PROJEK BASED LEARNING 1

APLIKASI NILAI EIGEN DAN EIGENFACE PADA PENGENALAN WAJAH

Kelompok 7 :

1. **Wiwid Widyaningsih (L0124123)**
2. **Alena Mashia Qolby (L0124129)**
3. **Maria Dewi Handayani (L0124132)**
4. **Nayyara Aqila Azra (L0124138)**

Dosen Pengampu :

Prof. Drs. Bambang Harjito M.App.Sc., Ph.D.

PROGRAM STUDI INFORMATIKA

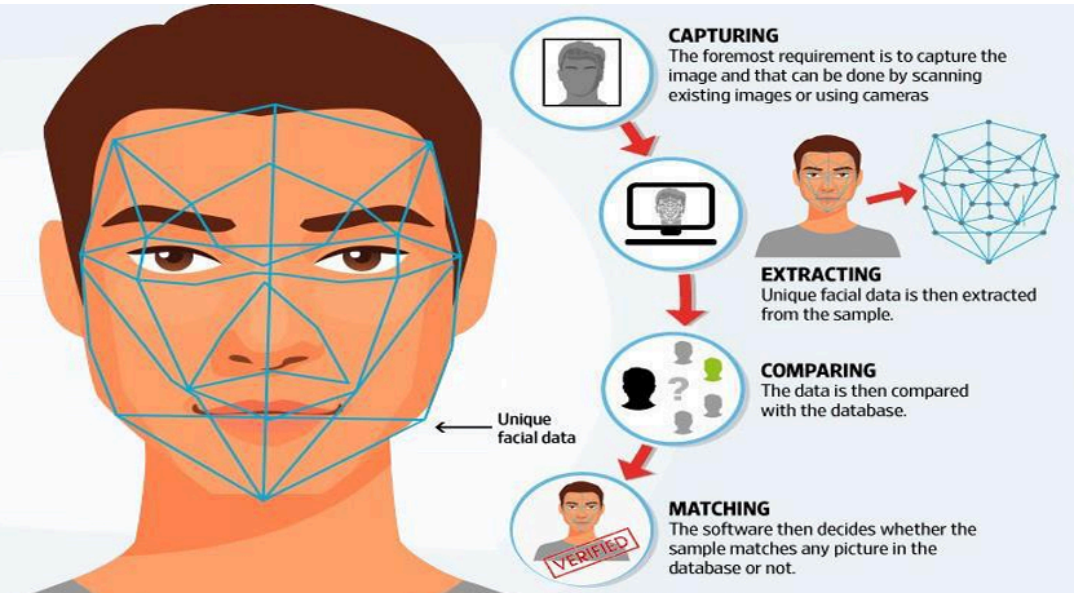
FAKULTAS TEKNOLOGI INFORMASI DAN SAINS DATA

UNIVERSITAS SEBELAS MARET

2025

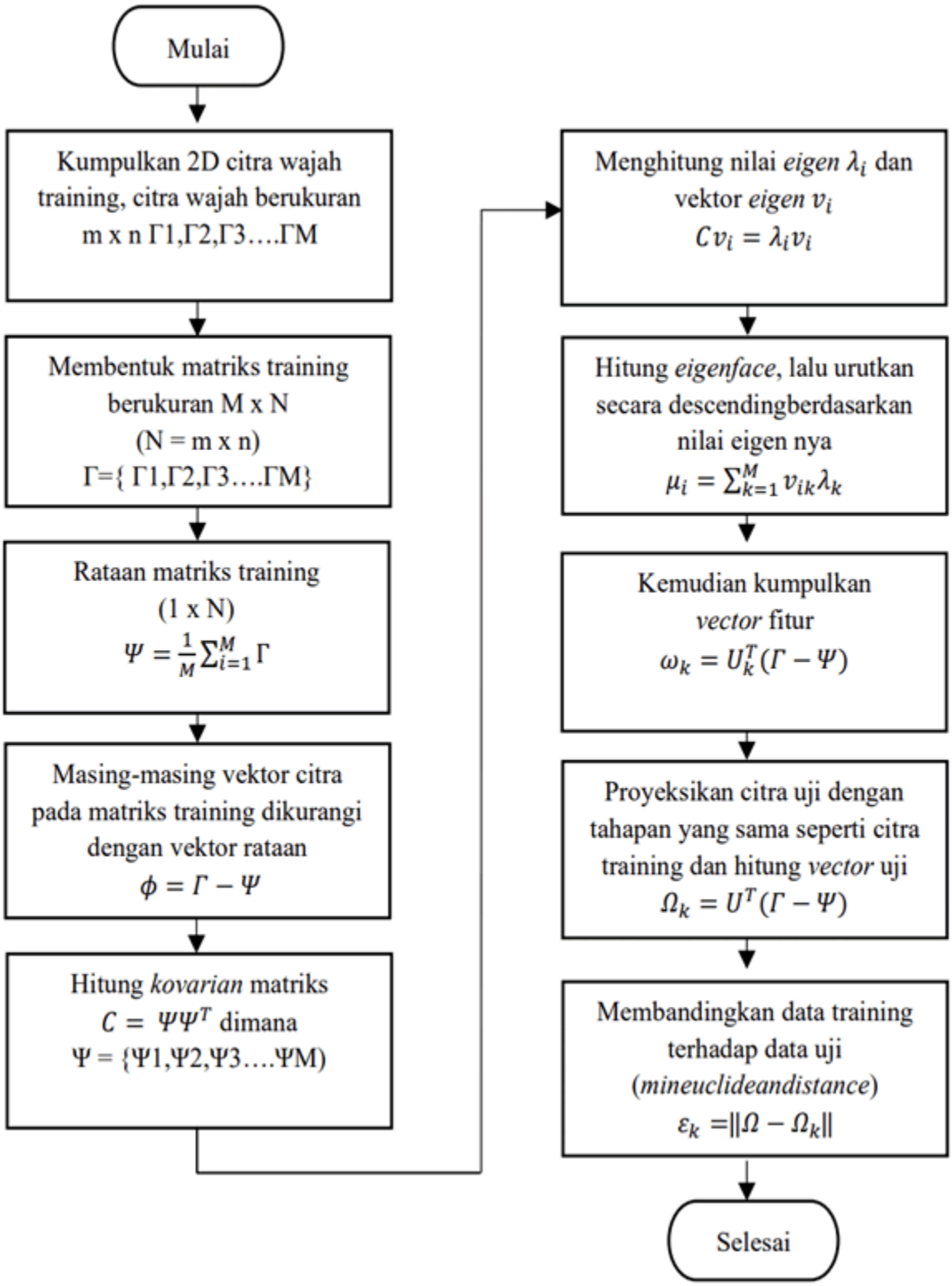
BAB 1
DESKRIPSI MASALAH

Pengenalan wajah (Face Recognition) adalah teknologi biometrik yang bisa dipakai untuk mengidentifikasi wajah seseorang untuk berbagai kepentingan khususnya keamanan. Program pengenalan wajah melibatkan kumpulan citra wajah yang sudah disimpan pada database lalu berdasarkan kumpulan citra wajah tersebut, program dapat mempelajari bentuk wajah lalu mencocokkan antara kumpulan citra wajah yang sudah dipelajari dengan citra yang akan diidentifikasi. Alur proses sebuah sistem pengenalan wajah diperlihatkan pada Gambar 1.



Terdapat berbagai teknik untuk memeriksa citra wajah dari kumpulan citra yang sudah diketahui seperti jarak Euclidean dan cosine similarity, principal component analysis (PCA), serta Eigenface. Pada Tugas ini, akan dibuat sebuah program pengenalan wajah menggunakan Eigenface. Sekumpulan citra wajah akan digunakan dengan representasi matriks. Dari representasi matriks tersebut akan dihitung sebuah matriks Eigenface. Program pengenalan wajah dapat dibagi menjadi 2 tahap berbeda yaitu tahap training dan pencocokkan. Pada tahap training, akan diberikan kumpulan data set berupa citra wajah. Citra wajah tersebut akan dinormalisasi dari RGB ke Grayscale (matriks), hasil normalisasi akan digunakan dalam perhitungan eigenface. Seperti namanya, matriks eigenface menggunakan eigenvector dalam pembentukannya. Berikut merupakan langkah rinci dalam pembentukan eigenface.

Flowchart Algoritma eigenface



Pada tahapan akhir, akan ditemui gambar dengan euclidean distance paling kecil maka gambar tersebut yang dikenali oleh program paling menyerupai test face selama nilai kemiripan di bawah suatu nilai batas. Jika nilai minimum di atas nilai batas maka dapat dikatakan tidak terdapat citra wajah yang mirip dengan test face Program dibuat dengan Bahasa Python dengan memanfaatkan sejumlah library di OpenCV (Computer Vision) atau library pemrosesan gambar lainnya (contoh PIL). Fungsi untuk mengekstraksi fitur dari sebuah citra wajah tidak perlu anda buat lagi, tetapi menggunakan fungsi ekstraksi yang sudah tersedia di dalam library. Fungsi Eigen dilarang import dari library dan harus diimplementasikan, sedangkan untuk operasi matriks lainnya silahkan menggunakan library.

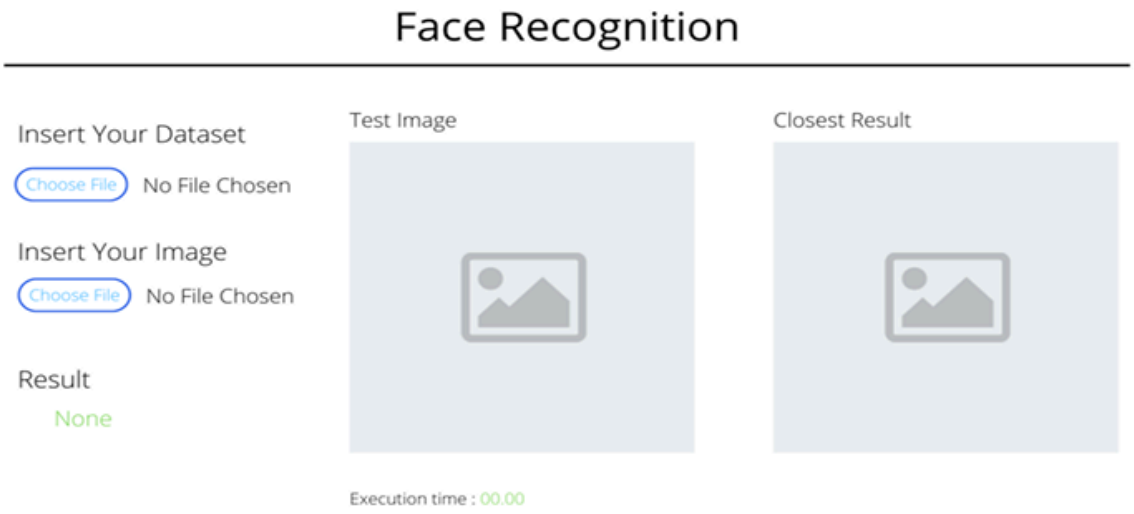
Kode program untuk ekstraksi fitur dapat dibaca pada artikel ini: Feature extraction and similar image search with OpenCV for newbies, pada laman: https://medium.com/machine-learning-world/feature-extraction-and-similar-image-search-with-opencv-for-newbies-3c59796bf774.

Nilai batas kemiripan citra test face dapat ditentukan oleh pembuat program melalui percobaan. Berikut merupakan referensi pengenalan wajah dengan metode https://jurnal.untan.ac.id/index.php/jcskommipa/article/download/9727/9500.
https://www.geeksforgeeks.org/ml-face-recognition-using-eigenfaces-pca-algorithm/.

Penggunaan Program

Berikut ini adalah input yang akan dimasukkan pengguna untuk eksekusi program.

1. Folder dataset, berisi folder atau directory yang berisi kumpulan gambar yang digunakan sebagai training image
2. File gambar, berisi file gambar input yang ingin dikenali dengan format file yang bebas selama merupakan format untuk gambar. Tampilan layout dari aplikasi web yang akan dibangun kurang lebih adalah sebagai berikut. Anda dapat mengubah layout selama layout masih terdiri dari komponen yang sama.



Catatan: Warna biru menunjukkan komponen yang dapat di klik. Warna hijau menunjukkan luaran yang didapat dari hasil eksekusi. Anda dapat menambahkan menu lainnya, gambar, logo, dan sebagainya. Tampilan GUI dibuat semenarik mungkin selama mencakup seluruh informasi pada layout yang diberikan di atas. Kreativitas menjadi salah satu komponen penilaian

BAB 2
TEORI SINGKAT

2.1 Perkalian Matriks

Terdapat dua jenis perkalian dalam matriks, yaitu perkalian skalar dengan matriks serta perkalian matriks dengan matriks. Perkalian skalar dengan matriks ialah mengalikan setiap elemen pada matriks dengan suatu skalar. Sedangkan perkalian matriks dengan matriks ialah mengalikan setiap elemen baris pada matriks pertama dengan setiap elemen kolom pada matriks kedua. Perkalian matriks dengan matriks bisa dilakukan dengan syarat kolom matriks pertama harus sama dengan jumlah baris matriks kedua. Contoh perkalian matriks yang bisa dilakukan adalah matriks dengan ordo m x n dikalikan dengan matriks berordo n x k sehingga dapat menghasilkan sebuah matriks dengan ordo m x n.

a. Perkalian skalar dengan matriks

$$3A = 3\begin{bmatrix} 2 & -1 & 2 \\ 0 & 4 & 3 \end{bmatrix} = \begin{bmatrix} 6 & -3 & 6 \\ 0 & 12 & 9 \end{bmatrix}$$

b. Perkalian matriks dengan matriks

$$A \times B = \begin{bmatrix} 2 & 1 \\ 2 & 2 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 \\ 3 & 2 & 1 \end{bmatrix} = \begin{bmatrix} 5 & 2 & 3 \\ 8 & 4 & 4 \end{bmatrix}$$

2.2 Nilai Eigen

Kata “eigen” berasal dari Bahasa Jerman yang artinya “asli” atau “karakteristik”. Eigen value atau nilai eigen (λ) merupakan nilai karakteristik suatu matriks yang menggambarkan matriks tersebut dalam bentuk perkalian dengan suatu vektor. Dengan kata lain, nilai eigen menyatakan nilai karakteristik dari sebuah 3 matriks yang berukuran n×n.

Secara matematis, untuk matriks bujursangkar A, nilai eigen λ dan vektor eigen x memenuhi persamaan:

$$Ax = \lambda x$$

di mana skalar λ disebut **nilai eigen** dari A, dan x dinamakan vektor eigen yang berkoresponden dengan. Atau juga dapat ditulis dalam bentuk homogen:

$$(A - \lambda I)x = 0$$

di mana I adalah matriks identitas. Persamaan ini konsisten jika dan hanya jika determinan $det(A - \lambda I) = 0$, yang disebut persamaan karakteristik. Nilai-nilai λ yang memenuhi persamaan ini adalah nilai eigen dari matriks A.

Nilai eigen bersama vektor eigen memiliki aplikasi luas dalam berbagai bidang seperti teori stabilitas, analisis getaran, orbital atom, pengenalan wajah, dan diagonalisasi matriks.

2.3 Vektor Eigen

Vektor eigen (atau eigenvector) adalah vektor non-nol yang arahnya tidak berubah, namun skalanya berubah ketika dikalikan dengan sebuah matriks kuadrat. Artinya, hasil perkalian matriks dengan vektor eigen adalah kelipatan skalar dari vektor tersebut. Persamaan karakteristik dari vektor eigen adalah :

Jika A adalah matriks kuadrat $n \times n$ dan v adalah vektor kolom $n \times 1$, maka v adalah vektor eigen dari A jika:

$$A \cdot v = \lambda \cdot v$$

dengan λ adalah nilai eigen (eigenvalue) yang menyatakan skala perubahan panjang vektor. Vektor eigen menunjukkan arah utama dalam transformasi matriks, yaitu arah di mana sebuah vektor akan mengalami perubahan skala tanpa perubahan arah (misalnya, arah peregangan atau perputaran). Dalam konteks pengenalan wajah menggunakan metode Eigenface, vektor eigen digunakan untuk merepresentasikan setiap wajah sebagai kombinasi linier dari wajah-wajah pelatihan. Dengan demikian, Eigenface dapat memadatkan informasi penting dari banyak citra wajah menjadi beberapa vektor eigen yang paling signifikan untuk keperluan klasifikasi atau pengenalan wajah.

2.4 Eigenface

Eigenface adalah suatu metode pengenalan wajah (face recognition) berdasarkan pada metode Principal Component Analysis (PCA). Metode ini menggunakan koleksi gambar wajah untuk menentukan variasi dari wajah lalu menggunakan varian tersebut untuk mengenkripsi dan mendekripsi sebuah wajah dengan cara machine learning tanpa mengurangi komputasi dan kompleksitas ruang. Dua algoritma dari eigenface:

2.4.1 Algoritma Training

2.4.1.1 Normalisasi dan Vektorisasi

Gambar akan diubah ke ukuran 256 x 256, lalu diubah menjadi vektor satu dimensi dan semua vektor akan digabung menjadi matriks A.

2.4.1.2 Hitung Rata-rata

Hitung rata-rata seluruh vektor gambar

$$\psi = \frac{1}{M}\Sigma_{n=1}^{M}\Gamma_n$$

2.4.1.3 Pengurangan rata-rata

Kurangi setiap vektor gambaran dengan vektor rata-rata untuk mendapatkan vektor selisih:

$$\Phi = \Gamma_i - \psi$$

Lalu gabungkan seluruh vektor selisih menjadi satu matriks. A berukuran $M \times N^2$, di mana M adalah jumlah gambar dan N^2 adalah jumlah pixel.

2.4.2 Algoritma Testing

2.4.2.1 Preprocessing

Gambar baru diubah ke ukuran dan format yg sama, lalu diubah menjadi vektor.

2.4.2.2 Pengurangan Rata-rata

Kurangi vektor gambar baru dengan rata-rata:

$$(\Gamma_{new} - \Psi)$$

Proyeksikan ke ruang eigenface :

$$\mu_{new} = v\,x\,T_{new} - \Psi$$

BAB 3
IMPLEMENTASI PROGRAM

Program ini disusun menggunakan bahasa python dengan berbagai macam library nya untuk melakukan pengolahan data, interface, dan analisis citra.

3.1 Tech stack/kakas yang digunakan pada program

Library	Method	Deskripsi pada program
Numpy	Numpy.array()	Membuat matriks kosong untuk menyimpan vektor citra
	numpy.add()	Menjumlahkan vektor rata-rata sementara dengan setiap elemen dataset untuk menghitung rata-rata akhir.
	numpy.subtract()	Mengurangi elemen-elemen vektor dengan rata-rata untuk menghilangkan pengaruh faktor pencahayaan atau distraksi lainnya.
	numpy.dot()	Merepresentasikan gambar di ruang eigenspace (proyeksi dataset) serta menghitung bobot setiap gambar berdasarkan eigenface.
	numpy.linalg.norm()	Mengukur tingkat kemiripan antara gambar uji dengan gambar di dataset.
	numpy.transpose()	Membalik baris menjadi kolom (atau sebaliknya) agar dimensi matriks sesuai untuk operasi aljabar linear seperti perkalian matriks.
	numpy.diag()	Mengambil nilai diagonal dari matriks untuk mendapatkan nilai eigen.
	numpy.divide()	Membagi setiap elemen matriks kovarian dengan jumlah gambar dalam dataset untuk mendapatkan kovarian rata-rata.

	numpy.absolute()	Menghilangkan nilai negatif dari hasil selisih untuk jarak Euclidean.
	numpy.where()	Mendapatkan indeks gambar dalam dataset yang paling mirip dengan gambar uji.
OpenCV	cv2.imread()	Membaca gambar dari path yang diberikan dan mengubahnya menjadi array numerik yang dapat diproses oleh NumPy dan OpenCV.
	cv2.resize()	Mengatur ukuran gambar agar konsisten di seluruh dataset dan gambar uji
	cv2.cvtColor()	Konversi gambar ke grayscale untuk menyederhanakan data
Tkinter	-	Fungsi: 1. Membuat antarmuka grafis, seperti tombol, label, dan frame. 2. Membaca file dari sistem pengguna. 3. Menampilkan gambar dataset, gambar uji, dan hasil pengenalan.

3.2 Algoritma Program

3.2.1 Image pre-processing

Pada bagian ini gambar dari dataset dan gambar uji coba diubah menjadi grayscale dengan ukuran resolusi 256 x 256 pixel. Citra citra tersebut kemudian dikonversi ke dalam bentuk vektor satu dimensi yakni 256^2 = 65,536 elemen. Kemudian perhitungan rata rata vektor tersebut dilakukan dengan tujuan untuk mengurangi vektor vektor citra agar dapat menormalkan data. Normalisasi ini digunakan untuk menghilangkan beberapa distraksi seperti pencahayaan dan faktor konstan lainnya. Hasil dari normalisasi ini kemudian digunakan untuk menghitung matriks kovarian. Matriks kovarian ini merepresentasikan hubungan antar piksel pada citra, yang selanjutnya akan digunakan dalam proses dekomposisi eigen.

3.2.2 Eigenface processing

Hasil perhitungan matriks kovarian yang dilakukan, digunakan untuk menghitung nilai eigen (eigenvalues) dan vektor eigen (eigenvectors) menggunakan QR decomposition. Vektor eigen ini merepresentasikan fitur-fitur utama dalam dataset, yang disebut sebagai eigenface. Eigenface berfungsi sebagai basis untuk menggambarkan karakteristik unik dari setiap wajah.

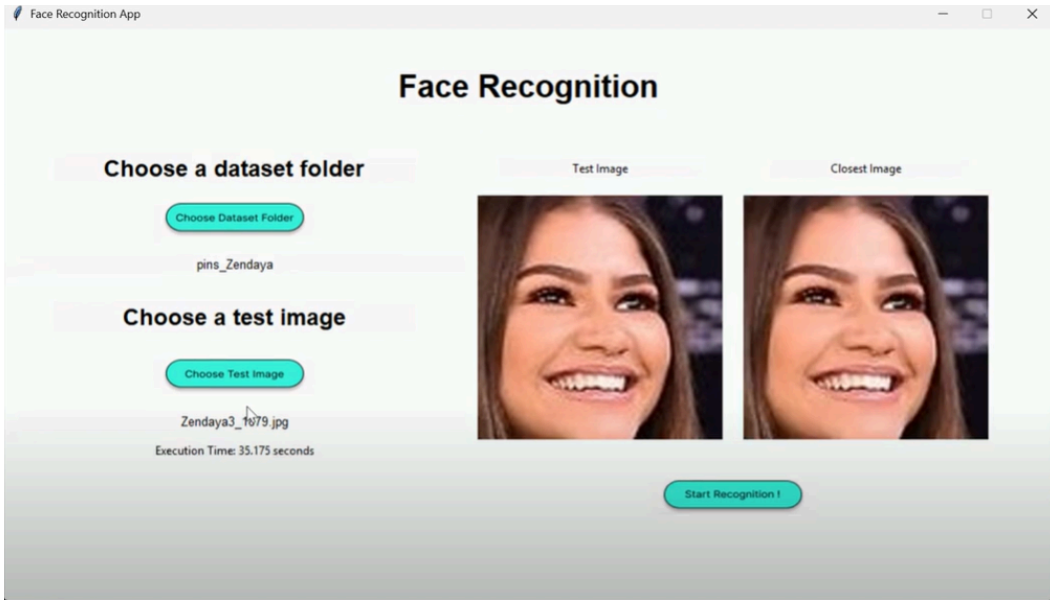
Matriks dataset yang sudah dihitung dan di normalisasi sebelumnya akan dikalikan dengan vektor eigen untuk mendapatkan vektor vektor eigenface. Vektor-vektor wajah kemudian diubah dalam bentuk vektor konstanta-konstanta kombinasi linear dari vektor-vektor eigenface tersebut.

3.2.3 Image Matching

Pada bagian ini, dilakukan pencocokan gambar uji yang telah diubah melalui proses pre-processing menjadi vektor wajah dan dalam kondisi “normal” setelah dilakukan proses pengurangan rata rata.Vektor wajah dari gambar uji kemudian dihitung bobotnya berdasarkan kombinasi linear eigenface. Bobot ini dibandingkan dengan bobot dari dataset menggunakan jarak Euclidean untuk mengukur tingkat kemiripan antara gambar uji dan gambar dalam dataset. Perhitungan gambar yang menghasilkan jarak euclidean terkecil akan dipilih kemudian ditampilkan dalam GUI beserta waktu eksekusi yang dibutuhkan pada program ini.

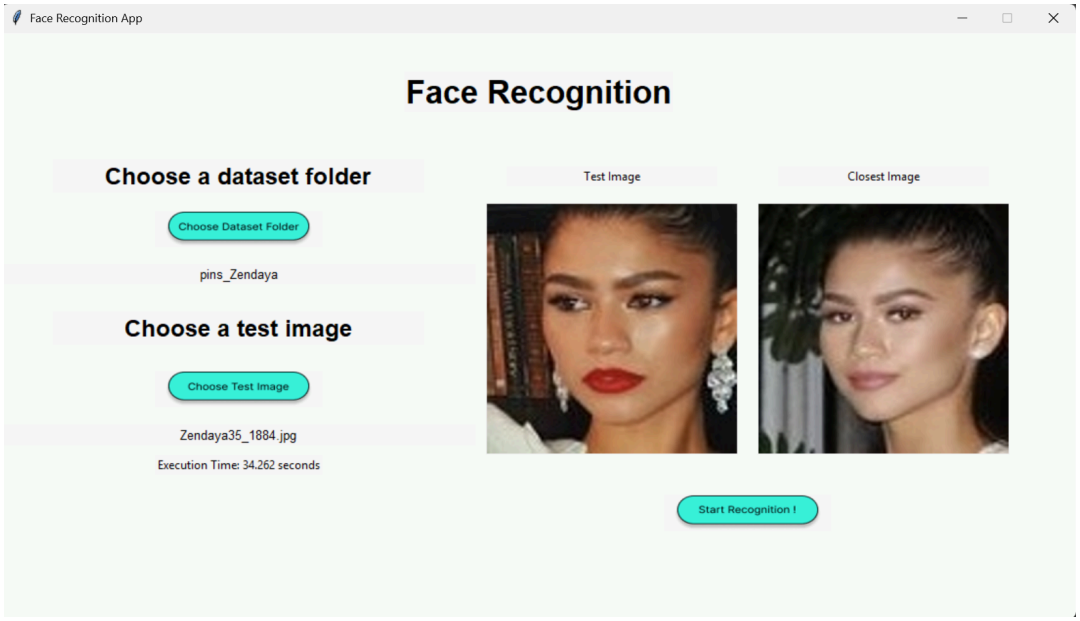
BAB 4
EKSPERIMEN

4.1 Gambar yang dijadikan test image terdapat dalam folder dataset



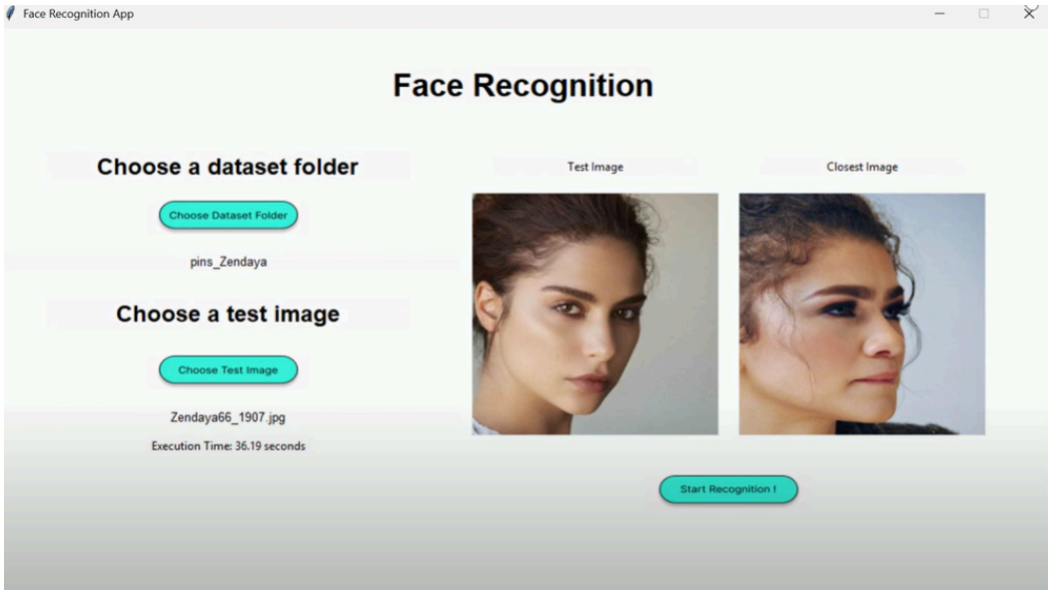
Gambar uji merupakan wajah yang sudah ada di dalam dataset, output menghasilkan ditemukan gambar cocok dengan waktu eksekusi 35.175 detik.

4.2 Gambar yang dijadikan test image tidak terdapat dalam folder dataset



Gambar uji merupakan wajah milik orang yang sama namun berada di luar dataset, output menampilkan gambar di dalam dataset dengan kemiripan terdekat dengan waktu eksekusi 34.262 detik.

4.3 Wajah yang berbeda dan tidak terdapat dalam folder dataset



Gambar uji merupakan wajah milik orang yang berbeda dan tidak terdapat dalam dataset. Output menampilkan gambar yang mendekati dengan input yang diberikan dengan waktu eksekusi 36.19 detik.

BAB 5
KESIMPULAN, SARAN, DAN REFLEKSI

5.1 Kesimpulan

Hasil implementasi program ini menunjukkan bahwa metode Eigenfaces berbasis Principal Component Analysis (PCA) dapat digunakan secara efektif untuk pengenalan wajah. Program telah berhasil memproses gambar wajah melalui tahap konversi menjadi vektor, normalisasi terhadap wajah rata-rata, perhitungan matriks kovarians, dekomposisi matriks menggunakan metode QR, serta proyeksi dataset ke subspace eigenspace. Proses ini kemudian dilengkapi dengan perhitungan jarak Euclidean untuk mengidentifikasi gambar uji terhadap dataset.

Dengan memanfaatkan bahasa pemrograman Python serta pustaka OpenCV dan NumPy, program ini mampu menjalankan seluruh pipeline pemrosesan gambar dan komputasi numerik secara efisien. Struktur kode yang disusun secara modular mempermudah pengembangan dan pemeliharaan, mulai dari tahap pre-processing hingga evaluasi hasil pengenalan.

Program telah memberikan output yang sesuai dengan rancangan dan harapan awal, di mana gambar uji dapat dicocokkan dengan gambar dataset dengan tingkat akurasi yang memadai. Implementasi yang berhasil ini menjadi bukti bahwa konsep reduksi dimensi melalui Eigenfaces dapat diimplementasikan secara praktis untuk aplikasi pengenalan wajah dalam skala terbatas. Secara keseluruhan, proyek ini menunjukkan pemahaman mendalam terhadap teori PCA dan aplikasinya pada pengolahan citra digital.

5.2 Saran

5.2.1 Optimasi Performa

Saat dataset bertambah besar, komputasi QR decomposition (di eigQR()) bisa menjadi bottleneck. Pertimbangkan menggunakan pustaka pustaka optimasi linear algebra seperti scipy.linalg.eigh() atau pustaka GPU acceleration (misalnya PyTorch atau CuPy) jika dataset-nya besar.

5.2.2 Error Handling & Logging

Tambahkan validasi input dan logging agar program lebih tahan terhadap error. Misalnya, cek apakah semua gambar memiliki ukuran yang sama, atau buat exception handling saat file tidak ditemukan.

5.2.3 Penyempurnaan UI/UX

Jika proyek ini dikembangkan untuk pengguna umum, antarmuka pengguna (GUI) bisa ditingkatkan agar lebih informatif, misalnya menampilkan hasil pengenalan dengan confidence level dan pratinjau wajah yang cocok.

5.2.4 Evaluasi Akurasi

Untuk dataset yang lebih besar dan variatif, dapat dilakukan evaluasi akurasi (misalnya confusion matrix, precision-recall) agar performa model lebih terukur.

5.3 Refleksi

Proyek ini memperlihatkan bagaimana teknik pengolahan citra (komputasi matriks dan PCA) dapat diaplikasikan untuk masalah dunia nyata seperti pengenalan wajah. Meski metode Eigenfaces cukup klasik, proyek ini memberi pemahaman yang baik tentang pipeline machine learning secara end-to-end—mulai dari data preprocessing, feature extraction, hingga prediksi.

Selain itu, proyek ini juga membuka ruang bagi pengembangan lebih lanjut, seperti penggunaan deep learning (CNN) atau metode machine learning lainnya untuk meningkatkan performa, terutama di lingkungan real-world yang lebih kompleks.

DAFTAR PUSTAKA

YUDHANTA, A. B. P. (2021). *DETEKSI KERUSAKAN STRUKTUR DENGAN MENGGUNAKAN MODAL ASSURANCE CRITERIONSYMBIOTIC ORGANISMS SEARCH ALGORITHM (MAC-SOS)* (Doctoral dissertation, Universitas Atma Jaya Yogyakarta). https://e-journal.uajy.ac.id/26185/

Khan Academy. (n.d.). *Introduction to eigenvalues and eigenvectors*. Retrieved June 10, 2025, from /www.khanacademy.org/math/linear-algebra/alternate-bases/eigen-everything/v/linear-algebra-introduction-to-eigenvalues-and-eigenvectors

Muliawan, M. R., Irawan, B., & Brianorman, Y. (2015). Implementasi pengenalan wajah dengan metode *Eigenface* pada sistem absensi. *Jurnal Coding, Sistem Komputer Untan*, 3(1), 41–50. Diakses dari https://jurnal.untan.ac.id/index.php/jcskommipa/article/download/9727/9500

Saputra, W. M., Wibawa, H. A., Bahtiar, N. (2013). Pengenalan Wajah Menggunakan Algoritma Eigenface dan Euclidean Distance. Diakses dari https://www.neliti.com/id/publications/135138/pengenalan-wajah-menggunakan-algoritma-eigenface-dan-euclidean-distance

Munir, R. (2023). *Nilai Eigen dan Vektor Eigen Bagian 1*. Retrieved June 10, 2025, from https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2023-2024/Algeo-19-Nilai-Eigen-dan-Vektor-Eigen-Bagian1-2023.pdf

LAMPIRAN

Link repository program : https://github.com/wiwidw/Face-Recognition/

Link youtube demonstrasi program : Demonstrasi Project Based Learning 01 kelompok 7