LAPORAN TUGAS BESAR 2

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

Laporan Ini Dibuat Untuk Memenuhi Tugas Perkuliahan
Mata Kuliah Aljabar Linier dan Geometri

KELAS 01
Dosen : Dr. Judhi Santoso, M.Sc.

DISUSUN OLEH:

Kelompok 46 (In Him We Trust)

Anggota:

(13521054) Wilson Tansil

(13521065) Mutawally Nawwar

(13521122) Ulung Adi Putra

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

SEMESTER I TAHUN 2021-2022

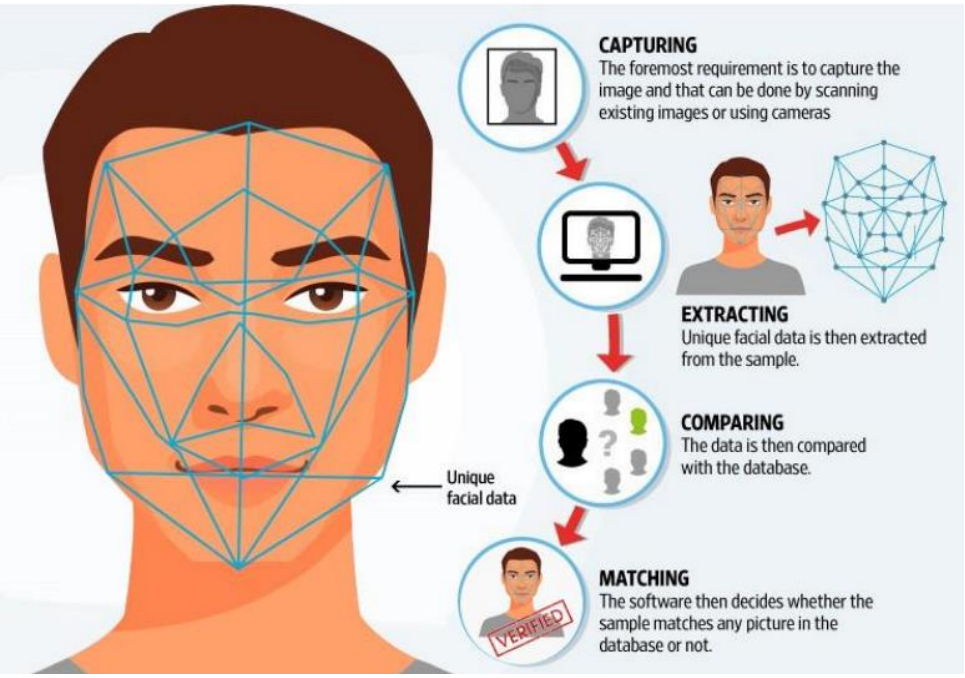
Daftar Isi

Daftar Gambar

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, principal component analysis (PCA), serta Eigenface. Pada Tugas ini, akan dibuat sebuah program pengenalan wajah menggunakan Eigenface. Sekumpulan citra wajah akan digunakan dengan representasi matriks. Dari representasi matriks tersebut akan dihitung sebuah matriks Eigenface. Program pengenalan wajah dapat dibagi menjadi 2 tahap berbeda yaitu tahap training dan pencocokkan. Pada tahap training, akan diberikan kumpulan data set berupa citra wajah. Citra wajah tersebut akan dinormalisasi dari RGB ke Grayscale (matriks), hasil normalisasi akan digunakan dalam perhitungan eigenface. Seperti namanya, matriks eigenface menggunakan eigenvector dalam pembentukannya.

BAB II
Teori Singkat

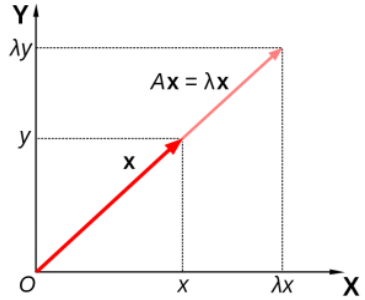
2.1. Nilai Eigen dan Eigen Vektor

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

$$\mathbf{Ax} = \mathbf{lx}$$

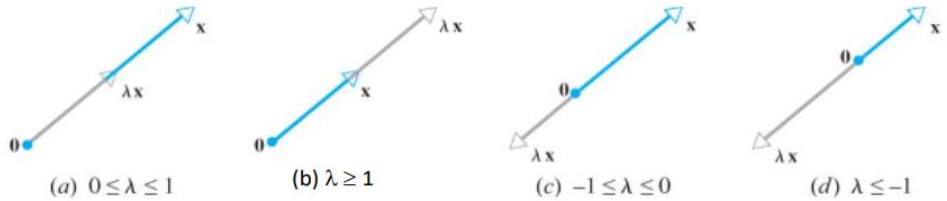
  Skalar l disebut nilai eigen dari A, dan **x** dinamakan vektor eigen yang berkoresponden dengan l.

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



Gambar 2 Ilustrasi vektor eigen

- Dengan kata lain, operasi **Ax** = **lx** menyebabkan vektor **x** menyusut atau memanjang dengan faktor l dengan arah yang sama jika l positif dan arah berkebalikan jika l negatif



Gambar 3 Ilustrasi nilai eigen

2.2. Diagonalisasi

- Matriks diagonal adalah matriks yang semua elemen di atas dan di bawah diagonal utama adalah nol.
- Matriks persegi A dikatakan dapat didiagonalisasi jika ia mirip dengan matriks diagonal.
- Misalkan E adalah matriks yang kolom-kolomnya adalah basis ruang eigen dari matriks A, yaitu:

$$E = (e1 \mid e2 \mid \ldots \mid en)$$

Misalkan D adalah matriks diagonal, maka

$$A = EDE^{-1} \rightarrow \boxed{D = E^{-1}AE}$$

2.3. Algoritma Eigenface

1. Langkah pertama adalah menyiapkan data dengan membuat suatu himpunan S yang terdiri dari seluruh training image, (Γ1, Γ2, …, ΓM)

$$S = (\Gamma 1, \Gamma 2, \ldots, \Gamma M) \quad (1)$$

2. Langkah kedua adalah ambil nilai rata-rata atau mean (Ψ)

$$\Psi = \frac{1}{M}\sum_{n=1}^{M}\Gamma_n \quad (2)$$

3. Langkah ketiga kemudian cari selisih (Φ) antara nilai training image (Γi) dengan nilai tengah (Ψ)

$$\phi i = \Gamma i - \Psi \quad (3)$$

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

$$C = \frac{1}{M}\sum_{n=1}^{M}\phi_n\phi_n^T = AA^T \quad (4)$$

$$L = A^T A \qquad L = \phi_m^T \phi_n$$

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

$$C \text{ x } vi = \lambda i \text{ x } vi \quad (5)$$

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

$$\mu_i = \sum_{k=1}^{M} v_{ik}\phi_l \quad (6)$$

$$l = 1, \ldots, M$$

2.4. Tahapan Pengenalan Wajah

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

$$\mu new = v \text{ x } \Gamma new - \Psi \ (7)$$

$$\Omega = \mu 1 \ , \mu 2 \ , \ldots \ , \mu M$$

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

$$\varepsilon k = \Omega - \Omega k \ (8)$$

BAB III
Implementasi Pustaka dan Program Java

3.1. Penjelasan Teknologi

Teknologi yang digunakan adalah python dengan library OpenCv. OpenCV adalah sebuah library yang ditujukan untuk pengolahan gambar *(image processing)* secara real-time yang dibuat oleh Intel. OpenCV pertama kali dibuat oleh Intel pada tahun 1999 oleh Gary Bradsky dan mulai dirilis keluar pada tahun 2000. Saat ini, OpenCV telah mendukung banyak algoritma yang terkait dengan Computer Vision dan Machine Learning. Selain itu, saat ini OpenCV juga dapat digunakan dalam berbagai macam bahasa pemrograman, seperti C++, Python, Java, dan lain sebagainya. Tidak hanya itu, OpenCV juga tersedia dalam berbagai platform, seperti Windows, Linux, OSX, Android, IOS, dan lain sebagainya.

Pada bagian GUI menggunakan library kivy. Kivy merupakan framework Python untuk membangun aplikasi berbasis NUI dengan ringkas, cepat dan mudah. Berbasis lisensi MIT, framework yang satu ini dapat dijalankan hampir di semua platform seperti Windows, Linux, iOS, Android dan Raspberry.

3.2. Penjelasan Alur Aplikasi dan Algoritma

Susunan folder pada dapat dilihat pada gambar berikut.

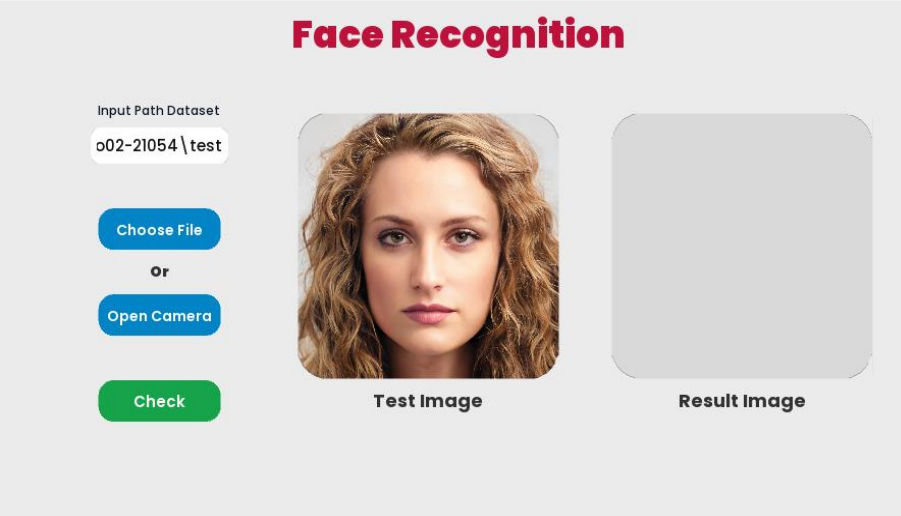
```
Algeo02-21054
|---doc
   | Algeo21054.docx
|---src
   |---fonts
       |   Poppins-Black.ttf
       |   Poppins-Bold.ttf
       |   Poppins-Medium.ttf
       |   Poppins-SemiBold.ttf
   |---assets
       |   background_textbox.p
       |   blank_image.png
       |   border_radius.png
   |   app.py
   |   eigen.py
   |   configdata.py
   |   my.ky
|---test
   |---get_data
   |---folder_dataset
```

Gambar 4 Susunan folder

Aplikasi ini memiliki 2 bagian besar yaitu tampilan aplikasi dan pencarian eigenface serta pencocokan.

3.2.1. Tampilan Aplikasi

Berikut adalah tampilan dari aplikasi.



Gambar 5 Tampilan aplikasi

Tampilan tersebut dibuat dengan menggunakan library kivy. kivy sendiri memiliki sangat banyak widget bawaan yang tinggal dipakai. Hal ini sangat mempermudha dalam pembangunan aplikasi. Untuk mempercantik tampilan, digunakan juga beberapa assets seperti gambar dan font yang bukan merupakan bawaan kivy.

Pada bagian kiri, terdapat input folder dataset yang berupa path. User harus menginputkan path folder dari dataset ke dalam kolom tersebut. Berikutnya user bisa memilih untuk gamber tes untuk menggunakan kamera atau langsung pilih dari lokal. Gambar tes otomatis akan muncul Ketika sudah dipilih. Setelah user menekan tombol check, maka aplikasi akan memproses beberapa detik sampai muncul gambar hasil.

3.2.2. Pencarian Eifenface serta Pencocokan

Langkah pertama untuk melakukan pencocokan gambar adalah dengan menyiapkan dataset gambar yang akan dijadikan training image. Kemudian dataset training image tersebut diconvert terelebih dahulu ukuruannya agar menjadi berukuran 256 x 256. Setekah di ubah ukurannya, gambar tersebut diubah nilainya agar menjadi gambar hitam putih (gray scale). Setelah dataset siap diproses, setiap gambar diubah menjadi vector dengan menggunakan fungsi flatten dari library numpy.

Jumlahkan semua vector training image dan bagi dengan banyaknya vector training image untuk menentukan vector rata rata. Setelah mendapatkan vector rata rata dari training image, kita akan mencari normalize dari setiap training image dengan cara mengurangkan training image dengan nilai rata ratanya.

Setekah mendapatkan data training image yang sudah di normalize, kita akan mencari matrix covarian dari training image tersebut. pertama kita transpose matrix dari vector vector training image yang sudah dinormalized. Kemudian kita kalikan matrix yang sudah di normalized dangen matrix yang sudah ditranspose tadi untuk menghasilkan matrix covarian.

Kemudian akan dicari nilai eigen dan vector eigen dari matrix covarian yang sudah didapat. Kelompok kami menggunakan metode QR decomposition untuk mencari eigen

vector dan nilai eigen. Pertama kita buat matrix identitas berukuran sama seperti matrix covarian dan. Kemudian lakukan QR dikomposisi pada matrix covarian sehingga menghasilkn matrix Q dan R. Setelah itu kalikan matrix R dan Q menjadi matrix A dan kalikan matrix QQ dengan matrix Q. Ulangi proses tersebut hingga nilai diagonal matrix A tidak berubah nilainya. Nilai nilai diagonal matrix A adalah nilai eigen dan matrix QQ adalah list atau kumpulan dari vector eigen.

Setelah vector eigen dan nilai eigen didapatkan, akan dicari eigen face dari setiap training image. Eigen face didapatkan dengan cara mengalikan vector eigen dengan setiap training image yang sudah di normalize (dikurangi dengan rata rata). Kemudian setelah mendapatkan eigen face, kita akan mencari weight setiap image dengan cara mengalikan eigen face dengan data training image yang sudah dikurangkan dengan rata rata.

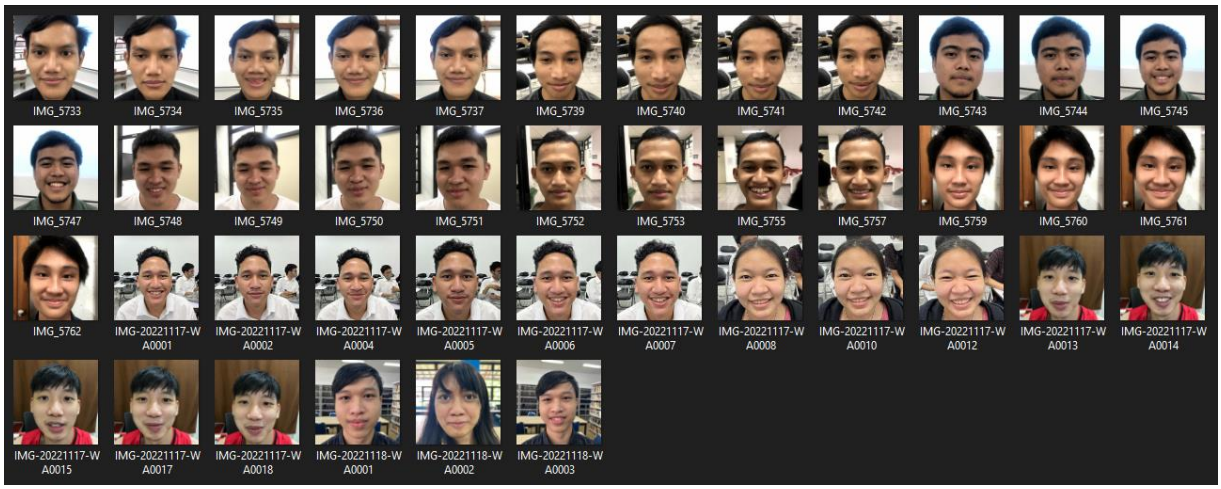
Kemudian kita akan menerima input gambar yang ingin kita cocokan dengan dataset training image. Ulangi cara yang sama untuk diterapkan ke foto yang diinputkan untuk mencari weightnya. Kemudian kita bandingkan nilai weight input dengan weight dari setiap dataset. weight gambar yang memiliki jarak paling sedikit dengan weight gambar input adalah gambar yang paling mirip.

BAB IV

Eksperimen

1. Mahasiswa Informatika

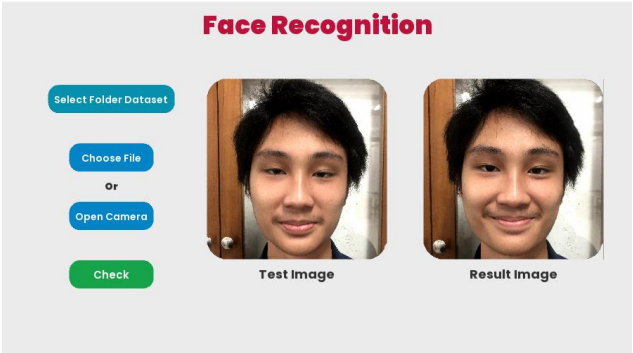
a. Dataset



Gambar 6 Dataset Mahasiswa Informatika

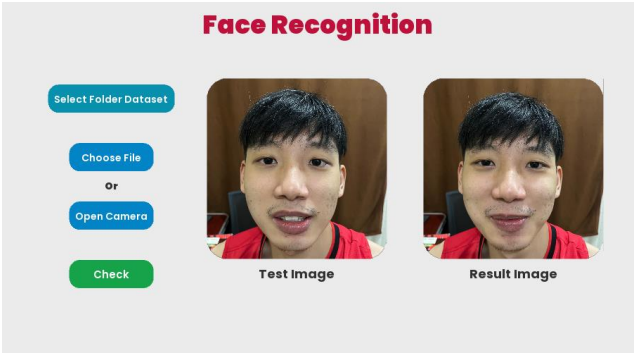
b. Test

i. Jimly



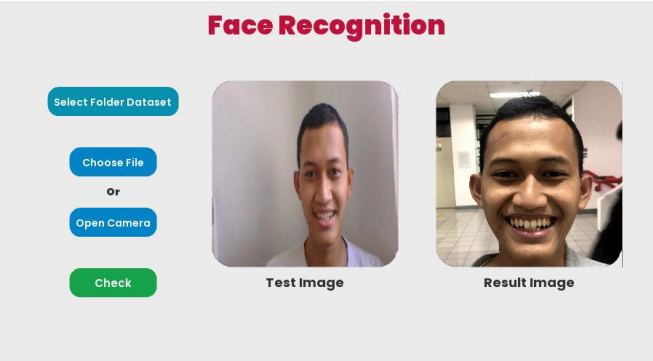
Gambar 7 Tes Jimly

ii. Wilson



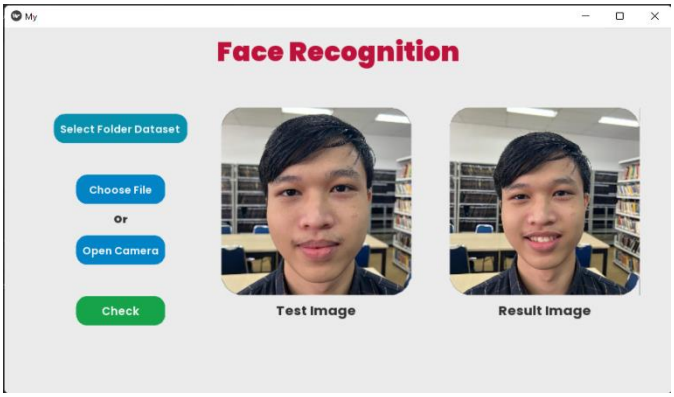
Gambar 8 Tes Wilson

iii. Nawwar



Gambar 9 Tes Nawwar

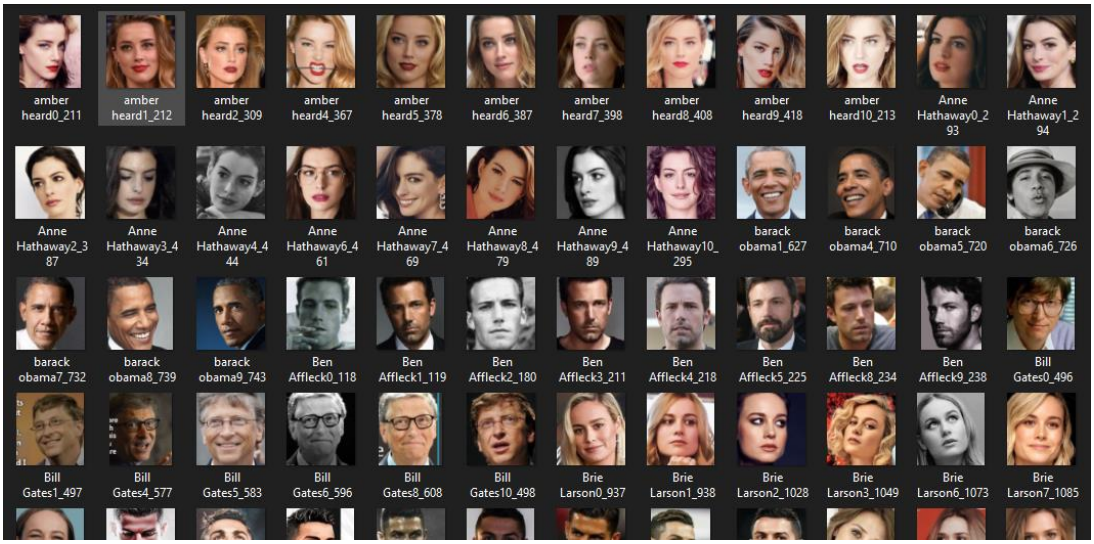
iv. Ulung



Gambar 10 Tes Ulung

2. Aktor/Aktris

a. Dataset



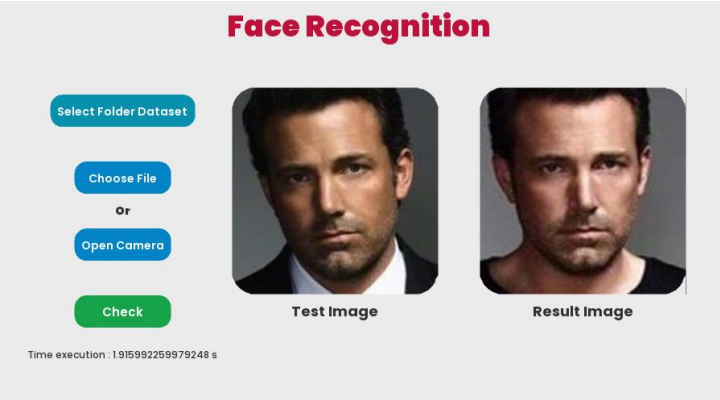
Gambar 11 Dataset Aktor/Aktris

b. Test

I. Bill Gates

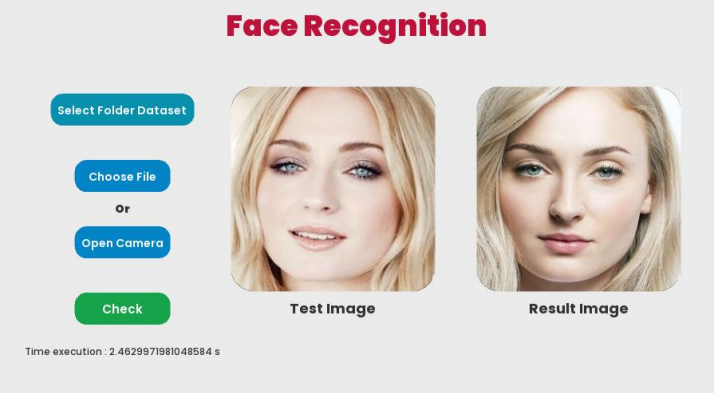
Gambar 12 Tes Bill Gates

II. Ben Affleck



Gambar 13 Tes Ben Affleck

III. Shopie Turner



Gambar 14 Tes Shopie Turner

BAB V
Kesimpulan, Saran, dan Refleksi

5.1. Kesimpulan

1. Kami berhasil membuat aplikasi dengan menerapkan materi eigen vektor dan eigen value.
2. Kami berhasil membuat aplikasi dengan menerapkan materi diagonalisasi.
3. Kami berhasil membuat aplikasi yang bisa mencocokan gambar wajah seseorang.
4. Kami berhasil mengeksplor lebih jauh mengenai dunia informatika dengan melalui Bahasa pemrograman python .
5. Kami berhasil membuat aplikasi yang dapat terintegrasi dengan kamera sebagai alat untuk mengambil gambar tes.

5.2. Saran

Kedepannya aplikasi sangat bisa dikembangkan lagi, apabila diberikan waktu tambahan.Contoh pengembangan yang bisa dilakukan adalah dengan memaksimalkan user interface dengan GUI, untuk mempermudah dan mempernyaman dalam menggunakan program. Dengan menerapkan tampilan yang lebih bagus, sangat mungkin bisa aplikasi tersebut dapat dipasarkan. Selain itu, bisa juga mengubah aplikasi yang berbasis GUI python menjadi berbasis website, sehingga semua orang bisa mengakses hanya dengan bermodal akses internet.

5.3. Refleksi

Permasalahan time-management menjadi masalah utama bagi kami, karena kami harus membagi waktu dengan mata kuliah lain, kegiatan diluar kuliah, sehingga seringkali Tugas Besar ini terbengkalai. Berikutnya, Kami tidak akan mungkin bisa menyelesaikan tugas ini jika hanya berbekal pengetahuan dari materi yang dosen berikan. Jadi, kami perlu eksplorasi lebih jauh untuk dapat membuat algoritma pencocokan wajah. Kemudian, dari GUI bukan merupakan hal yang sudah kami tahu, sehingga dalam pembuatan GUI memakan waktu yang lebih lama. Dari masalah dan tantangan yang kami hadapi tentu kami pasti mengintrospeksi diri kami sendiri dan sehingga masalah-masalah tersebut tidak akan terulang baik pada kegiatan selanjutnya

Referensi

Implementasi opencv pada industri Dan Kehidupan Sehari-Hari. Available at: https://crocodic.com/implementasi-opencv-pada-industri-dan-kehidupan-sehari-hari/ (Accessed: November 21, 2022).

Membangun aplikasi Android Dengan Python Kivy (no date) *Codepolitan*. Available at: https://codepolitan.com/blog/membangun-aplikasi-android-dengan-python-kivy-5b61d26882da1 (Accessed: November 21, 2022).

Opencv Python program for face detection (2017) *GeeksforGeeks*. Available at: https://www.geeksforgeeks.org/opencv-python-program-face-detection/ (Accessed: November 19, 2022).

Homepage Rinaldi Munir. Available at: https://informatika.stei.itb.ac.id/~rinaldi.munir/ (Accessed: November 21, 2022).

Lampiran

1. Link Repository : https://github.com/mutawalle/Algeo02-21054
2. Link Video : https://youtu.be/b_zi63THqTE