BAB 3

METODOLOGI PERANCANGAN

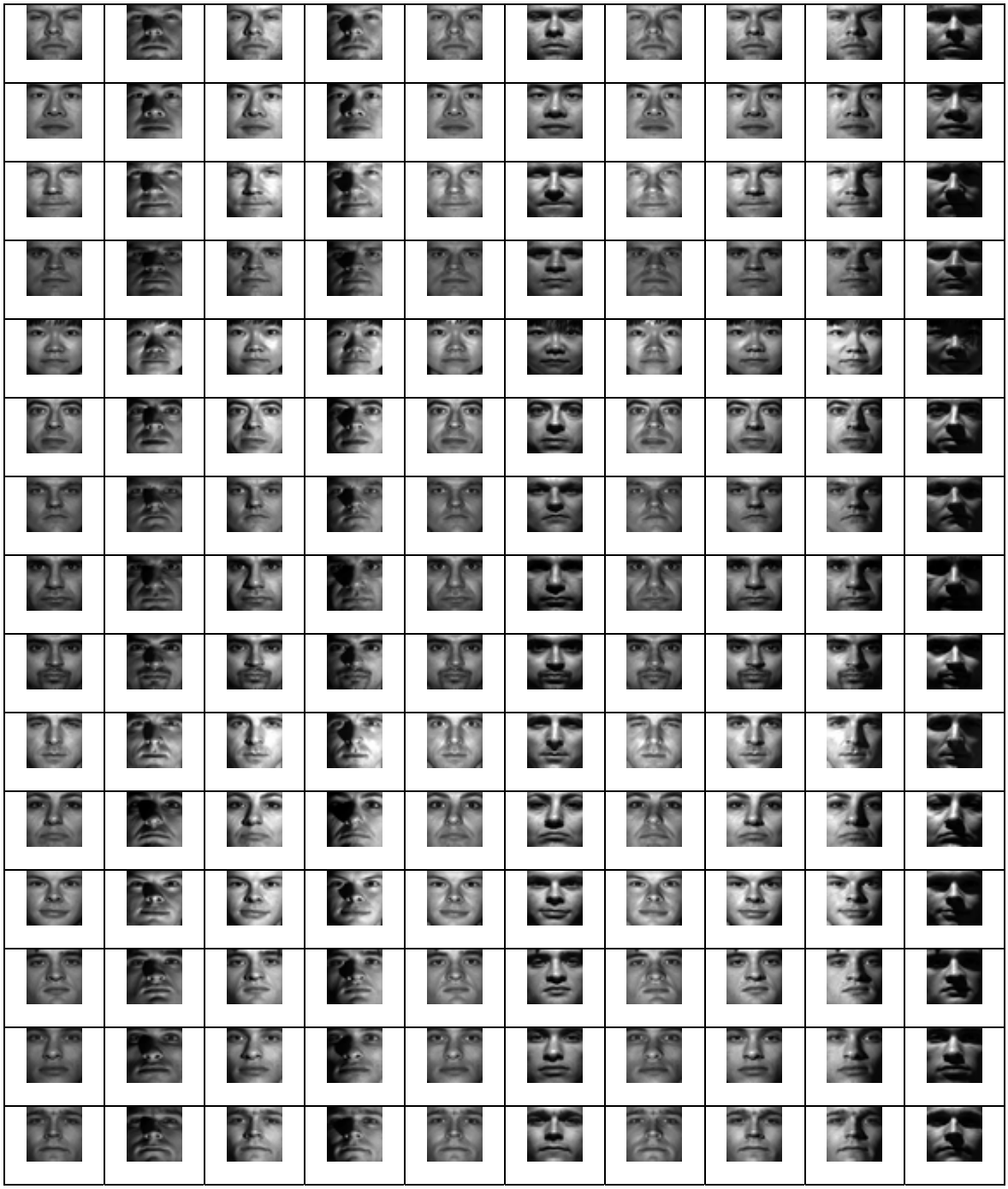
3.1 Input Data Citra Wajah

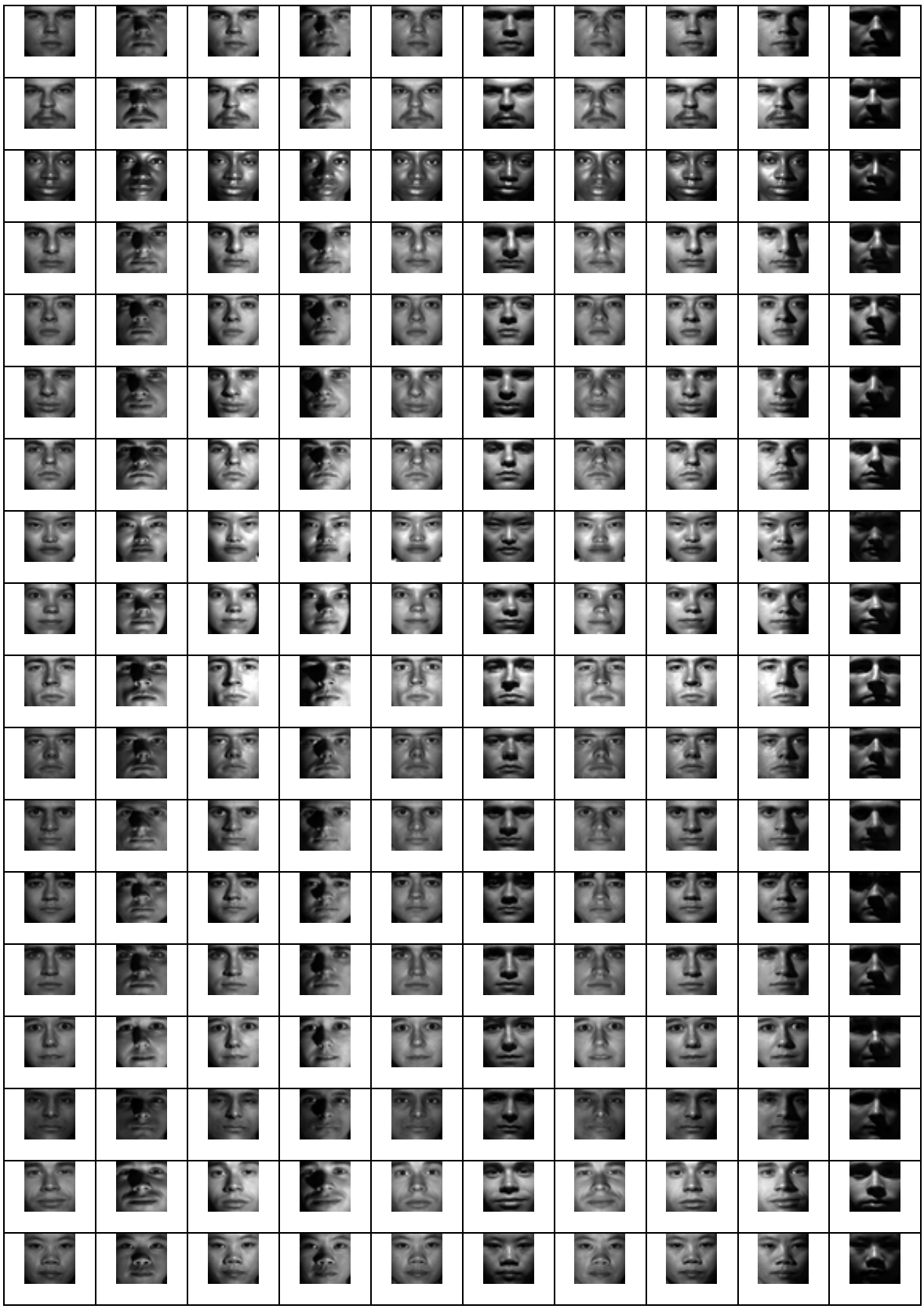
Pada penelitian ini, digunakan sebanyak 525 citra wajah yang terdiri dari 35 orang. Setiap orang diambil sampel sebanyak 15 citra wajah dengan pencahayaan yang berbeda – beda, dimana 10 citra wajah digunakan untuk input pada tahap pelatihan (*sample learning*) dan 5 citra wajah digunakan sebagai tahap pengujian (*sample detection*).

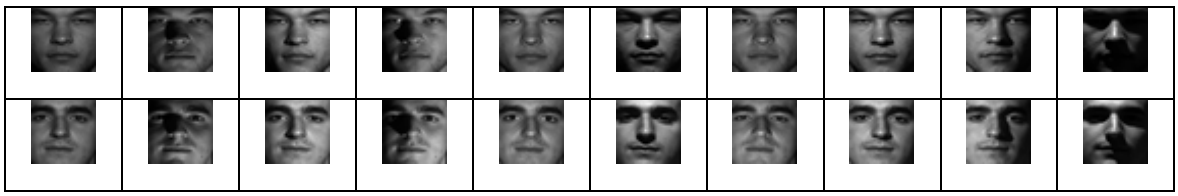
Setiap citra wajah menggunakan format *bitmap* berdimensi 32 x 32 *pixel* dan berwarna *greyscale* 8 bit (256 tingkat keabu – abuan). Karena format ini menyimpan citra *pixel per pixel* dan tanpa adanya kompresi data, maka tidak memungkinkan adanya kehilangan informasi pada citra yang dapat mengurangi keakuratan pengenalan.

Citra yang digunakan pada penelitian ini didapat dari *Extended Yale Face Database B* (mengacu pada "*the Extended Yale Face Database B and reference Athinodoros Georghiades, Peter Belhumeur, and David Kriegman*" dan "*From Few to Many: Illumination Cone Models for Face Recognition under Variable Lighting and Pose*", PAMI, 2001). *Extended Yale Face Database B m*erupakan pengembangan dari *Yale Face Database B* yang asli dengan 10 subjek yang pertama kali dibuat oleh Kuang-Chih Lee, Jeffrey Ho, dan David Kriegman pada

"*Acquiring Linear Subspaces for Face Recognition under Variable Lighting, PAMI*, *May*, 2005". Keseluruhan citra wajah secara *manual* diratakan, dipotong dan diubah ukurannya menjadi 32 x 32 *pixel*.







Tabel 3.1 Daftar citra wajah yang digunakan

3.2 Perancangan Program Aplikasi

3.2.1 Pengurangan Dimensi dengan PCA

Dalam analisis dan pengenalan wajah, data *singular* merupakan masalah utama. Dikarenakan, kadang – kadang jumlah citra pada *training set* sangat kecil dibandingkan jumlah *pixel* pada setiap citra. Untuk menangani masalah ini digunakanlah PCA sebagai pemecahannya sekaligus untuk mengurangi *noise* pada data.

Pada langkah pengurangan dimensi dengan PCA ini, nilai – nilai *pixel* pada citra sampel disimpan ke dalam vektor Tau. Dikarenakan citra sampel merupakan matriks dua dimensi berukuran n x m dimana n adalah tinggi dan m adalah lebar dari citra dan ukuran tiap citra sampel adalah 32 x 32 *pixel*, maka vektor Tau yang didapat berukuran 1x1024. Banyaknya vektor Tau yang dihasilkan sama banyaknya dengan jumlah citra sampel yang dipakai. $\tau_i = [\tau_1, \tau_2, \tau_3, \ldots, \tau_b]$, τ_i adalah vektor Tau ke-i.

Vektor – vektor Tau yang telah didapat kemudian digabung menjadi sebuah matriks Tau berukuran a x (n x m).

$$T = \begin{bmatrix} \tau_{11} & \tau_{12} & \tau_{13} & \dots & \tau_{1b} \\ \tau_{21} & \tau_{22} & \tau_{23} & \dots & \tau_{2b} \\ \dots & \dots & \dots & & \dots \\ \dots & \dots & \dots & & \dots \\ \tau_{a1} & \tau_{a2} & \tau_{a3} & \dots & \tau_{ab} \end{bmatrix}$$

T adalah matriks Tau berukuran a x b (350 x 1024), a adalah jumlah vektor atau jumlah citra (350), dan b adalah ukuran citra (32 x 32 = 1024).

Matriks Tau yang sudah didapat masih memiliki *noise* pada setiap elemen matriksnya. Untuk menghilangkan *noise*, sebuah vektor Psi perlu dihitung untuk mendapatkan besarnya *noise* pada matriks Tau, sehingga vektor – vektor Tau baru yang bebas noise, vektor Phi, dapat dibentuk. Vektor Psi dapat dihitung dengan rumus $\psi = \frac{1}{a}\sum_{i=1}^{a}\tau_i$, ψ adalah vektor Psi, a adalah jumlah citra (350), dan τ_i adalah vektor Tau ke-i, sehingga didapat sebuah vektor Psi dengan ukuran 1 x b (1 x 1024), $\psi = [\psi_1, \psi_2, \psi_3, \dots, \psi_b]$.

Setelah vektor Psi didapat, maka vektor Phi dapat dibentuk dengan rumus $\varphi_i = \tau_i - \psi$, φ_i adalah vektor Phi ke-i, τ_i adalah vektor Tau ke-I, dan ψ adalah vektor Psi.

Vektor – vektor Phi yang diperoleh kemudian disusun menjadi sebuah matriks Phi yang berukuran a x b (350 x 1024).

$$\Phi = \begin{bmatrix} \varphi_{11} & \varphi_{12} & \varphi_{13} & \cdots & \varphi_{1b} \\ \varphi_{21} & \varphi_{22} & \varphi_{23} & \cdots & \varphi_{2b} \\ \cdots & \cdots & \cdots & & \cdots \\ \cdots & \cdots & \cdots & & \cdots \\ \varphi_{a1} & \varphi_{a2} & \varphi_{a3} & \cdots & \varphi_{ab} \end{bmatrix}$$

Setelah matriks Phi (matriks Tau baru yang bebas *noise*) berhasil dibentuk, matriks kovarian dihitung untuk mendapatkan nilai eigen dengan rumus Cov = $\Phi\ \Phi^T$. Sehingga didapat matriks Cov dengan ukuran a x a (350 x 350).

$$\text{Cov} = \begin{bmatrix} c_{11} & c_{12} & c_{13} & \cdots & c_{1a} \\ c_{21} & c_{22} & c_{23} & \cdots & c_{2a} \\ \cdots & \cdots & \cdots & & \cdots \\ \cdots & \cdots & \cdots & & \cdots \\ c_{a1} & c_{a2} & c_{a3} & \cdots & c_{aa} \end{bmatrix}$$

Setelah matriks kovarian didapat, maka nilai eigen dapat dihitung dengan menggunakan rumus Det(λI – Cov) = 0, λ adalah nilai eigen dan I adalah matriks identitas. Diperoleh matriks berukuran 1 x a (1 x 350). Dikarenakan matriks Cov yang besar, maka digunakan bantuan *software* Matlab untuk mempermudah perhitungan, dengan sintaks [eigvector, eigvalue] = eig(Cov). Didapat matriks Veigen yang merupakan kumpulan vektor eigen dengan ukuran a x a (350 x 350).

$$\text{Veigen} = \begin{bmatrix} v_{11} & v_{12} & v_{13} & \dots & v_{1a} \\ v_{21} & v_{22} & v_{23} & \dots & v_{2a} \\ \dots & \dots & \dots & & \dots \\ \dots & \dots & \dots & & \dots \\ v_{a1} & v_{a2} & v_{a3} & \dots & v_{aa} \end{bmatrix}$$

Elemen – elemen vektor eigen pada matriks Veigen diurutkan berdasarkan nilai eigen-nya secara *descending* (mulai dari nilai yang paling besar menuju nilai yang paling kecil). Vektor eigen yang telah diurutkan akan mengalami pengurangan dimensi untuk mempermudah perhitungan. Pengurangan dimensi ini berdasarkan pada nilai vektor eigen yang lebih kecil daripada batasan yang ditentukan (*default*: e^{-12}) sehingga didapatkan vektor eigen dengan ukuran a x c, c adalah dimensi baru yang dimiliki setelah tahap pengurangan dimensi.

$$\text{Veigen} = \begin{bmatrix} v_{11} & v_{12} & v_{13} & \dots & v_{1c} \\ v_{21} & v_{22} & v_{23} & \dots & v_{2c} \\ \dots & \dots & \dots & & \dots \\ \dots & \dots & \dots & & \dots \\ v_{a1} & v_{a2} & v_{a3} & \dots & v_{ac} \end{bmatrix}$$

Matriks Veigen harus dibangun kembali agar mewakili himpunan citra awal, dapat dihitung dengan rumus $\text{Eigen}_{PCA} = \Phi^T$ x Veigen

$$\text{Eigen}_{PCA} = \begin{bmatrix} e_{11} & e_{12} & e_{13} & \dots & e_{1c} \\ e_{21} & e_{22} & e_{23} & \dots & e_{2c} \\ \dots & \dots & \dots & & \dots \\ \dots & \dots & \dots & & \dots \\ e_{b1} & e_{b2} & e_{b3} & \dots & e_{bc} \end{bmatrix}$$

Setiap kolom pada matriks tersebut mewakili karateristik citra awal. Matriks $Eigen_{PCA}$ tersebut kemudian dikalikan lagi dengan data awal (matriks Phi) untuk mendapatkan data yang non-*singular* untuk diproses pada tahap perhitungan *Orthogonal Laplacianface*, dengan rumus X = Phi x $Eigen_{PCA}$.

$$X = \begin{bmatrix} x_{11} & x_{12} & x_{13} & \dots & x_{1c} \\ x_{21} & x_{22} & x_{23} & \dots & x_{2c} \\ \dots & \dots & \dots & & \dots \\ \dots & \dots & \dots & & \dots \\ x_{a1} & x_{a2} & x_{a3} & \dots & c_{ac} \end{bmatrix}$$

Hasilnya adalah sebuah matriks X berukuran a x c.

3.2.2 Pembentukan Matriks Bobot

Pembentukan matriks bobot terdiri dari dua bagian yaitu pembentukan *nearest-neighbor graph* dan pembentukan titik berat.

3.2.2.1 Pembentukan Nearest-neighbor Graph

Dengan menggunakan matriks Tau baru yang didapat pada proses pengurangan dimensi dengan PCA, matriks jarak antar data dapat dibentuk dengan menyusun vektor jarak yang dihitung dengan rumus $d_i = \sqrt{\sum (T_i - T_j)^2}$, d_i adalah vektor jarak ke-i.

Vektor – vektor jarak yang sudah didapat, digabungkan menjadi sebuah matriks D dengan ukuran a x a (350 x 350).

$$D = \begin{bmatrix} d_{11} & d_{12} & d_{13} & \dots & d_{1a} \\ d_{21} & d_{22} & d_{23} & \dots & d_{2a} \\ \dots & \dots & \dots & & \dots \\ \dots & \dots & \dots & & \dots \\ d_{a1} & d_{a2} & d_{a3} & \dots & c_{aa} \end{bmatrix}$$

Setelah matriks D didapat, matriks tersebut dikalikan dengan *transpose*-nya untuk memulai pembentukan matriks *nearest-neighbor graph*, $D = D \times D^T$.

Matriks *nearest-neighbor graph* dibentuk dengan memasukkan nilai 1 pada setiap kolom ke-(knn+1) (*default* knn: 5) pada vektor D (yang sebelumnya sudah dikalikan dengan *transpose*-nya). Sehingga didapat matriks G dengan ukuran a x a (350 x 350).

$$G = \begin{bmatrix} g_{11} & g_{12} & g_{13} & \dots & g_{1a} \\ g_{21} & g_{22} & g_{23} & \dots & g_{2a} \\ \dots & \dots & \dots & & \dots \\ \dots & \dots & \dots & & \dots \\ g_{a1} & g_{a2} & g_{a3} & \dots & g_{aa} \end{bmatrix}$$

G adalah vektor *nearest-neighbor graph*.

3.2.2.2 Pembentukan Bobot

Setelah didapat matriks J dan matriks G pada tahap pembentukkan *nearest-neighbor graph* maka matriks bobot dapat dihitung dengan rumus W = $e^{-D(D)} * G$.

$$W = \begin{bmatrix} w_{11} & w_{12} & w_{13} & \dots & w_{1a} \\ w_{21} & w_{22} & w_{23} & \dots & w_{2a} \\ \dots & \dots & \dots & & \dots \\ \dots & \dots & \dots & & \dots \\ w_{a1} & w_{a2} & w_{a3} & \dots & w_{aa} \end{bmatrix}$$

Didapatkan sebuah matriks W (matriks bobot) berukuran a x a (350 x 350) yang akan digunakan dalam perhitungan *Orthogonal Laplacianfaces*.

3.2.3 Orthogonal Laplacianfaces

Dari tahap pengurangan dimensi dengan PCA kita mendapatkan dua buah matriks yang akan diolah selanjutnya di tahap *Orthogonal Laplacianfaces*, yaitu matriks $Eigen_{PCA}$ dan matriks X

$$Eigen_{PCA} = \begin{bmatrix} e_{11} & e_{12} & e_{13} & \dots & e_{1c} \\ e_{21} & e_{22} & e_{23} & \dots & e_{2c} \\ \dots & \dots & \dots & & \dots \\ \dots & \dots & \dots & & \dots \\ e_{a1} & e_{a2} & e_{a3} & \dots & e_{ac} \end{bmatrix}, \quad X = \begin{bmatrix} x_{11} & x_{12} & x_{13} & \dots & x_{1c} \\ x_{21} & x_{22} & x_{23} & \dots & x_{2c} \\ \dots & \dots & \dots & & \dots \\ \dots & \dots & \dots & & \dots \\ x_{a1} & x_{a2} & x_{a3} & \dots & c_{ac} \end{bmatrix}$$

dan pada tahap pembentukan *nearest-neighbor graph* didapatkan matriks bobot.

$$W = \begin{bmatrix} w_{11} & w_{12} & w_{13} & \dots & w_{1a} \\ w_{21} & w_{22} & w_{23} & \dots & w_{2a} \\ \dots & \dots & \dots & & \dots \\ \dots & \dots & \dots & & \dots \\ w_{a1} & w_{a2} & w_{a3} & \dots & w_{aa} \end{bmatrix}$$

Selanjutnya akan dilakukan perhitungan untuk mencari nilai *Orthogonal Laplacianfaces* data sampel. Pertama, kolom – kolom dari matriks W dijumlah dan hasilnya dibentuk menjadi matriks diagonal yaitu matriks D dan nilai dari matriks W dimasukkan ke dalam matriks L. Setelah itu, nilai utama dari data baru yang didapatkan pada tahap pengurangan dimensi dengan PCA bisa dihitung. Dprime = X^T * D * X dan Lprime = X^T * L * X.

Sebuah matriks W_{OLPP} yang merupakan susunan dari vektor basis ortogonal dibentuk dengan cara:

- Menghitung e_1 yang merupakan vektor eigen dari $Dprime^{-1}$ * $Lprime^T$, yang memiliki nilai eigen terkecil. Hasilnya merupakan sebuah vektor e_1 berukuran 1 x c. Penghitungan didapat menggunakan bantuan *software* Matlab dengan menggunakan sintaks [eigenvector, eigenvalue] = eig(M).

$$W_{OLPP} = \begin{bmatrix} w_{11} \\ w_{21} \\ \dots \\ w_{c1} \end{bmatrix}$$

- Menghitung e_k yang merupakan vektor eigen dari $M^{(k)}$ = { I – $Dprime^{-1}$ * $Eigen_{OLPP}$ * [$Eigen_{OLPP}{}^T$ * $Dprime^{-1}$ * $Eigen_{OLPP}$]$^{-1}$ * $Eigen_{OLPP}{}^T$ } *

$Dprime^{-1}$ * $Lprime^{T}$, yang memiliki nilai eigen terkecil. Hasilnya merupakan sebuah vektor e_k berukuran 1 x c. Vektor e_k tersebut digabungkan ke dalam vektor $Eigen_{OLPP}$ yang sudah ada sebelumnya sehingga jumlah kolomnya bertambah. Hal ini dilakukan hingga terbentuk sebuah matriks $Eigen_{OLPP}$ berukuran c x c. Penghitungan didapat menggunakan bantuan *software* Matlab dengan menggunakan sintaks M = (I – invDprime * wOLPP * inv(transpose(wOLPP) * invDprime * wOLPP) * transpose(wOLPP)) * Q dan [eigenvector, eigenvalue] = eig(M).

$$W_{OLPP} = \begin{bmatrix} w_{11} & w_{12} & w_{13} & \dots & w_{1c} \\ w_{21} & w_{22} & w_{23} & \dots & w_{2c} \\ \dots & \dots & \dots & & \dots \\ \dots & \dots & \dots & & \dots \\ w_{c1} & w_{c2} & w_{c3} & \dots & w_{cc} \end{bmatrix}$$

Kemudian untuk mendapatkan nilai *Orthogonal Laplacianfaces*, matriks $Eigen_{PCA}$ yang sudah didapat sebelumnya harus dikalikan dengan matriks W_{OLPP}, $Eigen_{OLPP} = Eigen_{PCA} * W_{OLPP}$.

$$Eigen_{OLPP} = \begin{bmatrix} e_{11} & e_{12} & e_{13} & \dots & e_{1c} \\ e_{21} & e_{22} & e_{23} & \dots & e_{2c} \\ \dots & \dots & \dots & & \dots \\ \dots & \dots & \dots & & \dots \\ e_{b1} & e_{b2} & e_{b3} & \dots & e_{bc} \end{bmatrix}$$

Matriks $Eigen_{OLPP}$ yang didapat berukuran b x c (1024 x c). Setiap kolom matriks mewakili karakteristik citra awal, yang jika dikonversikan akan menjadi

citra yang dinamakan *Orthogonal Laplacianfaces*. Kemudian matriks $Eigen_{OLPP}$ dikalikan dengan matriks Tau sehingga didapatkan matriks Y yang merupakan nilai *Orthogonal Laplacianfaces* dari sampel, $Y = T * Eigen_{OLPP}$.

$$Y = \begin{bmatrix} Y_{11} & Y_{12} & Y_{13} & \dots & Y_{1(a-1)} \\ Y_{21} & Y_{22} & Y_{23} & \dots & Y_{2(a-1)} \\ \dots & \dots & \dots & & \dots \\ \dots & \dots & \dots & & \dots \\ Y_{a1} & Y_{a2} & Y_{a3} & \dots & Y_{a(a-1)} \end{bmatrix}$$

3.2.4 Tahap Pemberian Identitas

Pemberian identitas dilakukan secara manual pada *source code* program. Jika indeks gambar yang dihasilkan kurang dari jumlah gambar tiap subyek maka akan diberi identitas 1, jika indeks gambar lebih besar dari jumlah gambar tiap subyek namun lebih kecil dari atau sama dengan k (dimulai dengan nilai 2) kali jumlah pola maka akan diberi identitas k dan seterusnya.

No.	Nama File	Sampel Wajah	Identitas Subyek
1.	1.bmp s/d 10.bmp		1
2.	11.bmp s/d 20.bmp		2
3.	21.bmp s/d 30.bmp		3
4.	31.bmp s/d 40.bmp		4
5.	41.bmp s/d 50.bmp		5

6.	51.bmp s/d 60.bmp		6
7.	61.bmp s/d 70.bmp		7
8.	71.bmp s/d 80.bmp		8
9.	81.bmp s/d 90.bmp		9
10.	91.bmp s/d 100.bmp		10
11.	101.bmp s/d 110.bmp		11
12.	111.bmp s/d 120.bmp		12
13.	121.bmp s/d 130.bmp		13
14.	131.bmp s/d 140.bmp		14
15.	141.bmp s/d 150.bmp		15
16.	151.bmp s/d 160.bmp		16
17.	161.bmp s/d 170.bmp		17
18.	171.bmp s/d 180.bmp		18
19.	181.bmp s/d 190.bmp		19
20.	191.bmp s/d 200.bmp		20
21.	201.bmp s/d 210.bmp		21
22.	211.bmp s/d 220.bmp		22
23.	221.bmp s/d 230.bmp		23

24.	231.bmp s/d 240.bmp		24
25.	241.bmp s/d 250.bmp		25
26.	251.bmp s/d 260.bmp		26
27.	261.bmp s/d 270.bmp		27
28.	271.bmp s/d 280.bmp		28
29.	281.bmp s/d 290.bmp		29
30.	291.bmp s/d 300.bmp		30
31.	301.bmp s/d 310.bmp		31
32.	311.bmp s/d 320.bmp		32
33.	321.bmp s/d 330.bmp		33
34.	331.bmp s/d 340.bmp		34
35.	341.bmp s/d 350.bmp		35

Tabel 3.2 Pemberian identitas pada citra wajah

3.2.5 Tahap Pengenalan

Pada tahap pengenalan, citra wajah yang baru dijadikan sebuah vektor X yang berukuran 1 x b (1 x 1024). Vektor X ini dikalikan dengan matriks F yang didapat pada tahap perhitungan *Orthogonal Laplacianfaces*, New_X = X * $\text{Eigen}_{\text{OLPP}}$, New_X adalah nilai X baru dalam *Orthogonal Laplacianfaces*.

Didapat sebuah vektor New_X dengan ukuran 1 x c. Vektor tersebut kemudian dihitung jaraknya menggunakan metode *Nearest-neighbor Classifier* dengan setiap vektor citra pada *training set* dengan rumus

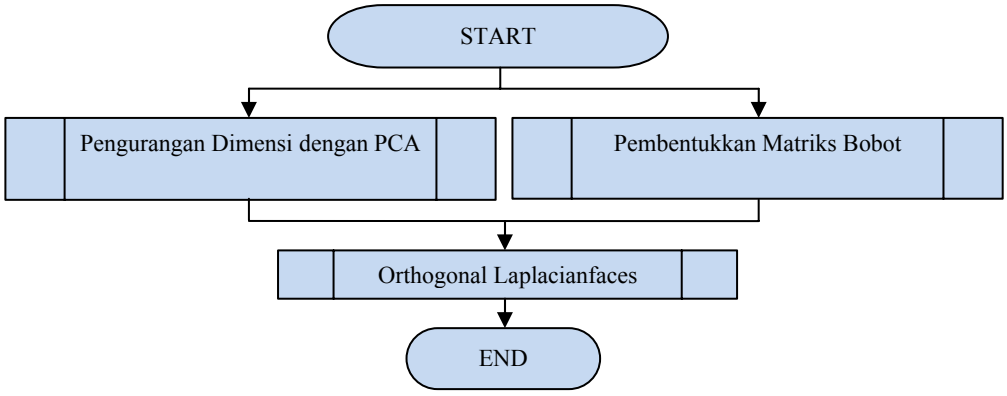
$d_{(x,y)} = \sqrt{\sum_{i=1}^{a-1} \left(new\_imageYi - Yi\right)^2}$, $d_{(x,y)}$ adalah jarak antar dua citra.

Pengenalan citra baru dari citra pada *training set* ditandai dengan nilai $d_{(x,y)}$ paling kecil.

3.3 Diagram Alir Modul

3.3.1 Diagram Alir Modul Perhitungan Orthogonal Laplacianfaces

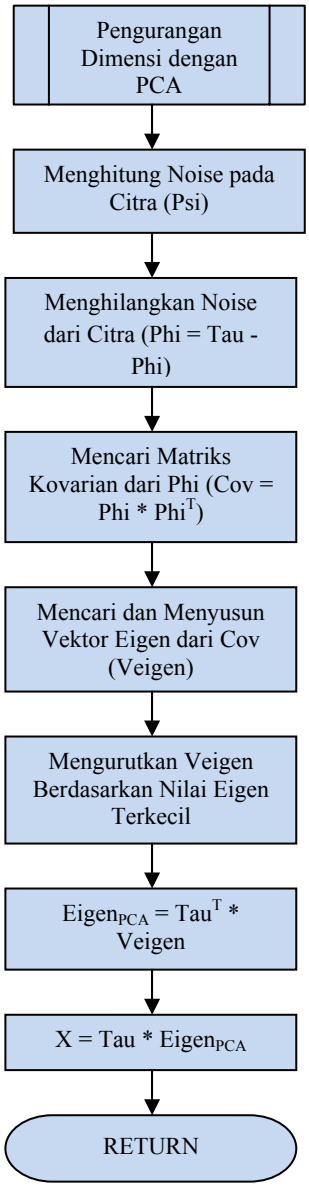
Berikut ini adalah diagram alir modul utama perhitungan *Orthogonal Laplacianfaces*, dibagi menjadi tiga buah diagram alir lagi yang menggambarkan tiga buah proses didalamnya, yaitu pengurangan dimensi dengan PCA, pembentukkan matriks bobot, dan *Orthogonal Laplacianfaces*.



Gambar 3.1 Diagram alir modul perhitungan Orthogonal Laplacianfaces

3.3.1.1 Diagram Alir Pengurangan Dimensi dengan PCA

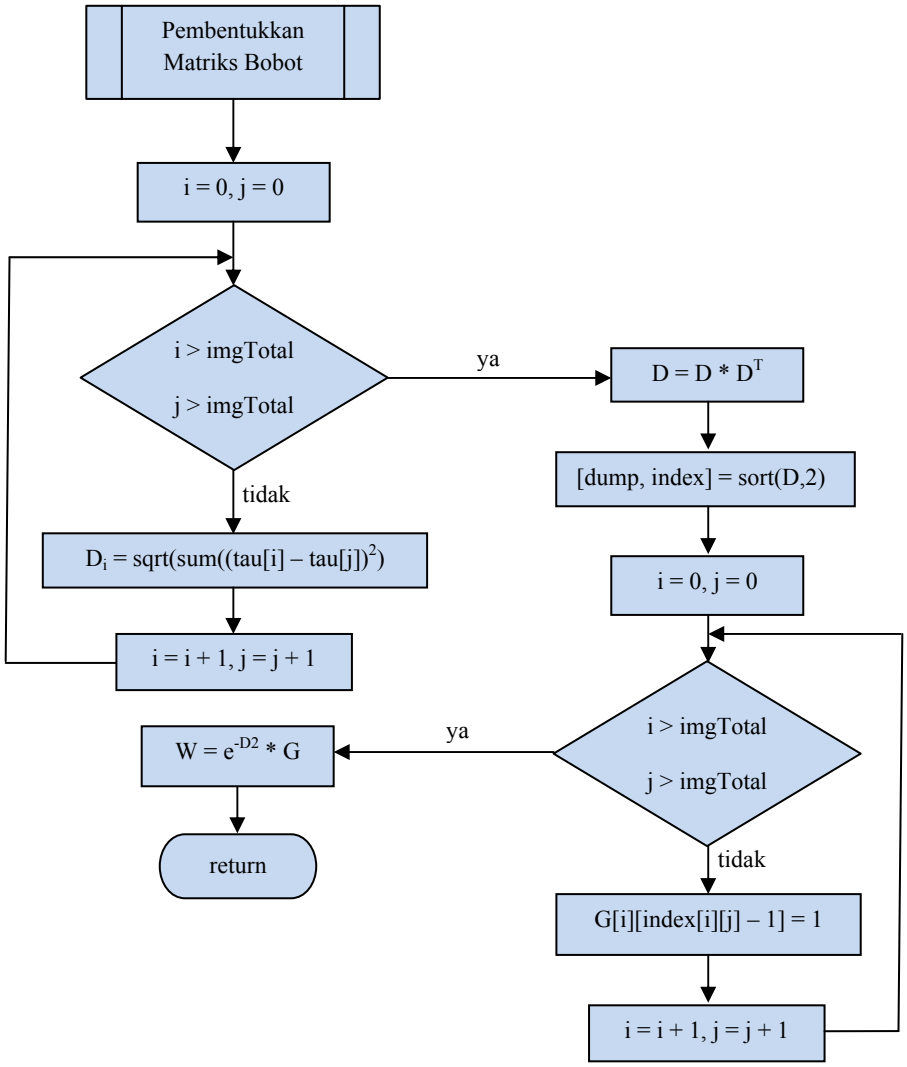
Berikut ini adalah diagram alir perhitungan *Orthogonal Laplacianfaces* pada bagian pengurangan dimensi dengan PCA sampai tahap perhitungan data baru untuk diproses pada langkah pembentukkan matriks bobot.



Gambar 3.2 Diagram alir pengurangan dimensi dengan PCA

3.3.1.2 Diagram Alir Pembentukan Matriks Bobot

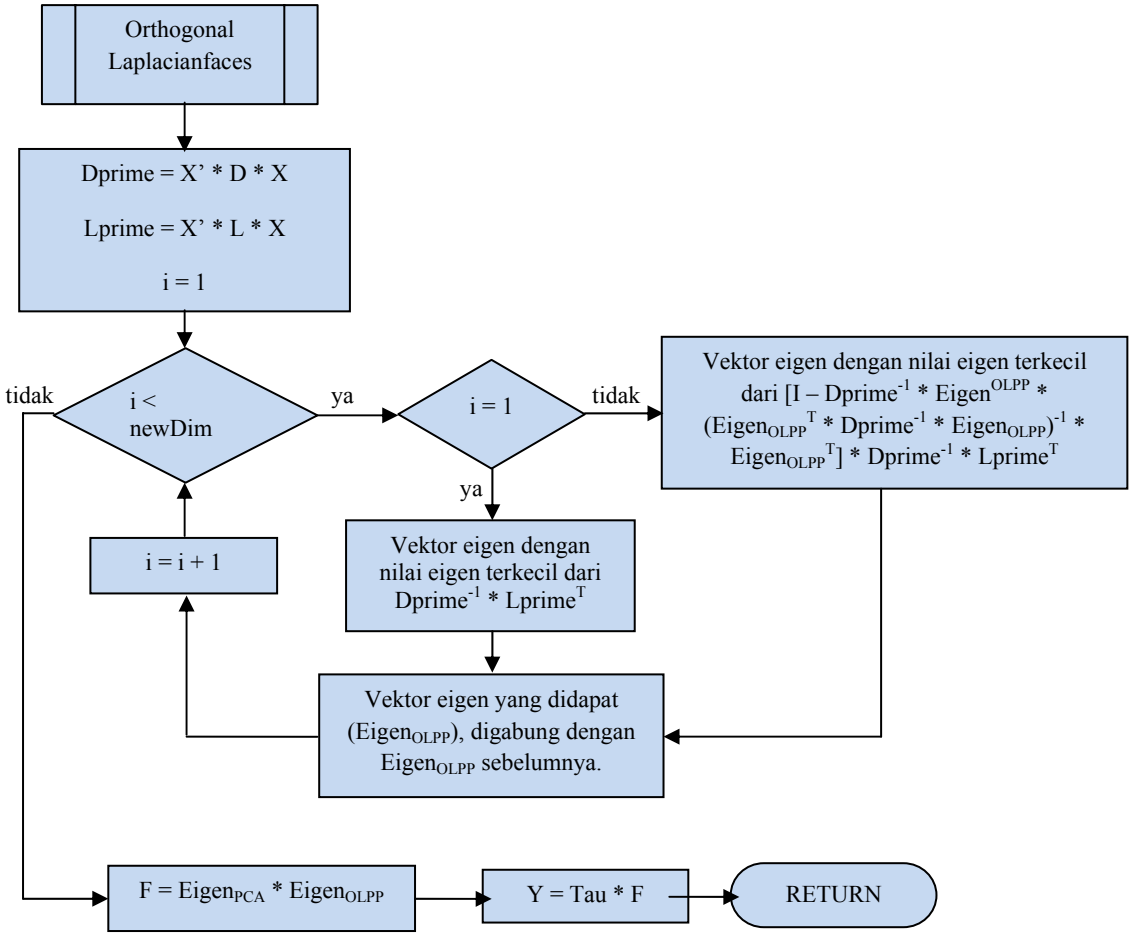
Berikut ini adalah diagram alir pembentukkan matriks bobot yang akan digunakan pada tahap *Orthogonal Laplacianfaces*.



Gambar 3.3 Diagram alir pembentukan matriks bobot

3.3.1.3 Diagram Alir Orthogonal Laplacianfaces

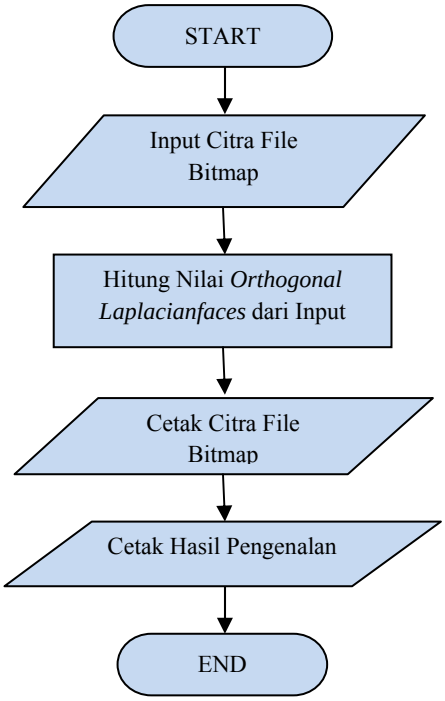
Berikut ini adalah diagram alir *Orthogonal Laplacianfaces*. Data yang dihasilkan pada modul ini akan digunakan pada proses pengenalan.



Gambar 3.4 Diagram alir modul Orthogonal Laplacianfaces

3.3.2 Diagram Alir Modul Pengenalan Citra Baru

Berikut ini adalah diagram alir modul pengenalan citra baru yang digunakan untuk mencocokkan citra baru dengan citra pada data training yang telah diproses dengan metode Orthogonal Laplacianfaces.



Gambar 3.5 Diagram alir modul pengenalan citra baru