BAB 3

ANALSIS DAN PERANCANGAN SISTEM

3.1 Analisis Sistem

Analisis sistem merupakan tahap yang bertujuan untuk memahami sistem, mengetahui kekurangan sistem, dan menentukan kebutuhan hasil proses pada perangkat lunak pengendali pointer. Dengan menganalisis prosedur sistem yang digunakan dan melakukan pengujian hasil, maka sistem dapat dievaluasi sehingga dapat dijadikan sebagai acuan untuk dalam proses pembentukan kesimpulan.

Perangkat lunak yang sudah ada contohnya adalah *Viacam* , perangkat lunak ini harus memilih fungsi terlebih dahulu untuk melakukan proses klik kiri maupun kanan sehingga membuat pengguna masih memerlukan mouse untuk memilih fungsi tersebut. Perangkat lunak lain yang menggunakan kepala adalah *HeadMouse*, perangkat lunak ini tidak dapat melakukan proses klik. Sehingga yang menjadi tujuan dari pembuatan perangkat lunak ini adalah untuk mengimplementasikan *haarcascade* dan metode Fisherface (FLDA) pada sistem pengenalan pola wajah untuk menggerakkan pointer *mouse* dan dapat melakukan proses klik. Kemudian, menguji performa dan kemampuan metode Fisherface dalam melakukan pengenalan pola wajah dan gerak yang telah dikondisikan sesuai kebutuhan.

Dan pada akhirnya *software* ini akan menghasilkan sebuah hasil yang dapat dianalisis kemudian untuk dibuat kesimpulan dan rekomendasi, sehingga dapat diketahui keefektifitasan perangkat lunak yang akan di bangun dan juga dapat

digunakan dalam kondisi yang telah ditentukan dan dalam ruang lingkup batasan masalah.

3.1.1 Analisis Masalah

Berdasarkan hasil analisis terhadap sistem yang ada, maka diidentifikasikan masalah-masalah yang ada sebagai berikut :

1. Metode Fisher's *Linear Discriminant Analysis* (FLDA) harus dapat diimplementasikan kedalam sistem perangkat lunak.
2. Performa dari metode yang diterapkan harus dapat diuji dengan kondisi yang telah ditentukan.
3. Sistem perangkat lunak harus dapat digunakan oleh user yang dituju yaitu para penyandang cacat pada bagian tangan.
4. Harus dapat menarik kesimpulan apakah sistem perangkat lunak yang dibangun dapat bermanfaat atau tidak bagi para pengguna.

3.1.2 Analisis Algoritma

Analisis algoritma dilakukan untuk dapat mengetahui alur proses dari algoritma yang digunakan dan dapat diterapkan kedalam sistem perangkat lunak. Analisis yang pertama kali dilakukan adalah analisis deteksi obyek yang akan dibagi menjadi 5 tahapan, yaitu :

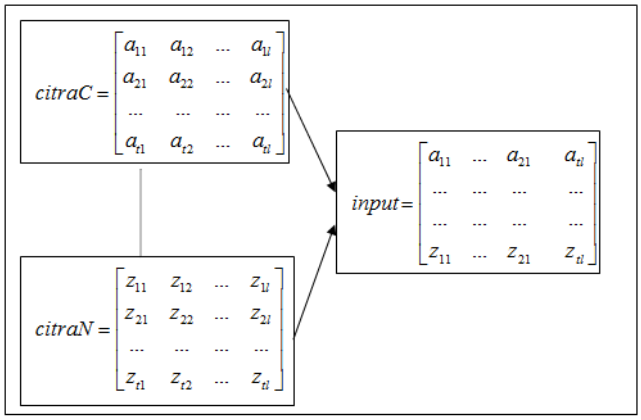
1. Tahap pertama adalah tahap pengambilan gambar secara *real time* yang kemudian akan di ubah menjadi gambar digital.
2. Tahap kedua adalah tahap pengenalan pola mata yang terdiri dari proses penskalaan (*scaling), grayscale* dan *tresholding.*

3. Tahap ketiga adalah integral proyeksi untuk mencari daerah lokasi dari mata.
4. Tahap keempat adalah tahap pengambilan nilai *training* menggunakan metode *haar cascade.*
5. Tahap kelima adalah proses pendeteksian wajah dan indera penglihatan menggunakan metode Fisher's *Linear Discriminant analisys* (FLDA).

3.1.2.1 Metode FLDA (Fisher's Linear Discriminant Analisys)

Dalam proses pengenalan pola wajah menerima masukan berupa citra bitmap (hasil *capture*) yang diubah dalam format *grey-level*. Kumpulan citra yang menjadi masukan pada proses pencarian matriks transformasi disebut *training set*, dan N = jumlah citra *training set*. Kumpulan citra yang menjadi masukan uji pengenalan disebut *testing set*, dan T = jumlah citra *testing set*.

Citra masukan dipetakan kedalam nilai-nilai bertipe *float*, yang merepresentasikan tingkat ke-abuan dari *pixel* pada posisi yang bersesuaian. Nilai-nilai ke-abuan dari suatu citra kemudian disimpan dalam bentuk vektor baris, yang disebut vektor citra. Gambar 3.1 menggambarkan penyimpanan vektor-vektor citra kedalam matriks *input* berdimensi N x n.



Gambar 3. 1 Proses pemetaan citra

Algoritma untuk mengenali pola wajah pada metode FLDA adalah algoritma Fisherface, algoritma fisherface merupakan turunan dari metode FLDA yang akan dijelaskan sebagai berikut :

Langkah 1 : Cari rata-rata $input_{FLD}$

Untuk i = 1 … N

$$\mu_{FLD} = \frac{1}{N}\sum_{i=1}^{N} x'_i \qquad (1*N) \qquad (3.1)$$

Penjelasan : Nilai dari N dicari dengan cara mencari determinannya yang akan menghasilkan nilai bertipe float kemudian akan diolah untuk menentukan pixel gambar.

Berikut ini adalah contoh perhitungan :

$$Jika, N = \begin{bmatrix} 6 & 2 \\ 8 & 4 \end{bmatrix}, maka: \mu_{FLD} = \frac{1}{\det(N)} = \frac{1}{4.6-(2.8)}$$

$$= \frac{1}{8} = 0.12$$

Langkah 2 : Untuk j = 1..N , cari rata-rata kelas X_i

$$\mu_i = \frac{1}{N_i} \sum_{j=1, x'_j \in X_i}^{N_i} x'_j \qquad \text{(N-C)} \qquad (3.2)$$

dimana N_i adalah jumlah anggota kelas X_i.

Simpan μ_i, i = 1…N, dalam matriks μ_K, sehingga :

$$\mu_K = \begin{bmatrix} \mu_1 \\ \mu_2 \\ \dots \\ \mu_c \end{bmatrix} \qquad \text{(C*(N-C))} \qquad (3.3)$$

Langkah 3 : Cari matriks *between-class scatter*, S_B

$$S_B = \sum_{i=1}^{C} N_i (\mu_i - \mu_{FLD})^T (\mu_i - \mu_{FLD}) \qquad (3.4)$$

((N-C)*C)* (C*(N-C)) = ((N-C)*(N-C))

Penjelasan : Nilai μ_i Hasil perhitungan nilai rata2 image masukan yang ada

Langkah 4 : Cari matriks *within-class scatter*, S_W

$$S_W = \sum_{i=1}^{C} \sum_{j=1, x'_j \in X_i}^{N_i} (x'_j - \mu_i)^T (x'_j - \mu_i) \qquad (3.5)$$

((N-C)*N_i)* (N_i*(N-C)) = ((N-C)*(N-C))

Langkah 5 : Cari rasio antara S_B dengan S_W

$$rasio = \frac{S_B}{S_W} \qquad \text{((N-C)*(N-C))} \qquad (3.6)$$

Langkah 6 : Cari vektor fisher dan nilai fisherface dari matriks *rasio*, kemudian urutkan berdasarkan nilai fisherface (dari yang terbesar)

[v,d] = *fisher*(*rasio*) (3.7)

Nilai fisher dari *rasio* :

$$d = \begin{bmatrix} d_1 & d_2 & \dots & d_{(N-C)} \end{bmatrix} \qquad (1*(N-C)) \tag{3.8}$$

dimana $d_1 > d_2 > \dots > d_{(N-C)}$.

Vektor fisher yang bersesuaian dengan nilai fisherface (dalam bentuk vektor kolom) :

$$v = \begin{bmatrix} w_{11} & w_{12} & \dots & w_{1(N-C)} \\ w_{21} & w_{22} & \dots & w_{2(N-C)} \\ \dots & \dots & \dots & \dots \\ w_{(N-C)1} & w_{(N-C)2} & \dots & w_{(N-C)(N-C)} \end{bmatrix} ((N-C)*(N-C)) \tag{3.9}$$

Langkah 7 : Tentukan dimensi ruang ciri (vektor fisher yang efektif untuk pengenalan : maksimal C)

$$m = \min_r \left\{ \frac{\sum_{i=1}^{r} d_i}{\sum_{i=1}^{C} d_i} > \theta \right\} \tag{3.10}$$

Langkah 8 : Reduksi vektor fisher dari *rasio* sesuai dengan nilai m

$$v' = \begin{bmatrix} w_{11} & w_{12} & \dots & w_{1m} \\ w_{21} & w_{22} & \dots & w_{2m} \\ \dots & \dots & \dots & \dots \\ w_{(N-C)1} & w_{(N-C)2} & \dots & w_{(N-C)m} \end{bmatrix} ((N-C)*m) \tag{3.11}$$

Matriks transformasi $W_{fisherface}$ adalah :

$$W_{fisherface} = v'^{T} \qquad (m*(N-C)) \tag{3.12}$$

Langkah 9 : Transformasi vektor masukan dimensi-(N-C) menjadi vektor ciri dimensi-m

$$\text{Transformasi} = Input_{FLD} \, W_{fisherface}^{T} \tag{3.13}$$

(N*(N-C))*((N-C)*m) = (N*m)

Atau untuk i = 1…N :

$$y_i = x'_i W_{FLD}^{T} \quad (3.14)$$

$$(1*n)*(n*(N-C)) = (1*(N-C))$$

Psuedo Code Metode FLDA

```
Procedure metode_FLDA
{IS : objek gambar dan nilai fisherface,serta elemen matriks sudah
terdefinisi }
{FS : objek gambar dirubah menjadi vektor ciri }
Kamus :
Nomer_kelas     : real
Fisherface      : real
V_Fisher        : real
Jumlah_kelas    : Array [1…N,1…Z] of integer
A,L,C,A',P      : matriks
N,Z             : integer
Gbr_training    : real
Algoritma :
For N ← 1 to Z do
      For Z ← 1 to M do
      Write (Jumlah_kelas[N,Z])
      Endfor
Endfor
Jumlah_kelas ← [N,Z]      {Jumlah gambar dari kelas atau N}
Nomer_kelas  ← (Fisherface /2 ) / 2     { nomer kelas}
P ← Jumlah_kelas  x  Nomer_kelas   {jumlah gambar training}
m_datagambar   ← Fisherface / Jumlah_kelas   {menghitung rata2}
         A ← Fisherface - (m_datagambar x 1 x P)     { selisih dari
rata2}
         L ← A' x A      { nilai A' / A transpose sudah terdefinisi}
         C ← A x A'     { matriks covariance}
EndProcedure

Procedure menghitung_garis_diagonal_eigen
{IS : matriks eigen dan covariance sudah terdefinisi }
{FS : mencari vektor eigen value dari metode FLDA }
Kamus :
D     :array [1..N][1..Z] of integer {Garis diagonal dari nilai eigen
L dan covariance C}
P     :matriks
V_FLDA : Array [1…N,1…Z] of integer
ProyeksiGambar_FLDA : real
Z     : integer
Algoritma
for i ← 1 to P do
          L_eig_vec ← [D / i];
endfor
V_FLDA ← A  x  L_eig_vec     {menghitung nilai eigen matriks covariace
'C'}
{nilai V_FLDA'adalah V_FLDA transpose, Nilai sudah terdefinisi}
Z ← V_FLDA'  x  (fisherface - m_datagambar)   {proyeksi menjadi
eigenspace}
```

```
      ProyeksiGambar_FLDA ← [ ];
for i ← 1 to P do
      temp ← V_FLDA' x A
      ProyeksiGambar_FLDA ← temp
Endfor
Endprocedure


Procedure menghitung_nilai_rata2
{IS : nilai proyeksi pca sudah terdefinisi }
{FS : mencari nilai rata2 dari fisherspace }
Kamus :
S ,Sw2,Sw1, Sb2,Sb1, m_FLDA,m_datagambar : array of integer
ProyeksiGambar_FLDA : real
Algoritma
m_FLDA ← ProyeksiGambar_FLDA/2  {Total rata2 di fisherspace}
m ←P / Nomer_kelas
Sw1 ← P      {memulai Scatter Matrix}
Sb1 ←P- Nomer_kelas      {memulai Between Scatter Matrix}
for i ← 1 to Nomer_kelas do
    m ← ((ProyeksiGambar_FLDA(i-1) x Jumlah_kelas+1) / i x
Jumlah_kelas )
    for j ← ( (i-1) x Jumlah_kelas+1 ) to ( i x Jumlah_kelas ) do
    S ← Sb1 + (ProyeksiGambar_FLDA-m) x (ProyeksiGambar_FLDA-m)
endfor
    Sw2 ← Sw1 + S   {Within Scatter Matrix}
    Sb2 ← Sb1 + (m-m_FLDA)  x  (m-m_FLDA)   { Between Scatter Matrix}
Endprocedure

Procedure menghitung_nilai_proyeksi_metode_fisher
{IS : nilai proyeksi pca sudah terdefinisi }
{FS : mencari nilai proyeksi dari metode FLDA }
Kamus :
J_eig_vec, J_eig_val ,S ,Sw2 , Sb2 , m_FLDA,m_datagambar : array of
integer
ProyeksiGambar_FLDA : real
Jumlah_kelas   : Array [1…N,1…n] of integer
Nomer_kelas    : real
Y : Integer
Algoritma
J ←  Invers Sw2
J_eig_vec ← J x Sb2
for i ← 1 to Nomer_kelas-1 do
    V_Fisher← J_eig_vec x I  {mariks fisher vectors yang terbesar dari
J}
endfor
Y ← V_Fisher  x  V_FLDA  x  (Ti - m_datagambar)
for i ← 1 to Nomer_kelas x Jumlah_kelas do
    ProyeksiGambar_Fisher = V_Fisher  x  ProyeksiGambar_FLDA
Endfor
Endprocedure
```

3.1.2.2 Analisis Library

Berikut ini adalah tabel pemakaian library pada prosedur yang digunakan dalam perangkat lunak yang dibangun :

Tabel 3. 1 Tabel Analisis library

NO	Prosedur	Library
1	Pendeteksian Wajah	haarcascade_frontalface_alt.xml
2	Pendeteksian Indera penglihatan	haarcascade_eye.xml
3	Pendeteksian klik pada pointer dan pendeteksian gerakan pada pointer	Fisherface.xml

3.1.2.3 Proses Pengambilan Gambar

Pada analisis ini inputan yang digunakan adalah gambar langsung yang dikalibrasi oleh *webcam* secara *real time* dan hasilnya berupa gambar digital yang akan digunakan untuk tahap berikutnya. Proses kalibrasi gambar secara *real time*:

1. Mengidentifikasi sejumlah titik koordinat acuan dengan syarat koordinat gambar pada kamera dan koordinat pada gambar *real time* juga diketahui, sehingga :

$$\mathbf{U = u/t}$$
$$\mathbf{V = v/t}$$

Dimana : U = koordinat gambar pada sumbu x

V = koordinat gambar pada sumbu y

u = koordinat homogenus pada sumbu x

v = koordinat homogenus pada sumbu y

t = koordinat homogenus pada sumbu z

2. Konversi koordinat gambar menjadi koordinat *real time* dengan matriks 3 x 4 berikut :

$$(x, y, z, 1)\begin{bmatrix} a1 & b1 & c1 \\ a2 & b2 & c2 \\ a3 & b3 & c3 \\ a4 & b4 & b4 \end{bmatrix} = (u, v, t)$$

Jika dilanjutkan maka akan diperoleh :

$$x.a1 + y.a2 + z.a3 + a4 = u$$
$$x.b1 + y.b2 + z.b3 + b4 = v$$
$$x.c1 + y.c2 + z.c3 + c4 = t$$

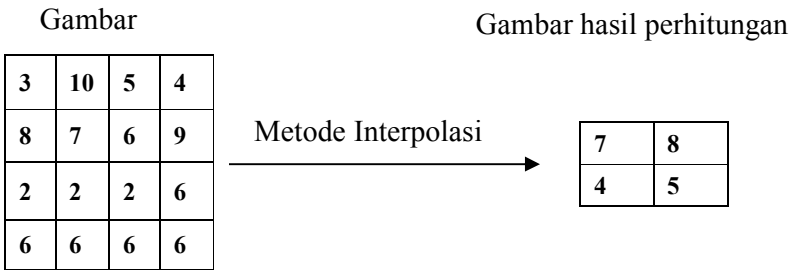
Bila keempat titik ini dikumpulkan, nilai – nilai di dalam matriks dapat diselesaikan menggunakan persamaan simultan dan didapatlah estimasi nilai – nilai pada matriks dengan koordinat (U,V).

3.1.2.4 Analisis Pengenalan Pola Wajah dan Indera Penglihatan

Dalam analisis deteksi obyek ini dibagi menjadi 3 tahap, tahap pertama yaitu penskalaan(*scaling*), tahap kedua proses *grayscaling*, dan tahap ketiga proses *tresholding.*

1. **Tahap Penskalaan (*Scaling)***

   Gambar digital yang diperoleh dari proses kalibrasi *webcam* dengan pengguna akan diperkecil skalanya dengan menggunakan metode interpolasi. Metode ini mengambil nilai rata – rata dari suatu daerah untuk mewakili daerah tersebut di dalam gambar yang lebih kecil ukurannya.



Gambar 3. 2 Metode interpolasi untuk memperkecil gambar

Nilai 7 pada koordinat kiri atas pada gambar hasil perhitungan metode interpolasi diperoleh dengan cara menghitung nilai rata – rata dari 4 nilai piksel pada pojok kiri atas gambar asli, yaitu (3+10+8+7)/4 = 7. Aturan ini juga diberlakukan untuk nilai koordinat piksel gambar yang lain pada gambar asli ketika proses *scaling* selesai maka gambar akan digunakan dalam proses berikutnya.

Berikut ini adalah contoh gambar hasil interpolasi :

Gambar Asli 400 x 200 *pixel*



Hasil *Scaling* 200 x 100 *pixel*



Gambar 3. 3 Penskalaan gambar menggunakan metode interpolasi

2. Tahap *Grayscaling*

Gambar digital yang telah melalui proses *scaling* kemudian diubah menjadi gambar dua warna dengan proses *grayscaling*. Proses *grayscaling* dilakukan oleh sistem perangkat lunak.

Berikut ini adalah gambar hasil *grayscaling* :

Gambar Asli **Gambar G*rayscale***



Gambar 3. 4 Proses pengubahan gambar RGB menjadi gambar *grayscale*

3. Tahap *Tresholding*

Selanjutnya adalah tahap *tresholding* yang digunakan untuk mengubah gambar hasil *grayscale* menjadi gambar biner. Tahap ini dilakukan karena sistem komputer hanya dapat membaca nilai biner dari setiap data yang diinputkan dari pengguna kedalam sistem.

Gambar Grayscale **Gambar Biner**



Gambar 3. 5 Proses pengubahan gambar *grayscale* menjadi gambar *biner*

3.1.2.5 Deteksi Wajah dan Indera Penglihatan

Metode yang digunakan untuk mendeteksi wajah dan indera penglihatan adalah metode Fisher's *Linear Discriminant Analisys* (FLDA) yang sebelumnya telah didapat nilai *training* dari *haar cascade classifier*. Metode ini menggunakan *statistical model* (*classifier*).

Tahap pendeteksian wajah dan indera penglihatan adalah sebagai berikut :

1. **Pengambilan Nilai *Haar training***

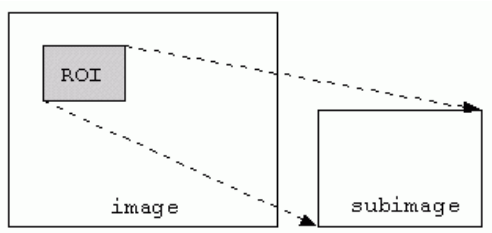
   Nilai *training* pada openCV yang dikenal dengan *Haar Training*. Proses *training* ini dilakukan untuk mendapatkan sampel dari gambar yang didapat dari kamera dan akan dibandingkan dengan nilai yang ada pada OpenCV. Kemudian nilai *frame* akan digunakan untuk mendeteksi apakah objek gambar yang masuk wajah atau bukan.

2. **Cara Kerja Metode**

   Cara kerja metode Haar Cascade Classifier untuk mendeteksi indera penglihatan adalah diawali dari pengambilan gambar yang dilakukan secara langsung melalui *webcam* kemudian dilakukan proses *tracking* untuk penentukan apakah objek yang ada adalah bentuk dari kepala atau bukan, setelah mendapatkan bentuk objek dan memastikan bahwa objek tersebut adalah kepala maka dilakukan pendeteksian objek wajah dan indera penglihatan . Sehingga didapatlah hasil berupa deskripsi gambar yang kemudian akan diolah menggunakan metode fisher's *linear discriminant analisys* (FLDA).

a. Menentukan ROI (*Region Of Interest*) Pada Gambar

Region Of Interest adalah suatu daerah persegi panjang yang digunakan untuk menentukan daerah atau posisi objek yang diperlukan untuk proses selanjutnya pada sistem. Contohnya adalah pengambilan daerah posisi indera penglihatan pada objek kepala.



Gambar 3. 6 *Region of Interest*

Setelah mendapatkan Nilai *Region Of Interest* maka perangkat lunak akan mendapatkan posisi atau letak objek kemudian nantinya akan digunakan untuk proses pengendalian *pointer* dan juga pengguna dapat melakukan klik kiri maupun klik kanan.

b. Mendeteksi Wajah

Fungsi utama cukup sederhana. Menginisialisasi beberapa variabel, memasukkan gambar, mengubah gambar menjadi *grayscale*, membuat fungsi *detecteyes* dan menampilkan gambar.



Gambar 3. 7 Gambar untuk deteksi wajah

Kemudian gambar masukan diubah menjadi gambar *grayscale* menggunakan fungsi yang ada pada openCV:

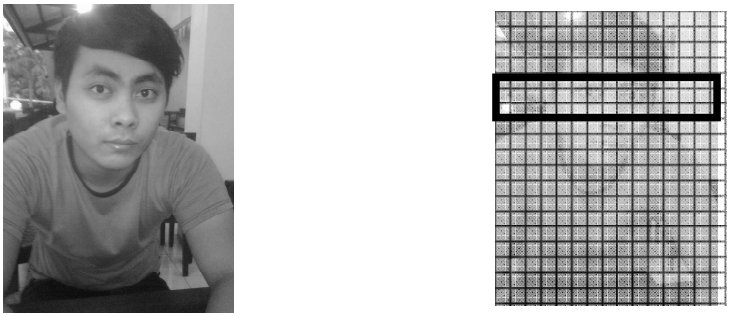


Gambar 3. 8 Gambar wajah yang telah melaui tahap *grayscale*

c. Integral Proyeksi

Setelah wajah terdeteksi, kemudian dilakukan proses integral proyeksi untuk menentukan daerah lokasi dari mata. Di sini diperkirakan lokasi mata 1/3 dari puncak wajah. Pencarian obyek dimulai pada sub window dengan ukuran skala 24x24 pada seluruh daerah gambar grayscale. Pencarian obyek diulangi kembali dengan ukuran sub window yang telah diskala ulang.

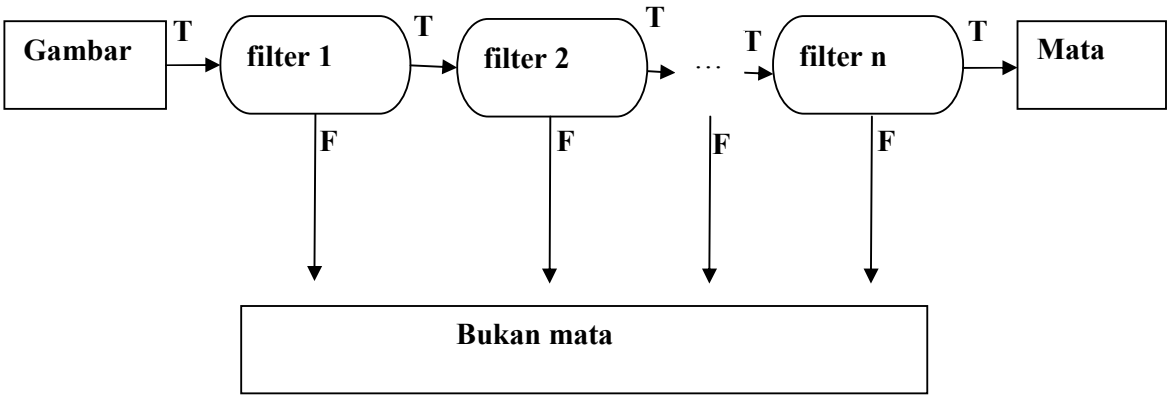
Gambar Grayscale **Perkiraan Daerah Lokasi Mata**



Gambar 3. 9 Perkiraan daerah lokasi mata

d. Mendeteksi Mata

Setelah menentukan daerah lokasi dari mata kemudian dilakukan pencarian obyek mata. Pencarian obyek mata pada suatu sub window berdasarkan data XML Classifier (*haarcascade_eye.xml*) yang telah dibuat (*cascade classifier*). Untuk membuat *cascade classifier* digunakan metode *gentle adaboost*, yaitu dengan cara mencari fitur-fitur yang memiliki tingkat pembeda yang tinggi. Hal ini dilakukan dengan mengevaluasi setiap fitur terhadap data latih dengan menggunakan nilai dari fitur tersebut. Fitur yang memiliki batas terbesar antara mata dan bukan mata dianggap sebagai fitur terbaik.



Gambar 3. 10 *Cascade Classifier*

Pada klasifikasi tingkat pertama (filter 1), tiap subgambar akan diklasifikasi menggunakan satu fitur. Klasifikasi ini kira-kira akan menyisakan 50% subgambar untuk diklasifikasi di tahap kedua (tahap filter 2). Seiring dengan bertambahnya tingkatan klasifikasi, maka

diperlukan syarat yang lebih spesifik sehingga fitur yang digunakan menjadi lebih banyak. Jumlah subgambar yang lolos klasifikasi pun akan berkurang hingga mendekati gambar yang ada pada sample (data .xml) atau hingga klasifikasi menyisakan 2% subgambar.

e. *Integral Image*

Kotak Haar-like feature dapat dihitung secara cepat menggunakan "*integral image*" nilai dari *feature* dihitung menggunakan integral image pada gambar dibawah ini :



Gambar 3. 11 Deskripsi haar feature

Dibandingkan dengan nilai *treshold* (-0.0202897991985083), jika lebih kecil maka mengembalikan nilai *left_value* (0.5025501251220703); dan jika lebih besar maka mengembalikan nilai right_value (-0.4522908926010132). Kemudian memproses *feature* berikutnya sampai *feature* terakhir. Jika total kembalian untuk semua feature lebih kecil dari stage_threshold (-1.3585350513458252), maka sub window ditolak (*rejected*), jika tidak maka dilakukan pemrosesan stage berikutnya.

3.1.2.6 Analisis *Tracking* wajah dan indera penglihatan

Tracking wajah dan indera penglihatan dilakukan untuk proses pergerakan dan proses klik pada *pointer mouse*. Tahap ini dilakukan menggunakan algoritma

template machine. Gambar objek dari *webcam* akan dibandingkan dengan gambar objek yang sudah disimpan dalam sebuah file, dalam hal ini disebut metode prinsip pecocokan bentuk atau metode *matching template*. Dalam *OpenCV* digunakan *cvnorm* yaitu mencari nilai perubahan bentuk membandingkan gambar objek dalam bentuk gray yang di-*capture* dari *webcam* dengan gambar mata dalam bentuk *gray* yang disimpan sebagai *file*, sehingga didapatkan arah posisi wajah ke arah kanan, ke kiri, ke atas, maupun ke bawah.

3.1.2.7 Analisis Deteksi Kedipan

Pada tahap ini sistem akan mendeteksi kedipan mata pengguna untuk digunakan sebagai proses klik pada *pointer*. Deteksi kedipan dilakukan dengan cara memberikan nilai *grayscale* dari gambar mata, kemudian dilakukan proses *tresholding* untuk mendapatkan gambar biner, sehingga dapat membedakan perubahan piksel yang terjadi pada gambar.

3.1.3 Spesifikasi Kebutuhan Perangkat Lunak

Spesifikasi kebutuhan perangkat lunak yang akan dibangun berdasarkan kebutuhan pengguna pada jurnal dan artikel yang ada. Spesifikasi kebutuhan perangkat lunak akan dibagi kedalam dua bagian yaitu SKPL-F (Spesifikasi kebutuhan perangkat lunak fungsional) dan SKPL-NF (Spesifikasi kebutuhan perangkat lunak non-fungsional). Berikut ini adalah tabel dari Spesifikasi kebutuhan perangkat lunak pengendali pointer mouse :

Tabel 3. 2 Spesifikasi kebutuhan perangkat lunak fungsional

Kode	**Kebutuhan**
SKPL-F001	Sistem harus dapat melakukan kalibrasi dengan

	webcam
SKPL-F002	Sistem harus dapat menggerakan pointer mouse dengan wajah
SKPL-F003	Sistem harus dapat melakukan proses klik pada pointer mouse dengan indera penglihatan
SKPL-F004	Sistem menyediakan tampilan profil tentang pembuat
SKPL-F005	Sistem menyediakan fungsi pengaturan cahaya pada *webcam*

Tabel 3. 3 Spesifikasi kebutuhan perangkat lunak non-fungsional

SKPL-NF001	Pengguna atau *user* yang menggunakan perangkat lunak ini adalah *user* yang mengalami cacat fisik pada bagian motorik , khususnya pada bagian tangan
SKPL-NF002	Pengguna atau *user* menggunakan bagian wajah dan indera penglihatan untuk menjalankan perangkat lunak
SKPL-NF003	Kondisi lingkungan pengguna harus dalam keadaan cukup terang agar memudahkan proses kalibrasi
SKPL-NF004	Perangkat keras yang digunakan adalah Komputer dengan spesifikasi minimal Processor 2.00 GHz dan memory 2 GB serta *webcam* minimal 1.3 Megapixel
SKPL-NF005	Perangkat lunak ini akan dibangun dengan menggunakan bahasa pemrograman C#

3.1.4 Analisis Kebutuhan Non Fungsional

Analisis kebutuhan non fungsional digunakan untuk memastikan bahwa perangkat lunak yang dibuat sesuai dengan kebutuhan pengguna dan layak untuk dikembangkan. Tahapan analisis kebutuhan non fungsional dibagi menjadi beberapa tahap yaitu analisis perangkat pikir , perangkat keras , dan perangkat lunak. Tahapan-tahapan ini sangat penting untuk membangun perangkat lunak agar berhasil secara keseluruhan.

3.1.4.1 Analisis kebutuhan perangkat pikir

Analisis perangkat pikir menjelaskan ketentuan pengguna maupun lingkunan dimana perangkat lunak akan digunakan.

1. Analisis Pengguna

   Pengguna Perangkat lunak pengendali *pointer* yang akan dibangun ini adalah pengguna yang mengalami cacat fisik pada bagian motorik berikut ini adalah bagian yang digunakan dalam menjalankan perangkat lunak :

   a. Kepala atau Wajah

      Posisi kepala pengguna harus tegak lurus dengan kamera yang akan di gunakan sehingga dapat memudahkan proses pendeteksian wajah dan indera penglihatan. Wajah akan digunakan untuk proses penggerakan pointer *mouse.*

   b. Indera Penglihatan

      Indera penglihatan atau mata akan digunakan untuk proses klik pada perangkat lunak dengan cara mengkedipkan mata tersebut. Kedipan mata sebelah kanan untuk proses klik kanan sedangkan kedipan mata

sebelah kiri untuk proses klik kiri. Jika pengguna melakukan kedipan dengan kedua matanya maka proses klik tidak akan terjadi.

c. Lingkungan

Kondisi background harus jelas dan pencahayaan harus cukup terang agar memudahkan proses tracking wajah dan indera penglihatan Karen aapabila kondisi gelap maka sistem akan sulit mendeteksi pola wajah dan indera penglihatan.

3.1.4.2 Analisis Kebutuhan Perangkat Keras

Pembangunan perangkat lunak pengendali *pointer* ini memerlukan beberapa perangkat keras seperti :

1. **Komputer** dengan spesifikasi sebagai berikut :
   a. Prosessor Pentium Dual Core T4200 2.00GHz.
   b. RAM 2 GB.
   c. VGA 512 MB.
2. ***Webcam*** yang digunakan untuk proses deteksi wajah dan mata untuk pengenalan pola, *tracking* wajah dan mata untuk menggerakkan *pointer*, dan deteksi kedipan (*blink detection*) untuk proses klik pada pointer.

3.1.4.3 Analisis Kebutuhan Perangkat Lunak

Sebelum menggunakan perangkat lunak pengendali *pointer* ini *user* harus terlebih dahulu menginstalasi aplikasi pendukung untuk mengaktifkan *webcam* dan perangkat lunak pendukung pembangunan perangkat lunak pengendali pointer ini antara lain :

1. Sistem Operasi Windows XP Profesional.

2. Microsoft Visual C# 2010 Express, digunakan untuk pengkodean sistem.
3. Microsoft Visio 2007, digunakan untuk memodelkan sistem.

3.1.5 Analisis File

Metode yang digunakan untuk mendeteksi wajah dan indera penglihatan adalah metode Fisher's *Linear Discriminant Analisys* (FLDA) yang sebelumnya telah didapat nilai *training* dari *haar cascade classifier*. Metode ini menggunakan *statistical model* (*classifier*). Tahap pendeteksian wajah dan indera penglihatan dimulai dari masuknya gambar *real time* oleh pengguna kemudian sistem akan mengolah data gambar yang masuk.

Dalam tahap ini frame-frame video atau gambar akan dibandingkan setiap beberapa detik terhadap frame-frame yang baru masuk ke sistem melalui kamera. Untuk mencari dan meng-*update* perubahan pada objek yang berada di frame-frame tersebut digunakan fungsi dari OpenCVSharp yaitu *queryfame,* alasan yang mendasari pembangun aplikasi menggunakan fungsi ini adalah untuk mengurangi beban memory pada hardisk karena *image* yang masuk kedalam sistem relatif besar, sehingga apabila user semakin lama memakai aplikasi ini maka file-file image akan bertambah terus dan akan semakin menumpuk pada *harddisk*.

Fungsi *queryframe* ini adalah fungsi yang digunakan untuk meng-*update* frame tiap 33 miliseconds sekali sehingga sistem tidak akan terlalu terbebani dengan data image yang masuk.

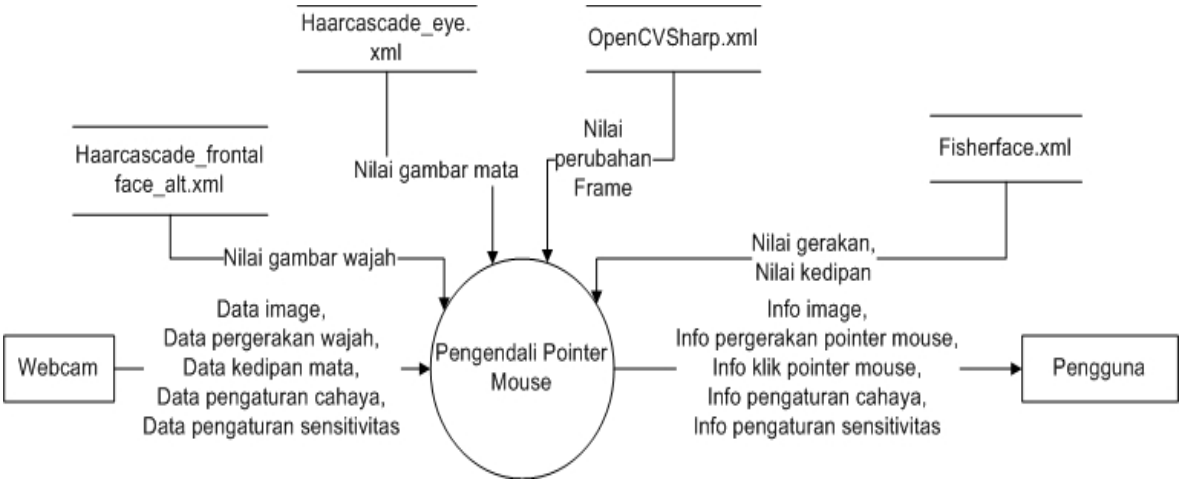
3.1.6 Analisis Kebutuhan Fungsional

Kebutuhan fungsional dianalisis dengan memodelkan sistem. Pendetakan pemodelan analisis yang digunakan untuk membangun perangkat lunak

pengendali *pointer mouse* ini adalah pemodelan prosedural. Perangkat lunak ini dimodelkan menggunakan *Tools* DFD (*Data Flow Diagram*). Perangkat lunak yang digunakan untuk memodelkan perangkat lunak ini adalah Microsoft Visio 2007.

3.1.6.1 Diagram konteks

Diagram konteks adalah diagram yang menggambarkan *input* , *process* dan *output* secara umum yang terjadi pada sistem perangkat lunak yang akan di bangun . Berikut ini adalah diagram konteks perangkat lunak pengendali pointer mouse :



Gambar 3. 12 Gambar diagram konteks pengendali pointer mouse

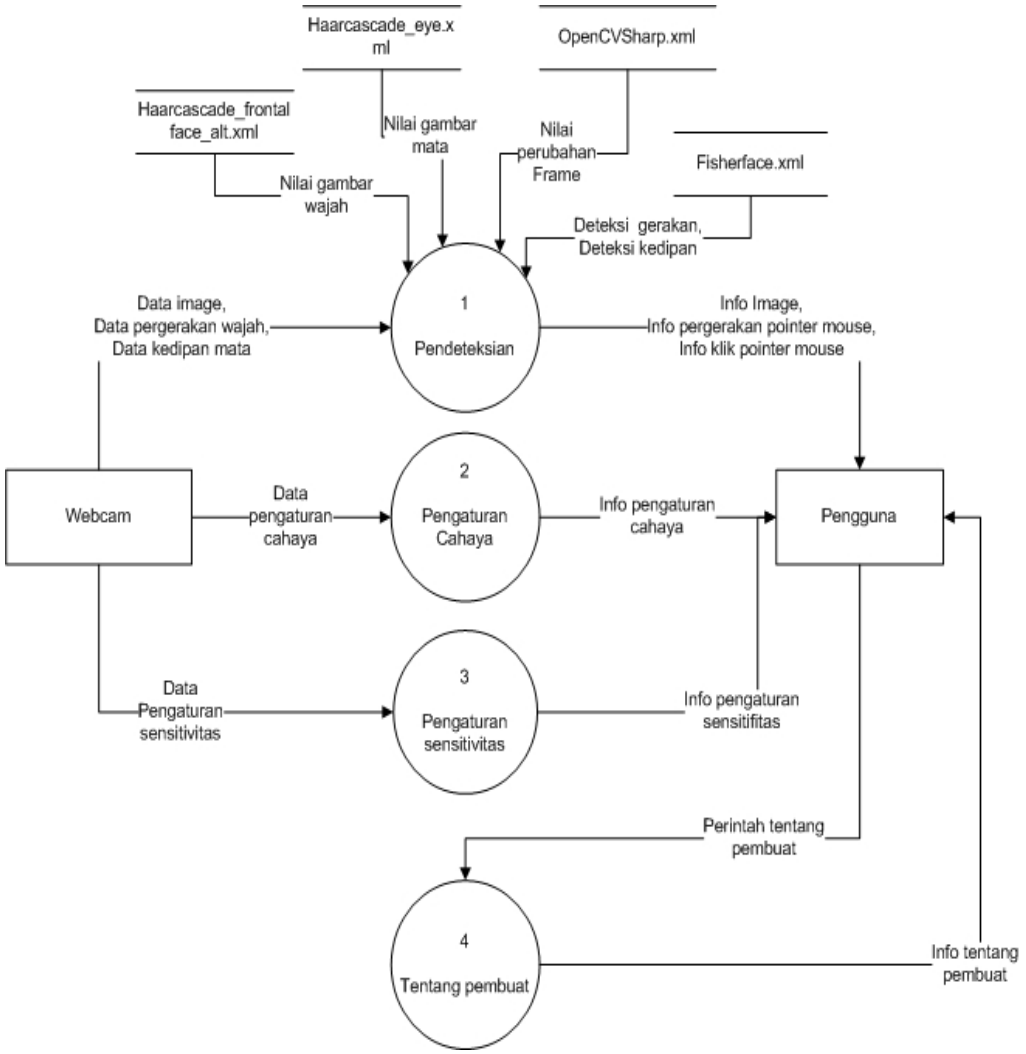
3.1.6.2 Data Flow Diagram (DFD)

Data Flow Diagram merupakan suatu media yang digunakan untuk menggambarkan aliran data yang mengalir pada suatu sistem informasi. Dalam Data Flow Diagram (DFD) terdiri dari entitas luar, aliran data, proses, dan

penyimpanan data. Salah satu keuntungan menggunakan DFD adalah memudahkan pemakai yang kurang menguasai bidang komputer untuk mengerti sistem yang akan dikerjakan. Adapun Data Flow Diagram Level 1 dari pembangunan perangkat lunak pengendali pointer mouse adalah sebagai berikut :

1. **DFD level 1 Pengendali pointer mouse**

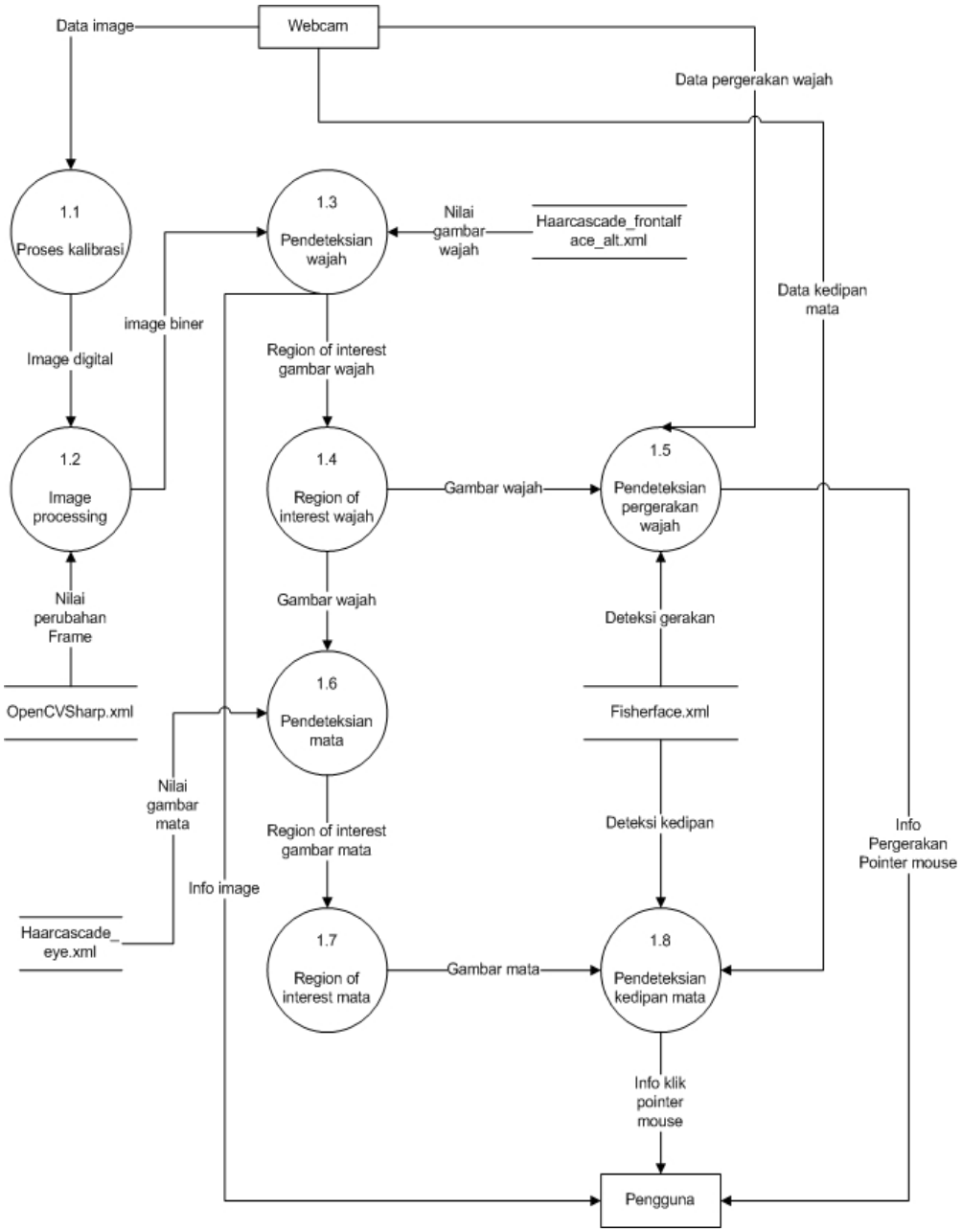
   Berikut ini adalah gambar DFD level 1 pengendali pointer mouse :



Gambar 3. 13 Gambar DFD level 1 pengendali pointer mouse

2. DFD level 2 Pendeteksian

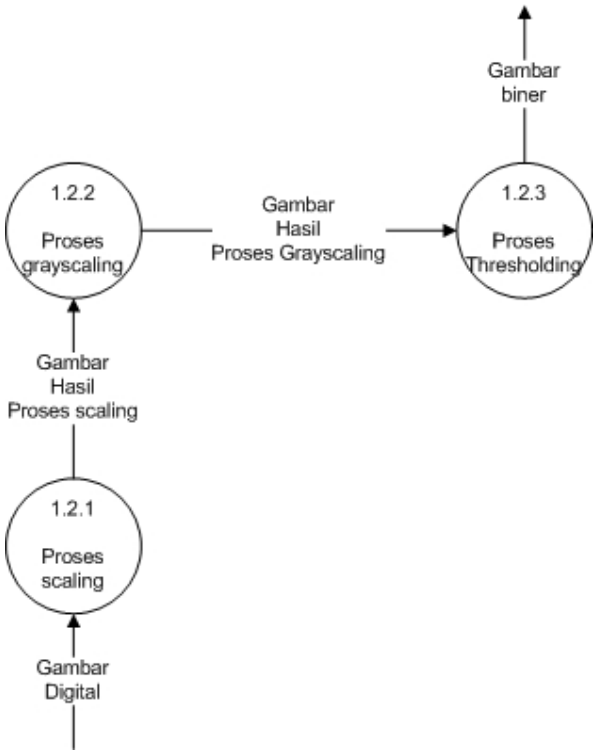
Berikut ini adalah gambar DFD level 2 proses pendeteksian :



Gambar 3. 14 Gambar DFD level 2 pendeteksian

3. DFD level 3 *image processing*

Berikut ini adalah gambar DFD level 3 dari proses *image processing* :



Gambar 3. 15 Gambar DFD level 3 proses *image processing*

Sistem akan mendeteksi gambar secara *realtime* kemudian mendeskripsikan gambar itu sebagai wajah atau bukan , lalu sistem akan melakukan proses *scaling* dilanjutkan dengan *grayscaling* dan tahap terakhirnya adalah proses *thresholding*.

3.1.6.3 Spesifikasi Proses

Spesifikasi proses merupakan deskripsi dari setiap elemen proses yang terdapat dalam program, yang meliputi nama proses, input, output dan keterangan dari proses. Berikut ini spesifikasi proses dari perangkat lunak pengendali pointer mouse :

Tabel 3. 4 Spesifikasi Proses

No	Proses	Keterangan
1	No.Proses	1
	Nama	Pendeteksian
	Source	Webcam
	Input	Data image, data pergerakan wajah, data kedipan mata
	Output	Info image, info pergerakan pointer mouse, info klik pointer mouse
	Destination	Pengguna
	Logika Proses	1. Webcam memasukkan Data image, data pergerakan wajah, data kedipan mata pada sistem perangkat lunak. 2. Apabila data image , data pergerakan wajah , dan data kedipan mata di kenali oleh sistem sebagai wajah dan indera penglihatan maka sistem akan menampilkan pergerakan pointer sesuai pergerakan wajah tersebut pada layar

		pengguna. 3. Apabila data-data masukan tidak dikenali oleh sistem maka sistem akan terus mendeteksi sampai mendapatkan objek yang menyerupai wajah.
2	No.Proses	2
	Nama	Pengaturan Cahaya
	Source	Webcam
	Input	Data pengaturan cahaya
	Output	Info pengaturan cahaya
	Destination	Pengguna
	Logika Proses	1. Pengguna melakukan pengaturan cahaya pada perangkat lunak. 2. Apabila sudah selesai pengaturan , maka sistem akan menampilkan layar sesuai dengan yang pengaturan yang telah dilakukan. 3. Apabila tidak melakukan pengaturan maka tidak akan ada perubahan pada layar
3	No.Proses	3
	Nama	Pengaturan sensitivitas
	Source	Webcam
	Input	Data pengaturan sensitivitas
	Output	Info pengaturan sensitivitas

	Destination	Pengguna
	Logika Proses	1. Pengguna melakukan pengaturan sensitivitas pada aplikasi 2. Sistem akan menyimpan pengaturan sensitivitas 3. Pengguna akan melihat reaksi dari sistem secara langsung pada form aplikasi
3	No.Proses	4
	Nama	Tentang pembuat
	Source	Pengguna
	Input	Perintah tentang pembuat
	Output	Info tentang pembuat
	Destination	Pengguna
	Logika Proses	1. Pengguna memilih perintah tentang pembuat. 2. Sistem akan menampilkan data tentang pembuat.
4	No.Proses	1.1
	Nama	Proses kalibrasi
	Source	Webcam
	Input	Data image
	Output	Image digital
	Destination	Pengguna
	Logika Proses	1. *Webcam* menerima masukan yang berupa data image 2. Data image yang menjadi masukan akan diubah

		bentuknya menjadi image digital
5	No.Proses	1.2
	Nama	Image processing
	Source	Webcam
	Input	Image digital
	Output	Image biner
	Destination	Pengguna
	Logika Proses	1. *Webcam* menerima *image digital* menjadi masukan 2. Kemudian sistem akan mengubah *image digital* menjadi *image* biner
6	No.Proses	1.3
	Nama	Pendeteksian wajah
	Source	Webcam
	Input	Image biner, nilai gambar wajah
	Output	Region of interest, info image
	Destination	Pengguna
	Logika Proses	1. Sistem akan membandingkan *image* biner dengan nilai gambar wajah yang ada pada library. 2. Apabila nilai wajah di temukan maka akan ditentukan region of interestnya 3. Info image akan langsung di tampilkan pada

		layar pengguna
7	No.Proses	1.4
	Nama	Region Of interest wajah
	Source	Webcam
	Input	Region of interest wajah
	Output	Gambar wajah
	Destination	Pengguna
	Logika Proses	1. Sistem akan menentukan nilai ROI yang sesusai dengan nilai yang terdapat pada library. 2. Kemudian sistem akan menandai posisi ROI dan tanda tersebut digunakan untuk mendeteksi mata
8	No.Proses	1.5
	Nama	Pendeteksian pergerakan wajah
	Source	Webcam
	Input	Gambar wajah
	Output	Info pergerakan pointer mouse
	Destination	Pengguna
	Logika Proses	1. Sistem akan menerima gambar wajah 2. Kemudian pointer mouse akan mengikuti gerakan wajah kemanapun arah gerakannya.
9	No.Proses	1.6

	Nama	Pendeteksian mata
	Source	Webcam
	Input	NIlai gambar mata,Gambar mata
	Output	Region of interest gambar mata
	Destination	Pengguna
	Logika Proses	1. Sistem akan menerima gambar mata kemudian gambar mata tersebut akan dibandingkan dengan library. 2. Apabila gambar mata tersebut serupa maka sistem akan menentukan letak ROI nya.
10	No.Proses	1.7
	Nama	Region of interest mata
	Source	Webcam
	Input	Region of interest gambar mata
	Output	Gambar mata
	Destination	Pengguna
	Logika Proses	1. Sistem akan menentukan letak ROI pada mata sesuai dengan library yang dipakai 2. Kemudian sistem akan memberikan kotak pada daerah ROI yang didapatkan
11	No.Proses	1.8
	Nama	Pendeteksian kedipan mata
	Source	Webcam

	Input	Gambar mata, data kedipan mata
	Output	Info klik pada pointer mouse
	Destination	Pengguna
	Logika Proses	1. Sistem akan menerima masukan gambar mata dan data kedipan mata 2. Kemudian jika terjadi kedipan maka pointer mouse akan melakukan proses klik saat itu juga.
12	No.Proses	1.2.1
	Nama	Proses scaling
	Source	Webcam
	Input	Gambar digital
	Output	Gambar hasil proses scaling
	Destination	Pengguna
	Logika Proses	1. Sistem akan melakukan proses scaling atau mengecilkan gambar digital 2. Kemudian gambar yang telah diperkecil akan digunakan untuk proses grayscaling
13	No.Proses	1.2.2
	Nama	Proses Grayscaling
	Source	Webcam
	Input	Gambar hasil proses scaling
	Output	Gambar hasil proses grayscaling

	Destination	Pengguna
	Logika Proses	1. Gambar hasil proses scaling akan dirubah menjadi grayscale melalui proses grayscaling 2. Gambar grayscale akan digunakan untuk proses thresholding
15	No.Proses	1.2.3
	Nama	Proses Thresholding
	Source	Webcam
	Input	Gambar hasil proses grayscaling
	Output	Gambar biner
	Destination	Pengguna
	Logika Proses	1. Gambar hasil grayscaling akan dirubah menjadi gambar biner dengan proses thresholding 2. Setelah mendapatkan nilai biner maka akan dideteksi pergerakan dan perubahannya

3.2 Perancangan sistem

Perancangan sistem dilakukan setelah analisis sistem selesai. Perancangan sistem terdiri dari perancangan data , perancangan arsitektur dan prosedural.

3.2.1 Perancangan Data

Perancangan data dilakukan untuk mengembangkan sistem dengan merancang data apa saja yang berkaitan dengan perangkat lunak. Perancangan data pada perangkat lunak ini memanfaatkan *function* dari OpenCV yaitu

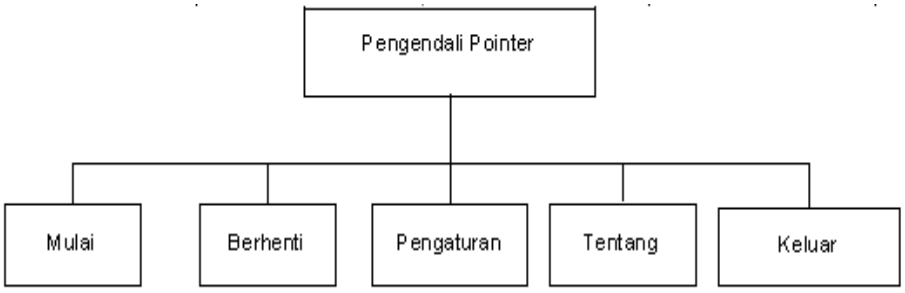
queryframe. *Function* ini berfungsi untuk meng-*update* frame tiap beberapa detik sekali, dari frame ini akan dicari perubahan dari objek sehingga didapatkan nilai *threshold* untuk melakukan proses pergerakan pada pointer dan proses klik pada pointer mouse.

3.2.2 Perancangan Arsitektur

Perancangan arsitektur adalah tahap yang dilakukan dalam merancang stuktur menu , merancang antarmuka bagi pengguna, perancangan pesan dalam perangkat lunak serta jaringan semantik.

3.2.2.1 Perancangan Stuktur Menu

Berikut ini adalah perancangan struktur menu program pengendali pointer mouse.



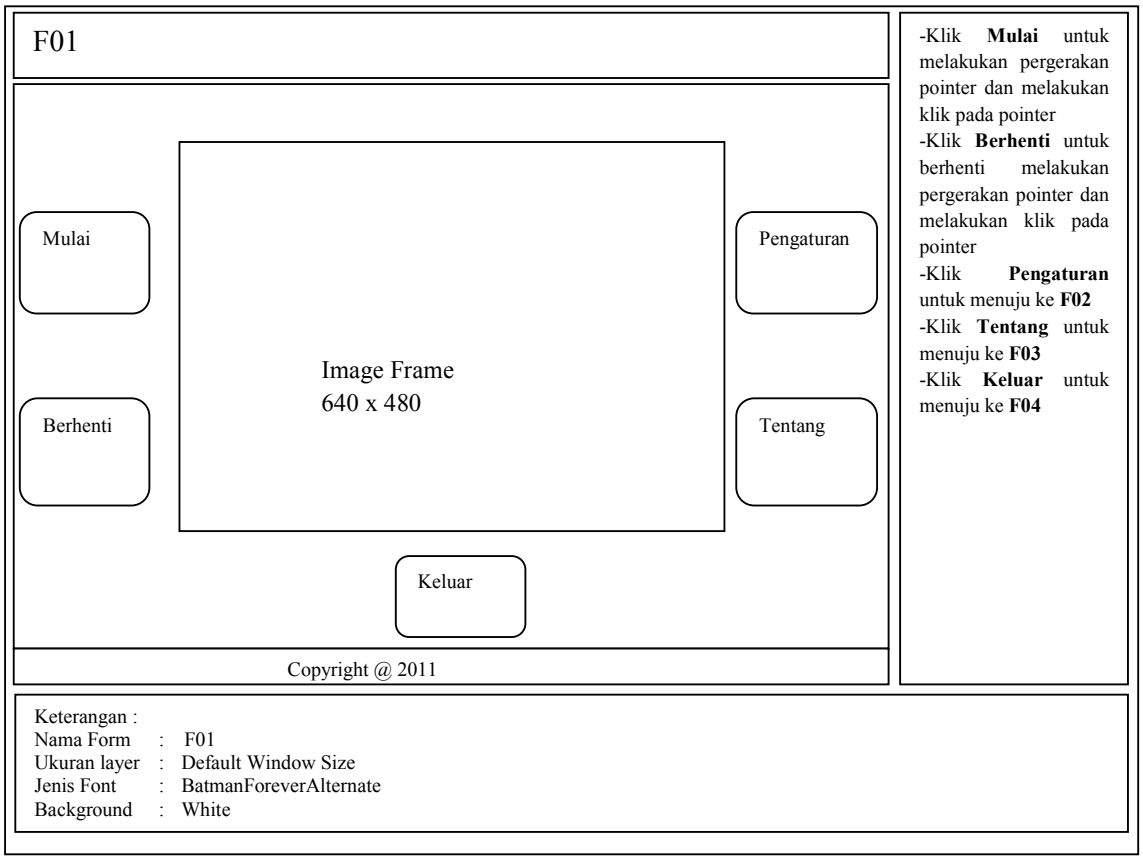
Gambar 3. 16 Struktur Menu Program Pengendali Pointer Dengan Mata

3.2.2.2 Perancangan Antarmuka

Perancangan antarmuka merupakan sebuah penggambaran, perencanaan, dan pembuatan sketsa atau pengaturan dari beberapa elemen yang terpisah ke dalam satu kesatuan yang utuh dan berfungsi. Adapun perancangan antarmuka perangkat lunak pengendali pointer adalah sebagai berikut :

1. Desain *Form* Aplikasi

Form aplikasi merupakan *form* yang digunakan sebagai tampilan pada saat pengguna membuka aplikasi. Desain tampilan form dan deskripsi objek dari aplikasi ini dapat dilihat pada Gambar 3. 17 dan Tabel 3. 5.



Gambar 3. 17 Desain tampilan *form* Aplikasi

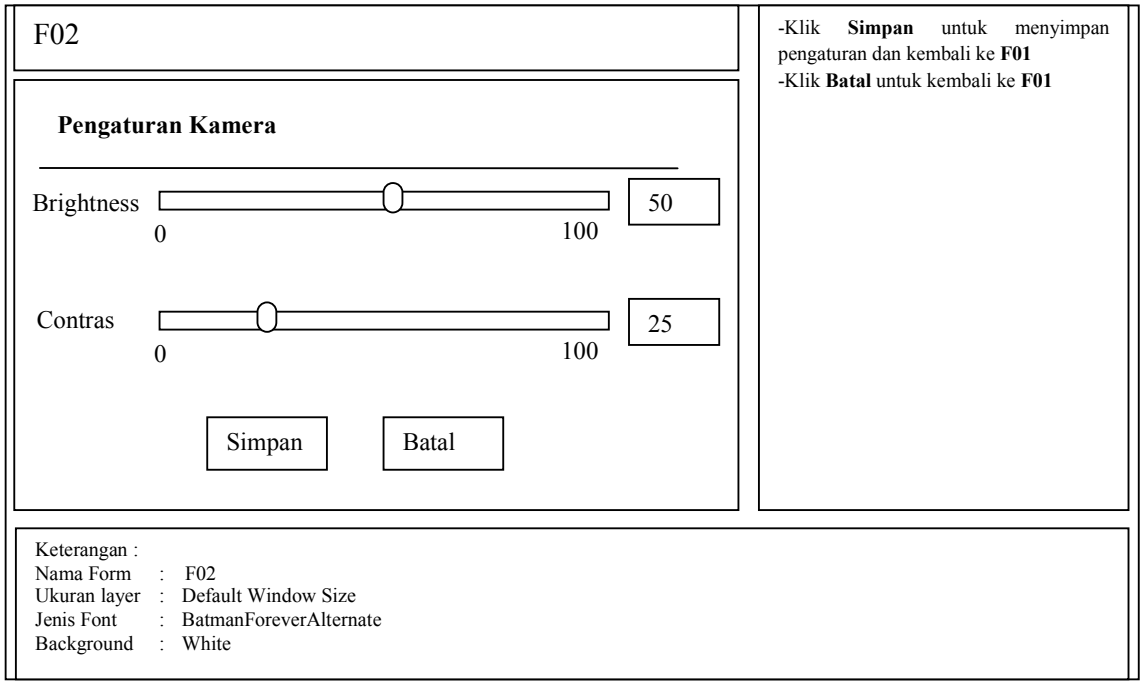
Tabel 3. 5 Tabel Deskripsi objek

Objek	Jenis	Keterangan
Mulai	Button	Memulai mendeteksi
Berhenti	Button	Berhenti mendeteksi

Pengaturan	Button	Konfigurasi sistem (F02)
Tentang	Button	Link ke form Tentang (F03)
Keluar	Button	Keluar Aplikasi (F04)
Image Frame	Frame	Tampilan objek yang sedang di deteksi

2. Desain *Form* Pengaturan

Form pengaturan merupakan *form* yang digunakan untuk mengkonfigurasi pengaturan suara dan pengaturan kamera. Berikut ini adalah desain tampilan form pengaturan :



Gambar 3. 18 Perancangan Antarmuka Pengaturan

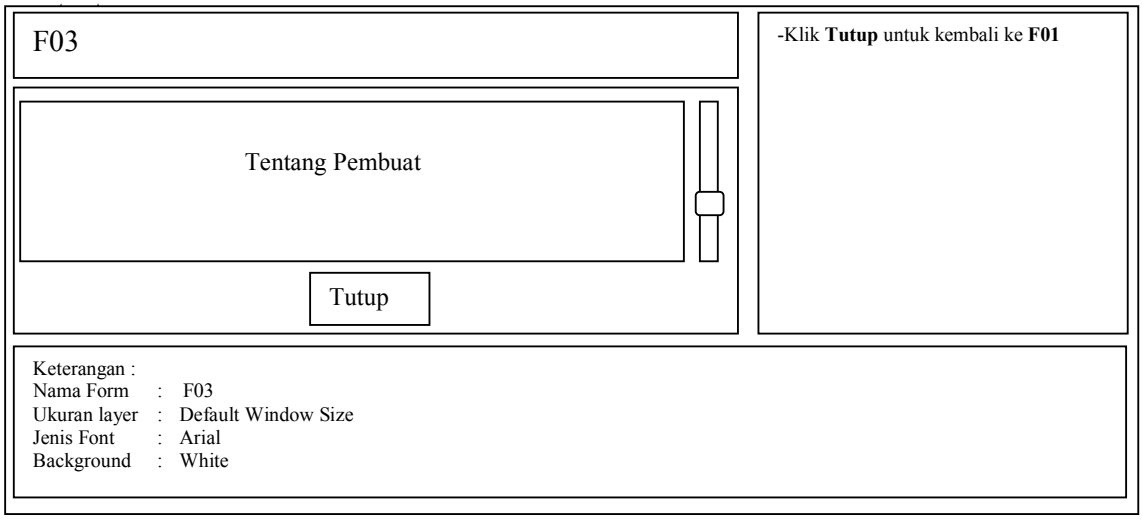
Berikut ini adalah deskripsi objek *form* pengaturan :

Tabel 3. 6 Deskripsi objek form pengaturan

Objek	Jenis	Keterangan
Brightness	Control Box	Mengatur Brightness
Contrast	Control Box	Mengatur Contrast
Simpan	Button	Menyimpan pengaturan dan kembali ke (F01)
Batal	Button	Kembali ke (F01)

3. Desain *Form* Tentang

Form tentang merupakan *form* yang digunakan untuk menampilkan tentang pembuat aplikasi. Berikut ini adalah desain tampilan form



Gambar 3. 19 Perancangan Antarmuka Pengaturan

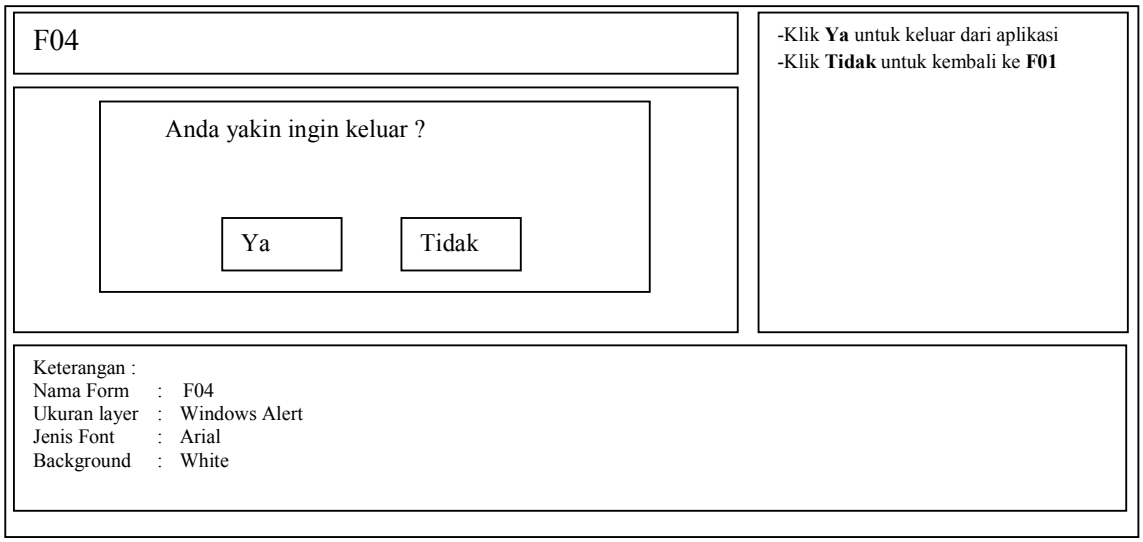
Berikut ini adalah deskripsi objek *form* tentang :

Tabel 3. 7 Deskripsi objek form tentang

Objek	Jenis	Keterangan
Tentang Pembuat	Memo	Deskripsi mengenai pembuat aplikasi
Tutup	Button	Kembali ke (F01)

3.2.2.3 Perancangan Pesan

Perancangan pesan digunakan untuk merancang pesan konfirmasi dari tindakan yang dilakukan pengguna terhadap perangkar lunak. Berikut ini adalah perancangan tampilan pesan :



Gambar 3. 20 Perancangan Antarmuka Pesan

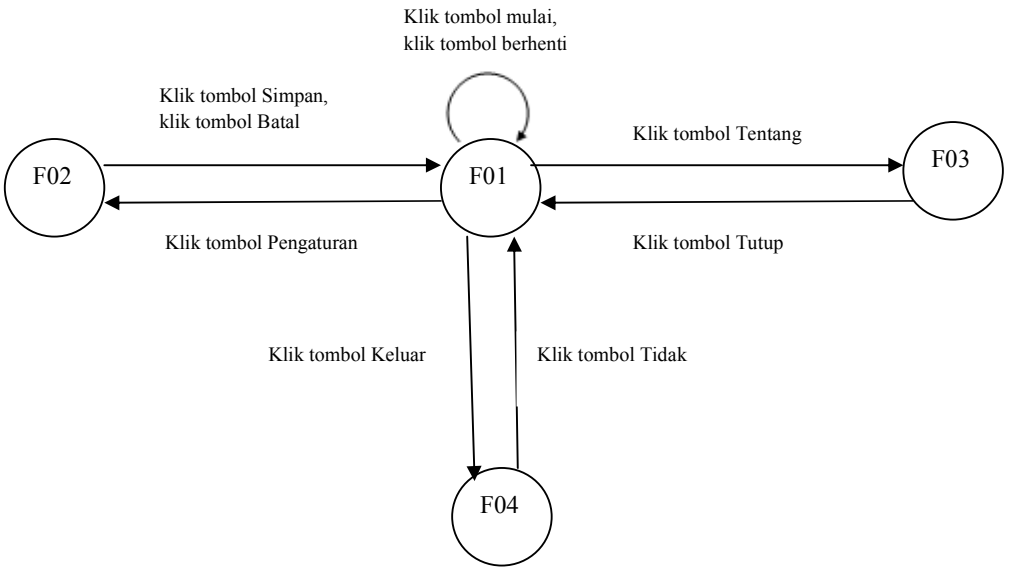
Berikut ini adalah deskripsi perancangan pesan :

Tabel 3. 8 Deksripsi perancangan

Objek	Jenis	Keterangan
Ya	Button	Keluar dari aplikasi
Tidak	Button	Kembali ke (F01)

3.2.2.4 Jaringan Semantik

Berikut ini adalah gambar jaringan semantik yang menggambarkan hubungan antar modul perangkat lunak pengendali pointer :

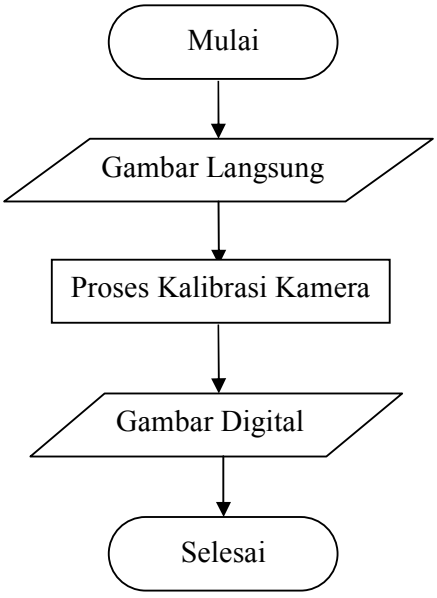


Gambar 3. 21 Jaringan semantik

3.2.3 Perancangan Prosedural

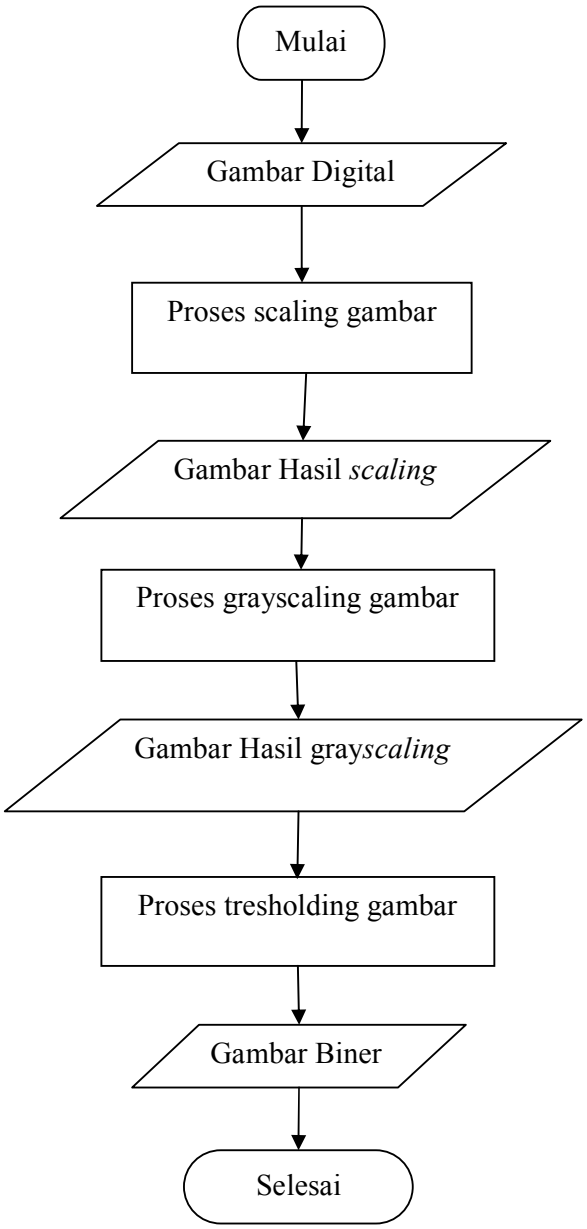
Langkah terakhir dalam perancangan yaitu merancang prosedural yang akan diimplementasikan ke dalam sistem. Prosedur ini akan digunakan sebagai algoritma dasar dalam mengkodekan prosedur yang ada. Adapun perancangan prosedural untuk pembangunan perangkat lunak pengendali pointer yang akan dibangun adalah sebagai berikut :

1. Prosedur kalibrasi kamera



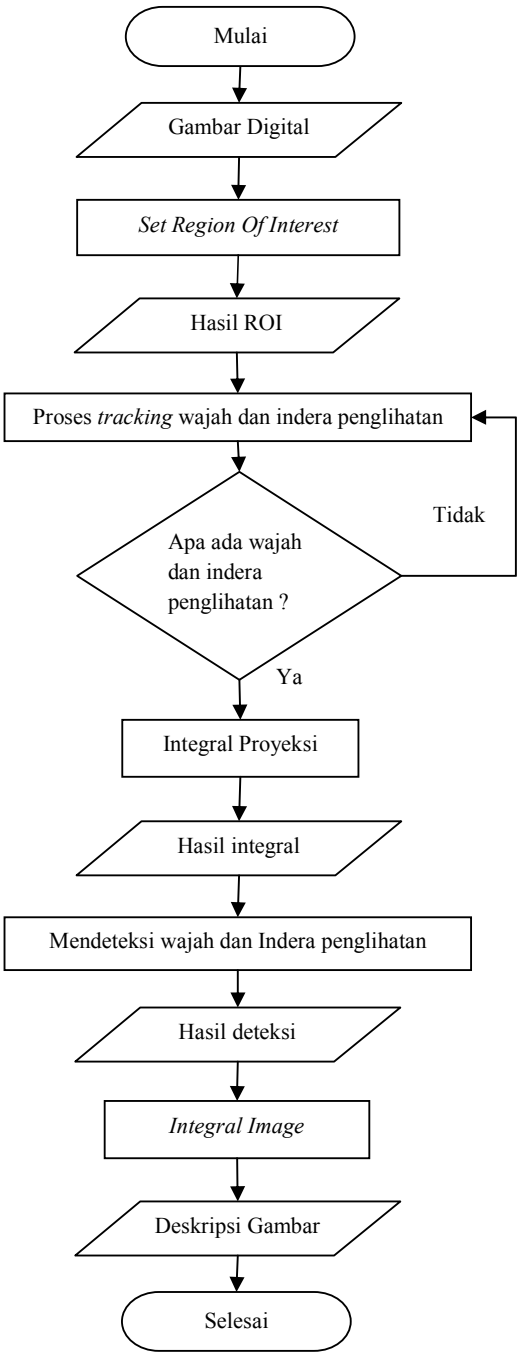
Gambar 3. 22 Prosedur kalibrasi kamera

2. Prosedur pengenalan objek



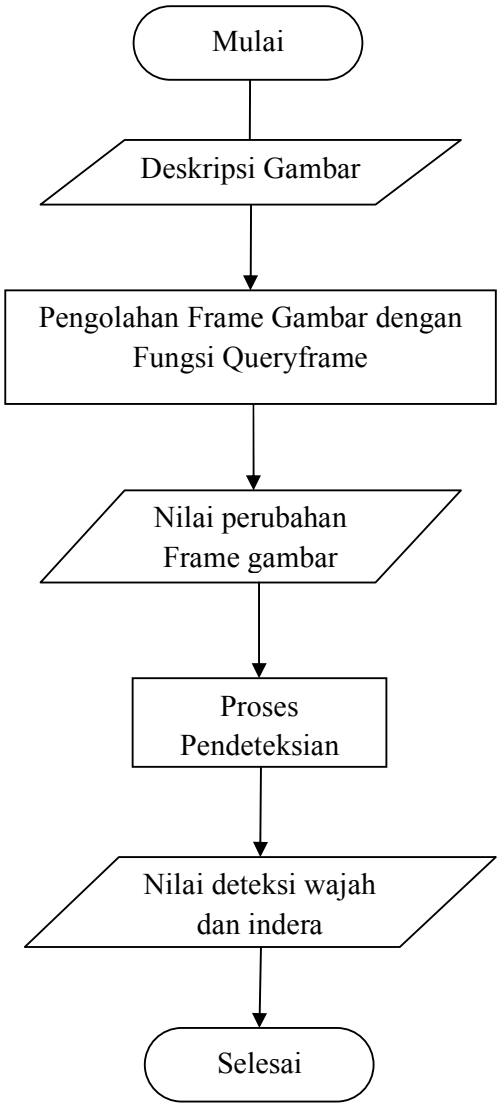
Gambar 3. 23 Prosedur pengenalan objek

3. Prosedur cara kerja metode pendeteksian



Gambar 3. 24 Prosedur cara kerja metode pendeteksian

4. Prosedur pengambilan nilai *threshold*



Gambar 3. 25 Prosedur pengambilan nilai *threshold*