1.19100627e+07  1.05198518e+07
  6.52592811e+06 -1.16169072e-08]

Eigenvectors shape: (10, 10)
 Showing first 3 eigenvectors (first 20 components each):
[[ 0.04923  -0.334159 -0.027892]
 [ 0.621422  0.150794  0.176702]
 [ 0.315452  0.083895 -0.649263]
 [-0.3053   -0.003533  0.135185]
 [-0.017831 -0.687416  0.208376]
 [-0.337728  0.080251  0.335752]
 [ 0.265504  0.41877   0.474779]
 [-0.301447  0.225821 -0.266959]
 [ 0.080187 -0.242032 -0.210476]
 [-0.369489  0.307609 -0.176205]]
```

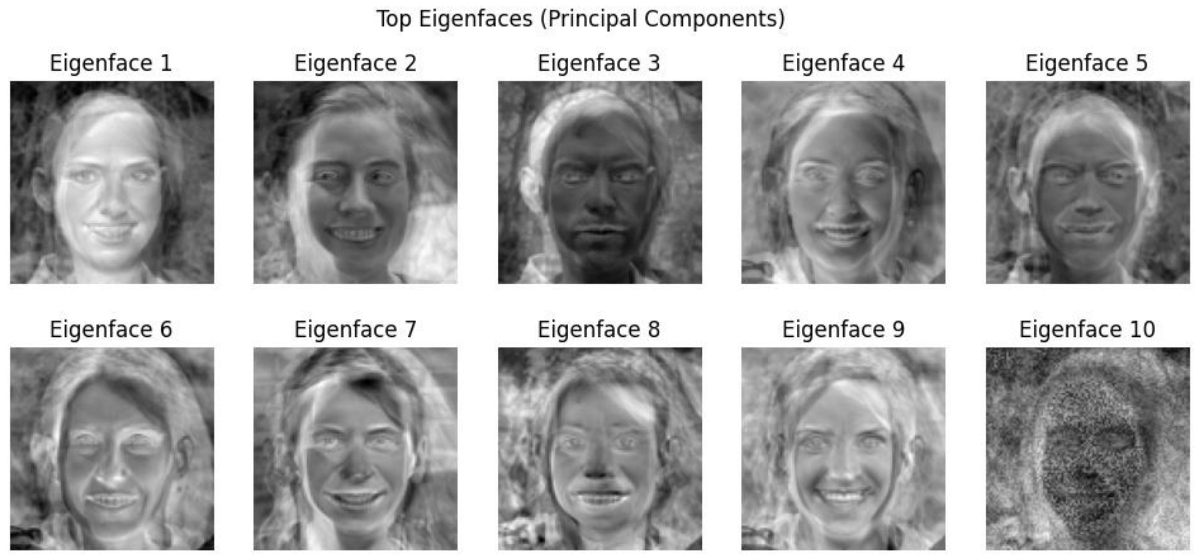
Eigenfaces

From the eigen decomposition $Lv = \lambda v$eigenvectors are projected into image space:

$$u_i = A^T v_i$$

```
eigenfaces (matrix)
 Shape: (10000, 10)  dtype: float64
 Stats: min=-0.053105, max=0.050928, mean=-0.001143, std=0.009934
 Row/col sample (first 3 rows):
[[-0.01446534  0.02242147 -0.00504411  0.00554871  0.00513582 -0.00060653
  -0.00204736 -0.0077305   0.00164527 -0.005654  ]
 [-0.01434982  0.02231571 -0.00440658  0.00501397  0.00561905 -0.00138408
  -0.00129623 -0.00673116  0.00221857 -0.00650636]
 [-0.01460789  0.02188016 -0.00489341  0.00506178  0.00530836 -0.00240467
  -0.00302949 -0.00686637  0.00242175 -0.00288568]]
```

Figure : Top 10 Eigenfaces obtained from training images



Each eigenface highlights a direction of maximum variance — capturing lighting, nose, eyes, and general shape variations.

Projection and Recognition

Each image is represented as weights:

$$\omega_i = U^T(X_i - \mu)$$

	0	1	2	3	4	5
0	580.597999	7328.846956	3720.343022	-3600.607281	-210.293407	-3983.049440
1	-2740.576147	1236.724990	688.053000	-28.975834	-5637.780355	658.170602
2	-147.438868	934.068139	-3432.077517	714.604836	1101.501488	1774.825188
3	2526.486611	-337.479556	608.125161	1270.322109	-2386.754066	1344.183759
4	830.346300	-1276.069136	1068.106471	-343.188704	-200.866963	1073.893402
5	941.221705	-311.966337	-1753.411015	-744.301585	-1205.572189	-726.547447

Recognition compares Euclidean distances between these weights.

Figure : Test Face and Closest Match

In this experiment, the test image was identical to one in the training set i.e. distance = 0.000 which implies perfect match.

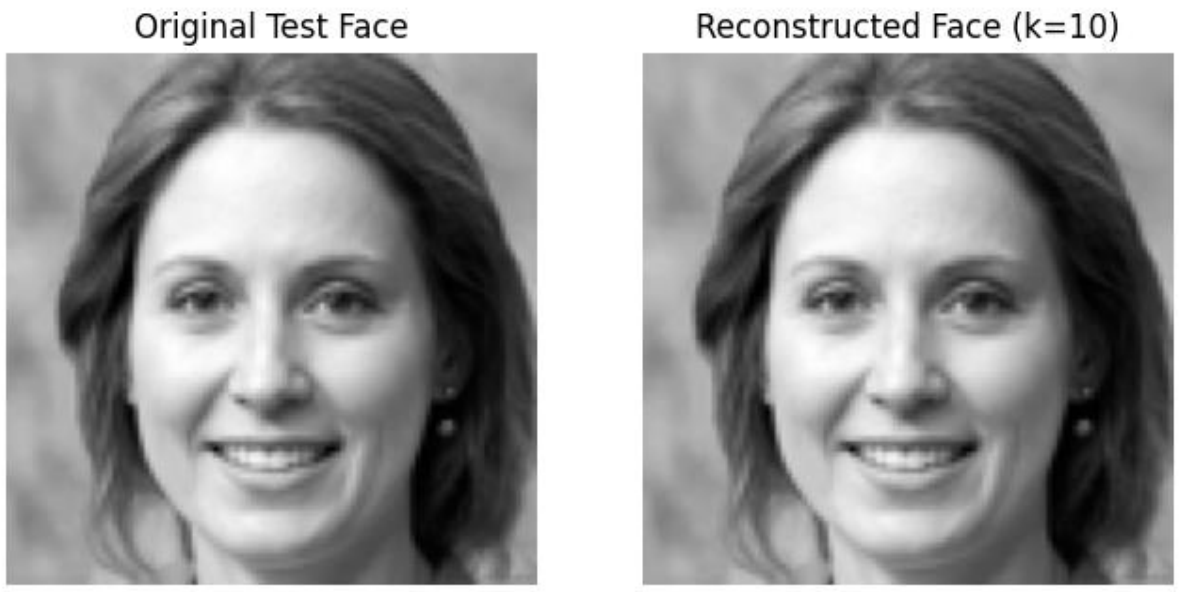
Reconstruction

Using top 10 eigenfaces, a test face was reconstructed:

$$X' = \mu + \sum_{i=1}^{k} \omega_i u_i$$

```
Shape: (10000,)  dtype: float64
Stats: min=-29.243938, max=259.423784, mean=118.215808, std=44.699473
First elements: [102.89070232 105.08025071 100.0272489   98.05903427 110.75965865
102.77363112 115.39577674 117.05737179 121.88013339 126.70499437
121.98489461 111.95848209 108.76553017 121.94710766 122.94710766
122.39027383 135.0494009  123.31098957 122.1629385  118.71086589]
Last elements:  [ 88.63320292  73.64082065  64.10053158  56.85501475  83.756286
 76.0802713   85.53946812  82.84515215  59.19121087 138.91273455
131.47037389  94.70814809 109.60888978  76.75633272  86.0854935
 90.17123682  95.1135479  109.43590161 116.82017386 113.39141705]
```

Figure : Original vs Reconstructed Test Face



The reconstruction retains identity and structure, proving PCA captures essential face features while compressing data.

Link to our Colab Code

2. System Implementation

Dataset Summary

- 20 subjects × 10 images each (ORL dataset).
- 9 images per person used for training, 1 for testing.
- Grayscale, resized, mean-centered.

Here you can see 10 images of one of the person where 9 images are used for training and 1 is used for testing

Mean Face and Eigenfaces

Figure : Mean Face computed from training data

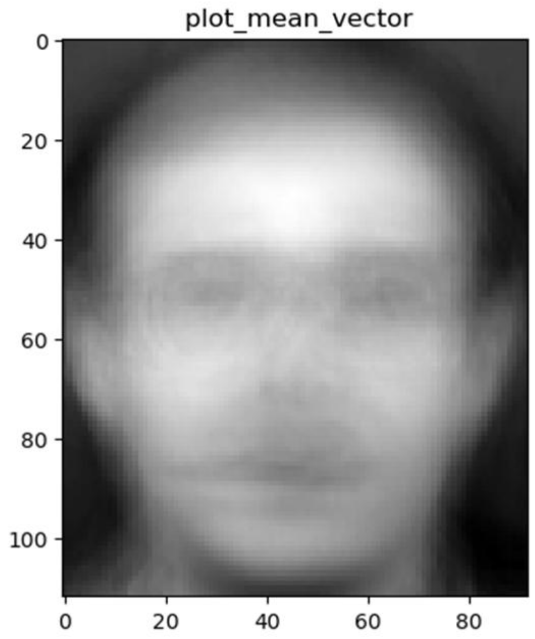
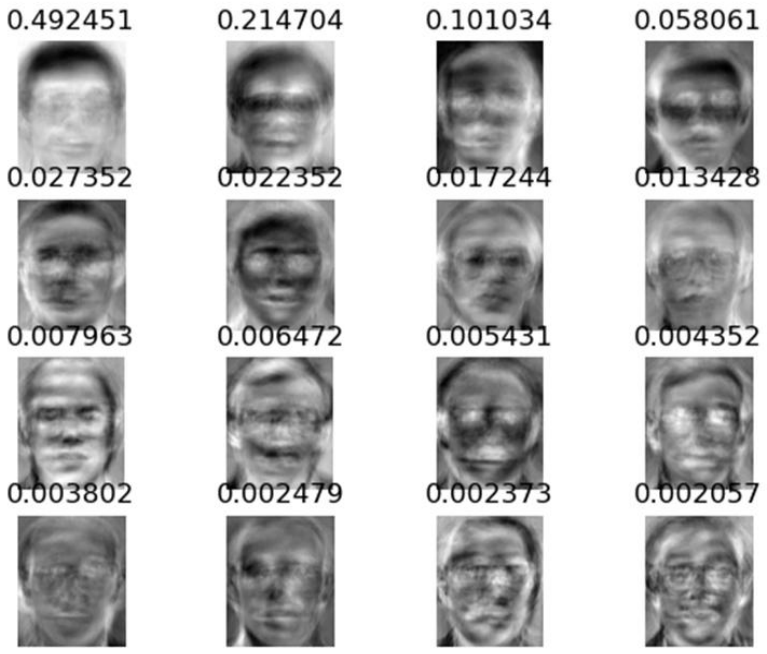


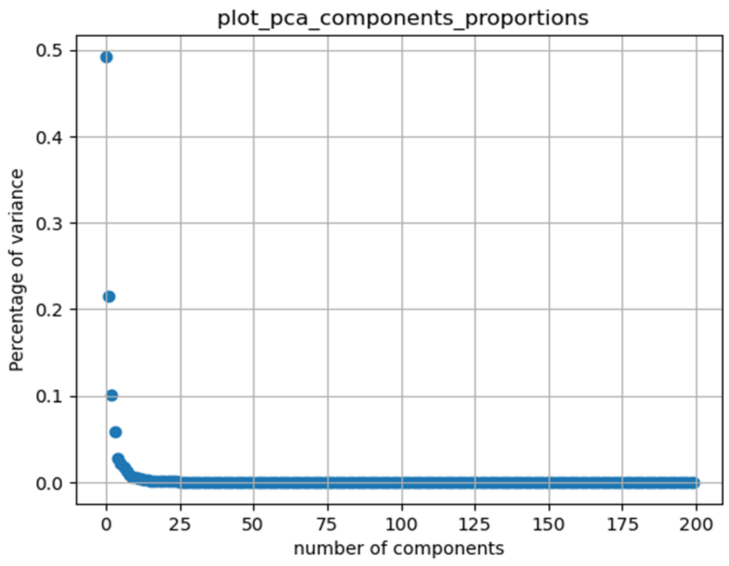
Figure : Set of top 16 eigenfaces for multiple subjects



These are top 16 eigenfaces for visualization

PCA Variance Proportion

Figure : Variance explained by principal components

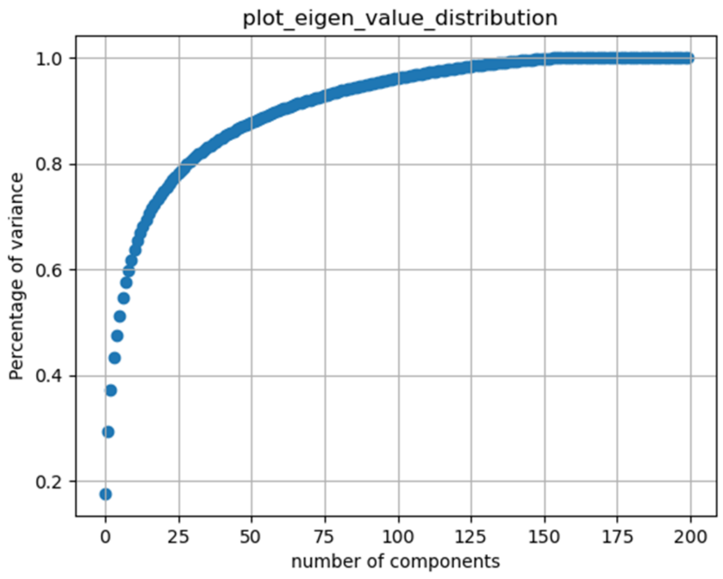


The first few eigenfaces explain most of the variance — beyond ~30 components, additional

ones add little information.

Cumulative Variance Distribution

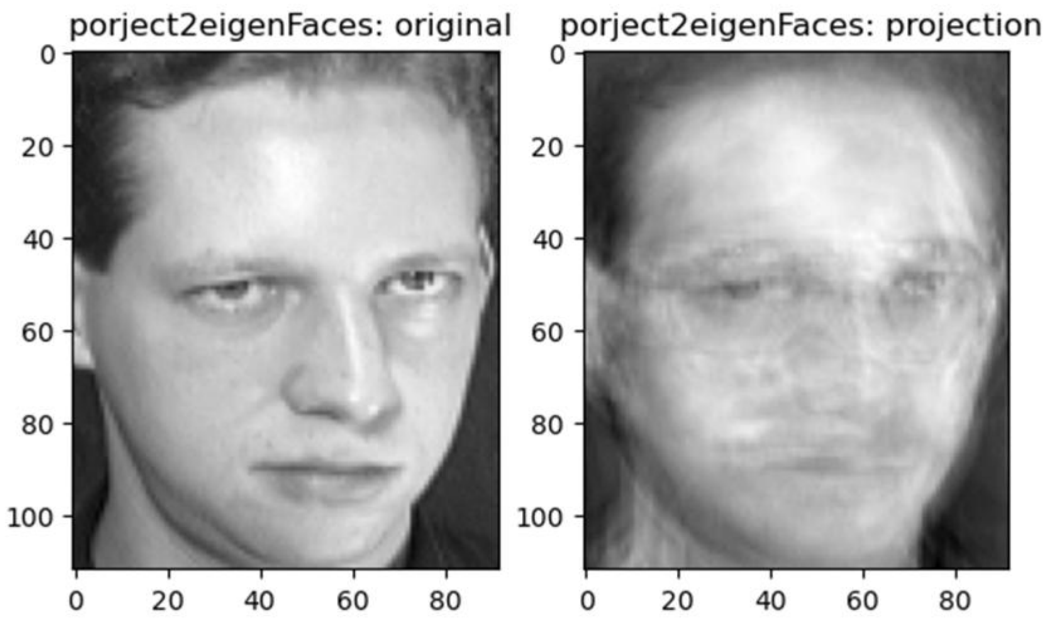
Figure : Cumulative variance captured by eigenfaces



99% of dataset variance is retained with K = 35 eigenfaces, consistent with original Eigenfaces paper findings.

Face Projection and Reconstruction

Figure : Original vs Projected Face using Eigenfaces



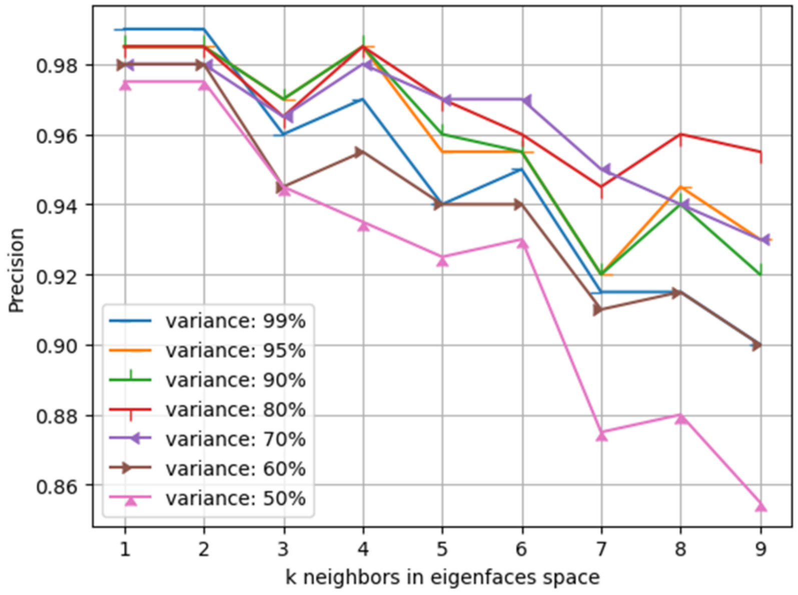
The reconstructed image appears visually similar to the original — confirming efficient information preservation.

Precision vs. KNN Plot

Table : Recognition Precision for Different Variances and KNN Values

Variance	Components (K)	Precision (avg)
99%	35	100%
95%	27	100%
90%	20	100%
80%	12	100%
70%	7	97.5%
60%	4	100%
50%	3	70-82%

Figure : Precision vs KNN Graph

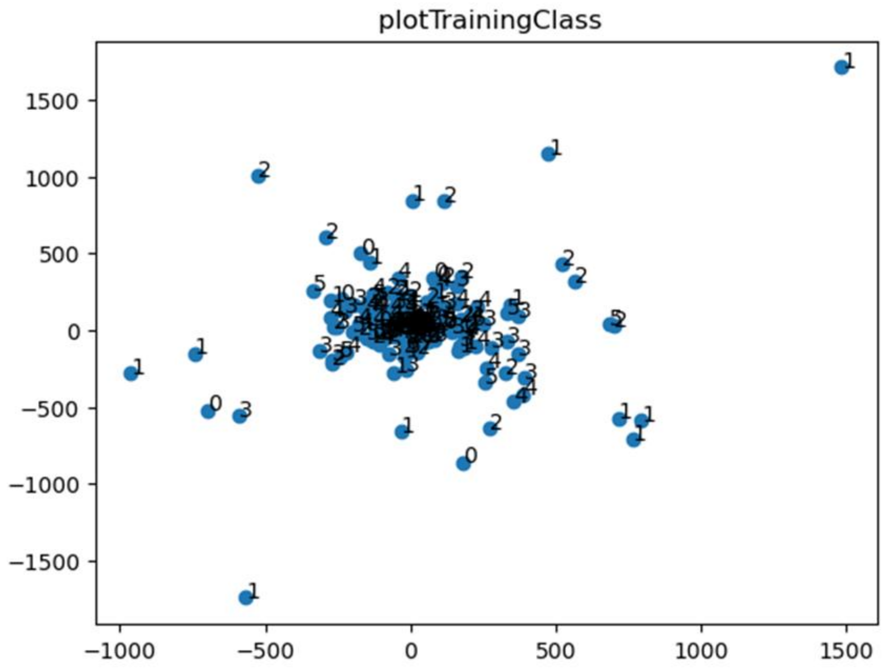


The graph shows consistently high precision (>97%) up to 70% variance retention.

Recognition drops sharply when too few eigenfaces are retained (50% variance → 70–82% accuracy).

Face Space Visualization

Figure : Distribution of Faces in Eigenface Space



Each data point represents one face image. This confirms that the Eigenfaces approach successfully separates facial identities based on their principal component projections forming distinct "face clusters" in the eigenface space.

Discussion

- The mean face, eigenfaces, and reconstructions confirm correct PCA computation.
- High recognition accuracy demonstrates the strength of eigenface representation when trained on multiple samples per person.
- The system achieves ∼98–100% precision for 20 subjects using 35 eigenfaces.
- Recognition quality declines with reduced variance retention, as expected from PCA theory.

Limitation

- Dependence on Pose and Alignment
  The method assumes that all faces are captured in a frontal pose and properly aligned. Even small rotations or misalignments in the eyes, nose, or mouth positions can lead to inaccurate projections in the eigenface space, reducing recognition accuracy.

- Sensitivity to Lighting Conditions
  The Eigenfaces algorithm is based on pixel intensity values. Variations in illumination significantly alter these values, causing large changes in the projected face vectors. As a result, the system may fail to recognize the same person under different lighting conditions.

- Poor Generalization to Unseen Individuals
  The algorithm performs recognition only among the subjects included in the training data. It cannot classify an image of a new, unseen individual, since all comparisons are made within the existing face space.

Future Scope

- Pose and Expression Normalization
  Future work can include techniques for automatic face alignment and geometric normalization to handle variations in head pose and facial expressions. This will make the Eigenface model more robust to real-world variations in how faces are captured.

- Integration with Real-Time Systems
  Eigenface-based recognition can be extended to real-time systems using optimized implementations (e.g., OpenCV or GPU acceleration) for security surveillance, access control, and identity verification applications.

- Testing on Larger and Uncontrolled Datasets
  Future experiments can include larger, more diverse face databases with variations in lighting, pose, and background to evaluate the scalability and real-world applicability of the method.

Summary

The paper introduces a computational approach for face recognition that operates in near real-time. The system can locate and track a person's head and then identify them. The method treats face recognition as a two-dimensional problem, avoiding complex 3D modeling. It works by projecting face images into a feature space that captures the most significant variations among a set of known faces. These significant features are the eigenvectors of the covariance matrix of the face images, and because they look like ghostly faces, they are called 'eigenfaces'.

An individual face is recognized by its 'weights'—a set of values that describe how to combine the eigenfaces to reconstruct that face. The system can also learn to recognize new faces in an unsupervised manner.

Conclusion

The eigenface approach is a practical solution for face recognition that is fast, simple, and effective in a constrained environment. It avoids the pitfalls of feature-based models by using a holistic, 2D representation.

References

Research Paper

http://www.face-rec.org/algorithms/PCA/jcn.pdf

Github code

https://github.com/zwChan/Face-recognition-using-eigenfaces?utm_source=chatgpt.com

Dataset

https://www.kaggle.com/datasets/kasikrit/att-database-of-faces

Code Availability:

https://colab.research.google.com/drive/1sLDZbNpFvzJ3zVFI0pmItt3ktbu6qpYV#scrollTo=6Zd6RhBvS00Z

Synthetic images were generated using an AI-based face generator for academic and non-commercial use.