

# PROJEK ALJABAR LINIER

## “Aplikasi Nilai Eigen dan EigenFace pada Pengenalan Wajah”



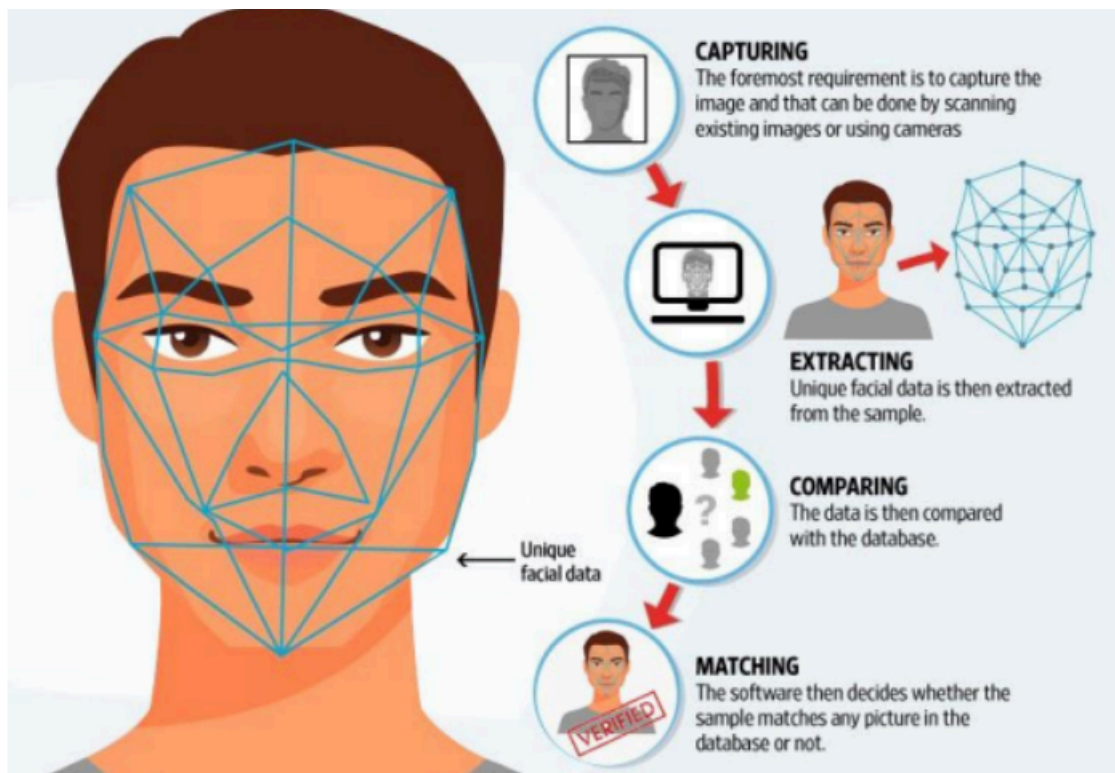
Di susun oleh:

- 1, Intan Trinanda (L0124018)
2. Izanahda Nurkhasna (L0124019)
3. Waldani Nabila Tamamah (L0124122)

FAKULTAS TEKNOLOGI INFORMASI DAN  
SAINS DATA  
2025

## BAB I 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.



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*, prinsip 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.

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.

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

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

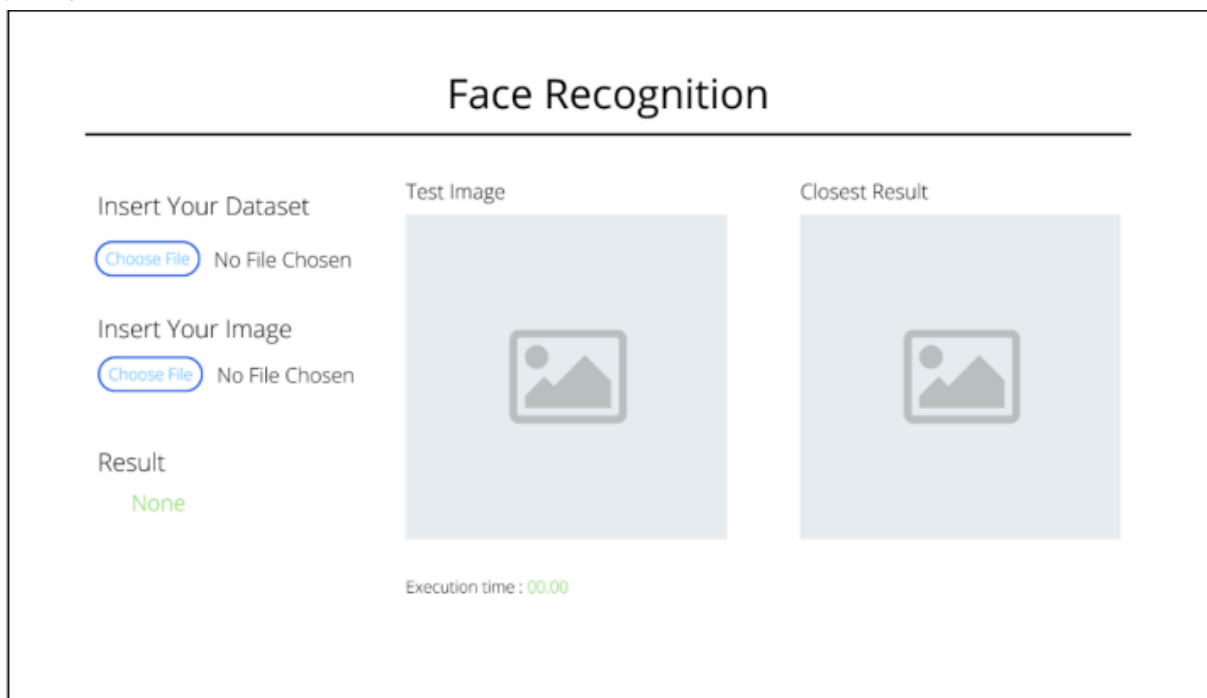
## PENGUNAAN PROGRAM

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 diklik. 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.

## **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/hereisburak/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.

## BAB II TEORI

### 1. Perkalian Matrix

Pada matriks sendiri terdapat 2 jenis perkalian matriks yaitu, perkalian matriks dengan sebuah skalar dan perkalian matriks dengan matriks lainnya. Perkalian matriks dengan skalar dapat dilakukan dengan mengalihkan setiap elemen pada matriks dengan sebuah skalar. sedangkan perkalian matriks dengan matriks lainnya memiliki beberapa syarat agar dapat dilakukan yaitu, jumlah kolom matriks pertama harus sama dengan jumlah baris matriks kedua. contoh perkalian matriks yang memungkinkan adalah matriks dengan ukuran  $m \times n$  dikalikan dengan matriks berukuran  $k \times n$  yang akan menghasilkan matriks berukuran  $m \times n$

a. Misalkan matriks A dan B sebagai berikut:

$$A = \begin{bmatrix} 1 & 2 & 4 \\ 2 & 6 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 4 & 1 & 4 & 3 \\ 0 & -1 & 3 & 1 \\ 2 & 7 & 5 & 2 \end{bmatrix}$$

b. Maka kita misalkan C sebagai matriks hasil perkalian matriks A dan B sehingga matriks C akan menjadi matriks berikut:

$$\begin{bmatrix} 12 & 27 & 30 & 13 \\ 8 & -4 & 26 & 12 \end{bmatrix}$$

### 2. Nilai Eigen (Eigenvalue) dan Vektor Eigen

Kata “eigen” berasal dari bahasa jerman yang artinya “asli” atau “karakteristik” yang berarti nilai eigen adalah sebuah nilai yang menyatakan nilai karakteristik dari sebuah matriks yang berukuran  $n \times n$ . Vektor eigen adalah vektor yang menyatakan sebuah matriks kolom yang apabila dikalikan dengan matriks berukuran  $n \times n$  maka akan menghasilkan vektor lain yang merupakan kelipatan vektor itu sendiri.

Korelasi antara nilai eigen dan vektor eigen dari sebuah matriks dapat diperoleh dari rumus berikut:

$$A\mathbf{x} = \lambda\mathbf{x}$$

A: Matriks ukuran  $n \times n$ ,  $\mathbf{x}$  : vektor eigen,  $\lambda$  : nilai eigen

Ukuran mencari nilai eigen dan vektor dari sebuah matriks berukuran  $n \times n$  adalah dengan menggunakan persamaan berikut:

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

A: Matriks ukuran  $n \times n$ ,  $\mathbf{x}$ : vektor eigen,  $\lambda$ : nilai eigen,  $I$ : matriks identitas

Agar persamaan  $(\lambda I - A)\mathbf{x} = 0$  memiliki solusi tidak nol, maka determinan dari matriks  $(\lambda I - A)$  haruslah 0. Persamaan  $\det(\lambda I - A) = 0$  disebut persamaan karakteristik dari matriks A, dan akar-akar dari persamaan tersebut adalah nilai-nilai eigen dari matriks tersebut.

Untuk mencari vektor eigen dari matriks A, kita dapat mensubstitusikan nilai eigen yang kita dapatkan dari persamaan karakteristik matriks A kepada matriks  $(\lambda I - A)$  kemudian lakukan operasi OBE kepada matriks tersebut dan kemudian mencari solusi dari matriks OBE tersebut.

Namun, karena perhitungan nilai eigen dan vektor eigen menggunakan cara diatas kurang efisien jika dilakukan oleh komputer, maka pada tugas besar ini akan digunakan metode lain dalam mencari nilai eigen dan vektor eigen yaitu QR-Algorithm. Algoritma QR-Algorithm akan melakukan dekomposisi QR secara berulang-ulang untuk mencari nilai eigen dan vektor eigen. Dekomposisi QR adalah sebuah metode untuk membagi matriks menjadi 2 bagian yaitu Q dan R (Q adalah matriks

ortogonal dan R adalah matriks segitiga atas). Setelah dilakukan dekomposisi QR secara berulang akan didapatkan matriks  $R^*Q$  yang berbentuk matriks segitiga atas. Elemen diagonal pada matriks ini adalah nilai-nilai eigen dari matriks. Lalu vektor eigen dari matriks dapat dicari dengan hasil perkalian semua matriks Q. Berikut persamaan dari QR-Algorithm :

$$A_k = Q_k R_k$$

$$A_{k+1} = R_k Q_k$$

$$A_{k+1} = R_k Q_k = Q_k^T Q_k R_k Q_k = Q_k^T A_k Q_k = Q_k^{-1} A_k Q_k$$

Nilai  $k \geq 0$ , R: Matriks segitiga atas, Q: Matriks ortogonal , A: Matriks yang nilai eigennya dicari

### 3. Eigenface

Eigenface adalah metode pengenalan wajah (face recognition) dengan menentukan varian-varian dari wajah di dalam sebuah koleksi gambar-gambar wajah dan menggunakan varian-varian tersebut untuk mengenkripsi dan mendekripsi sebuah wajah dengan cara machine learning tanpa mengurangi komputasi dan kompleksitas ruang. Berikut adalah algoritma dari eigenface :

- Algoritma untuk Training image :
  - a. Apabila di dalam training set kita terdapat m gambar dengan setiap gambar berukuran  $m \times n$ , maka tiap gambar kita ubah terlebih dahulu dimensinya menjadi  $256 \times 256$  dan kemudian ubah tiap gambar menjadi vektor berukuran  $1 \times n^2$ . Kemudian vektor-vektor dari tiap gambar akan digabung semuanya menjadi matriks berukuran  $m \times n^2$ .
  - b. Hitung vektor image rata-rata dari semua vektor image dengan menggunakan persamaan berikut:



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

$\Gamma$ : vektor image

- c. Cari selisih antara vektor image dengan vektor image rata-rata menggunakan persamaan berikut :

$$\phi_i = \Gamma_i - \Psi$$

$\Phi$ : selisih,  $\Gamma$ : vektor image,  $\Psi$ : vektor image rata-rata

- d. Setelah mendapatkan selisih untuk tiap vektor image, gabungkan tiap selisih image menjadi 1 matriks berukuran  $m \times N^2$ . Kemudian cari nilai matriks kovarian dengan menggunakan persamaan berikut:

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

A: matriks selisih

- e. Setelah matriks kovarian didapatkan, akan dicari nilai eigen dan vektor eigen dari matriks kovarian menggunakan QR-Algorithm.
- f. Kalikan matriks image dengan matriks eigen vektor yang didapat dari matriks kovarian. Hasil perkalian matriks image dengan matriks eigen vektor kemudian dikalikan dengan matriks selisih untuk mendapatkan eigenface dari tiap image.
- g. Simpan nilai-nilai eigenface image ke dalam sebuah matriks.

- Algoritma testing :
  - a. Ubah dimensi gambar baru sesuai dengan dimensi gambar-gambar yang ada di dalam training set apabila gambar baru berbeda dan ubah gambar baru menjadi vektor
  - b. Kurangi vektor image baru dengan vektor rata-rata sehingga di peroleh vektor selisih baru

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

$\Gamma$ : vektor image baru,  $\Psi$ : vektor image rata-rata

- c. Cari nilai eigenface gambar baru dengan mengalikan matriks selisih baru dengan matriks proyeksi vektor

$$\mu_{new} = v \times \Gamma_{new} - \Psi$$

$\Gamma$ : vektor image baru,  $\Psi$ : vektor image rata-rata,  
 $v$ : matriks proyeksi vektor

- d. Hitung jarak euclidean gambar baru dengan tiap gambar dalam training set. Gambar dengan jarak euclidean paling kecil dan kurang dari batas yang telah ditentukan adalah gambar yang paling mirip.

### BAB III

#### IMPLEMENTASI PROGRAM

Program ini dibuat dengan menggunakan beberapa library yaitu, numpy (untuk operasi-operasi perhitungan matriks), opencv (untuk mengekstrak fitur-fitur pada gambar), time (untuk mendapat waktu eksekusi), tkinter dan PIL dari pillow (untuk GUI), os (untuk mengakses *directory*). Berikut beberapa fungsi yang terdapat di dalam *source code*.

| Nama           | Parameter                                       | Kembalian                        | Deskripsi   |
|----------------|---|----------------------------------|---|
| imagetoVector  | Path file image                                 | Vektor dari image                | Mengubah image menjadi grayscale dan vektor                     |
| vectortoMatrix | Path folder dataset                             | Matrix training image            | Mengubah folder berisi training image menjadi matriks           |
| mean           | Matrix <i>training image</i>                    | Vektor rata-rata                 | Mencari vektor rata-rata dari semua image                       |
| selisih        | Matrix <i>training image</i> , vektor rata-rata | Matrix selisih image             | Mencari selisih tiap vektor image                               |
| covariance     | Matrix selisih image                            | Matrix covarian                  | Mencari matrix kovarian   |
| QR             | Matrix covarian                                 | Matrix Q dan R                   | Melakukan dekomposisi QR dengan <i>Householder reflection</i>   |
| eigQR          | Matrix covarian                                 | eigen value, matrix eigen vektor | Mencari nilai eigen dan vektor eigen dengan <i>QR-Algorithm</i> |

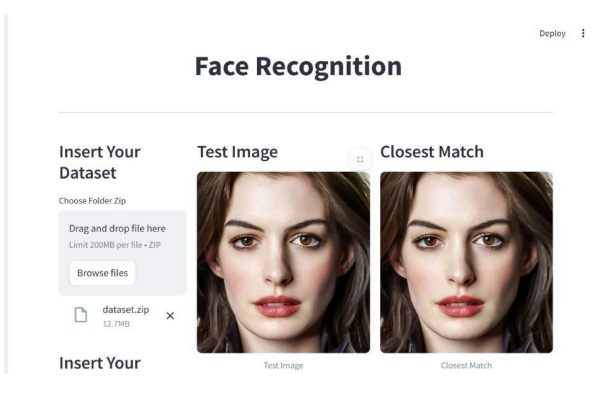
|                        |  |  |  |
|------------------------|--|--|--|
| eig                    | Matrix covarian  | Matrix eigenvector transpos                      | Mencari nilai eigen dan vektor eigen dengan <i>QR-Algorithm</i>  |
| projection             | Matrix training image, matrix eigenvector  | Matrix proyeksi untuk eigen space                | Mencari matrix proyeksi yang nantinya akan membentuk eigen space |
| weightDataset          | weightDataset  | Matrix eigenface                                 | Mencari nilai eigenface dari tiap image                          |
| recogniseUnknownFace   | Path folder dataset, path file image baru, Vektor rata-rata, Vektor image baru, Matrix eigenface | Gambar image paling mirip dan persentase akurasi | Mencari image termirip   |
| uploadDatasetFile      | -  | Directory dataset                                | Mendapatkan path folder dataset                                  |
| uploadTestFile         | -  | Directory test face                              | Mendapatkan path dari test face                                  |
| captureTestFaceWithCam | -  | Gambar test face yang ditangkap dari kamera      | Menyimpan hasil potret test face dari kamera                     |
| closeCam               | -  | State kamera                                     | Menutup kamera dan meyimpan hasil potret test face               |
| startRecognize         | -  | Image hasil dan waktu eksekusi                   | Menampilkan wajah yang   |

|  |  |         |  |
|--|--|---------|--|
|  |  | program | di-recognize<br>dari dataset<br>serta<br>menampilkan<br>lama eksekusi<br>program |
|--|--|---------|--|

## BAB IV EKSPERIMEN DAN ANALISIS

### Eksperimen

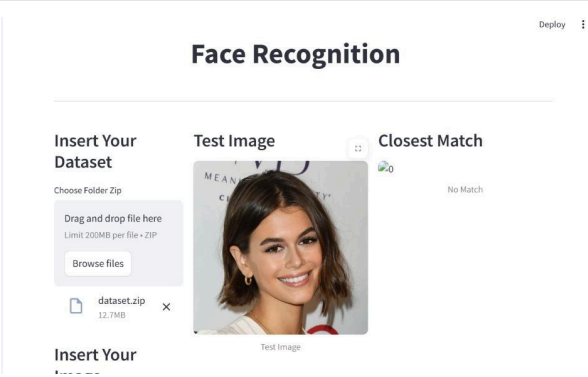
#### A. Apabila Gambar yang dijadikan test image terdapat dalam folder dataset

| Hasil tes  | Penjelasan  |
|--|---|
|  | Pada eksperimen ini, di dalam folder dataset training image kami terdapat foto yang sama persis dengan foto yang menjadi test image, oleh karena itu foto yang keluar juga merupakan foto yang sama dengan foto test image dengan persentase akurasi sebesar 100%. Folder training image yang kami gunakan memiliki 29 image di dalamnya dan membutuhkan waktu sekitar 17 detik untuk eksekusi program. |

#### B. Apabila Gambar yang dijadikan test image tidak ada di dalam folder dataset

| Hasil tes   | Penjelasan   |
|---|--|
|  | Pada eksperimen ini, di dalam folder dataset training image tidak terdapat foto yang sama dengan foto training image, namun terdapat foto orang yang sama dengan orang pada test image. Dapat dilihat program kami akan mengeluarkan foto orang yang sama dengan persentase akurasi 91.89%. Folder training image yang kami gunakan memiliki 28 image di dalamnya dan membutuhkan waktu sekitar 14,9 detik untuk eksekusi program. |

### C. Apabila tidak ada foto yang mirip dengan foto test image

| Hasil tes   | Penjelasan  |
|---|---|
|  | <p>Pada eksperimen kali ini, karena di dalam folder dataset kami tidak ada nilai euclidean distance yang nilainya dibawah threshold yang kami berikan, maka GUI kami tidak akan mengeluarkan foto. Folder training image yang kami gunakan memiliki 29 image di dalamnya dan membutuhkan waktu sekitar 16,5 detik untuk eksekusi program.</p> |

## BAB V

### KESIMPULAN DAN REFLEKSI

#### **Kesimpulan**

Dalam mengenali sebuah wajah (face recognition), terdapat berbagai macam metode yang dapat digunakan dan salah satunya adalah dengan algoritma eigenface yang dapat mengenali sebuah wajah dengan varian-varian dari wajah di dalam sebuah koleksi gambar-gambar wajah dan menggunakan varian-varian tersebut untuk mengenkripsi dan mendeskripsi sebuah wajah

Pada projek ini, kami membuat sebuah program pengenalan wajah yang didasarkan dengan algoritma eigenface. Kami mengimplementasikan program ini dalam bentuk GUI yang di buat frngan menggunakan library tkinter. Program kami berhasil mengenali foto-foto dari orang yang sama dan didukung dengan eksekusi program yang cepat.

#### **Saran**

Saran untuk kelompok kami yaitu:

1. Pembagian tugas yang harusnya dilakukan lebih baik lagi
2. Pemberian komen kepdaa kode program yang lebih jelas agar kode program dapat lebih mudah di debug dan lebih mudah dimengerti

#### **Refleksi**

Dengan selesainya projek pertama ini, kami mendapatkan banyak manfaat dan pengalaman baru. Mulai dari algoritma pengenalan wajah menggunakan opencv python yang sebelumnya belum pernah kami pelajari sama sekali. Selain menambah skil-skil dalam hal pemograman,



kami juga dapat meningkatkan soft skill melalui projek pertama ini seperti kerjasama, time management, problem solving, dan masih banyak lainnya. Namun, kami juga merasa masih harus mengembangkan time management karena pengerjaan tugas yang sangat mepet dengan deadline yang di berikan.

## DAFTAR REFERENSI

1. <https://www.geeksforgeeks.org/ml-face-recognition-using-eigenfaces-pca-algorithm>
2. <https://medium.com/machine-learning-world/feature-extraction-and-similar-image-search-with-opencv-for-newbies-3c59796bf774>
3. <https://jurnal.untan.ac.id/index.php/jcskommipa/article/download/9727/9500>

## LAMPIRAN

GitHub: <https://github.com/Izanahdanur/Projek-Eigenface>

Youtube: <https://youtu.be/Zw5xS2vnkY4?si=XvH3NiC4QhdnfB7I>