

LAPORAN TUGAS BESAR 2 IF2123

ALJABAR LINIER DAN GEOMETRI

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



Kelompok 11 – Parahlimpik

Rizky Abdillah Rasyid – 13521109

Muhammad Zaki Amanullah – 13521146

Muhammad Dimas Sakti Widyatmaja – 13521160

Semester 1 Tahun 2022/2023

DAFTAR ISI

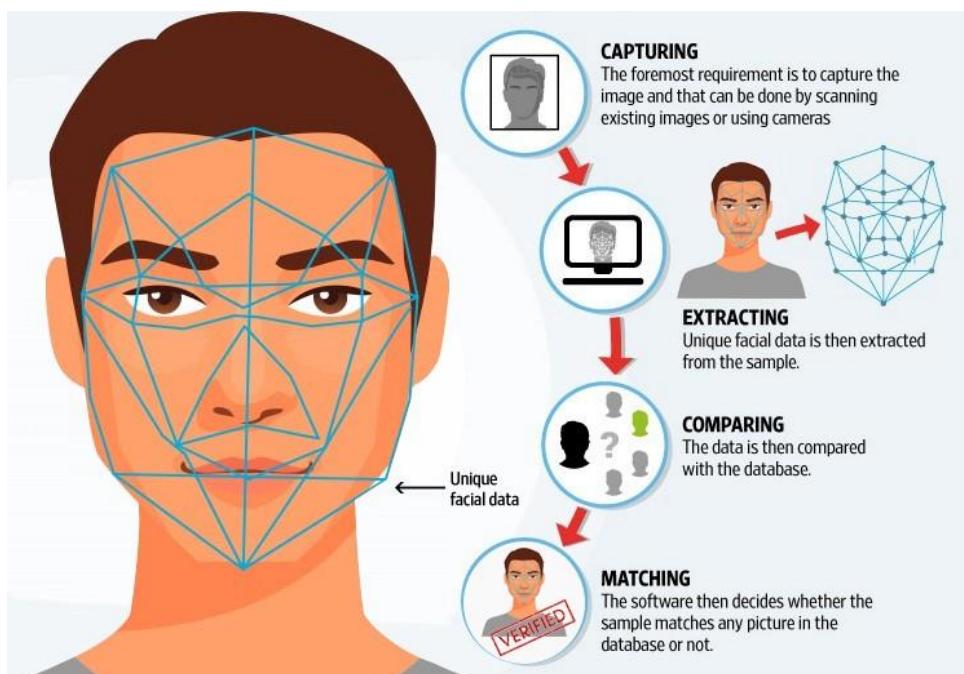
DAFTAR ISI	1
BAB 1	
DESKRIPSI MASALAH	2
1.1 Abstraksi	2
1.2 PENGGUNAAN PROGRAM	5
1.3 SPESIFIKASI TUGAS	5
BAB 2	
TEORI SINGKAT	8
2.1. Perkalian Matriks	8
2.2. Vektor Eigen	8
2.3. Nilai Eigen	9
2.4. Eigenface	10
BAB 3	
IMPLEMENTASI	12
3.1. Kakas/ Tech Stack	12
3.2. Algoritma	12
3.2.1. averageface.py	12
3.2.2. getcovariant.py	12
3.2.3. main.py	13
3.2.4. app.py	14
3.2.5. Garis Besar Program	15
BAB 4	
EKSPERIMEN	18
4.1. Eksperimen 1	18
4.1.1. Gambar Masukan Terdapat di Folder Dataset	18
4.1.2. Gambar Masukan Tidak Terdapat di Dataset	19
4.2. Eksperimen 2	20
4.2.1. Gambar masukan terdapat di Folder Dataset	20
4.2.2. Gambar masukan tidak terdapat di Folder Dataset	21
4.3. Gambar Masukan dari Kamera	22
4.4. Analisis Hasil Eksperimen	23
BAB 5	
KESIMPULAN, SARAN, DAN REFLEKSI	25
5.1. KESIMPULAN	25
5.2. SARAN	25
5.3. REFLEKSI	25
LAMPIRAN	28

BAB 1

DESKRIPSI MASALAH

1.1 Abstraksi

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.



Gambar 1. Alur proses di dalam sistem pengenalan wajah (Sumber:

<https://www.shadowsystem.com/page/20>

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 pencocokan. 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.

Algoritma Eigenface

1. Langkah pertama adalah menyiapkan data dengan membuat suatu himpunan S yang terdiri dari seluruh training image, $(\Gamma_1, \Gamma_2, \dots, \Gamma_M)$

$$S = (\Gamma_1, \Gamma_2, \dots, \Gamma_M) \quad (1)$$

2. Langkah kedua adalah ambil nilai tengah atau mean (Ψ)
- (2)
3. Langkah ketiga kemudian cari selisih (Φ) antara nilai training image (Γ_i) dengan nilai tengah (Ψ)

$$\phi_i = \Gamma_i - \Psi \quad (3)$$

4. Langkah keempat adalah menghitung nilai matriks kovarian (C)

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

$$L = A^T A \quad L = \phi_m^T \phi_n \quad (4)$$

5. Langkah kelima menghitung eigenvalue (λ) dan eigenvector (v) dari matriks kovarian (C)

$$C \times vi = \lambda i \times vi \quad (5)$$

6. Langkah keenam, setelah eigenvector (v) diperoleh, maka eigenface (μ) dapat dicari dengan:

$$\mu_i = \sum_{k=1}^M v_{ik} \phi_k$$

$$l = 1, \dots, M \quad (6)$$

$$l = 1, \dots, M$$

Tahapan Pengenalan wajah:

1. Sebuah image wajah baru atau test face (Γ_{new}) akan dicoba untuk dikenali, pertama terapkan cara pada tahapan pertama perhitungan eigenface untuk mendapatkan nilai eigen dari image tersebut.

$$\mu_{new} = v \times \Gamma_{new} - \Psi \quad (7)$$

$$\Omega = \mu_1, \mu_2, \dots, \mu_M$$

2. Gunakan metode euclidean distance untuk mencari jarak (distance) terpendek antara nilai eigen dari training image dalam database dengan nilai eigen dari image testface.

$$\varepsilon_k = \Omega - \Omega_k \quad (8)$$

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-searchwith-opencv-for-newbies-3c59796bf774>

Nilai batas kemiripan citra test face dapat ditentukan oleh pembuat program melalui percobaan.

Berikut merupakan referensi pengenalan wajah dengan metode eigenface:

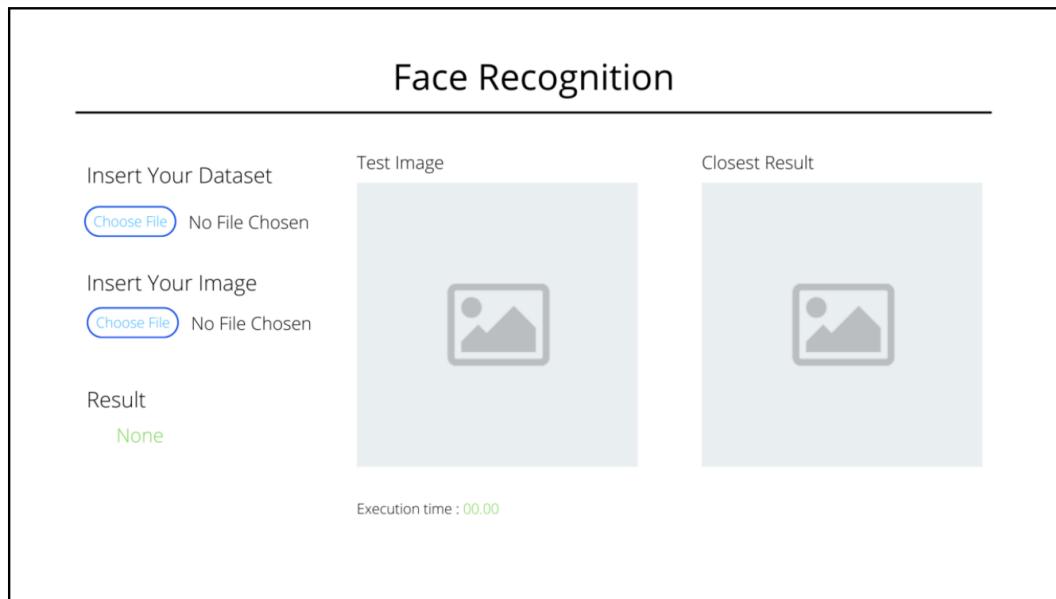
<https://jurnal.untan.ac.id/index.php/jcskommipa/article/download/9727/9500>

[https://www.geeksforgeeks.org/ml-face-recognition-using-eigenfaces-pca-a](https://www.geeksforgeeks.org/ml-face-recognition-using-eigenfaces-pca-algorithm/)
[lgorithm/](#).

1.2 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.



Gambar 3. Contoh tampilan layout dari aplikasi web yang dibangun.

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.

1.3 SPESIFIKASI TUGAS

Buatlah 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.

Bonus:

Bagian A (Kamera)

1. Terdapat fitur kamera yang dapat mengenali wajah secara realtime menggunakan webcam ketika program dijalankan.
2. Teknis pengenalan wajah melalui kamera dibebaskan oleh pembuat program. (contoh: program dieksekusi setiap 10 detik sekali).
3. Fitur kamera merupakan fitur tambahan, fitur utama upload gambar melalui GUI tetap harus ada.

Bagian B (Video)

1. Video penjelasan algoritma dan aplikasi program yang diunggah ke youtube.
2. Video dibuat sekreatif mungkin dengan target untuk mensosialisasikan ilmu yang kalian sudah pelajari dan terapkan pada program ini. Bukan hanya video mengenai penggunaan aplikasi.
 - A. Membuat program pengenalan wajah dalam Bahasa Python berbasis GUI.
 - B. Program dapat menerima input folder dataset dan sebuah gambar citra wajah.

- C. Program dapat menampilkan gambar citra wajah yang dipilih pnegguna
- D. Program melakukan pencocokan wajah dengan koleksi wajah yang ada di folder yang telah dipilih. Metrik untuk pengukuran kemiripan menggunakan eigenface + jarak euclidean.
- E. Program menampilkan 1 hasil pencocokan pada dataset yang paling dekat dengan gambar input atau memberikan pesan jika tidak didapatkan hasil yang sesuai.
- F. 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.

BAB 2

TEORI SINGKAT

2.1. Perkalian Matriks

Dalam matematika, perkalian matriks adalah suatu operasi biner dari dua matriks yang menghasilkan sebuah matriks. Agar dua matriks dapat dikalikan, banyaknya kolom pada matriks pertama harus sama dengan banyaknya baris pada matriks kedua. Matriks hasil perkalian keduanya, akan memiliki baris sebanyak baris matriks pertama, dan kolom sebanyak kolom matriks kedua. Perkalian matriks A dan B dinyatakan sebagai AB.

Penjumlahan dua buah matriks $C_{m \times n} = A_{m \times n} + B_{m \times n}$.

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1p} \\ b_{21} & b_{22} & \cdots & b_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \cdots & b_{np} \end{pmatrix}$$

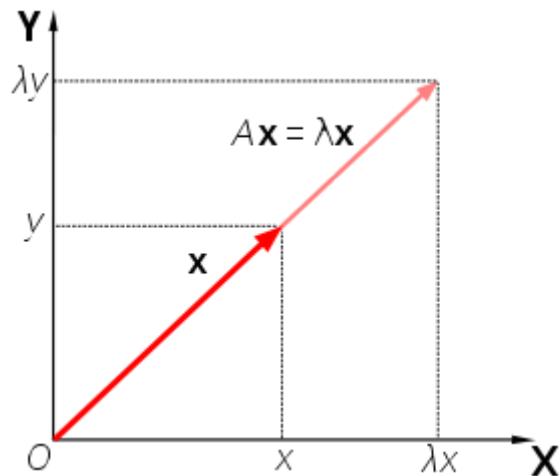
Misal $A = [a_{ij}]$ dan $B = [b_{ij}]$, maka $C = A \times B = [c_{ij}]$, $c_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \dots + a_{in}b_{nj}$ syarat: jumlah kolom A sama dengan jumlah baris B.

$$\mathbf{C} = \begin{pmatrix} c_{11} & c_{12} & \cdots & c_{1p} \\ c_{21} & c_{22} & \cdots & c_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ c_{m1} & c_{m2} & \cdots & c_{mp} \end{pmatrix}$$

2.2. Vektor Eigen

Kata "eigen" berasal dari Bahasa Jerman yang artinya "asli" atau "karakteristik". Jika A adalah matriks $n \times n$, maka vektor taknol x dinamakan vektor eigen (eigenvector) dari A jika Ax adalah kelipatan skalar dari x; yakni $Ax=\lambda x$.

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



Sumber gambar: Wikipedia.

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.

Diberikan sebuah matriks A berukuran $n \times n$. Vektor eigen dan nilai eigen dari matriks A dihitung sebagai berikut:

$$Ax = \lambda x$$

$$I Ax = \lambda I x \text{ (kalikan kedua ruas dengan } I = \text{matriks identitas)}$$

$$Ax = \lambda I x$$

$(\lambda I - A)x = 0$ 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. Nilai Eigen

Skalar λ dinamakan nilai eigen (eigenvalue) dari A dan x dikatakan vektor eigen yang bersesuaian dengan λ . Dengan kata lain, nilai eigen menyatakan nilai karakteristik dari sebuah matriks yang berukuran $n \times n$.

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. Dikembangkan oleh Sirovich and Kirby dan digunakan oleh Matthew Turk dan Alex Pentland untuk klasifikasi 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.

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

Eigenface menggunakan metode Principal Component Analysis (PCA) dan dapat digunakan untuk mereduksi dimensi gambar wajah sehingga menghasilkan variabel yang lebih sedikit yang lebih mudah untuk diobservasi dan ditangani. Hasil yang diperoleh kemudian akan dimasukkan ke suatu pattern classifier untuk menentukan identitas pemilik wajah

Pengenalan wajah dapat diartikan sebagai pengelompokan sebuah wajah sebagai “wajah dikenali” atau “wajah tidak dikenali”, setelah membandingkannya ke dalam wajah individu dikenali yang tersimpan sebelumnya (*dataset wajah*).

Terdapat 6 komponen utama dalam pengenalan wajah:

1. Acquisition Module

Citra wajah dihadirkan ke sistem di modul ini melalui media *input* atau dari file di penyimpanan.

2. Pre-processing Module

Hasil dari modul akuisisi akan dinormalisasi dan apabila diperlukan akan dilakukan pengolahan lanjutan agar didapatkan hasil pengenalan yang lebih baik.

3. Feature Extraction Module

Modul ini bertanggung jawab terhadap penyusunan sebuah feature vector yang terbaik untuk merepresentasikan citra wajah.

4. Classification Module

Fitur terekstraksi dari sebuah wajah dapat dibandingkan dengan yang lainnya yang tersimpan di perpustakaan wajah (databasewajah) dengan bantuan pattern classifier.

5. Training set

Modul ekstraksi fitur dan modul klasifikasi menyesuaikan parameter mereka agar dapat mencapai performa pengenalan optimum dengan penggunaan training set.

6. Face library or face database

Modul ini menyimpan fitur vektor dari citra wajah di himpunan data pelatihan (training set).

BAB 3

IMPLEMENTASI

3.1. Kakas/ Tech Stack

Pada program ini kami menggunakan bahasa python. Untuk frontend program, kami menggunakan library tkinter, customtkinter dan pillow untuk bagian GUI pada program. Alasan dalam penggunaan tkinter adalah mudah digunakan dan banyak dokumentasi yang dapat membantu pembuatan program, library cutomtkinter digunakan karena dengan library ini bisa memberikan tampilan yang lebih baik karena mendukung highDPI pada windows dan Mac dan bisa menggunakan fungsi dari tkinter biasa sehingga sangat mudah untuk mendapatkan pengalaman pengguna yang baik. Pillow digunakan untuk menampilkan citra pada GUI program. Untuk backend program kami menggunakan bantuan library openCV, Numpy dan Natsort. Library openCV digunakan untuk mengolah citra masukan dan library numpy untuk melakukan operasi matriks sederhana untuk memudahkan pengembangan algoritma pengenalan wajah dengan eigenface.

3.2. Algoritma

3.2.1. averageface.py

File ini memiliki fungsi bernama getAvgFace yang berfungsi untuk mencari nilai tengah atau mean dari gambar-gambar pada dataset.

3.2.2. getcovariant.py

File ini memiliki fungsi bernama getCovariant yang berfungsi untuk menghitung nilai matriks covariant.

3.2.3. main.py

File ini terdiri atas beberapa fungsi yang memiliki tujuan utama untuk memprediksi gambar yang tersedia di folder dataset berdasarkan masukan gambar dari pengguna. Beberapa file tersebut antara lain:

1. qr : Fungsi ini digunakan untuk mencari matriks Q dan R yang merupakan pemfaktoran dari matriks gambar masukan yang akan digunakan untuk mencari eigen value dan eigen vector.
2. make_householder : Fungsi ini digunakan untuk mencari matriks householder yang akan digunakan untuk mencari matriks QR di fungsi qr.
3. eigenValVec : Fungsi ini bertugas untuk mencari eigen value dan eigen vector dengan memanfaatkan matriks QR yang didapat dari fungsi qr.
4. sortEigenVectors : Fungsi ini digunakan untuk menyortir eigen value yang vector berdasarkan urutan eigen value yang mengecil.
5. getAdjustedVector : Fungsi ini mencari edjusted vector dari eigen vector yang telah didapat, fungsi ini akan mengubah dimensi dari matriks eigen vector menjadi 65536×1 .
6. displayEigenFaces : Fungsi ini digunakan untuk menampilkan hasil eigen face dalam bentuk gambar.
7. addKZeros : Fungsi ini menambahkan 0 pada array dengan jumlah tertentu berdasarkan masukan pengguna
8. getLinearCombination : Fungsi ini mencari kombinasi linier dari matriks average dan eigen vector dan diambil sejumlah k terbaik nilai yang paling merepresentasikan.
9. getClosest : Fungsi ini memberi keluaran indeks di dataset dari gambar yang memiliki *absolute difference* paling rendah dengan gambar masukan dari pengguna.
10. predictImageIndex : Fungsi ini memiliki keluaran berupa minValue, endTime, resultPath. minValue adalah ..., endTime merupakan waktu berakhirnya proses menghitung eigen face dari suatu gambar, dan resultPath adalah path berbentuk string yang menunjukkan letak gambar prediksi di folder dataset.

11. `getImagePath` : Fungsi ini memberi keluaran path dari gambar hasil prediksi.

3.2.4. app.py

File terdiri dari atas class bernama App dengan attribute dan method sebagai berikut :

1. Attribute:
 - a. `res` : citra hasil prediksi dengan menggunakan metode eigenface
 - b. `img` : citra yang akan menjadi masukan fungsi prediksi
 - c. `imgTk` : citra yang akan ditampilkan sebagai citra keluaran
 - d. `imgtk` : citra yang akan ditampilkan sebagai citra masukan
 - e. `cam` : nomor camera yang akan digunakan
 - f. `cap` : frame/citra dari camera yang akan digunakan sebagai masukan program
 - g. `status_cam` : true jika kamera sedang menyala dan false jika kamera menyala
 - h. `dir` : directory dataset yang dipilih
 - i. `result` : selisih mutlak dari kombinasi linear test image dengan gambar yang diprediksi
 - j. `count_time` : runtime program hingga citra hasil prediksi muncul
 - k. `frame_left` : base frame untuk setting button
 - l. `label_1` : label judul aplikasi
 - m. `title_insert_img` : label “Insert Image”
 - n. `button_insert_img` : button “choose image”
 - o. `title_insert_fld` : label “Insert Folder”
 - p. `button_insert_fld` : button “choose folder”
 - q. `title_oncam` : label “camera”
 - r. `button oncam` : switch on/of camera
 - s. `label_mode` : option untuk tema aplikasi
 - t. `optionmenu_1` : option untuk nomor kamera yang akan digunakan

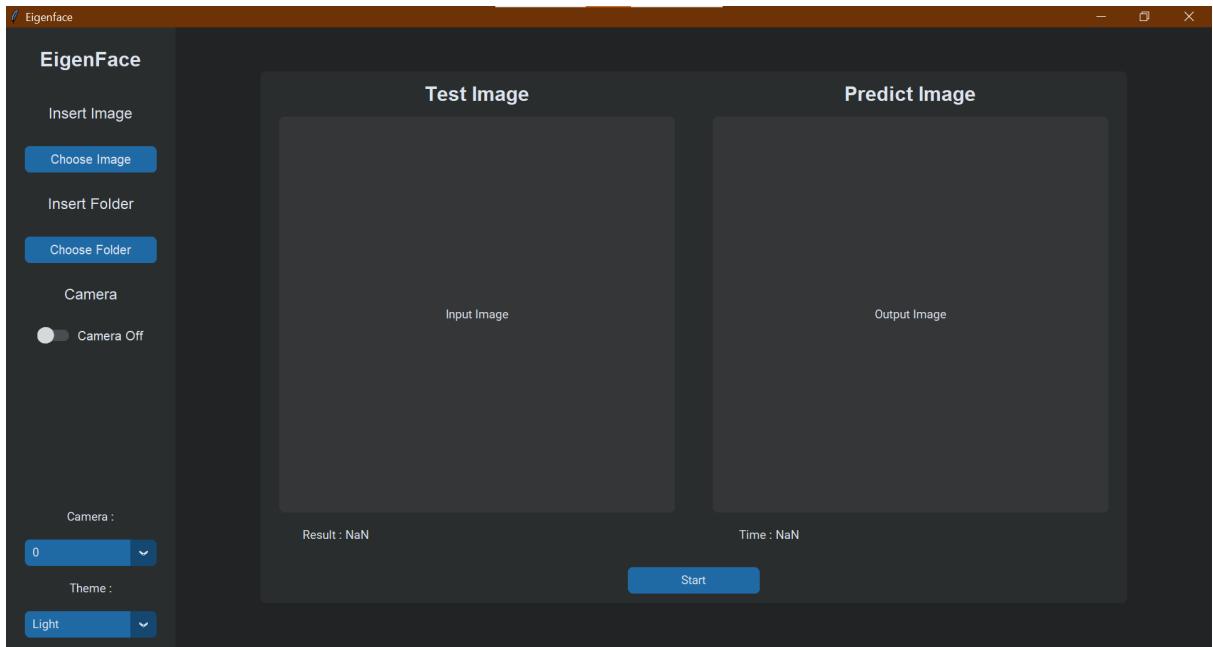
- u. frame_right : base frame untuk tampilan gambar
 - v. frame_footer1 : label “result”
 - w. frame_footer2 : label “runtime”
 - x. frame_footer3 : button “start”/ mulai program
 - y. title_R1 : label “Test Image”
 - z. title_R2 : label “Predict Image”
- aa.subframeR_1 : base frame untuk tampilan citra masukan
bb. subframeR_2 : base frame untuk tampilan citra keluaran
cc.image_input : tampilan citra masukan
dd. image_output : tampilan citra keluaran

2. Method :

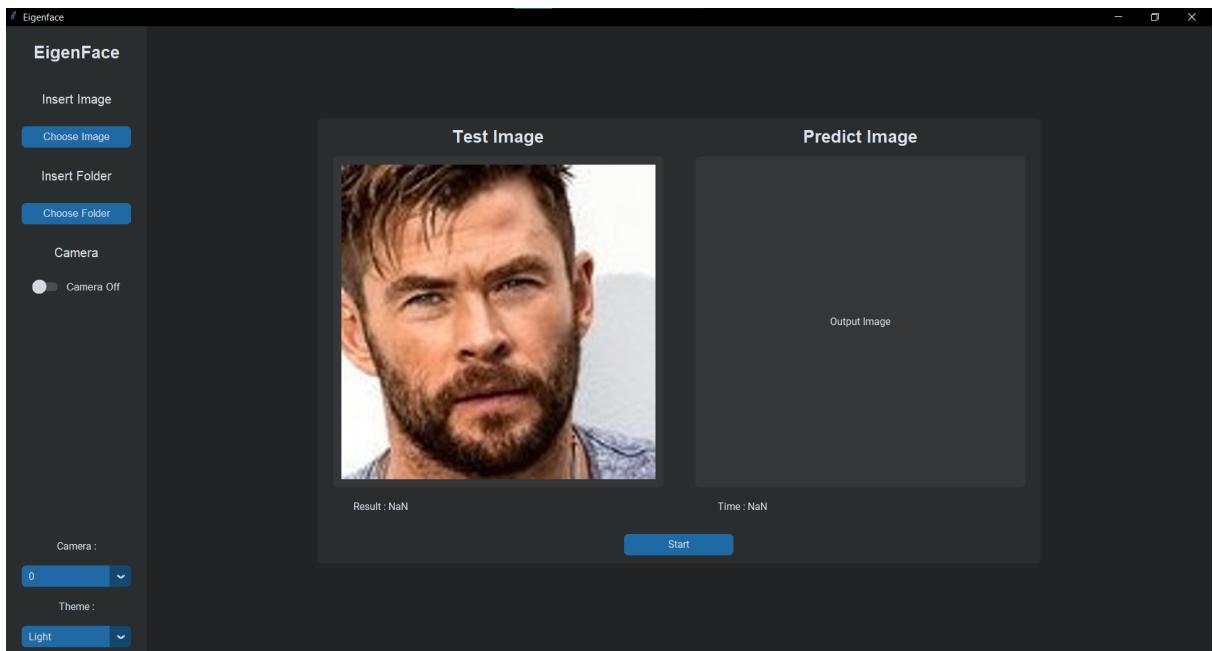
- a. on_cam : prosedur yang akan mengaktifkan dan menampilkan kamera sebagai masukan fungsi
- b. open_image : prosedur yang akan membuka file dialog dan menerima masukan pengguna berupa directory file citra masukan
- c. open_folder : prosedur yang akan membuka file dialog dan menerima directory folder dataset
- d. change_cam : prosedur yang mengubah pilihan camera pengguna
- e. change_appearance_mode : prosedur yang mengubah tema warna aplikasi
- f. start : memulai program face recognition dengan metode eigen face
- g. on_closing : prosedur ketika program ditutup

3.2.5. Garis Besar Program

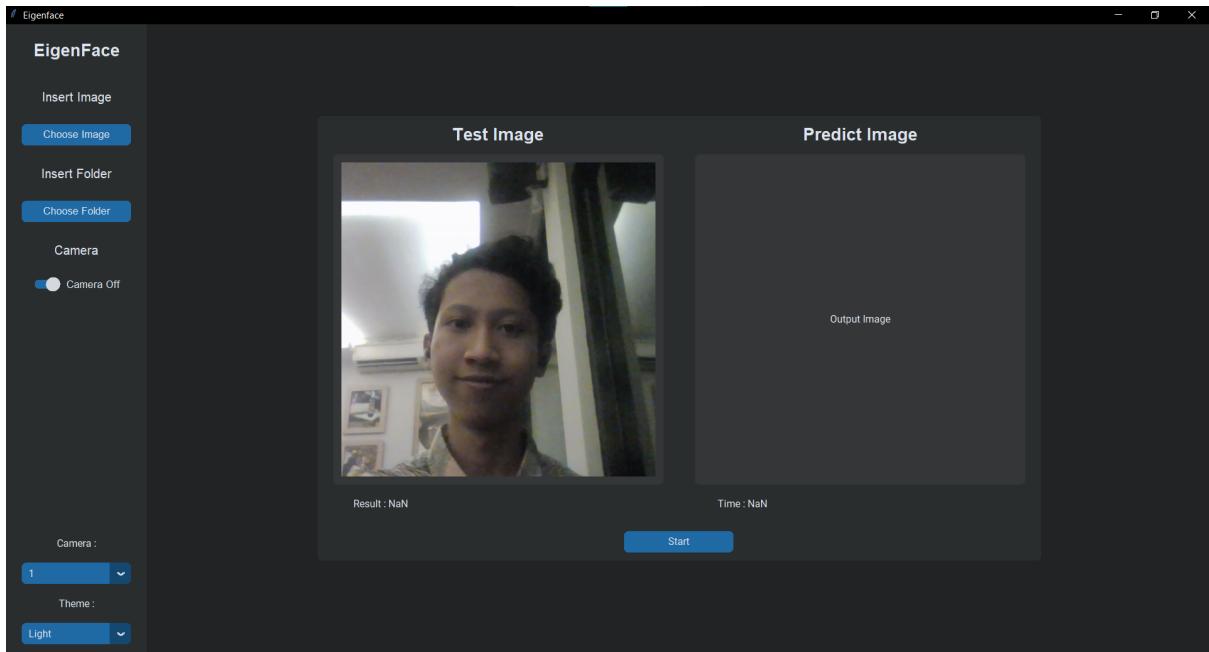
Program dijalankan dengan mengeksekusi command python ./src/app.py di dalam direktori Algeo02-21109. Setelah menunggu beberapa saat, akan muncul pop up dari GUI program.



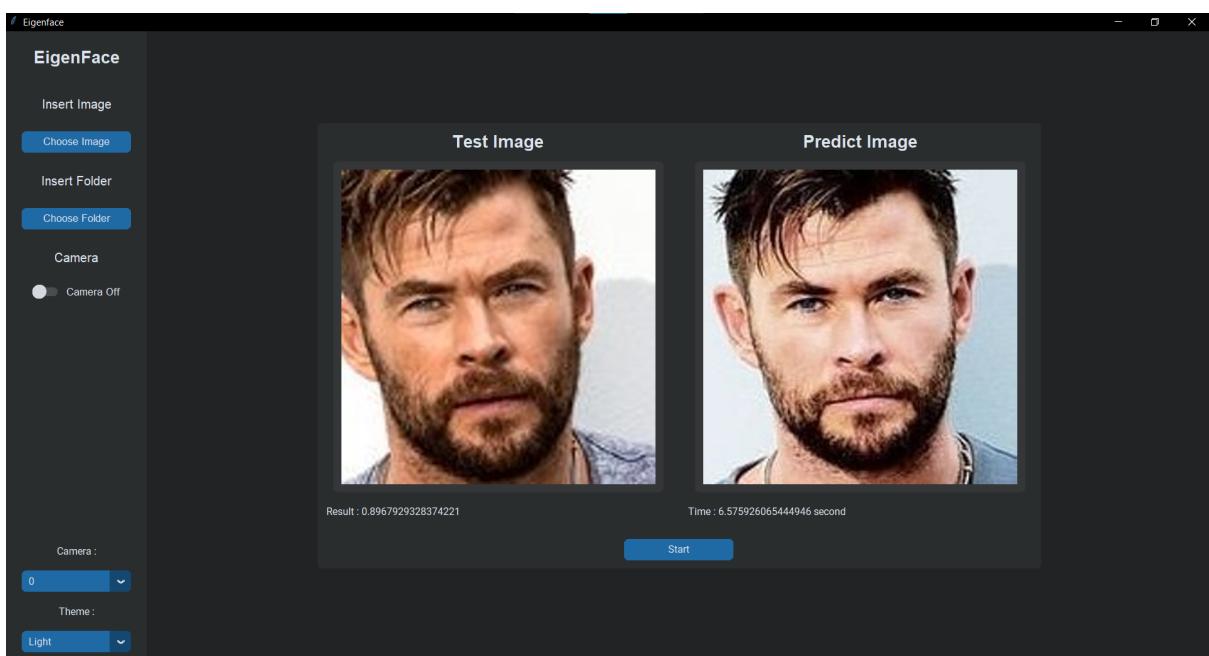
Setelah itu, pengguna bisa memilih folder dataset dengan mengklik tombol di bawah input folder. Setelah itu, pengguna bisa memasukkan test image dengan mengklik tombol choose image di bawah insert image.



Selain menggunakan masukan gambar dari direktori lokal komputer, pengguna juga bisa memasukkan gambar melalui kamera dengan menekan *switch* yang berada di samping tulisan *Camera On*.



Tekan tombol Start untuk memulai pencarian gambar prediksi di folder dataset.
Tunggu beberapa saat hingga muncul gambar prediksi.



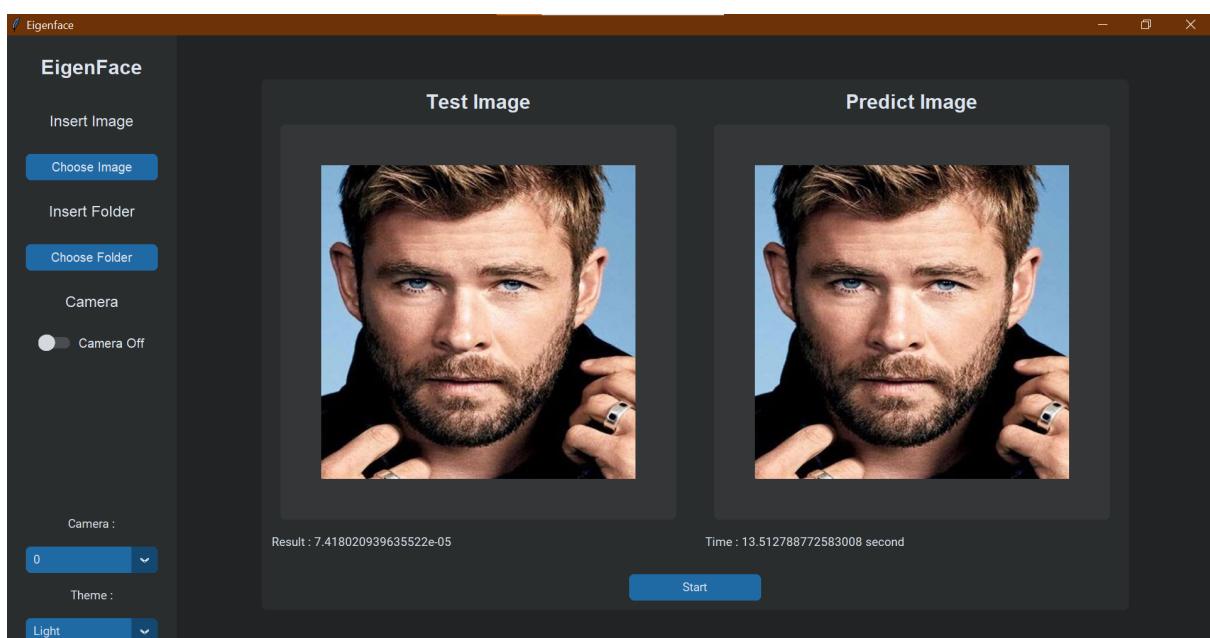
BAB 4

EKSPERIMEN

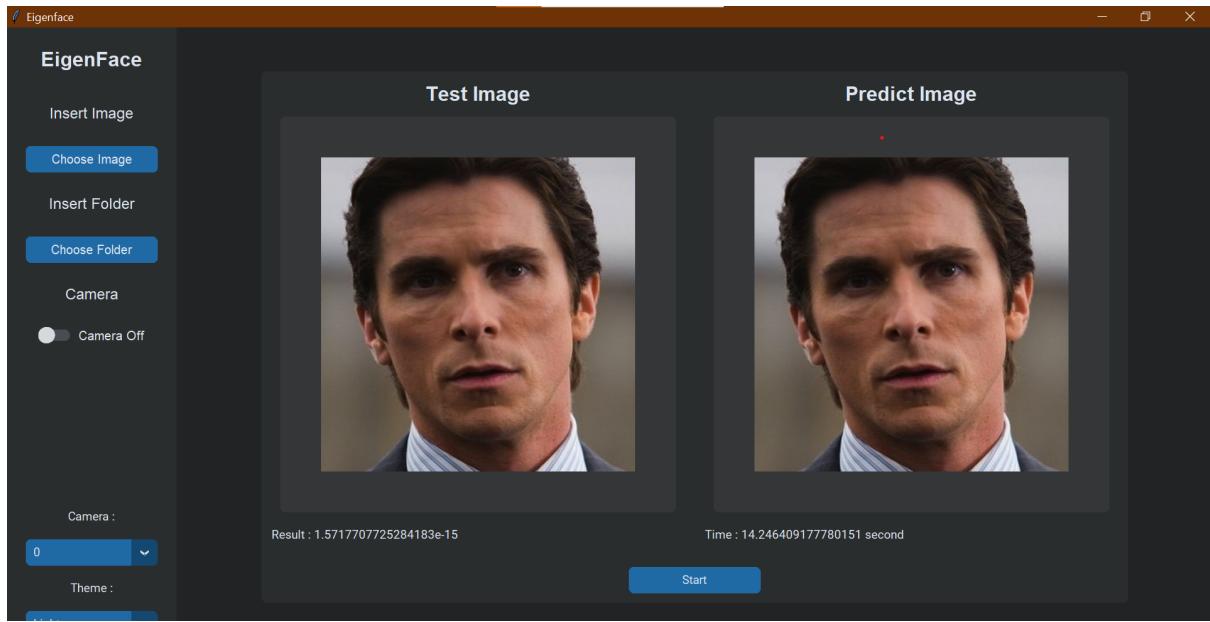
4.1. Eksperimen 1

Pada percobaan ini, kami menggunakan dataset yang terdiri atas foto dari 5 individu berbeda. Masing-masing individu memiliki 10 gambar dirinya. Gambar-gambar pada dataset ini diambil dari: <https://www.kaggle.com/datasets/herveisburak/pins-face-recognition>.

4.1.1. Gambar Masukan Terdapat di Folder Dataset

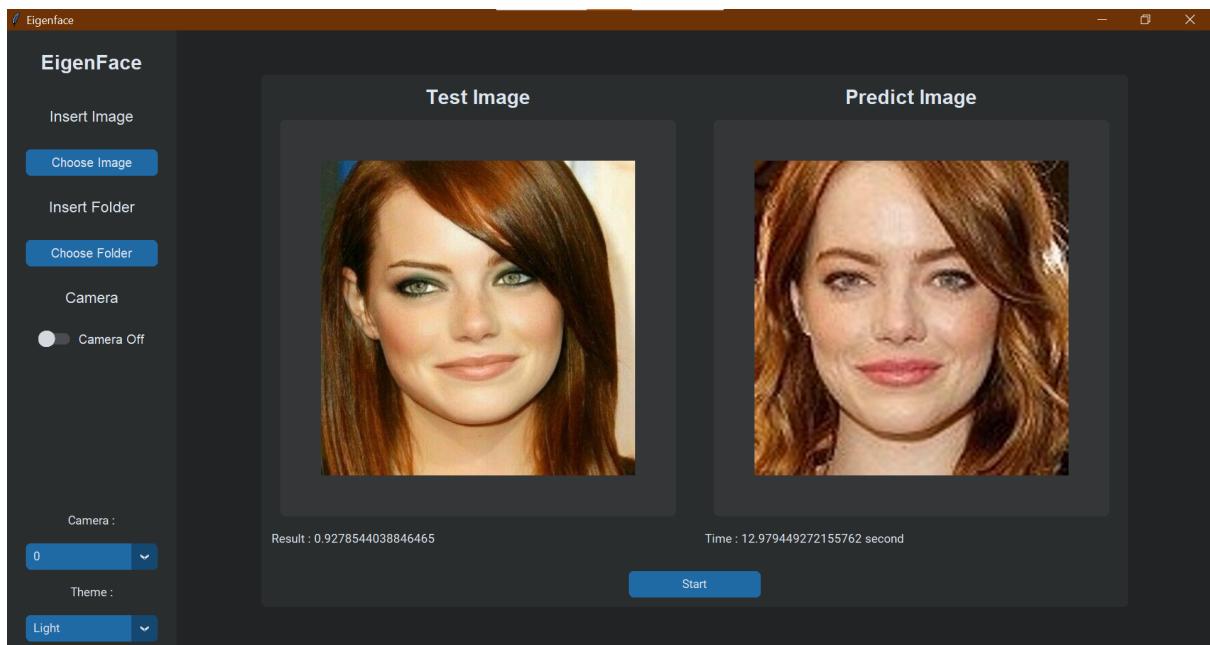


Ukuran masukan gambar: 564x668.

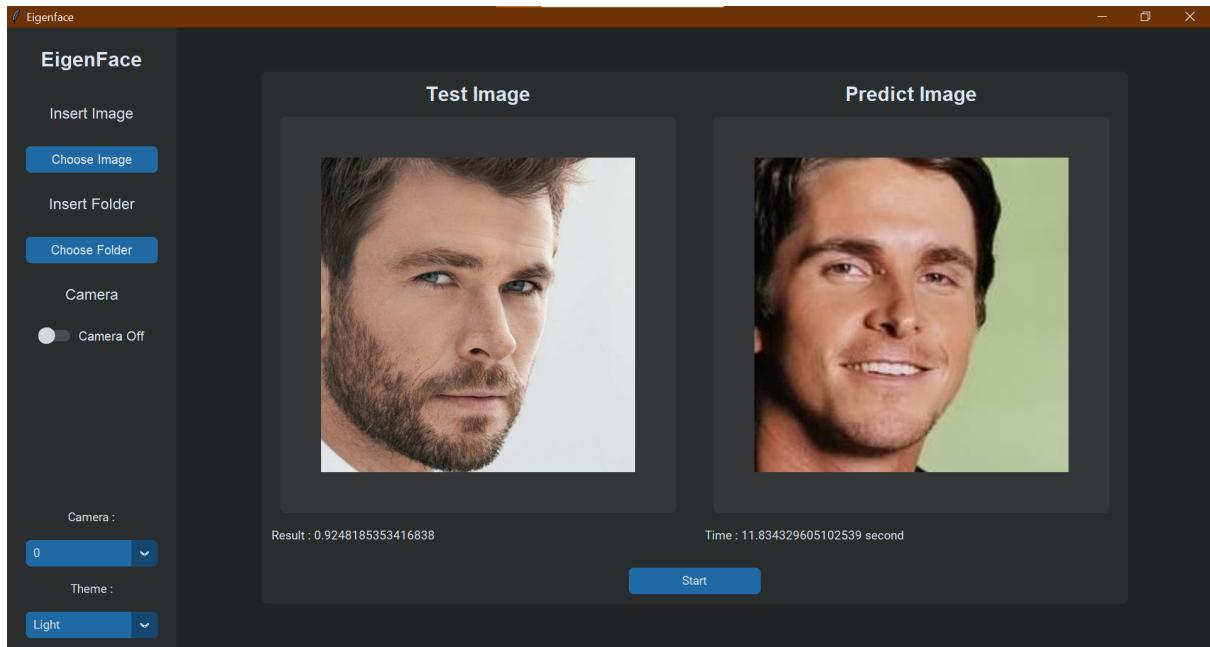


Ukuran masukan gambar: 365 x 387.

4.1.2. Gambar Masukan Tidak Terdapat di Dataset



Ukuran masukan gambar: 436x463.

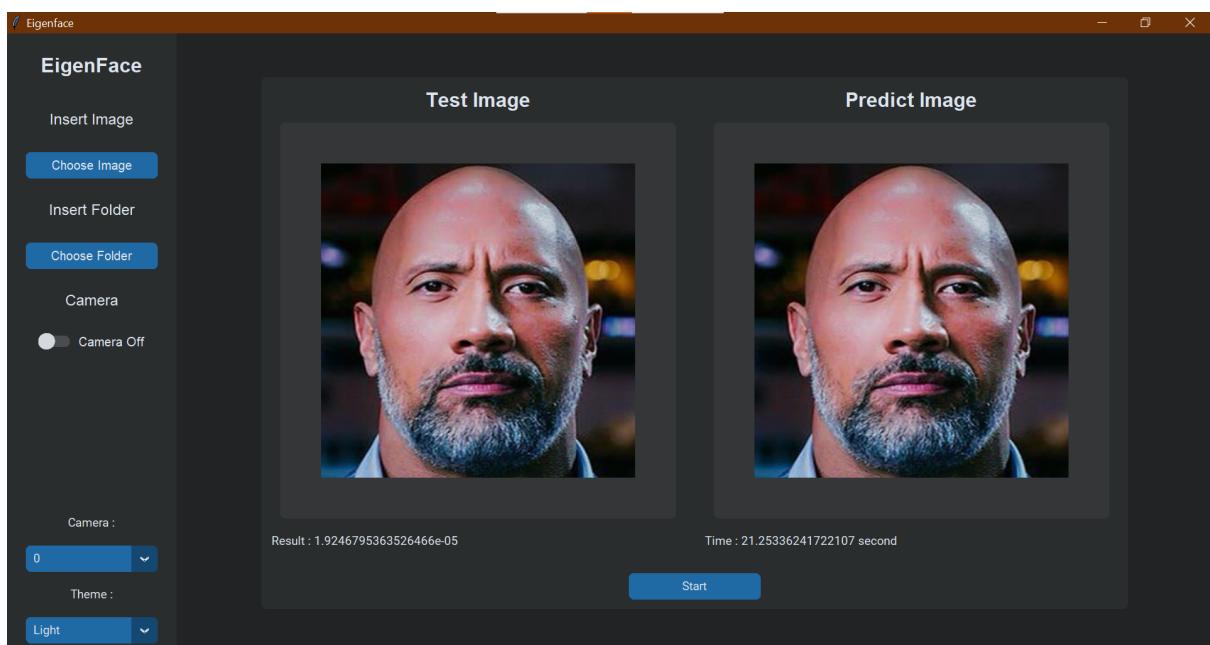


Ukuran masukan gambar: 437x462.

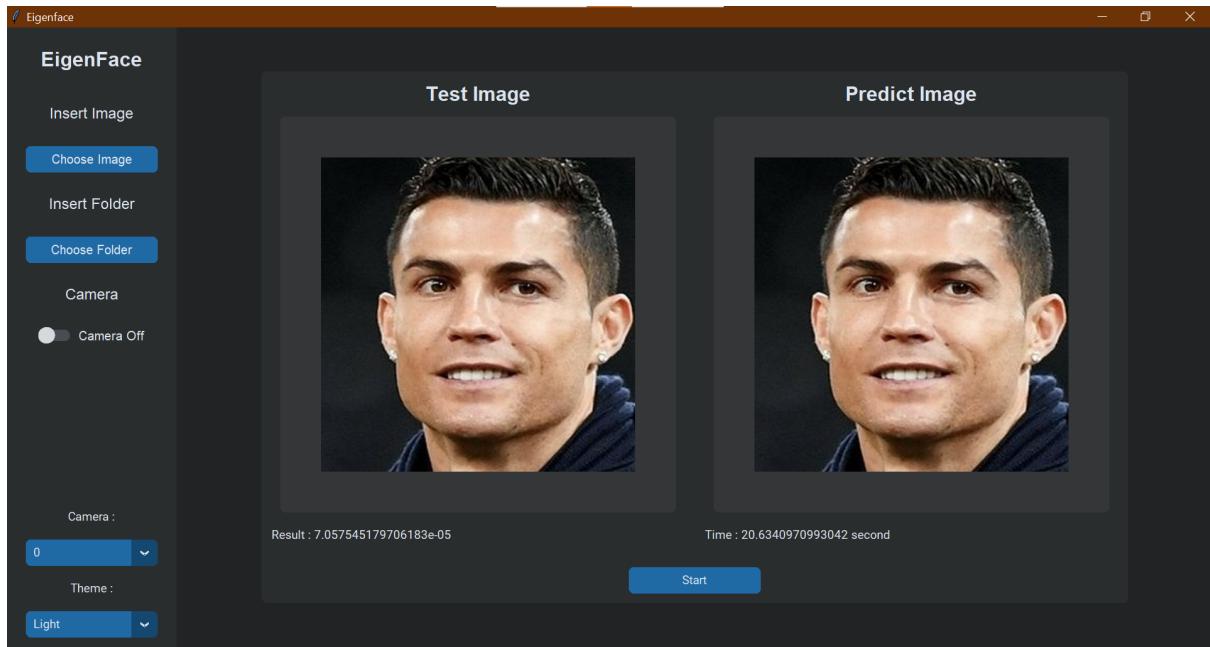
4.2. Eksperimen 2

Pada percobaan ini, kami menggunakan dataset yang terdiri atas foto dari 8 individu berbeda. Masing-masing individu memiliki 10 gambar dirinya. Gambar-gambar pada dataset ini diambil dari: <https://www.kaggle.com/datasets/herveisburak/pins-face-recognition>.

4.2.1. Gambar masukan terdapat di Folder Dataset

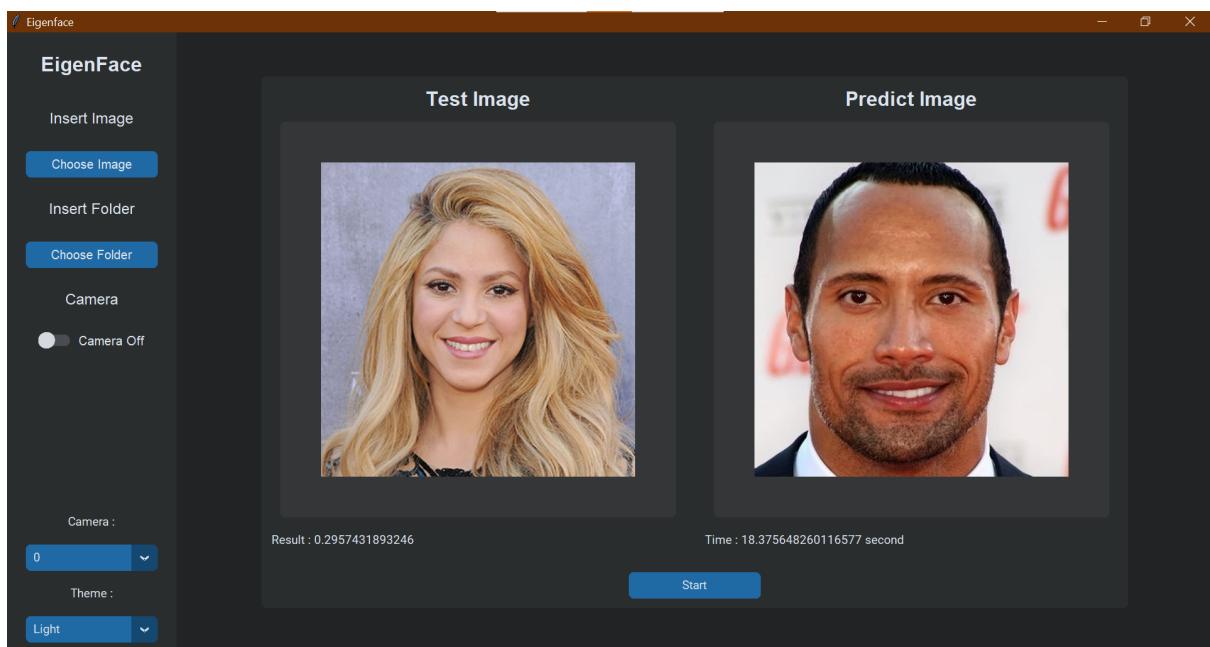


Ukuran masukan gambar: 510x554.

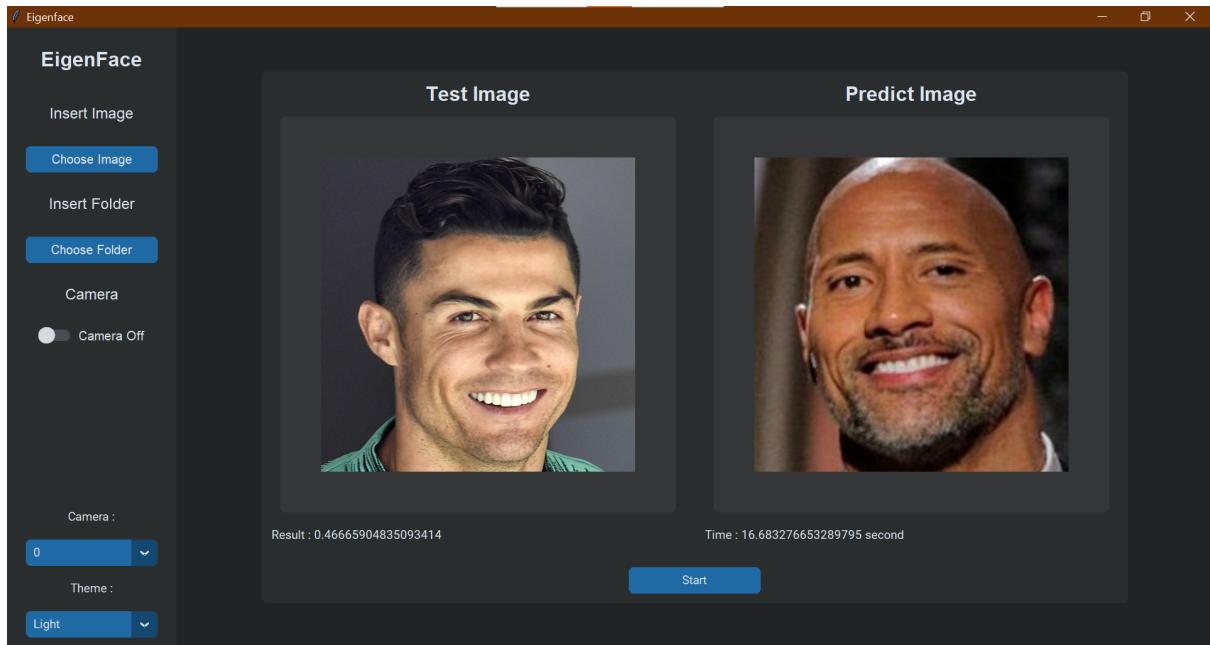


Ukuran masukan gambar: 302x320.

4.2.2. Gambar masukan tidak terdapat di Folder Dataset



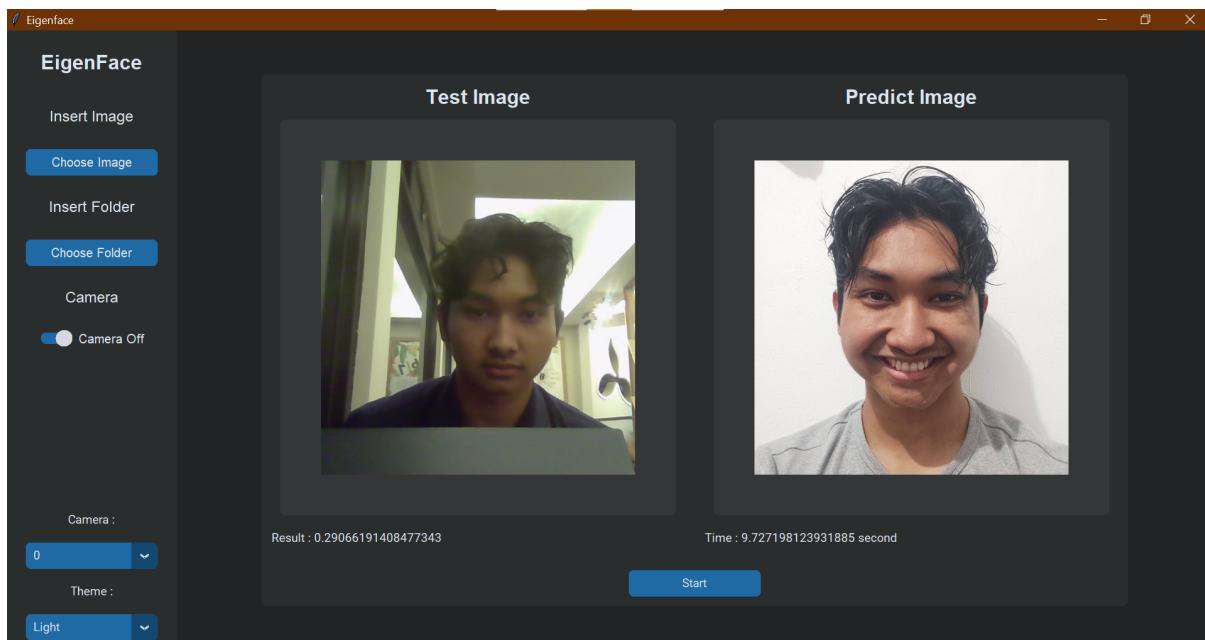
Ukuran masukan gambar: 891x891.

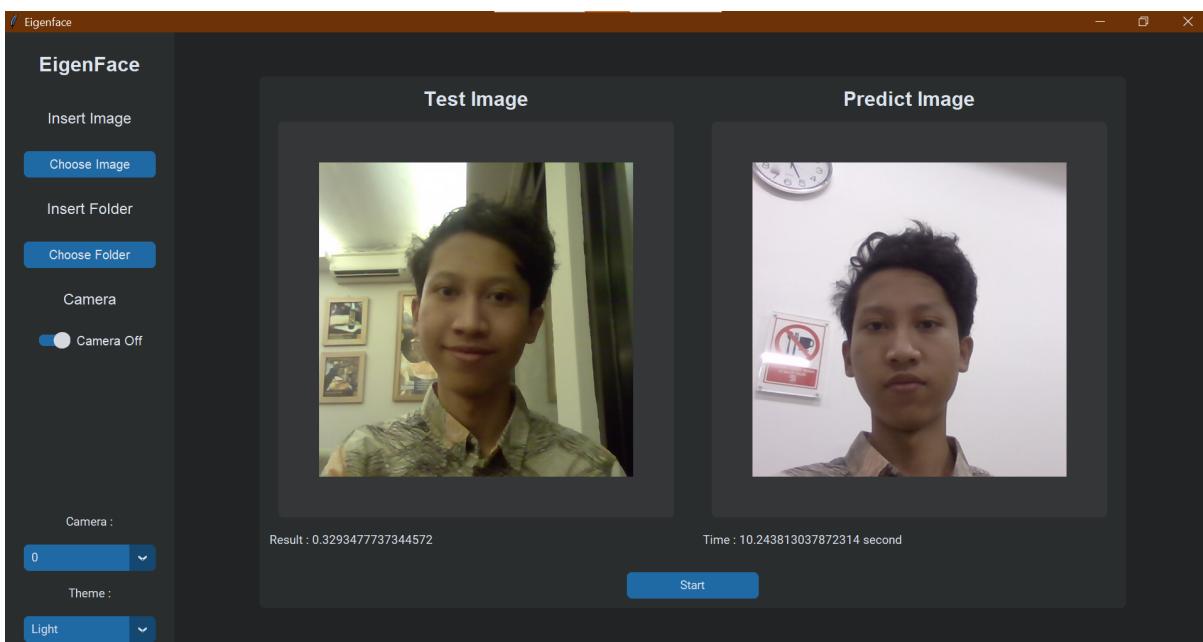
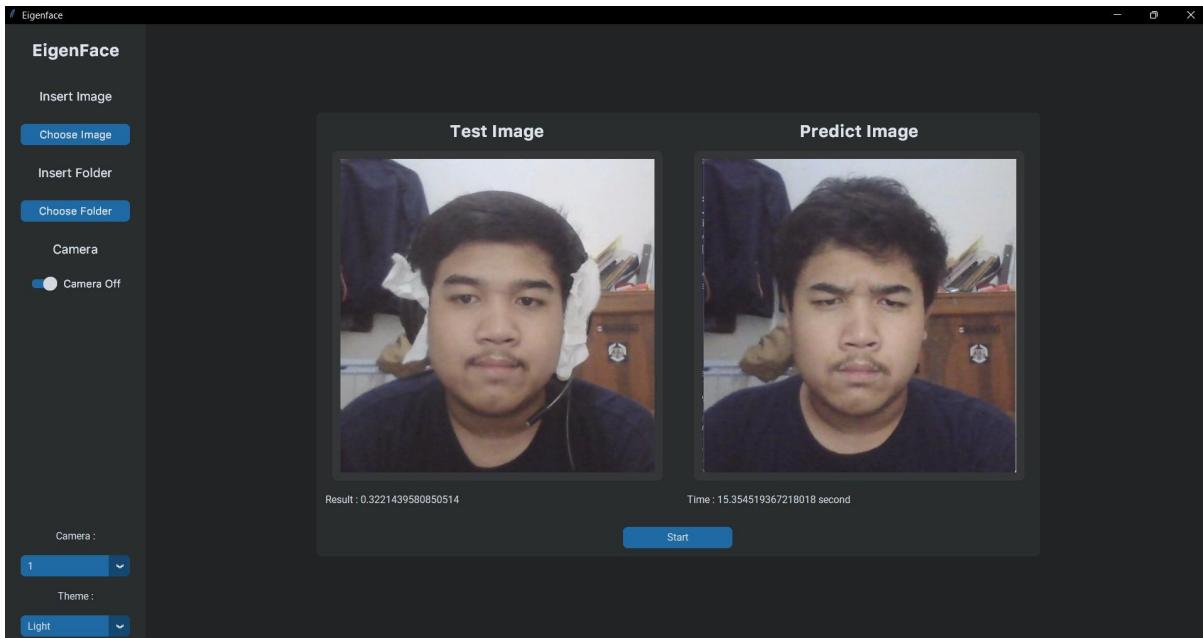


Ukuran masukan gambar: 1152x1220.

4.3. Gambar Masukan dari Kamera

Pada percobaan ini, kami menggunakan dataset yang terdiri atas foto kami. Masing-masing terdapat 10 gambar. Gambar-gambar pada dataset ini diambil dari kamera.





4.4. Analisis Hasil Eksperimen

Berikut merupakan hasil tabularisasi dari beberapa percobaan yang telah kami lakukan dengan program kami.

Jumlah individu di dataset	Jumlah Gambar	Ke-ada-an gambar di dataset	Waktu (detik)	Results	Ketepatan

5	50	ada	13.51	7.42e-5	Tepat
5	50	ada	14.25	1.57e-15	Tepat
5	50	tidak ada	12.98	0.92	Tepat
5	50	tidak ada	11.83	0.92	Tidak tepat
8	80	ada	21.25	1.93e-5	Tepat
8	80	ada	20.63	7.06e-5	Tepat
8	80	tidak ada	18.38	0.30	Tidak tepat
8	80	tidak ada	12.68	0.47	Tidak tepat
3	30	tidak ada	9.73	0.29	Tepat
3	30	tidak ada	15.35	0.32	Tepat
3	30	tidak ada	10.24	0.33	Tepat

Berdasarkan percobaan-percobaan di atas, hasil prediksi gambar yang didapat dari program face recognition yang kami buat berdasarkan algoritma eigen face, menunjukkan hasil yang cukup baik apabila dataset yang dimiliki “bersih”.

BAB 5

KESIMPULAN, SARAN, DAN REFLEKSI

5.1. KESIMPULAN

Hasil program yang kami buat dapat digunakan untuk :

5.2. SARAN

Penulis menyadari bahwa dalam pengerjaan dan pembuatan program dalam tugas besar ini dapat dikembangkan lebih baik lagi. Hal-hal yang dapat dikembangkan menjadi lebih baik lagi adalah sebagai berikut:

1. Dalam pembuatan suatu spesifikasi program, lebih baik untuk melakukan dekomposisi persoalan yang mendalam dan menyeluruh sebelum memulai membuatnya dalam bentuk *code*. Hal ini dilakukan supaya program yang dibuat menjadi lebih mudah dalam proses *debugging* dan pembacaan serta meningkatkan *reusability* dari prosedur atau fungsi yang dibuat.
2. Setiap fungsi atau prosedur lebih baik dituliskan beserta dengan komen singkat untuk menjelaskan apa kegunaan dari fungsi atau prosedur tersebut. Selain itu, penulisan langkah-langkah singkat apa yang dilakukan fungsi atau prosedur juga sangat diperlukan. Hal tersebut berguna untuk memudahkan dalam hal *debugging* dan juga pembacaan.
3. Menggunakan bantuan library tambahan seperti numba, CUDA atau OpenCL yang dapat mengoptimisasi dan mempercepat program dengan mengalokasikan proses operasi matriks yang besar secara paralel sehingga mendapatkan waktu yang lebih sedikit dalam menjalankan program.

5.3. REFLEKSI

Penulis menemukan berbagai pelajaran berharga dalam pembuatan tugas besar ini. Dari sisi nonteknis penulis belajar untuk bekerja sama dalam menyelesaikan pekerjaan sehingga pekerjaan tersebut dapat selesai dengan cepat. Pembagian tugas masing-masing individu juga penting supaya jelas siapa yang menegrejakan suatu bagian dan juga dapat

mempertanggungjawabkan bagian tersebut. Selain itu, penulis juga menyadari pentingnya untuk tidak menunda-nunda suatu pekerjaan. Dari sisi teknis, penulis mendapat pelajaran berharga yaitu dapat memahami dan menggunakan bahasa pemrograman baru yaitu java. Pelajaran mengenai *control-flow* dari pembuatan suatu program juga didapat dalam menggunakan git dan github.

REFERENSI

Algeo-01-Review-Matriks.pdf

Algeo-18-Nilai-Eigen-dan-Vektor-Eigen-Bagian1.pdf

<https://jagostat.com/aljabar-linear/nilai-eigen-dan-vektor-eigen> (diakses pada 19 November 2022)

https://id.wikipedia.org/wiki/Perkalian_matriks#cite_note-5 (diakses pada 19 November 2022)

<https://jagostat.com/aljabar-linear/nilai-eigen-dan-vektor-eigen> (diakses pada 19 November 2022)

<https://www.neliti.com/publications/135138/pengenalan-wajah-menggunakan-algoritma-eigenface-dan-euclidean-distance> (diakses pada 19 November 2022)

https://en.wikipedia.org/wiki/Householder_transformation (diakses pada 22 November 2022)

LAMPIRAN

Github : <https://github.com/rizkyrsyd28/Algeo02-21109>

YouTube : <https://youtu.be/tfyLhPN1A4U>