

LAPORAN TUGAS BESAR II

“Aplikasi Nilai Eigen dan EigenFace pada Pengenalan Wajah (*Face Recognition*)”

Mata Kuliah Aljabar Linier dan Geometri (IF2123)

Dosen Pengampu : Dr. Ir. Rinaldi Munir, M.T.



Disusun Oleh :

Kelompok bisain

Anggota :

1. Maggie ZR Simangunsong (13521117)
2. Ferindya Aulia Berlianty (13521161)
3. Muhammad Habibi Husni (13521169)

**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG
SEMESTER I TAHUN 2022-2023**

Daftar Isi

Daftar Isi	2
BAB I Deskripsi Masalah	4
BAB II Teori Singkat	
2.1 Perkalian Matriks	5
2.2 Nilai Eigen	5
2.3 Vektor Eigen	6
2.4 Eigenface	7
BAB III Implementasi Program	
3.1 Mengekstrasi Fitur-Fitur dari Foto Training	10
3.2 Mencari Eigenface	10
3.3 Mencocokkan Fitur-Fitur Sebuah Gambar dengan Basis Data	11
3.4 Graphical User Interface (GUI)	12
3.5 Insert Photo and Dataset	13
3.6 Show Test Image and Closest Result	13
BAB IV Eksperimen	
4.1 Eksperimen 1	14
4.2 Eksperimen 2	14
4.3 Eksperimen 3	15
4.4 Eksperimen 4	15
4.5 Eksperimen 5	15
4.6 Eksperimen 6	16
4.7 Eksperimen 7	16
4.8 Eksperimen 8	16
4.9 Eksperimen 9	17
BAB V Kesimpulan, Saran, dan Refleksi	
5.1 Kesimpulan	18
5.2 Saran	18
5.3 Refleksi	18
Daftar Referensi	19

BAB I

Deskripsi Masalah

Membuat program pengenalan wajah dalam Bahasa Python berbasis GUI dengan spesifikasi sebagai berikut:

1. Program menerima input *folder dataset* dan sebuah gambar citra wajah.
2. Basis data wajah dapat diunduh secara mandiri melalui <https://www.kaggle.com/datasets/herveisburak/pins-face-recognition>.
3. Program menampilkan gambar citra wajah yang dipilih oleh pengguna.
4. Program melakukan pencocokan wajah dengan koleksi wajah yang ada di folder yang telah dipilih. Metrik untuk pengukuran kemiripan menggunakan eigenface + jarak *euclidean*.
5. Program menampilkan 1 hasil pencocokan pada dataset yang paling dekat dengan gambar input atau memberikan pesan jika tidak didapatkan hasil yang sesuai.
6. Program menghitung jarak *euclidean* dan nilai eigen & vektor eigen yang ditulis sendiri. Tidak boleh menggunakan fungsi yang sudah tersedia di dalam *library* atau Bahasa Python.
7. Input gambar akan berukuran MINIMAL 256 x 256.
8. Hasil training dapat disimpan, namun tetap wajib dapat dilakukan training kembali jika diperlukan user.

BAB II

Teori Singkat

2.1 Perkalian Matriks

Dua buah matriks dapat dikalikan jika banyak kolom pada matriks pertama mempunyai nilai yang sama dengan banyak baris pada matriks kedua. Perkalian dua buah matriks akan menghasilkan sebuah matriks baru dengan ukuran baris sama dengan matriks pertama dan ukuran kolom sama dengan matriks kedua. Jika A dan B adalah suatu matriks, maka hasil perkalian dari kedua matriks A dan B dapat dinyatakan sebagai $C = AB$.

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \times \begin{bmatrix} e & f \\ g & h \end{bmatrix} = \begin{bmatrix} ae + bg & af + bh \\ ce + dg & cf + dh \end{bmatrix}$$

Dapat dinyatakan dengan $C_{R \times T} = A_{R \times S} \times B_{S \times T}$

$$C = C_{ij} = A_{i1}B_{1j} + \dots + A_{is}B_{sj} = \sum_{k=1}^s A_{ik}B_{kj},$$

S adalah banyaknya kolom matriks pertama atau banyaknya baris matriks kedua.

Sedangkan untuk rumus perkalian skalar matriks dilakukan dengan cara konstanta, yaitu nilai matriks dikalikan dengan cara mengalikan setiap elemen atau komponen nilai matriks dengan skalar. Misalnya nilai Matriks A dikalikan dengan skalar K maka setiap elemen atau komponen Matriks A dikali dengan k.

$$k \times \begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} ka & kb \\ kc & kd \end{pmatrix}$$

2.2 Nilai Eigen

Jika A adalah matriks $n \times n$, maka vektor tidak-nol x di R^n disebut vektor eigen dari A jika Ax sama dengan perkalian suatu skalar λ dengan x , yaitu

$$Ax = \lambda x$$

Skalar λ disebut nilai eigen dari A , dan x dinamakan vektor eigen yang berkoresponden dengan λ , dengan kata lain, nilai eigen menyatakan nilai karakteristik dari sebuah matriks yang berukuran $n \times n$. Untuk menghitung nilai eigen dari suatu matriks A dapat dihitung sebagai berikut,

$$\begin{aligned} Ax &= \lambda x \\ IAx &= \lambda Ix \\ Ax &= \lambda Ix \\ (\lambda I - A)x &= 0 \end{aligned}$$

$x = 0$ adalah solusi trivial dari $(\lambda I - A)x = 0$

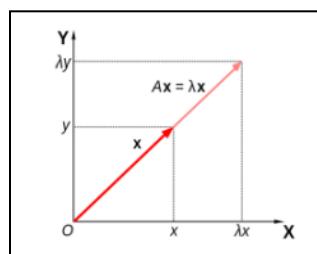
agar $(\lambda I - A)x = 0$ memiliki solusi tidak-nol, maka haruslah

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

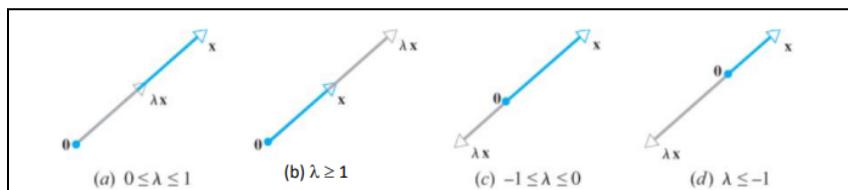
Persamaan $\det(\lambda I - A) = 0$ disebut persamaan karakteristik dari matriks A , dan akar-akar persamaan tersebut, yaitu λ , dinamakan akar-akar karakteristik atau nilai-nilai eigen.

2.3 Vektor Eigen

Vektor eigen berkaitan dengan nilai eigen. Vektor eigen x menyatakan vektor kolom yang apabila dikalikan dengan sebuah matriks $n \times n$ menghasilkan vektor lain yang merupakan kelipatan vektor itu sendiri.



Dengan kata lain, operasi $Ax = \lambda x$ menyebabkan vektor x menyusut atau memanjang dengan faktor λ dengan arah yang sama jika λ positif dan arah berkebalikan jika λ negatif.



Dengan pemecahan solusi yang telah dijelaskan pada bagian sebelumnya, setelah didapat nilai dari λ (nilai eigen) kita dapat mensubstitusi nilai setiap λ yang ada pada persamaan $(\lambda I - A)x = 0$ untuk mendapatkan vektor eigen.

Sebagai contoh, untuk

$$(\lambda I - A)x = 0 \rightarrow \begin{bmatrix} \lambda - 3 & 0 \\ -8 & \lambda + 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

didapatkan nilai-nilai eigen dari matrik A adalah $\lambda = 3$ dan $\lambda = -1$.

Untuk $\lambda = 3$,

$$\begin{bmatrix} 0 & 0 \\ -8 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \rightarrow -8x_1 + 4x_2 = 0 \rightarrow 8x_1 = 4x_2 \rightarrow x_1 = \frac{1}{2}x_2$$

Solusi :

$$x_1 = \frac{1}{2}t, x_2 = t, t \in \mathbb{R}$$

Vektor eigen :

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \frac{1}{2}t \\ t \end{bmatrix} = t \begin{bmatrix} \frac{1}{2} \\ 1 \end{bmatrix}$$

Membentuk ruang eigen (*eigen space*).

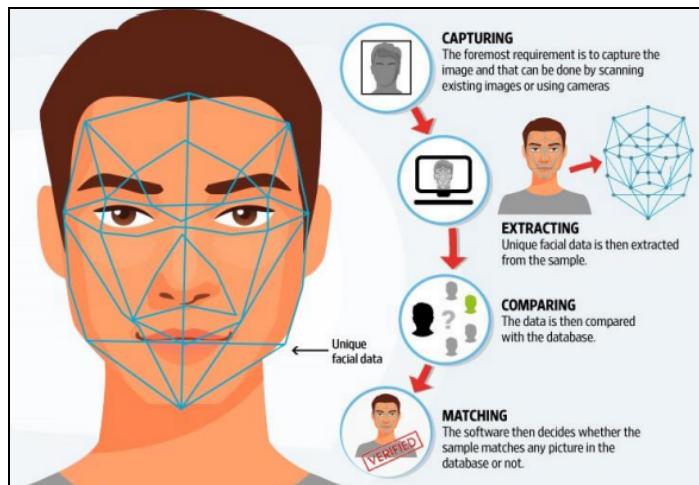
Sehingga, $\begin{bmatrix} \frac{1}{2} \\ 1 \end{bmatrix}$ adalah basis untuk ruang eigen dengan $\lambda = 3$. Ruang eigen dapat ditulis sebagai berikut

$$E(3) = \{ x = t \begin{bmatrix} \frac{1}{2} \\ 1 \end{bmatrix}, t \in \mathbb{R} \}$$

Untuk $\lambda = -1$ dapat digunakan cara yang serupa untuk mendapatkan vektor eigennya.

2.4 Eigenface

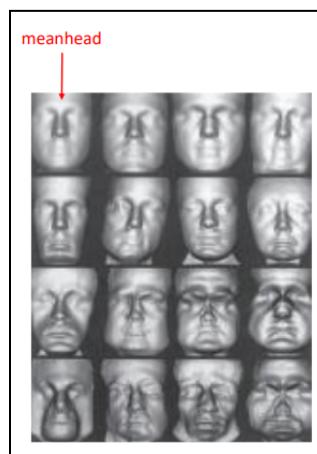
Eigenface adalah metode pengenalan wajah (face recognition) di berbasis vektor eigen dan nilai eigen yang digunakan untuk persoalan-persoalan di dalam computer vision.



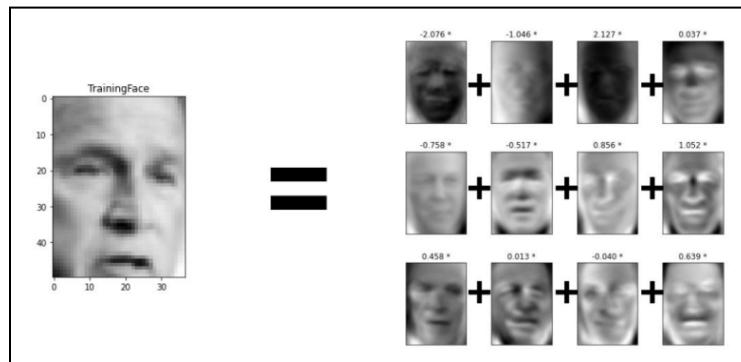
Alur proses di dalam sistem pengenalan wajah

Vektor eigen diturunkan dari matriks kovarian dari sejumlah citra wajah latih (training image). Eigenface membentuk himpunan basis dari semua gambar yang digunakan untuk membangun matriks kovarian. Ini menghasilkan pengurangan dimensi dengan memungkinkan kumpulan gambar dasar yang lebih kecil untuk mewakili gambar pelatihan asli. Klasifikasi dapat dicapai dengan membandingkan bagaimana wajah direpresentasikan oleh himpunan basis. Gagasan ini adalah bahwa setiap wajah manusia dalam kelompok ras adalah kombinasi dari beberapa lusin bentuk primer.

Misalnya, dengan menganalisis pemindaian tiga dimensi dari banyak wajah, para peneliti di Universitas Rockefeller telah menghasilkan bentuk kepala rata-rata pada ras Kaukasia—dijuluki meanhead (gambar pojok kiri atas)—dan satu himpunan berisi variasi wajah dari bentuk itu, yang disebut eigenheads (15 diantaranya ditunjukkan pada gambar).



Dinamakan eigenface karena merupakan vektor eigen dari matriks kovarian yang menyimpan informasi citra wajah. Bentuk wajah direpresentasikan secara matematis sebagai kombinasi linier dari eigenheads.



BAB III

Implementasi Program

3.1 Mengekstrasi Fitur-Fitur dari Foto Training

Dari setiap input gambar dan dataset yang dimiliki akan diekstrak menjadi matriks untuk kemudian dilakukan pengolahan. Foto-foto yang pada awalnya berwarna, akan diambil foto potongan wajah dari masing-masing foto tersebut serta diubah menjadi grayscale sebelum dicari eigenface-nya. Untuk setiap proses ekstraksi yang dilakukan, kami menggunakan library OpenCV (Open Source Computer Vision). Library ini dipilih karena sangat cocok digunakan untuk program yang memerlukan proses pengolahan citra gambar. Proses ekstraksi secara singkat dilakukan sebagai berikut :

1. Membaca data foto training menjadi matriks BGR
2. Melakukan pengambilan matriks wajah dari masing-masing foto tersebut dengan memanfaatkan Cascade Classifier dari OpenCV.
3. Melakukan resize matriks foto wajah menjadi ukuran 100 x 100

3.2 Mencari Eigenface

Setelah mendapatkan kumpulan matriks wajah dari proses pengekstrakan, maka selanjutnya akan dilakukan pencarian eigenface. Pada pencarian eigenface secara singkat dilakukan sebagai berikut :

1. Mengubah skema warna matriks wajah menjadi grayscale
2. Melakukan pencarian matriks rata-rata, $\Psi = \frac{1}{m} \sum_{i=1}^m x_i$
3. Mendapatkan matriks normalisasi wajah dengan melakukan pengurangan oleh matriks rata-rata, $a_i = x_i - \Psi$
4. Menjadikan masing-masing matriks wajah normalisasi sebagai vektor kolom dari $A = [a_1 | a_2 | \dots | a_m]$
5. Mencari matriks kovarian dari kumpulan matriks wajah yang sudah dinormalisasikan dengan rumus, $Kov = A^T \times A$, sehingga didapat matriks berukuran $m \times m$.

6. Mencari vektor eigen dari matriks Kov lalu mengambil k buah eigenvector terbesar. $V = [v_1 | v_2 | \dots | v_k]$ dengan V berukuran $m \times k$.
7. Mengembalikan dimensi vektor eigen sesuai panjang foto dengan menggunakan rumus : $u_i = A \times v_i$. Matriks $U = [u_1 | u_2 | \dots | u_k]$ ini merupakan kumpulan eigenface yang ukurannya $n^2 \times k$.
8. Merepresentasikan masing-masing foto normalisasi menjadi kombinasi linier dari eigenface yang sudah ditemukan. Untuk pencarian koefisien w_i dapat dilakukan dengan melakukan proyeksi skalar dari a_i terhadap u_1 .

$$a_i = w_{1i}u_1 + w_{2i}u_2 + \dots + w_{ki}u_k$$

Kumpulan koefisien tersebut dapat dijadikan sebuah vektor koefisien yang akan merepresentasikan foto awal.

$$w_i = \begin{bmatrix} w_{1i} \\ w_{2i} \\ \vdots \\ \vdots \\ w_{ki} \end{bmatrix}$$

3.3 Mencocokkan Fitur-Fitur Sebuah Gambar dengan Basis Data

Langkah selanjutnya yaitu melakukan pencocokkan fitur-fitur terhadap gambar yang dimasukkan ke dalam program melalui aplikasi GUI sederhana yang telah kami buat, proses ini dilakukan setelah proses ekstraksi selesai dilakukan. Pencocokkan dilakukan dengan menggunakan fungsi-fungsi yang telah kami buat yang berfungsi untuk mencocokkan fitur-fitur sebuah gambar dengan dataset yang keduanya telah disimpan dalam tipe gambar (.jpg, .png, dll). Proses pencocokan adalah sebagai berikut :

1. Mendapatkan matriks wajah dari foto yang akan dites
2. Mengubah skema warna matriks wajah tes menjadi grayscale
3. Mengurangi matriks wajah tes dengan rata-rata dari foto yang ada di basis data sebelumnya. $a = x - \Psi$

4. Mendapatkan vektor koefisien dengan basis eigenface sebelumnya.

$$w = \begin{bmatrix} w_1 \\ w_2 \\ \cdot \\ \cdot \\ w_k \end{bmatrix}$$

5. Mendapatkan foto terdekat dengan mencari jarak euclidean terkecil.
 $dist = \min ||w_i - w||$, dalam hal ini foto ke-i merupakan foto yang paling mirip dengan foto tes.

3.4 Graphical User Interface (GUI)

Program yang kami buat memanfaatkan library python yaitu Tkinter untuk membuat aplikasi sederhana berbasis GUI dalam pencocokan wajah dengan dataset yang ada. Penggunaan library ini didasarkan dengan pertimbangan kemudahan karena Tkinter sudah banyak digunakan menjadi library standar GUI dari python. Tkinter memiliki beberapa fitur yang kami gunakan, seperti fitur entry input untuk masukan dari pengguna, button sebagai media interaktif yang menandakan program dimulai, canvas berfungsi untuk menampilkan gambar, serta text dan message sebagai media komunikasi dengan pengguna terhadap respons input yang dimasukkan.

Entry input digunakan untuk masukan test image dan folder dataset. Pengguna dapat memasukkan directory tempat file/folder berada ataupun langsung meng-klik icon folder yang ada disebelahnya. Dengan meng-import class filedialog pada Tkinter, button icon folder ini akan otomatis menampilkan directory dari file/folder yang dipilih pada entry input. Khusus untuk masukan test image, pengguna dapat memilih menggunakan webcam untuk input test image. Auto capture ini menggunakan library OpenCV, dan akan meng-capture secara otomatis.

Ketika pengguna telah memasukkan input test image dan folder dataset, pengguna dapat meng-klik button ‘Test’ untuk langsung mengeksekusi proses pencocokan gambar pada dataset. Jika berhasil, foto *test image* dan foto hasil pengenalan akan dimunculkan keduanya, dan muncul dan *execution time* akan ikut ditampilkan. Namun jika tidak ditemukan, yang dimunculkan hanya foto *test image* saja tanpa ada *execution time* yang tercatat.

3.5 Insert Photo and Dataset

Fitur yang berada di panel sebelah kiri ini berisi 2 slot yaitu untuk menerima test image dan menerima dataset gambar. Penerimaan gambar dapat dilakukan dengan memilih secara manual pada folder local ataupun menggunakan fitur kamera untuk mengambil gambar secara langsung. Sedangkan, untuk penerimaan dataset gambar hanya dapat dilakukan dengan memilih folder dataset secara manual pada folder local yang digunakan.

3.6 Show Test Image and Closest Result

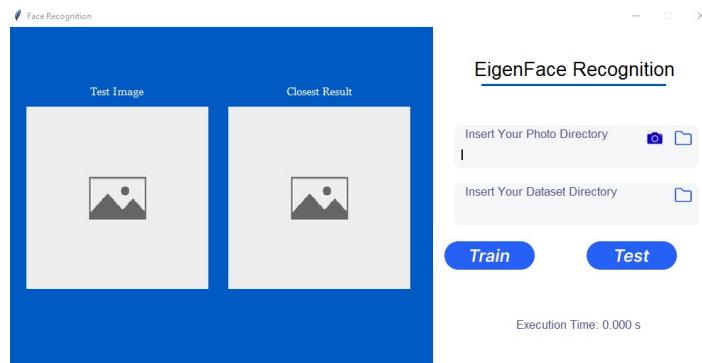
Fitur yang berada di sebelah kanan ini terdiri dari 2 slot untuk menampilkan test image dan hasil gambar yang paling mendekati. Path gambar yang telah diinput sebelumnya, akan ditampilkan pada slot test image setelah user menekan button ‘Test’. Akan terdapat keterangan “Gambar Ditemukan!” saat ditemukan gambar yang cocok dan “Wajah tidak ditemukan dalam dataset” jika tidak ditemukan gambar yang cocok. Selain itu, terdapat “Execution Time” yang akan menampilkan durasi yang dibutuhkan untuk melakukan pencocokan gambar. Ada juga button ‘Train’ yang berfungsi untuk membuat dataset baru agar kita tidak perlu mengulangi input dataset.

BAB IV

Eksperimen

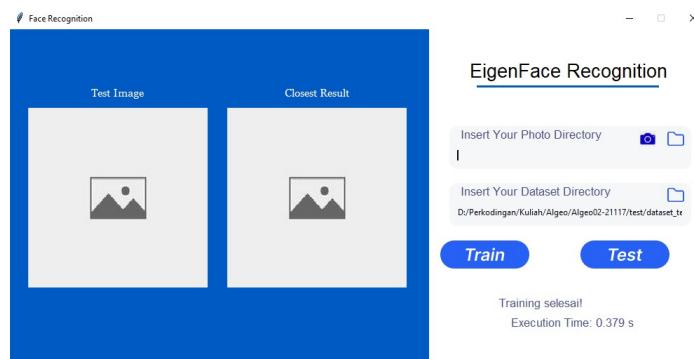
Tampilan website dan pada saat pemrosesan gambar sebagai berikut.

4.1 Eksperimen 1



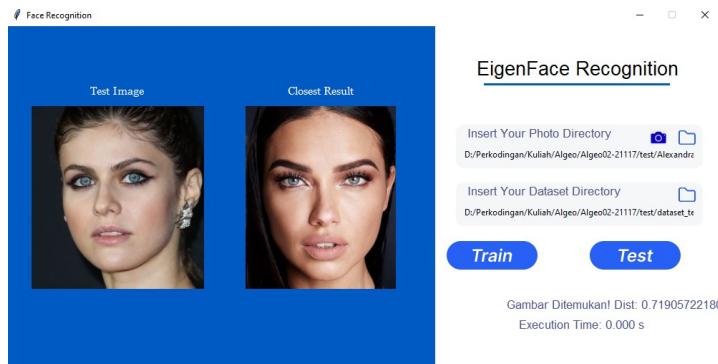
Gambar 4.1.1 Tampilan Layar GUI Sebelum Menerima Input Foto dan Dataset

4.2 Eksperimen 2



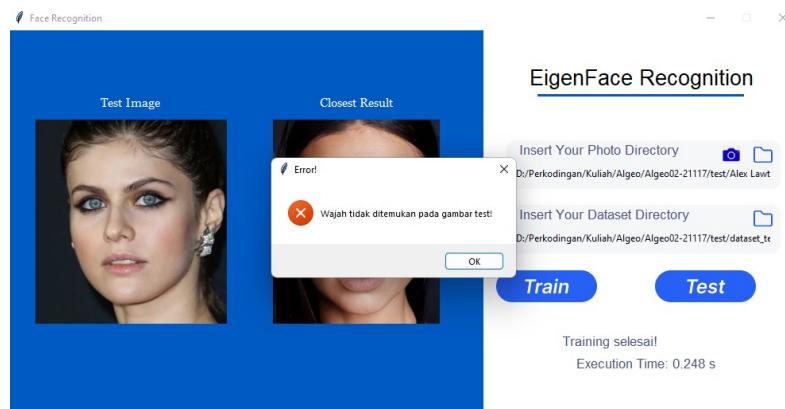
Gambar 4.1.2 Tangkapan Layar GUI Setelah Melakukan Training

4.3 Eksperimen 3



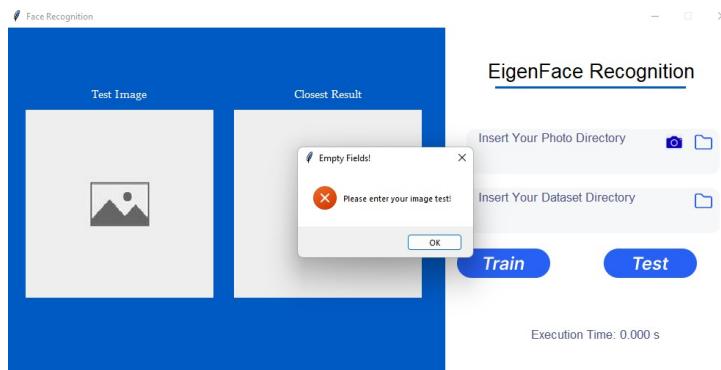
Gambar 4.1.3 Tangkapan Layar GUI Hasil Pencocokan dengan Metode EigenFace

4.4 Eksperimen 4



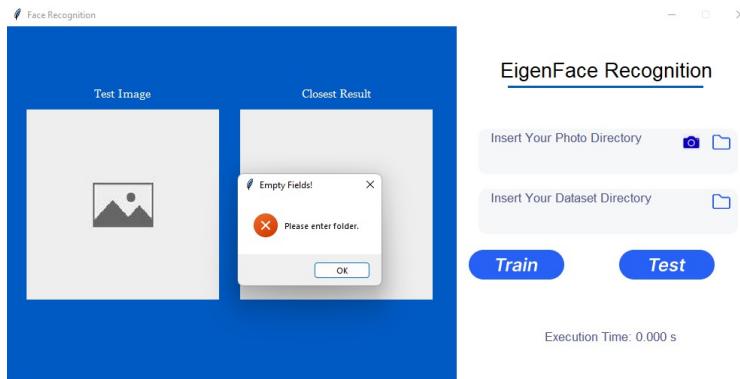
Gambar 4.1.4 Tangkapan Layar GUI Hasil Pencocokan Jika Tidak Ditemukan Wajah Pada Foto yang Akan Dites

4.5 Eksperimen 5



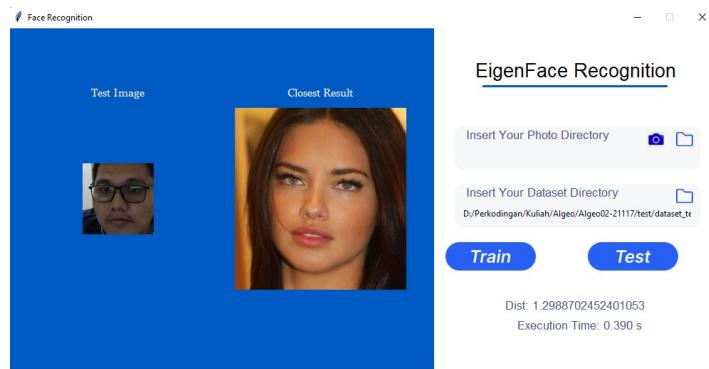
Gambar 4.1.5 Tangkapan Layar GUI Hasil Pencocokan Jika Tidak Ada Input Foto

4.6 Eksperimen 6



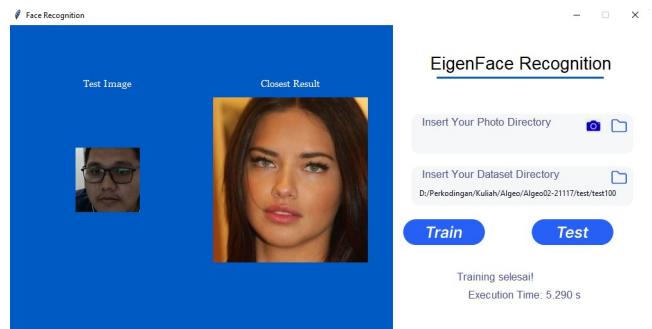
Gambar 4.1.6 Tangkapan Layar GUI Hasil Pencocokkan Jika Tidak Ada Input Folder

4.7 Eksperimen 7

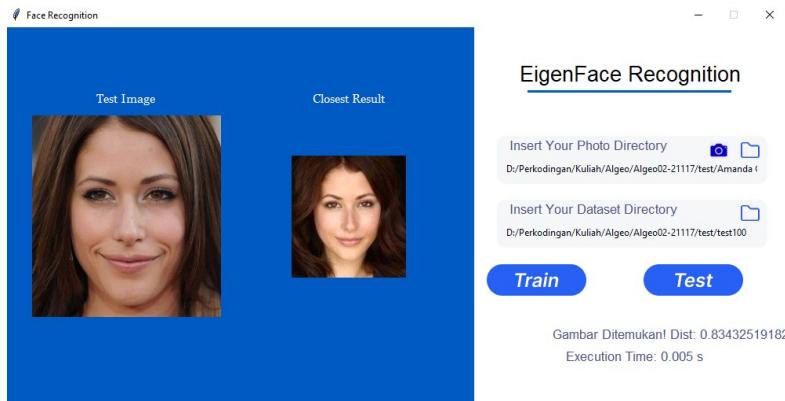


Gambar 4.1.7 Tangkapan Layar GUI Hasil Pencocokkan Foto Dari Kamera

4.8 Eksperimen 8

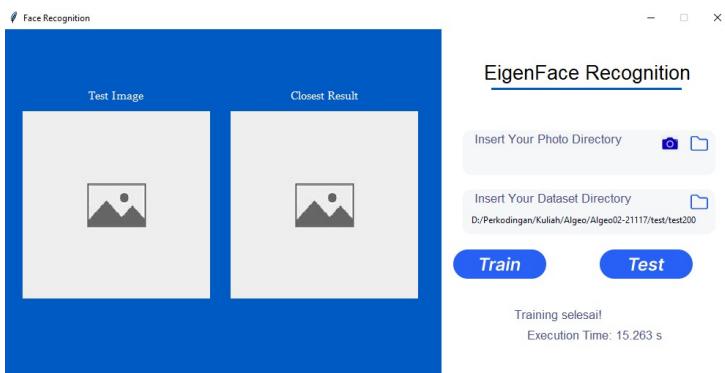


Gambar 4.1.8 Tangkapan Layar GUI Hasil Melakukan Training dari 100 Gambar Dataset

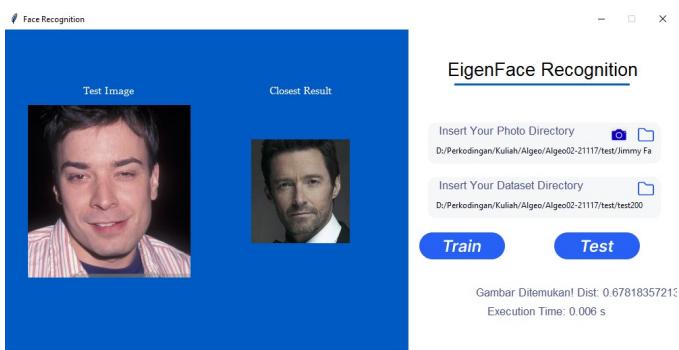


Gambar 4.1.9 Tangkapan Layar GUI Hasil Pencocokkan Foto dengan 100 Gambar Dataset

4.9 Eksperimen 9



Gambar 4.1.9 Tangkapan Layar GUI Hasil Training dengan 200 Gambar Dataset



Gambar 4.1.10 Tangkapan Layar GUI Hasil Pencocokkan Foto dengan 200 Gambar Dataset

BAB V

Kesimpulan, Saran, dan Refleksi

5.1 Kesimpulan

Pada Tugas Besar II IF2123 Aljabar Linier dan Geometri ini, kami berhasil membuat sebuah program menggunakan bahasa pemrograman Python dengan memanfaatkan sejumlah library di OpenCV untuk membuat suatu program pengenalan wajah menggunakan Eigenface. Fitur-fitur dalam program kami diantaranya insert test image dan dataset, execution time, serta tampilan gambar test image dan hasil pencocokan. Tingkat akurasi program kami dapat dikatakan menyerupai dengan baseline yang diberikan dalam spesifikasi walaupun kami tidak menggunakan KAZE features.

5.2 Saran

Dalam pelaksanaan tugas besar ini, kami menyadari segala kendala dan keterbatasan program yang telah kami buat. Kami menyadari bahwa program yang kami buat masih belum maksimal dan masih dapat terdapat kekurangan. Tentunya dengan menyadari adanya kekurangan ini, program ini masih dapat dikembangkan untuk optimalisasi lebih lanjut dan dapat dilakukan efisiensi kerja.

5.3 Refleksi

Setelah menyelesaikan tugas besar ke-2 IF2123 Aljabar Linier dan Geometri, kami dapat merefleksikan bahwa komunikasi antar anggota kelompok berjalan cukup baik sehingga tidak terjadi miskomunikasi dan kesalahpahaman dalam pelaksanaan tugas besar ini, sebelum dimulainya pelaksanaan tugas besar ini, sudah dilakukan diskusi untuk membahas pembagian kerja untuk setiap anggota kelompok, dan terakhir kami menyadari perlunya mempelajari web development agar untuk kedepannya dapat menciptakan sebuah website yang lebih fungsional serta user-friendly.

Daftar Referensi

<https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2020-2021/Algeo-18-Nilai-Eigen-dan-Vektor-Eigen-Bagian1.pdf>

<https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2022-2023/Algeo-19-Nilai-Eigen-dan-Vektor-Eigen-Bagian2-2022.pdf>

<https://www.geeksforgeeks.org/ml-face-recognition-using-eigenfaces-pca-algorithm/>

<https://onionessquareality.wordpress.com/2009/02/11/face-recognition-using-eigenfaces-and-distance-classifiers-a-tutorial/>

<https://pyimagesearch.com/2021/05/10/opencv-eigenfaces-for-face-recognition/>

Lampiran

<https://github.com/ferindya/Algeo02-21117>