

# LAPORAN TUGAS BESAR

**IF-2123 Aljabar Linier dan Geometri  
Kelas Mahasiswa (K-1) / Kelompok HananGeming**

**Dosen : Dr. Judhi Santoso, M.Sc.**



## **Anggota Kelompok :**

<b>Muhammad Hanan</b>	<b>(13521041)</b>
<b>Bagas Aryo Seto</b>	<b>(13521081)</b>
<b>Mohammad Farhan Fahrezy</b>	<b>(13521106)</b>

**Sekolah Teknik Elektro dan Informatika  
Institut Teknologi Bandung  
2022**

# Daftar Isi

<b>Daftar Isi</b>	<b>1</b>
<b>Bab 1: Deskripsi Masalah</b>	<b>2</b>
<b>1.1 Deskripsi Masalah</b>	<b>2</b>
<b>1.2 Solusi Permasalahan</b>	<b>2</b>
<b>Bab 2: Teori Singkat</b>	<b>3</b>
<b>2.1 Perkalian Matriks</b>	<b>3</b>
<b>2.2 Eigenvalue</b>	<b>3</b>
<b>2.3 Eigenvector</b>	<b>3</b>
<b>2.4 Power Iteration Method</b>	<b>4</b>
<b>2.5 QR Decomposition</b>	<b>4</b>
<b>2.6 Euclidean Distance</b>	<b>5</b>
<b>2.7 Eigenface</b>	<b>6</b>
<b>2.8 Tkinter</b>	<b>6</b>
<b>Bab 3: Implementasi pustaka dan program</b>	<b>7</b>
<b>3.1 Penjelasan Library yang Digunakan</b>	<b>7</b>
<b>3.2 Penjelasan Fungsi atau Prosedur yang Digunakan</b>	<b>7</b>
<b>Bab 4: Eksperimen</b>	<b>10</b>
<b>4.1 Tampilan GUI</b>	<b>10</b>
<b>4.2 Hasil Face Recognition</b>	<b>11</b>
<b>Bab 5: Kesimpulan, saran, dan refleksi</b>	<b>13</b>
<b>5.1 Kesimpulan</b>	<b>13</b>
<b>5.2 Saran</b>	<b>13</b>
<b>5.3 Refleksi</b>	<b>13</b>
<b>Daftar Referensi</b>	<b>14</b>
<b>Lampiran</b>	<b>15</b>

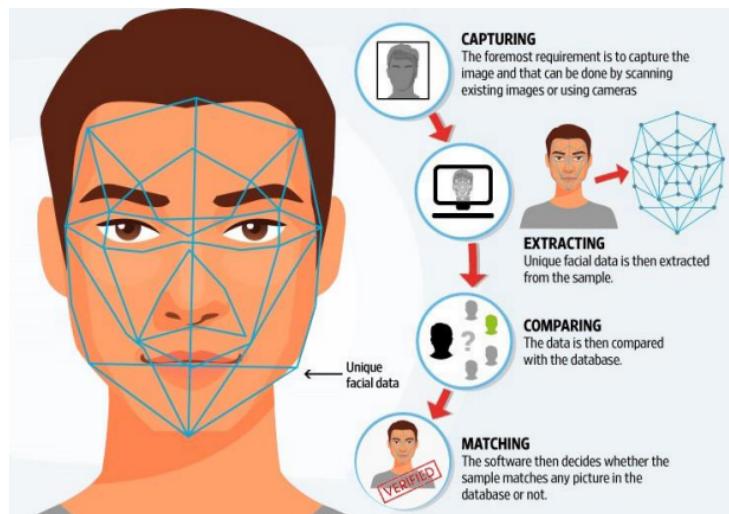
# Bab 1: Deskripsi Masalah

## 1.1 Deskripsi Masalah

Pengenalan wajah atau *face recognition* merupakan teknologi biometrik yang dapat digunakan untuk mengidentifikasi wajah seseorang untuk berbagai kepentingan khususnya di bidang keamanan. Program pembuatan pengenalan wajah melibatkan kumpulan citra wajah tersebut yang sudah disimpan pada suatu database yang kemudian berdasarkan kumpulan citra wajah tersebut, program dapat mempelajari bentuk suatu wajah dan mencocokkan antara kumpulan citra wajah tadi dengan citra wajah baru yang akan diidentifikasi. Berbagai teknik dapat digunakan untuk membuat program *face recognition* seperti *cosine similarity* dan jarak Euclidean, *principal component analysis* (PCA), dan Eigenface.

## 1.2 Solusi Permasalahan

Pada tugas besar ini, kami akan mengimplementasikan penggunaan Eigen untuk pembuatan program *face recognition*. Alur program pengenalan wajah adalah sebagai berikut:



Program ini terdiri atas 2 tahapan utama yaitu *training image* dan pencocokan. Pada tahap *training image*, diperlukan *database* atau *dataset* yang terdiri dari citra wajah yang telah dinormalisasi dari RGB ke Grayscale yang kemudian akan dihitung nilai matriks Eigenface-nya. Selanjutnya, pencocokan akan dilakukan dengan cara mencari selisih terdekat antara nilai eigen dari wajah yang berada pada *database* dengan nilai eigen dari wajah yang ingin dikenali. Pembuatan program ini akan dibuat dengan menggunakan bahasa pemrograman Python dengan memanfaatkan beberapa *library* seperti OpenCV, numpy, dan PIL.

## Bab 2: Teori Singkat

### 2.1 Perkalian Matriks

Perkalian antara dua matriks misal matriks A dan B, hanya bisa dilakukan jika dilakukan jika jumlah kolom A sama dengan jumlah baris B. Hasil dari perkalian matriks AxB memiliki ukuran baris yang sama dengan matriks A dan kolom yang sama dengan matriks B.

Misalkan matriks A memiliki ordo (3x3) dan matriks B memiliki ordo (3x3), maka hasil perkalian matriks A x B misal C juga memiliki ordo (3x3). Elemen matriks C pada baris ke-i dan kolom ke-j atau  $C_{ij}$  diperoleh dari jumlah hasil perkalian elemen-elemen baris ke-i pada matriks A dan kolom ke-j pada matriks B.

$$\begin{aligned}A \times B &= \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} \times \begin{pmatrix} 9 & 8 & 7 \\ 6 & 5 & 4 \\ 3 & 2 & 1 \end{pmatrix} \\&= \begin{pmatrix} 1 \cdot 9 + 2 \cdot 6 + 3 \cdot 3 & 1 \cdot 8 + 2 \cdot 5 + 3 \cdot 2 & 1 \cdot 7 + 2 \cdot 4 + 3 \cdot 1 \\ 4 \cdot 9 + 5 \cdot 6 + 6 \cdot 3 & 4 \cdot 8 + 5 \cdot 5 + 6 \cdot 2 & 4 \cdot 7 + 5 \cdot 4 + 6 \cdot 1 \\ 7 \cdot 9 + 8 \cdot 6 + 9 \cdot 3 & 7 \cdot 8 + 8 \cdot 5 + 9 \cdot 2 & 7 \cdot 7 + 8 \cdot 4 + 9 \cdot 1 \end{pmatrix} \\&= \begin{pmatrix} 9 + 12 + 9 & 8 + 10 + 6 & 7 + 8 + 3 \\ 36 + 30 + 18 & 32 + 25 + 12 & 28 + 20 + 6 \\ 63 + 48 + 27 & 56 + 40 + 18 & 49 + 32 + 9 \end{pmatrix} \\&= \begin{pmatrix} 30 & 24 & 18 \\ 84 & 69 & 54 \\ 138 & 114 & 90 \end{pmatrix}\end{aligned}$$

Gambar 2.1.1 Contoh Perkalian Matriks

Perkalian matriks tidak bersifat komutatif,  $A \times B \neq B \times A$ .

### 2.2 Eigenvalue

Nilai eigen, atau biasa disebut *eigenvalue*, sering dilambangkan dengan “ $\lambda$ ” yang artinya sebagai karakteristik dari matriks berukuran  $N \times N$ . Apabila suatu nilai eigen dari matriks X dikalikan dikalikan dengan matriks X, hasilnya akan berupa kelipatan matriks X itu sendiri. Nilai eigen ini dapat digunakan untuk membentuk vektor eigen nantinya.

### 2.3 Eigenvector

Jika matriks A berukuran  $N \times N$ , maka vektor tidak nol  $x$  di dalam  $R^n$  dinamakan **vektor eigen (eigenvector)** dari A jika  $Ax$  adalah kelipatan skalar dari  $x$ . Kata “eigen” berasal dari bahasa Jerman yang berarti ‘asli’ atau ‘karakteristik’. Dengan kata lain vektor eigen merupakan vektor karakteristik dari sebuah matriks berukuran  $N \times N$ .

Eigenvector dari suatu matriks berukuran  $N \times N$  dapat dicari dengan persamaan berikut.

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

Gambar 2.3.1 Persamaan Karakteristik

Dengan  $I$  adalah matriks identitas berukuran  $N \times N$ . Agar  $(\lambda I - A)x = 0$  memiliki solusi tidak nol, maka haruslah  $\det(\lambda I - A) = 0$ . Persamaan  $\det(\lambda I - A) = 0$  disebut sebagai persamaan karakteristik dan akar-akar dari persamaan tersebut  $(\lambda)$  disebut sebagai akar-akar karakteristik atau *eigenvalue*.

## 2.4 Power Iteration Method

**Power Iteration** merupakan salah satu metode untuk mengaproksimasi nilai dominan dari *eigenvalues* dan *eigenvector*. Suatu *eigenvalue* dapat dinyatakan dominan apabila tiap  $\lambda_i$  dari matriks  $A$ , maka  $|\lambda_1| > |\lambda_i|$  untuk semua  $i = 2, 3, 4, \dots$  dan *eigenvector*  $v_1$  juga disebut dominan.

Untuk menemukan *eigenvector* dominan, dapat digunakan **Power Method** dengan mengambil sebuah vektor random  $x_0$  lalu melakukan iterasi  $x_n = A \cdot x_{n-1}$  hingga mendapatkan nilai sebenarnya.

Metode tersebut hanya dapat digunakan untuk menemukan **eigenvector dominan**, tetapi karena kita mengetahui bahwa *eigenvector* lain ortogonal terhadap *eigenvector* dominan, **Power Iteration** dapat digunakan terus menerus sampai ditemukan seluruh *eigenvector* dari suatu matriks  $A$ .

Untuk menemukan seluruh *eigenvector* dari suatu matriks  $A$  berukuran  $N \times N$ , dibuat sebuah matriks sembarang  $Q_0$  yang berukuran sama dengan matriks  $A$ . Lalu gunakan **QR Decomposition** untuk mendapatkan matriks orthogonal  $Q_1$  dan matriks *upper-triangular*  $R_1$ . Lalu, lakukan perkalian  $Q_1$  dan  $A$  yang menghasilkan matriks baru  $QR$ . Ulangi metode **QR Decomposition** untuk mendapatkan  $Q_i$  dan  $R_i$  sampai nilai dari diagonal matriks  $R_i$  (*eigenvalue*) tidak berubah. Hasil  $Q_i$  akhir dari metode ini merupakan matriks yang berisi  $n$  kolom yang masing-masing kolom merupakan aproksimasi dari tiap *eigenvector*.

## 2.5 QR Decomposition

**QR Decomposition** merupakan salah satu metode untuk menyelesaikan masalah dalam aljabar linear yang berhubungan dengan *eigenvalue* dan *eigenvector*. Suatu matriks  $A$  dapat difaktorisasikan menjadi matriks ortogonal  $Q$  dan matriks *upper-triangular*  $R$ . Algoritma **Gram-Schmidt** merupakan salah satu metode untuk melakukan **QR Decomposition**.

Misal suatu matriks  $A$ .

$$A = [a_1 \mid a_2 \mid \cdots \mid a_n]$$

Gambar 2.5.1 Matriks  $A$  dengan  $a_i$  sebagai Kolom

Algoritma ini secara garis besar melakukan 2 hal, yaitu me-**normalize** suatu vektor dan me-**orthogonalize** vektor selanjutnya. Pertama, kita set  $u_1 = a_1$ , lalu *normalize*.

$$u_1 = a_1, \quad e_1 = \frac{u_1}{\|u_1\|}$$

Lalu *orthogonalize* untuk mendapatkan  $u_2$  agar  $e_2$  dapat dihitung

$$u_2 = a_2 - (a_2 \cdot e_1)e_1, \quad e_2 = \frac{u_2}{\|u_2\|}$$

Dan untuk  $k = 2, 3, \dots, n-1$  dapat dihitung

$$u_{k+1} = a_{k+1} - (a_{k+1} \cdot e_1)e_1 - \cdots - (a_{k+1} \cdot e_k)e_k, \quad e_{k+1} = \frac{u_{k+1}}{\|u_{k+1}\|}$$

Maka didapatkan

$$A = [a_1 \mid a_2 \mid \cdots \mid a_n] = [e_1 \mid e_2 \mid \cdots \mid e_n] \begin{bmatrix} a_1 \cdot e_1 & a_2 \cdot e_1 & \cdots & a_n \cdot e_1 \\ 0 & a_2 \cdot e_2 & \cdots & a_n \cdot e_2 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_n \cdot e_n \end{bmatrix}$$

Dengan nilai Q dan R sebagai berikut

$$Q = [a_1 \mid a_2 \mid \cdots \mid a_n] = [e_1 \mid e_2 \mid \cdots \mid e_n]$$

$$R = \begin{bmatrix} a_1 \cdot e_1 & a_2 \cdot e_1 & \cdots & a_n \cdot e_1 \\ 0 & a_2 \cdot e_2 & \cdots & a_n \cdot e_2 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_n \cdot e_n \end{bmatrix}$$

## 2.6 Euclidean Distance

Euclidean Distance didefinisikan sebagai jarak antara dua titik. Dengan kata lain, jarak Euclidean antara dua titik dalam ruang Euclidean didefinisikan sebagai panjang ruas garis antara dua titik. Rumus jarak Euclidean, seperti namanya, memberikan jarak antara dua titik atau jarak garis lurus. Mari kita asumsikan bahwa  $(x_1, y_1)$ ,  $(x_2, y_2)$  adalah dua titik dalam bidang dua dimensi. Berikut adalah rumus jarak Euclidean.

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Dengan  $d$  adalah jarak dari kedua titik tersebut.

## **2.7 Eigenface**

Eigenface merupakan salah satu pengimplementasian eigen value dan eigen vektor yang didasari oleh PCA (*Principal Component Analysis*) dan ditemukan oleh Sirovich bersama Kirby. Pembuatan algoritma Eigenface didasari atas kumpulan vektor eigen. Dengan eigenface ini dapat digunakan untuk membuat program *face recognition*.

## **2.8 Tkinter**

Tkinter adalah graphic user interface (GUI) standar python digunakan untuk membuat tampilan aplikasi dengan komponen-komponen yang ada di modul tkinter seperti Button, Textbox, Label, Frame, Window yang mana sangat mendukung dalam penciptaan aplikasi GUI .

## Bab 3: Implementasi pustaka dan program

### 3.1 Penjelasan Library yang Digunakan

Nama Library	Penjelasan
tkinter	<ol style="list-style-type: none"><li>1. Untuk membangun window yang digunakan untuk keberjalan program.</li><li>2. Akan mengambil filedialog, yang berfungsi untuk pembacaan file.</li><li>3. Penulisan suatu kata atau kalimat, bersifat statis, yang akan didisplay pada window menggunakan canvas.</li><li>4. Mendisplay foto yang akan ditampilkan pada window menggunakan frame dengan bantuan library ImageTk yang ada pada PIL.</li></ol>
PIL	<ol style="list-style-type: none"><li>1. Membuka jenis file foto dengan menggunakan Image dan ImageTk</li><li>2. Untuk mengubah ukuran foto agar menjadi 256x256 dengan menggunakan fungsi resize.</li></ol>
os	<ol style="list-style-type: none"><li>1. Digunakan untuk mendapatkan nama file dalam folder.</li><li>2. Mengambil seluruh file dan folder dari path yang dipilih.</li><li>3. Mengambil nama folder saja tanpa directory secara lengkap.</li></ol>
time	<ol style="list-style-type: none"><li>1. Digunakan untuk perhitungan waktu, yaitu pada saat menghitung time execution.</li></ol>
cv2	<ol style="list-style-type: none"><li>1. Digunakan untuk mengubah mode gambar menjadi grayscale</li><li>2. Digunakan untuk mengubah gambar menjadi matriks</li></ol>
numpy	<ol style="list-style-type: none"><li>1. Digunakan untuk melakukan bermacam perhitungan matematis</li></ol>

### 3.2 Penjelasan Fungsi atau Prosedur yang Digunakan

#### A. File mainwindow.py

Nama	Penjelasan
def start()	Digunakan untuk menjalankan Algoritma utama dengan menghitung eigenface lalu mendisplay hasil closest result yang ditemukan, hasil dari execution time, dan nama file dari closest result yang ditemukan. Jika similarity antara testcase file dengan file closest result itu kecil dari 50, maka akan menampilkan pesan bahwa file tidak ditemukan. Program akan berjalan ketika button start

	diklik.
def cfile()	Digunakan untuk memilih file test case menggunakan fungsi askopenfilename, jika file test case sudah dipilih, maka foto testcase yang dipilih akan didisplay dan nama file yang dipilih juga akan didisplay. Fungsi ini akan berjalan jika button choose file diklik.
def cfolder()	Digunakan untuk memilih folder dataset yang akan digunakan dengan menggunakan fungsi askdirectory. Jika folder sudah dipilih, maka fungsi akan mendisplay nama folder yang dipilih tersebut. Fungsi akan berjalan jika button choose folder diklik.

## B. File main.py

Nama	Penjelasan
getDataset(foldername)	Digunakan untuk mengambil dataset gambar dari folder dan mengembalikan matriks berisi vektor-vektor gambar.
getCovarian(dataset)	Digunakan untuk mendapatkan matriks kovarian dari dataset.
writeImage(arr, name)	Digunakan untuk menampilkan vektor arr ke dalam file gambar.
getEigenfaces(dataset, v)	Digunakan untuk mendapatkan eigenfaces dari dataset dan eigenvector v.
getOmega(eigenFaces, subtractedArr)	Digunakan untuk mendapatkan vektor omega dari eigenfaces dan vektor subtracted.
getOmegaSet(eigenfaces, subtracted)	Digunakan untuk mendapatkan matriks vektor omega dari eigenfaces dan matriks vektor subtracted.
getEuclideanAndIndex(omega Set, omegaNew)	Digunakan untuk mendapatkan nilai euclidean distance terkecil serta indeks euclidean distance terkecil antara matriks vektor omega dan omega testing face baru.
getFileName(foldername, index)	Digunakan untuk mendapatkan nama file dari foldername urutan ke indeks.
getThreshold(omegaset)	Digunakan untuk mendapatkan nilai threshold dari matriks vektor omega.
getDistance(vector1, vector2)	Digunakan untuk mendapatkan nilai euclidean distance

	antara 2 vektor.
runprogram(foldername, filename)	Digunakan untuk mendapatkan gambar pada foldername yang memiliki euclidean distance terkecil dengan gambar filename, waktu lama pemrosesan, serta persentase kesamaan kedua gambar.

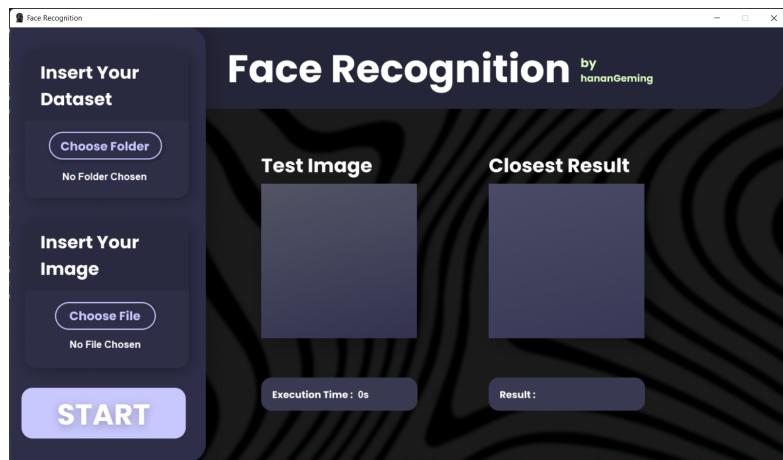
### C. File eigen.py

Nama	Penjelasan
def getEVEV(covarian)	Digunakan untuk mencari nilai eigenvector menggunakan <i>power iteration method</i> dan masih menggunakan algoritma <i>QR Decomposition</i> numpy, yaitu numpy.linalg.qr agar bisa meningkatkan perhitungan eigenvector.
def getEVEV2(covarian)	Memiliki fungsi yang sama dengan fungsi getEVEV(), tetapi tidak menggunakan <i>QR Decomposition</i> numpy melainkan dengan menggunakan implementasi sendiri, yaitu fungsi QRDecomp().
def QRDecomp(mat)	Merupakan fungsi untuk mengeluarkan <i>QR Decomposition</i> dari suatu matriks mat menjadi bentuk ortogonal (Q) dan upper-triangular (R) menggunakan algoritma Gram-Schmidt.
def norm(u)	Nilai normalisasi matriks dalam ruang L <sub>2</sub> yang digunakan dalam fungsi QRDecomp().

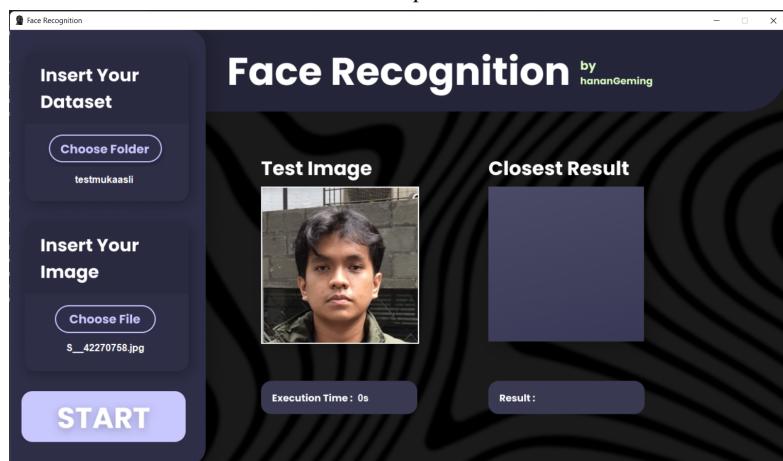
# Bab 4: Eksperimen

## 4.1 Tampilan GUI

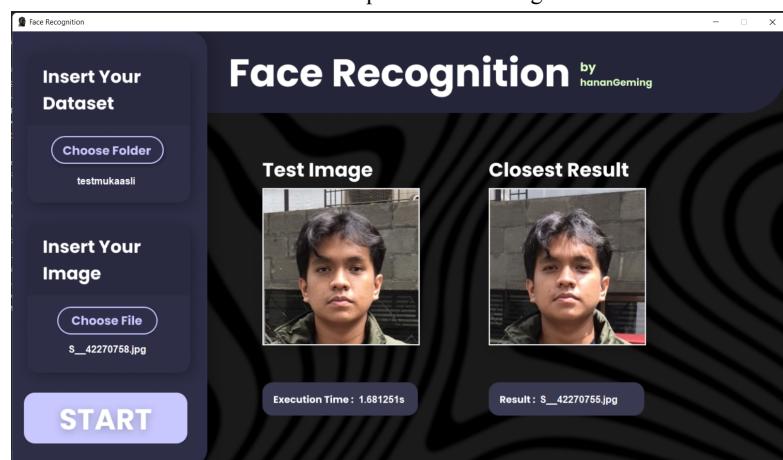
Berikut merupakan tampilan dari aplikasi pada saat awal digunakan, saat digunakan dan saat hasilnya ditampilkan.



Gambar 4.1.1 Tampilan Awal GUI

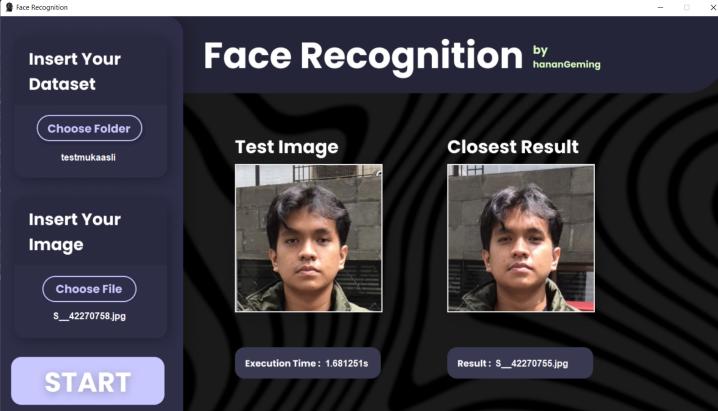
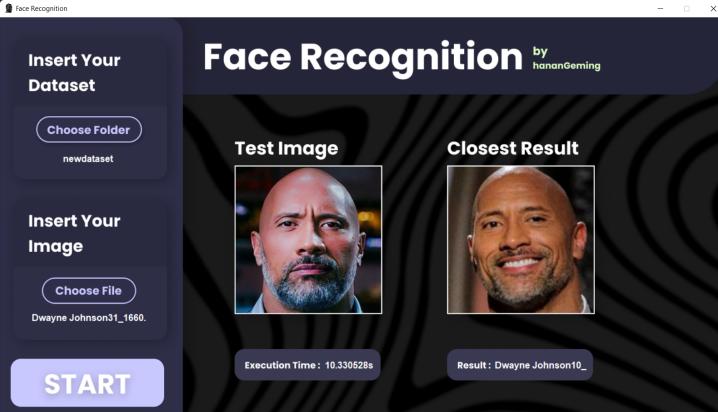
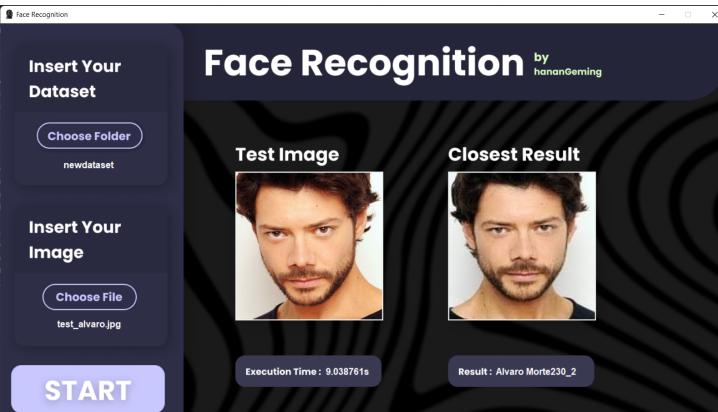


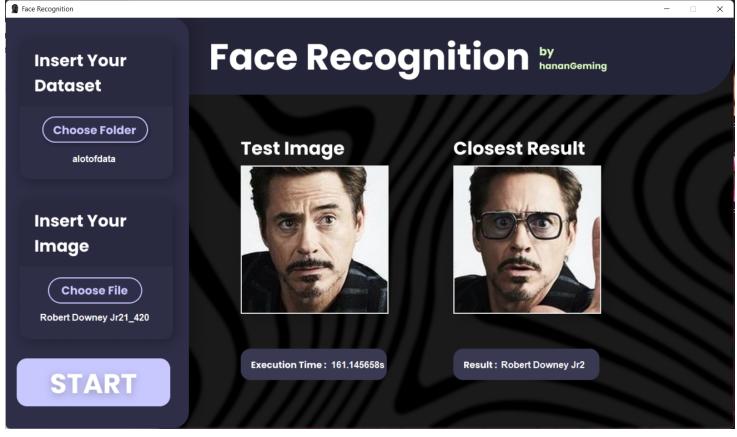
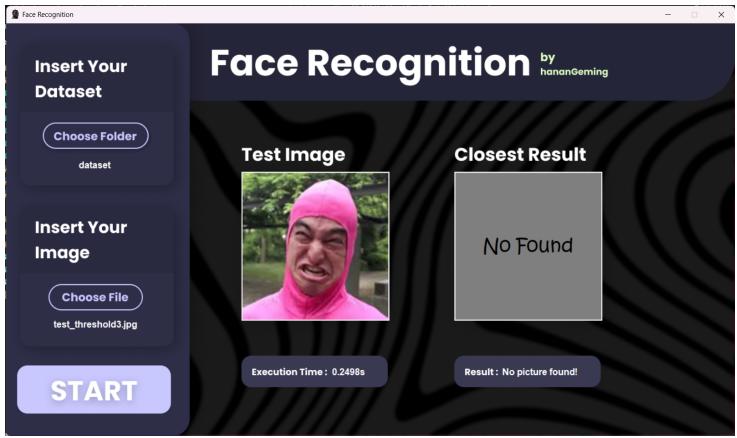
Gambar 4.1.2 Tampilan GUI saat digunakan



Gambar 4.1.3 Tampilan GUI saat Program Selesai

## 4.2 Hasil Face Recognition

No	Hasil Face Recognition	Deskripsi dan Analisis
1	 <p>The screenshot shows the Face Recognition application interface. On the left, there are two input fields: 'Insert Your Dataset' (with 'Choose Folder' button and 'testmukaasi' selected) and 'Insert Your Image' (with 'Choose File' button and 'S_42270755.jpg' selected). A large blue 'START' button is at the bottom. The main area displays the title 'Face Recognition by hananGeming'. It shows a 'Test Image' of a man's face and a 'Closest Result' of the same man's face. Below them are buttons for 'Execution Time: 1.681251s' and 'Result: S_42270755.jpg'.</p>	<p>Ukuran gambar : 361x361 px Banyak dataset : 69 Gambar Waktu eksekusi : 1,68 s</p> <p>Analisis : waktu eksekusi bisa sangat cepat karena dataset yang digunakan relatif sedikit</p>
2	 <p>The screenshot shows the Face Recognition application interface. The setup is identical to the first row: 'Insert Your Dataset' (Choose Folder, 'newdataset') and 'Insert Your Image' (Choose File, 'Dwayne Johnson31_1660.jpg'). The 'START' button is present. The main area shows a 'Test Image' of Dwayne Johnson and a 'Closest Result' of the same person. Buttons for 'Execution Time: 10.330528s' and 'Result: Dwayne Johnson10...' are shown.</p>	<p>Ukuran gambar : 510x554 px Banyak dataset : 300 Gambar Waktu eksekusi : 10,33 s</p> <p>Analisis : waktu eksekusi cenderung lama karena dataset yang sudah lebih banyak.</p>
3	 <p>The screenshot shows the Face Recognition application interface. The setup is identical to the first row: 'Insert Your Dataset' (Choose Folder, 'newdataset') and 'Insert Your Image' (Choose File, 'test_alvaro.jpg'). The 'START' button is present. The main area shows a 'Test Image' of Alvaro Morte and a 'Closest Result' of the same person. Buttons for 'Execution Time: 9.038761s' and 'Result: Alvaro Morte230_2' are shown.</p>	<p>Ukuran gambar : 437x462px Banyak dataset : 300 Gambar Waktu eksekusi : 9,03 s</p> <p>Analisis : waktu eksekusi cenderung lama karena dataset yang sudah lebih banyak.</p>

4	 <p>The screenshot shows the Face Recognition application interface. On the left, there are two input sections: 'Insert Your Dataset' with a 'Choose Folder' button and 'alotofdata' text, and 'Insert Your Image' with a 'Choose File' button and 'Robert Downey Jr21_420' text. A large blue 'START' button is at the bottom. In the center, under 'Face Recognition by hananGeming', it says 'Test Image' with a photo of Robert Downey Jr., 'Closest Result' with a photo of him wearing glasses, and 'Execution Time : 161.145658s' and 'Result : Robert Downey Jr2' below.</p>	<p>Ukuran gambar : 175x184px Banyak dataset : 1086 Gambar Waktu eksekusi : 161,14 s</p> <p>Analisis : waktu eksekusi lama karena dataset yang digunakan itu sangat banyak.</p>
5	 <p>The screenshot shows the Face Recognition application interface. On the left, there are two input sections: 'Insert Your Dataset' with a 'Choose Folder' button and 'dataset' text, and 'Insert Your Image' with a 'Choose File' button and 'test_threshold3.jpg' text. A large blue 'START' button is at the bottom. In the center, under 'Face Recognition by hananGeming', it says 'Test Image' with a photo of a person in a pink hoodie, 'Closest Result' with a gray box containing 'No Found', and 'Execution Time : 0.2498s' and 'Result : No picture found!' below.</p>	<p>Ukuran gambar : 600x600px Banyak dataset : 30 Gambar Waktu eksekusi : 0.2498s</p> <p>Analisis : waktu yang digunakan itu sedikit jadi waktu eksekusinya cepat, namun tidak ditemukan foto yang similarity nya lebih besar dari 50% jadi ditampilkan tidak ditemukannya foto yang mirip dengan test case.</p>

# Bab 5: Kesimpulan, saran, dan refleksi

## 5.1 Kesimpulan

Pada proses pembuatan program pengenalan wajah atau *face recognition* ini, kami berhasil membentuk suatu aplikasi pencocokan foto dengan dataset yang ditentukan sebelumnya. Tentunya aplikasi ini dapat diimplementasikan dengan menggunakan algoritma eigenface. Algoritma eigenface juga dari pengaplikasian beberapa materi perkuliahan Aljabar Linier dan Geometri, yaitu operasi vektor, nilai dan vektor eigen, dan masih banyak lainnya.

## 5.2 Saran

Kami berharap pada pelaksanaan tugas besar selanjutnya dapat disertakan asistensi untuk mempermudah kami dan mahasiswa yang lain dalam menyelesaikan tugas besar ini. Kemudian kami berharap spesifikasi dibuat di google docs agar mempermudah para mahasiswa jika ada update atau perubahan mengenai update terkait spesifikasi. Selain itu, kami juga berharap untuk pelampiran referensi diperbanyak agar mempermudah kami dalam mengerjakan tugas besar.

## 5.3 Refleksi

Pada tugas besar yang kedua ini, kami merasa lebih baik dalam menghadapi tugas besar seperti mencicil pengerjaannya dari minggu pertama tugas besar ini rilis, namun hasilnya masih belum maksimal seperti tidak sempat untuk membuat bonus kamera.. Tetapi, kami tetap mengapresiasi teman-teman sekelompok yang tidak menghilang dan cekatan ketika diajak mengerjakan tugas besar.

# Daftar Referensi

- [https://python.quantecon.org/qr\\_decomp.html](https://python.quantecon.org/qr_decomp.html) (diakses pada 20/11/2022)
- [http://mlwiki.org/index.php/Power\\_Iteration](http://mlwiki.org/index.php/Power_Iteration) (diakses pada 22/11/2022)
- <https://www.activestate.com/resources/quick-reads/how-to-add-images-in-tkinter/> (diakses pada 17/11/2022)
- <https://stackoverflow.com/questions/58625628/is-there-any-command-attribute-for-label-in-tkinter-python> (diakses pada 17/11/2022)
- <https://pythonguides.com/python-tkinter-image/> (diakses pada 17/11/2022)
- <https://www.educba.com/tkinter-messagebox/> (diakses pada 15/11/2022)
- <https://stackoverflow.com/questions/48306487/python-3-tkinter-center-label-text> (diakses pada 15/11/2022)
- <https://stackoverflow.com/questions/63145692/how-to-overwrite-an-image-in-tkinter> (diakses pada 14/11/2022)

# **Lampiran**

Link Repository Github: <https://github.com/bagas003/Algeo02-21041>

Link Video Youtube: [https://bit.ly/Algeo02\\_21041](https://bit.ly/Algeo02_21041)