MIND Journal | ISSN (p): 2528-0015 | ISSN (e): 2528-0902 | Vol. 1 | No. 2 | Halaman 62 - 75
DOI : https://doi.org/10.26760/mindjournal Juli 2016

Daftar Kehadiran Mahasiswa dengan Autentikasi Wajah Menggunakan Metode Eigenface

MIRA MUSRINI BARMAWI [1], ANDRIANA Z [2], MUHAMAD RIZKI A. F. [3]

[1] [3]JurusanTeknikInformatika, Fakultas Teknik Industri
Institut Teknologi Nasional Bandung
[2]Universitas Langlangbuana Bandung

Email : mmb0036@gmail.com

ABSTRAK

Sistem pengenalan atau autentikasi tidak hanya dengan menggunakan sidik jari, tetapi juga dapat menggunakan pengenalan wajah. Pengenalan wajah dapat dikembangkan sebagai media identifikasi dan memiliki berbagai manfaat, diantaranya tidak diperlukan kartu atau foto pada kartu identifikasi. Metode eigenface digunakan dalam daftar kehadiran dengan pengenalan wajah, media webcam digunakan untuk menangkap gambar secara real-time. Proses dari aplikasi ini adalah kamera menangkap gambar pada wajah, kemudian didapatkan sebuah nilai R, G, B. Dengan melakukan pemrosesan awal dilakukan penyesuaian ukuran, RGB ke Grayscale, dan histogram equalizer. Metode eigenface berfungsi untuk menghitung eigenvalue dan eigenvector yang digunakan sebagai fitur dalam melakukan pengenalan. Euclidean distance digunakan untuk mencari jarak dengan data fitur yang telah didapat, serta jarak terkecil dengan hasilnya. Berdasarkan pengujian aplikasi, tingkat keberhasilan pengenalan citra wajah mencapai 80%, sehingga aplikasi ini dapat dijadikan sebagai alternatif untuk autentikasi kehadiran mahasiswa.Kata Kunci : eigenface, pengenalan wajah, daftar kehadiran

Kata kunci: *eigenface, pengenalan wajah, daftar kehadiran.*

ABSTRACT

The authentication system not only by using a fingerprint but also with face recognition. Face recognition can be developed as identification media and have more benefit, there are not required any card or photo on identification card. In the attendance list with face recognition using Eigenface method, use a webcam as media to capturing real-time images. The application process is a camera did capturing the face, then obtain a value of R, G, B. By conducting the initial process, conducted size adjustments, RGB to Greyscale, and histogram equalization. Eigenface method used to calculate the eigenvalues and eigenvector that used as a feature to authentication perform. Euclidean distance was used to find the distance with data feature that has been obtained, then the smallest distance with the result. Based on test, the success rate reached 80%, so this application can be used as an alternative to authenticate the students presence.

Keywords: *eigenface, face authentication, attendance list*

1. PENDAHULUAN

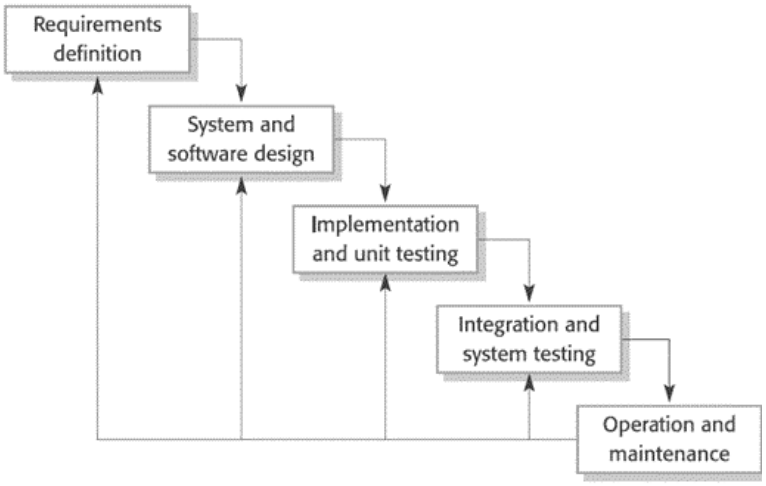
Sistem biometrika merupakan teknologi pengenalan diri dengan menggunakan bagian tubuh atau perilaku manusia, diantaranya sidik jari, tanda tangan, DNA, telinga, wajah, infrared, geometri tangan, selaput pelangi, gaya penekanan tombol, bau, telapak tangan, retina, suara, gigi, dan bibir. Tiap masing-masing pencirian tersebut merupakan karakteristik dari cabang ilmu biometrika. Sistem pengenalan diri ini merupakan sistem untuk mengenali identitas dari setiap makhluk hidup, khususnya manusia secara otomatis dengan menggunakan teknologi komputer, yang bertujuan untuk meningkatkan keamanan sistem dan untuk mengenali target secara cepat dan tepat.

Identifikasi wajah dalam sistem absensi mahasiswa pada sebuah universitas adalah sistem untuk mengontrol sumber daya manusia (SDM). Salah satu fungsi mengontrol SDM bertujuan meningkatkan potensi sumber daya manusia dan efisiensi kehadiran perkuliahan.

Pembangunan aplikasi daftar kehadiran dengan menggunakan autentikasi wajah dapat dijadikan alternatif untuk perekaman dan pengambilan data unik manusia, karena alat yang dipergunakan sangat umum dan biasanya sudah built-in pada setiap perangkat komputer.

2. METODOLOGI PENELITIAN

Metode penelitian yang digunakan dalam perancangan sistem pakar ini adalah metodologi *waterfall.* Proses-proses tersebut dapat dijelaskan sebagai berikut:



Gambar 1. *Waterfall Paradigm*

1. Analisa kebutuhan: Pengumpulan data dalam tahap ini bisa malakukan sebuah penelitian, wawancara atau study literature.
2. Perancangan Sistem: Menerjemahkan syarat kebutuhan sebuah perancangan perangkat lunak yang dapat diperkirakan sebelum dibuat coding
3. *Coding* & Testing : penggunaan computer akan dimaksimalkan dalam tahapan ini. Setelah pengkodean selesai maka akan dilakukan *testing* terhadap sistem yang telah dibuat tadi.

Tujuan *testing* adalah menemukan kesalahan-kesalahan terhadap sistem tersebut dan kemudian bisa diperbaiki

4. Pengujian Sistem: klien mengevaluasi *prototype* yang dibuat dan digunakan untuk memperjelas kebutuhan *software*.
5. Pengujian Program : Tahapan ini bisa dikatakan final dalam pembuatan sebuah sistem. Setelah melakukan analisa, design dan pengkodean maka sistem yang sudah jadiakan digunakan oleh user.
6. Pemeliharaan : Tahap ini merupakan penyusaian program mengenai perubahan yang terjadi pada studi kasus.

2. LANDASAN TEORI

2.1 *Principal Component Analysis*

Metode Principal Component Analysis (PCA) dibuat pertama kali oleh para ahli statistik dan ditemukan oleh Karl Pearson pada tahun l90l yang memakainya pada bidang biologi. Kemudian tidak ada perkembangan baru pada teknik ini, dan. Perkembangannya baru mulai pesat pada akhir tahun l930 dan awal 1940. Setelah itu perkembangannya berkurang sebentar sampai komputer telah berhasil didesain sehingga dapat mengaplikasikan teknik ini pada masalah-masalah yang masuk akal. Pada tahun 1947 teori ini muncul lagi dan cukup independen sebagai teori probabilitas yang ditemukan oleh Karhunen, dan kemudian dikembangkan oleh Loeve pada tahun 1963, sehingga teori ini juga dinamakan Karhunen-Loeve transform pada bidang ilmu telekomunikasi.

PCA adalah sebuah transformasi linier yang biasa digunakan pada kompresi data. PCA adalah sebuah teknik statistika yang berguna pada bidang pengenalan, klasifikasi dan kompresi data citra. PCA juga merupakan teknik yang umum digunakan untuk menarik fitur-fitur dari data pada sebuah skala berdimensi tinggi. Dengan cara mentransformasikan citra ke dalam eigenface secara linier, proyeksikan citra ke dalam bentuk skala berdimensi n, yang menampakkan properti dari sampel yang paling jelas sepanjang koordinat.
Fitur yang paling signifikan yang ada pada citra akan menjadi *principal component* yang akan digunakan untuk pengolahan selanjutnya. Dalam prosesnya *principal component analysis* menggunakan vektor-vektor yang disebut dengan *eigenvector* dan nilai-nilai yang disebut dengan *eigenvalue* untuk mendapatkan fitur yang paling signifikan pada dataset. *Principal component* dicari dengan hubungan [1]:

$$AC = \lambda C \tag{1}$$

di mana A adalah matriks yang akan dicari *principal component*-nya, C adalah *principal component* atau disebut dengan *eigenvector* dan λ adalah *eigenvalue*.

Andaikan A adalah sebuah matriks berdimensi n x n, *eigenvalue* dari matriks A

diperoleh dengan hubungan:

$$\det(A-\lambda I) = 0 \tag{2}$$

di mana I adalah matriks identitas dari A dan λ adalah *eigenvalue* dari matriks A.

Mencari nilai *Eigenvector* dapat dicari dengan memecahkan (A- λI) v = 0, dalam beberapa kasus dapat dijumpai suatu matriks tanpa *eigenvalues*, misalnya:

$$A=\begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \tag{3}$$

di mana karakteristik polinomial-nya adalah λ2 + 1 sehingga *eigenvalue*-nya adalah bilangan bilangan kompleks *i, -i*. Eigenvalue juga tidak riil. Sebagai contoh diberi matriks citra $A = \begin{bmatrix} 2 & 1 \\ 0 & 3 \end{bmatrix}$, polynomial karakteristiknya dapat dicari sebagai berikut:

$$\det \begin{bmatrix} 2-\lambda & -1 \\ 0 & (3-\lambda) \end{bmatrix} = \lambda^2 - 5\lambda + 6 = 0 \quad (4)$$

Ini adalah persamaan kuadrat dengan akarnya λ1 = 2 dan λ2 = 3.

Substitusikan $\lambda_2 = 3$ ke dalam persamaan. Misalnya Y_0 adalah *Eigenvector* yang berasosiasi dengan *Eigenvalue* $\lambda_2 = 3$. Set Y_0 dengan nilai:

$$Y_0 = \begin{bmatrix} X0 \\ Y0 \end{bmatrix} \quad (5)$$

Subtitusikan Y_0 dengan v pada persamaan :

$$(-2 - \lambda I)v = 0 \text{ , diperoleh}$$

$$\begin{bmatrix} (2-3)X0 + -Y0 = 0 \\ 0 + (3-3)Y0 = 0 \end{bmatrix} \quad (6)$$

Dapat disederhanakan lagi menjadi $Y_0 = -X_0$

Sehingga *Eigenvector* untuk *Eigenvalue* = 3 adalah $Y_0 = \begin{bmatrix} X0 \\ Y0 \end{bmatrix} = \begin{bmatrix} X0 \\ X0 \end{bmatrix}$

$$Y_0 = X_0 \begin{bmatrix} 1 \\ -1 \end{bmatrix} \quad (7)$$

2.2 *Eigenface*

Eigenface berasal dari prefix Bahasa Jerman "*eigen"*, yang berarti "sendiri / individual". Metode ini dianggap sebagai teknologi pengenalan wajah otomatis pertama yang pernah diciptakan. Teori *eigenface* dikembangkan dengan membagi sebuah citra wajah menjadi data set fitur karakteristik. Fitur karakteristik merupakan komponen utama (*principal component*) training set awal dari citra wajah. Prinsip dasar dari pengenalan wajah adalah dengan mengutip informasi unik wajah, kemudian di-*encode* dan dibandingkan dengan hasil *decode* yang sebelumnya dilakukan.

Dalam metode *eigenface, decoding* dilakukan dengan menghitung *eigenvector* kemudian direpresentasikan dalam sebuah matriks yang berukuran besar. *Eigenvector* juga dinyatakan sebagai karakteristik wajah, olehkarena itu metode ini disebut dengan *eigenface.* Setiap wajah direpresentasikan dalam kombinasi linear *eigenface.* Teknik yang dikembangkan untuk menghitung koordinat sistem wajah disebut dengan *eigenpicture* [2].

Dalam perhitungan *eigenvector*, citra diubah menjadi bentuk mean seperti yang ditunjukkan pada Gambar 2.

Gambar 2. Citra Mean

Sedangkan citra hasil proses *eigenface* ditunjukkan pada Gambar 3.

Gambar 3. Citra Hasil Proses Eigenface

Perbandingan proses dan hasil pada sistem pengenalan pola wajah yang dibuat harus memiliki fungsi sebagai pembanding antara metode Eigenface untuk digunakan sebagai acuan untuk membentuk kesimpulan dan rekomendasi mengenai metode mana yang efektif dalam sistem pengenalan pola wajah yang dapat digunakan dalam keadaan yang telah dikondisikan sebelumnya.

3. ANALISIS DAN PEMBAHASAN

Dalam membangun aplikasi Identifikasi Wajah, terdapat beberapa kebutuhan sistem yang digunakan.

3.1 Analisis Perangkat Keras

Perangkat keras yang mendukung pembangunan aplikasi Identifikasi Wajah adalah :

1. Laptop Acer Aspire 4810T dengan spesifikasi laptop:
   - ✓ *Processor* Intel Core 2 Duo U9400 1.40 GHz x 2
   - ✓ Memori *3 GB* RAM
   - ✓ Tipe sistem 64-bit *Operating System*
   - ✓ *Hardisk* 500 GB
2. *Device* yang digunakan untuk alat pengenalan wajah adalah *webcam built-in.* berikut spesifikasi *device* pengenalan wajah:
   - ✓ *Brand* : SuYin Optronics Corp.
   - ✓ *Interface* : USB *built-in*

- ✓ *Operating system* : Windows, Win CE, Linux
- ✓ *Resolution* : *up to* 5 *megapixel*
- ✓ *Sensor* : *Crystal Optical*
- ✓ *Power* : *bus-powered / 5v*
- ✓ *Working Current* : <150ma
- ✓ *Temp. / Humidity* : -10 – 70 *celcius. RH[95%]*

3.2 Analisis Perangkat Lunak

Software yang digunakan dalam pengembangan aplikasi pengenalan wajah, yaitu:

1. *Windows 8.1 Professional* 64-bit, dibutuhkan sebagai *operating system* yang digunakan untuk membuat dan menjalankan aplikasi.
2. *Scripting* dan *coding* yang digunakan untuk membuat *interface* dan *function* menggunakan Bahasa pemrograman Matlab R2014b.

Perancangan sistem merupakan bagian dari metodologi pembangunan suatu perangkat lunak yang harus dilakukan setelah melalui tahapan analisis. Pada bagian ini akan dijelaskan perancangan sistem yang dimaksudkan untuk menggambarkan sistem yang dibangun. Langkah-langkah yang dilakukan dalam tahapan perancangan sistem adalah sebagai berikut:

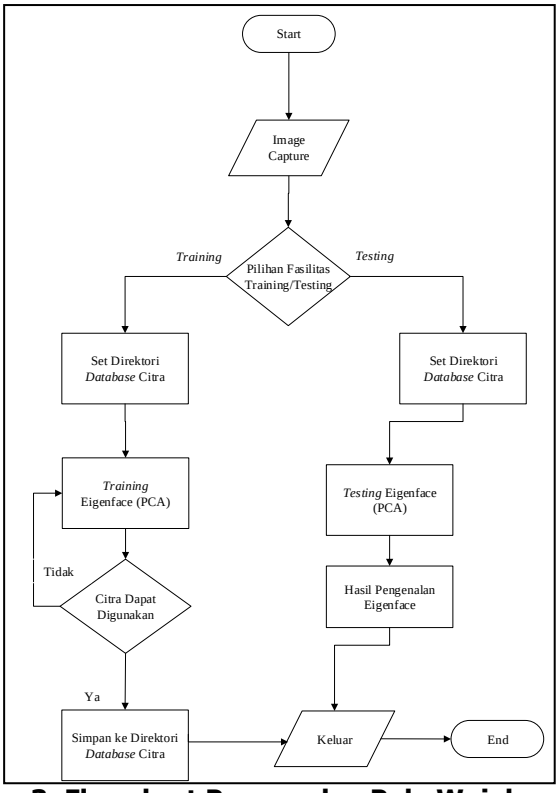
1. Perancangan Arsitektur (Perancangan Struktur Menu)
2. Perancangan Antarmuka (Perancangan *Form*)

3.3 Perancangan Sistem

Proses identifikasi wajah meliputi 2 tahapan proses, yakni training dan testing. Adapun permodelan sistem terlihat pada Gambar 3.

Saat awal aplikasi diaktifkan hal yang pertama yang dilakukan adalah melakukan proses pengambilan gambar yang dilakukan secara *real-time* dengan menggunakan perangkat webcam, gambar yang diperoleh satu persatu akan ditangkap, yang kemudian secara otomatis akan diproses sekaligus dengan menggunakan metode *Eigenface*, lalu selanjutnya memilih proses yang akan dikerjakan, yaitu proses *training* atau proses *testing*.
Pada proses *training*, hasil tangkap citra diproses untuk ditentukan apakah citra tersebut dapat digunakan atau tidak, jika dapat digunakan maka langsung disimpan ke dalam *database*. Sedangkan pada proses *testing*, hasil capture diproses untuk dikenali pola wajah didalam citra sesuai dengan yang ada dalam *database*.

Dasar dari metode Eigenface sering disebut juga transformasi Karhunen-Loeve. Metode ini bertujuan untuk mentransformasi vektor citra dari ruang citra dimensi-n ke ruang ciri dimensi-m.



Gambar 3. Flowchart Pengenalan Pola Wajah pada Eigenface

3.4 Tahapan Proses Sistem

Berikut tahapan untuk dapat mengetahui nilai *eigenface* dari beberapa kumpulan gambar. Misalkan terdapat tiga buah himpunan gambar dengan ukuran 3 x 3 piksel (Γ_1, Γ_2, Γ_3).

Piksel adalah unsur gambar atau representasi sebuah titik terkecil dalam sebuah grafis yang dihitung per inci.



$$A = \begin{bmatrix} 0 & 4 & 3 \\ 1 & 4 & 2 \\ 0 & 0 & 0 \end{bmatrix} \quad B = \begin{bmatrix} 2 & 2 & 1 \\ 3 & 2 & 4 \\ 0 & 0 & 0 \end{bmatrix} \quad C = \begin{bmatrix} 1 & 3 & 2 \\ 2 & 3 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

1. Menentukan nilai tengah atau *mean* (Ψ). Berfungsi untuk mencari nilai rata-rata dari semua sampel yang diuji.

$$\Psi = \frac{1}{3}\sum_{n=1}^{3} \Gamma_n = \frac{1}{3}\left[\begin{bmatrix} 0 & 4 & 3 \\ 1 & 4 & 2 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 2 & 2 & 1 \\ 3 & 2 & 4 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 1 & 3 & 2 \\ 2 & 3 & 0 \\ 0 & 0 & 0 \end{bmatrix}\right] = \begin{bmatrix} 1 & 3 & 2 \\ 2 & 3 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

2. Menentukan selisih (Φ) antara *training image* (Γ) dengan nilai tengah (Ψ). Berfungsi untuk mencari persamaan nilai antara citra uji dengan nilai rata-rata sampel yang diuji.

$$\Phi_1 = \Gamma_1 - \Psi = \begin{bmatrix} 0 & 4 & 3 \\ 1 & 4 & 2 \\ 0 & 0 & 0 \end{bmatrix} - \begin{bmatrix} 1 & 3 & 2 \\ 2 & 3 & 0 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\Phi_2 = \Gamma_2 - \Psi = \begin{bmatrix}2 & 2 & 1\\ 3 & 2 & 4\\ 0 & 0 & 0\end{bmatrix} - \begin{bmatrix}1 & 3 & 2\\ 2 & 3 & 0\\ 0 & 0 & 0\end{bmatrix} = \begin{bmatrix}1 & 0 & 0\\ 1 & 0 & 1\\ 0 & 0 & 0\end{bmatrix}$$

$$\Phi_3 = \Gamma_3 - \Psi = \begin{bmatrix}1 & 3 & 2\\ 2 & 3 & 0\\ 0 & 0 & 0\end{bmatrix} - \begin{bmatrix}1 & 3 & 2\\ 2 & 3 & 0\\ 0 & 0 & 0\end{bmatrix} = \begin{bmatrix}0 & 1 & 0\\ 0 & 0 & 0\\ 0 & 0 & 0\end{bmatrix}$$

3. Menghitung nilai matriks kovarian (C). Digunakan untuk mengoptimalkan representasi distribusi data antara citra uji dengan selisih dari keseluruhan citra yang diuji. Berikut rumus yang digunakan:

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

   $C = AA^T \; A = [\Phi_1, \Phi_2, \Phi_3]$

   $L = AA^T$, dimana $L_{m,n} = \Phi_n \Phi_n^T$

$$\begin{bmatrix} \begin{bmatrix}0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0\\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0\\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0\end{bmatrix} & \\ \begin{bmatrix}4 & 2 & 1\\ 2 & 3 & 1\\ 1 & 1 & 1\end{bmatrix} & x \begin{bmatrix}0 & 0 & 0\\ 1 & 1 & 0\\ 1 & 0 & 0\\ 1 & 1 & 1\\ 0 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 0\\ 1 & 0 & 0\\ 0 & 0 & 0\end{bmatrix} \end{bmatrix}$$

4. Menghitung *eigenvalue* (λ) dan *eigenvector* (v) dari matriks kovarian (C). *Eigenvalue* adalah nilai yang dihasilkan dari nilai rata-rata sampel dikurangi dengan selisih keseluruhan citra yang diuji, sedangkan *eigenvector* merupakan matriks yang mendefinisikan nilai antara citra penguji dengan selisih nilai *training image* dengan nilai tengah. Rumus yang digunakan adalah:

   $C \times v_i = \lambda_i \times v_i$

   Mencari nilai *eigenvalue* (λ) dan *eigenvector* (v).

   $L \times v = \lambda \times v$

   $L \times v = \lambda I \times v$

   $(L - \lambda I) = 0$ atau $(\lambda I - L) = 0$

   Maka *eigenvalue* (λ) dihitung dengan, det $(\lambda I - L) = 0$

$$= \lambda \begin{bmatrix}1 & 0 & 0\\ 1 & 0 & 1\\ 0 & 0 & 0\end{bmatrix} - \begin{bmatrix}4 & 2 & 1\\ 2 & 3 & 1\\ 1 & 1 & 1\end{bmatrix} = \begin{bmatrix}\lambda - 4 & 2 & 1\\ 2 & \lambda - 3 & 1\\ 1 & 1 & \lambda - 1\end{bmatrix}$$

   Maka dihasilkan nilai λ = 1 dan λ = 4

$$v = \begin{bmatrix}v1\\ v2\\ v3\end{bmatrix}$$

   eigenvector (v) dihasilkan dengan mensubtitusi *eigenvalue* (λ) ke dalam persamaan (λI – L) v = 0

untuk λ = 4, maka $\begin{bmatrix}4-4 & 2 & 1\\ 2 & 4-3 & 1\\ 1 & 1 & 4-1\end{bmatrix}\begin{bmatrix}v1\\ v2\\ v3\end{bmatrix} = \begin{bmatrix}0\\ 0\\ 0\end{bmatrix}$

$$\begin{bmatrix}2 & 2 & 1\\ 2 & 1 & 1\\ 1 & 1 & 3\end{bmatrix}\begin{bmatrix}v1\\ v2\\ v3\end{bmatrix} = \begin{bmatrix}2v1 & 2v2 & v3\\ 2v1 & v2 & v3\\ v1 & v2 & 3v3\end{bmatrix} = \begin{bmatrix}0\\ 0\\ 0\end{bmatrix}$$

Dihasilkan *eigenvector* $\begin{bmatrix}1\\-1\\0\end{bmatrix}$ dan $\begin{bmatrix}-3\\0\\1\end{bmatrix}$

Untuk λ = 1, maka $\begin{bmatrix}1-4 & 2 & 1\\ 2 & 1-3 & 1\\ 1 & 1 & 1-1\end{bmatrix}\begin{bmatrix}v1\\v2\\v3\end{bmatrix}=\begin{bmatrix}0\\0\\0\end{bmatrix}$

$\begin{bmatrix}-3 & 2 & 1\\ 2 & -2 & 1\\ 1 & 1 & 0\end{bmatrix}\begin{bmatrix}v1\\v2\\v3\end{bmatrix}=\begin{bmatrix}-3v1 & 2v2 & v3\\ 2v1 & -2v2 & v3\\ v1 & v2 & 0\end{bmatrix}=\begin{bmatrix}0\\0\\0\end{bmatrix}$

Dihasilkan *eigenvector* $\begin{bmatrix}1\\1\\0\end{bmatrix}$

sehingga matriks yang dihasilkan matriks L adalah $\begin{bmatrix}1 & -3 & 1\\ -1 & 1 & 1\\ 0 & 1 & 0\end{bmatrix}$

Eigenface (μ) didapatkan dengan

$\mu_1 = \sum_{k=1}^{M} v_{lk}\,\Phi_k$

$\mu_1 = v.\Phi_1 = \begin{bmatrix}1 & -3 & 1\\ -1 & 1 & 1\\ 0 & 1 & 0\end{bmatrix}\begin{bmatrix}0 & 1 & 1\\ 0 & 1 & 0\\ 0 & 0 & 0\end{bmatrix}=\begin{bmatrix}0 & -3 & 1\\ 0 & 1 & 0\\ 0 & 1 & 0\end{bmatrix}$

$\mu_2 = v.\Phi_2 = \begin{bmatrix}1 & -3 & 1\\ -1 & 1 & 1\\ 0 & 1 & 0\end{bmatrix}\begin{bmatrix}1 & 0 & 0\\ 1 & 0 & 1\\ 0 & 0 & 0\end{bmatrix}=\begin{bmatrix}1 & 0 & 0\\ -1 & 0 & 1\\ 0 & 0 & 0\end{bmatrix}$

$\mu_3 = v.\Phi_3 = \begin{bmatrix}1 & -3 & 1\\ -1 & 1 & 1\\ 0 & 1 & 0\end{bmatrix}\begin{bmatrix}0 & 1 & 0\\ 0 & 0 & 0\\ 0 & 0 & 0\end{bmatrix}=\begin{bmatrix}0 & 1 & 0\\ 0 & -1 & 0\\ 0 & 0 & 0\end{bmatrix}$

Untuk proses pengenalan gambar, tahapannya sebagai berikut.

1. Sebuah *test face* (Γ_{new}) dicoba untuk dikenali pada tahapan pertama perhitungan *eigenface* untuk mendapatkan nilai *eigenface* dari gambar tersebut.
   a. Mencari selisih (Φ) antara *test face* (Γ_{new}) dengan nilai tengah (Ψ), apabila terdapat nilai dibawah nol ganti nilainya dengan nol.
      Misal *test face* (Γ_{new}) terdiri dari matriks 3 x 3 pixel.

Test Face	$\begin{bmatrix}2 & 6 & 5\\ 0 & 1 & 3\\ 1 & 0 & 0\end{bmatrix}$

      $\Phi_{new} = \begin{bmatrix}2 & 6 & 5\\ 0 & 1 & 3\\ 1 & 0 & 0\end{bmatrix} - \begin{bmatrix}1 & 3 & 2\\ 2 & 3 & 2\\ 0 & 0 & 0\end{bmatrix} = \begin{bmatrix}1 & 3 & 3\\ 0 & 0 & 1\\ 1 & 0 & 0\end{bmatrix}$

   b. Mencari nilai *eigenface* dari *test face*

      $\mu_{new} = v.\ \Phi_{new}$

      $\Phi_{new} = \begin{bmatrix}1 & -3 & 1\\ -1 & 1 & 1\\ 0 & 1 & 0\end{bmatrix}\begin{bmatrix}1 & 3 & 3\\ 0 & 0 & 1\\ 1 & 0 & 0\end{bmatrix} = \begin{bmatrix}1 & -9 & 3\\ 0 & 0 & 1\\ 0 & 0 & 0\end{bmatrix}$

2. *Euclidean Distance* digunakan untuk mencari selisih terkecil antara *eigenface training image* (Γ_1) dalam database dengan *eigenface test face* (Γ_{new}).
   $\varepsilon_k = \|\Omega - \Omega new\|$

$$\varepsilon_1 = \|\Omega 1 - \Omega new\| = \left\| \begin{bmatrix} 0 & -3 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \end{bmatrix} - \begin{bmatrix} 1 & -9 & 3 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \right\| = 12$$

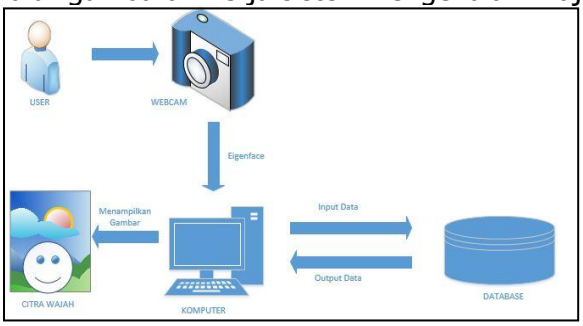
$$\varepsilon_2 = \|\Omega 1 - \Omega new\| = \left\| \begin{bmatrix} 1 & 0 & 0 \\ -1 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} - \begin{bmatrix} 1 & -9 & 3 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \right\| = 13$$

$$\varepsilon_3 = \|\Omega 1 - \Omega new\| = \left\| \begin{bmatrix} 0 & 1 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \begin{bmatrix} 1 & -9 & 3 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \right\| = 16$$

Karena jarak *eigenface* pada *image* 1 dengan *test face* paling kecil, maka hasil identifikasi menyimpulkan bahwa *test face* lebih mirip dengan *image* 1.

3.5 Gambaran Sistem

Gambar 4 menunjukkan alur gambaran kerja sistem Pengenalan Wajah yang dibangun.



Gambar 4. Gambaran Cara Kerja Sistem

Cara kerja dari aplikasi *face recognition* dalam mengidentifikasi wajah manusia, yaitu :

1. Proses akuisisi citra dilakukan dengan menggunakan *webcam* yang sudah *built-in* pada perangkat komputer. Proses ini dilakukan dengan menempatkan kamera tepat didepan wajah yang akan dilakukan *snapshot*.
2. Proses berikutnya adalah sistem mengubah nilai RGB dari gambar yang ditangkap menjadi *grayscale*, kemudian dilakukan perhitungan *eigenvector* pada gambar.
3. Selanjutnya membandingkan citra uji dengan citra pembanding dengan cara menghitung dua matriks yang berbeda menggunakan *euclidean distance*, kemudian diambil nilai terkecil atau nilai yang paling mendekati 0.
4. Setelah citra uji terdeteksi, kemudian sistem akan mengetahui dan mencocokkan dengan citra pembanding yang terdapat di *database.*
5. Pada proses selanjutnya, hasil dari perhitungan tersebut akan menampilkan nama dan keterangannya pada sistem.

3.5.1 Tahap *Grayscale*

Grayscale merupakan sebuah hasil dari proses pengolahan citra karena dapat menyederhanakan proses yang harus dilakukan dibandingkan menggunakan citra berwarna [3]. Perhitungan secara manual konversi RGB menjadi *grayscale*. Misalkan sebuah citra warna berukuran 3×3 piksel dengan nilai-nilai RGB pada perhitungan matriks berikut :

$$R = \begin{bmatrix} 32 & 45 & 50 \\ 66 & 78 & 80 \\ 104 & 123 & 138 \end{bmatrix} \quad G = \begin{bmatrix} 31 & 41 & 52 \\ 61 & 75 & 87 \\ 109 & 124 & 133 \end{bmatrix} \quad B = \begin{bmatrix} 39 & 40 & 51 \\ 60 & 77 & 89 \\ 106 & 128 & 130 \end{bmatrix}$$

Dengan menggunakan persamaan, yaitu :

$$fo(x,y) = \frac{fi^R(x,y) + fi^G(x,y) + fi^B(x,y)}{3}$$

$$Grayscale = \begin{bmatrix} 34 & 42 & 51 \\ 62 & 76 & 85 \\ 106 & 125 & 133 \end{bmatrix}$$

Gambar 5. Proses *Greyscale*

3.5.2 *Euclidean Distance*

Dalam identifikasi pola wajah antara data penguji dengan data uji dicari nilai kemiripan antara tiap-tiap citra [4]. Model matematis *euclidean distance* dapat diekspresikan menggunakan persamaan :

$$D(A,B) = \sqrt{\sum_{i=1}^{n}(Ai - Bi)^2}$$

Berikut adalah contoh perhitungan nilai *euclidean distance* dengan terdapat dua buah vektor *image*:

A = [0, 3, 4, 5]

B = [7, 6, 3, -1]

Maka euclidean distance dari dua vektor image tersebut adalah:

$$\|A\| = \left[\sum_i A_i^2\right]^{1/2} = \sqrt{0^2+3^2+4^2+5^2} = \sqrt{50}$$

$$\|B\| = \left[\sum_i B_i^2\right]^{1/2} = \sqrt{7^2+6^2+3^2+(-1)^2} = \sqrt{95}$$

$$\overline{A} = \left[\frac{0}{\sqrt{50}}, \frac{3}{\sqrt{50}}, \frac{4}{\sqrt{50}}, \frac{5}{\sqrt{50}}\right] = [0,\ 0.42,\ 0.56,\ 0.7]$$

$$\overline{B} = \left[\frac{7}{\sqrt{95}}, \frac{6}{\sqrt{95}}, \frac{3}{\sqrt{95}}, \frac{-1}{\sqrt{95}}\right] = [0.72,\ 0.61,\ 0.3,\ -0.1]$$

$$\overline{d}(A,B) = \sqrt{(0-0.72)^2 + (0.42-0.61)^2 + (0.56-0.3)^2 + (0.7+0.1)^2}$$

$$\overline{d}(A,B) = \sqrt{1.25}$$

$$\overline{d}(A,B) = 1.11$$

3.6 Implementasi Akuisisi Citra

Dalam sistem ini dilakukan beberapa pengujian yang terdapat dalam aplikasi autentikasi wajah manusia. Berikut ini adalah salah satu tahapan dan hasil pengujian yang telah dilakukan. Gambar 6 merupakan *source code* dari proses akuisisi citra, sedangkan pada Gambar 7 menunjukkan *screenshot* proses akuisisi citra yang dilakukan dalam aplikasi dengan kamera dan citra wajah, menangkap citra wajah lalu dimasukkan ke dalam *database.*

```
if(statVideo == true)
        trigger(handles.video);
        img = getdata(handles.video,1);
        stop(handles.video);
    else
        img = imread(imgstr);
    end

        imgInput=imresize(img,[200 180]);

        imgstrr=strcat(TrainDB,'/',num2str(TN+1),'.jpg');
        imwrite(imgInput,imgstrr,'jpg');

        msgbox(strcat(imgstrr,' berhasil tersimpan'),'Informasi');
        set(handles.edGambar1,'String','');
set(handles.edGambar2,'String','');

        dtz.no = TN+1;
        dtz.file = imgstrr;
        dtz.nama = nama;

        dt = [dt dtz];
```

Gambar 6. *Source Code* Input Citra Wajah ke *Database*

Gambar 7. Pengujian Input Citra Wajah ke *Database*

3.7 Pengujian Aplikasi

Pengujian dilakukan dengan mengambil sampel 5 wajah berbeda yang dilakukan sebanyak satu kali akusisi. Total 5 data citra wajah disimpan dalam lokal *folder*. Tabel 1 merupakan hasil pengujian proses *matching* dengan sampel citra menggunakan metode *Eigenface*.

Tabel 1. Hasil Pengujian

Data Uji	Terdeteksi	Keterangan
Rizki	Rizki	*Match*
Luqman	Indra	*Unmatch*
Indra	Indra	*Match*
Dinal	Dinal	*Match*
Ersa	Ersa	*Match*

Berdasarkan hasil uji yang dilakukan dapat disimpulkan bahwa sistem dapat mengenali wajah manusia dengan kategori keberhasilan yang didapat adalah baik. Banyak faktor yang

mempengaruhi sistem dalam mencapai tujuan yang diinginkan. Berikut adalah faktor yang dapat mempengaruhi keberhasilan sistem :
Perancangan antar muka merupakan perancangan yang dibuat sebelum program aplikasi dibuat, dimana perancangan ini bertujuan untuk menggambarkan tampilan program secara umum, yang nantinya diharapkan dapat memudahkan proses pembuatan aplikasi.
Perancangan *form-form* yang terdapat dalam aplikasi perbandingan hasil dan proses metode *Eigenface* pada sistem pengenalan pola wajah, adalah sebagai berikut:

- Resolusi kamera, yaitu semakin tinggi resolusi kamera maka semakin tinggi akurasi nilai yang didapatkan.
- Perbedaan intensitas cahaya antara data uji dan data penguji.
- Posisi wajah antara data uji dan data penguji.

3.8 Pengujian *Matching* Citra

Proses pengujian ekstraksi ciri dan matching citra wajah sebagai sampel pengujian. Pada proses *matching* citra sistem dilakukan dengan mencocokan citra uji dengan citra latih menggunakan *euclidean distance*. Gambar 8 menunjukkan *screenshot* hasil proses uji citra terhadap citra *database.*

Gambar 8. Hasil Proses Uji Citra

4. KESIMPULAN

Untuk menjawab kesimpulan hasil analisis dan perancangan sistem dari aplikasi Daftar Kehadiran dengan Autentikasi Wajah Menggunakan Metode *Eigenface* diambil kesimpulan berdasarkan dari implementasi dan pengujian sistem yang telah dilakukan, yaitu:

- Berdasarkan Tabel 1 Metode Eigenface dapat dijadikan alternatif untuk membangun aplikasi daftar kehadiran dengan pengolahan citra wajah manusia, karena sistem yang dirancang mencapai hasil 80% dari 5 sampel uji citra wajah.

DAFTAR RUJUKAN

[1] Ika Made. 2003. "Face Recognition dengan Menggunakan Metode Principal Component Analysis (PCA), Makalah Skripsi Universitas Gunadarma Jakarta

[2] Prasetyo Eri dan Rahmatun Isna. 2005. "Desain Variasi Wajah dengan Variasi Ekspresi dan Posisi Menggunakan Metode Eigenface". Makalah Skripsi Universitas Gunadarma Jakarta

[3] Abdul Kadir dan Adhi Susanto. 2013. Teori dan Aplikasi Pengolahan Citra. Yogyakarta

[4] R. H. Sianipar, S.T., M.T., M.Eng., Ph.D. 2013. Pemrograman Matlab

[5] Sandra Agustyan Putra. 2010. Sistem Absensi Mahasiswa Secara Visual Menggunakan Webcam Dengan Dynamic Times Warping