BAB III

ANALISIS DAN PERANCANGAN SISTEM

III.1 Analisis Sistem

Analisis sistem merupakan penguraian dari suatu sistem yang utuh ke dalam beberapa komponen dengan maksud untuk mengidentifikasikan dan mengevaluasi permasalahan-permasalahan yang terdapat dalam suatu sistem.

Dalam membangun perangkat lunak ini dilakukan beberapa tahap analisis yaitu :

a. Menentukan masalah yang akan dibangun untuk perangkat lunak perbandingan algoritma.
b. Mengumpulkan data yang diperlukan untuk membangun perangkat lunak melalui studi literatur dan observasi.
c. Analisis Kebutuhan Non Fungsional yang merupakan batasan-batasan dari layanan-layanan dan fungsi-fungsi dari sebuah perangkat lunak.
d. Analisis Kebutuhan Fungsional berupa fungsionalitas dan layanan yang terdapat dalam perangkat lunak.

III.1.1 Analisis Masalah

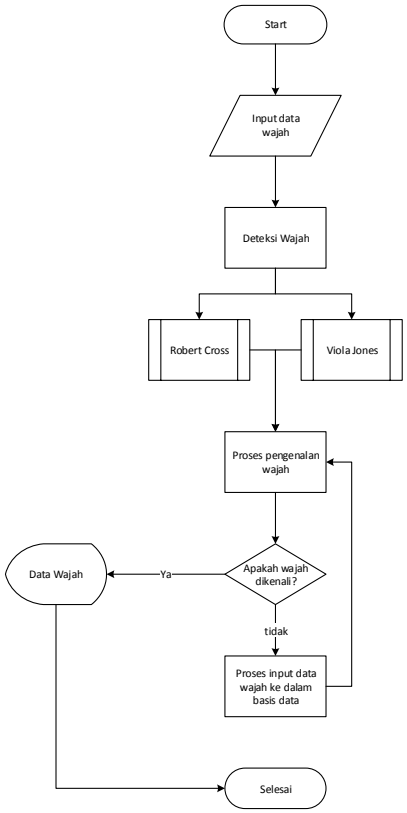
Proses pengenalan wajah harus memiliki tingkat keamanan yang tinggi sehingga dibutuhkan algoritma yang memiliki kemampuan proses yang baik. Terdapat beberapa masalah yang terdapat dalam sistem keamanan dalam media pengenalan wajah :

1. Belum adanya penelitian yang menghasilkan kesimpulan metode mana terbaik dalam perangkat lunak pengenalan wajah sehingga sulit untuk membangun perangkat lunak pengenalan wajah dengan kecepatan yang baik.
2. Kurangnya kecepatan dalam proses pengenalan wajah yang dapat menyebabkan terjadinya kesalahan dalam proses pengenalan wajah**.**

III.1.2 Analisis Algoritma

Penelitian ini menggunakan dua metode, metode tersebut adalah *metode Roberts Cross dan metode Viola Jones.*

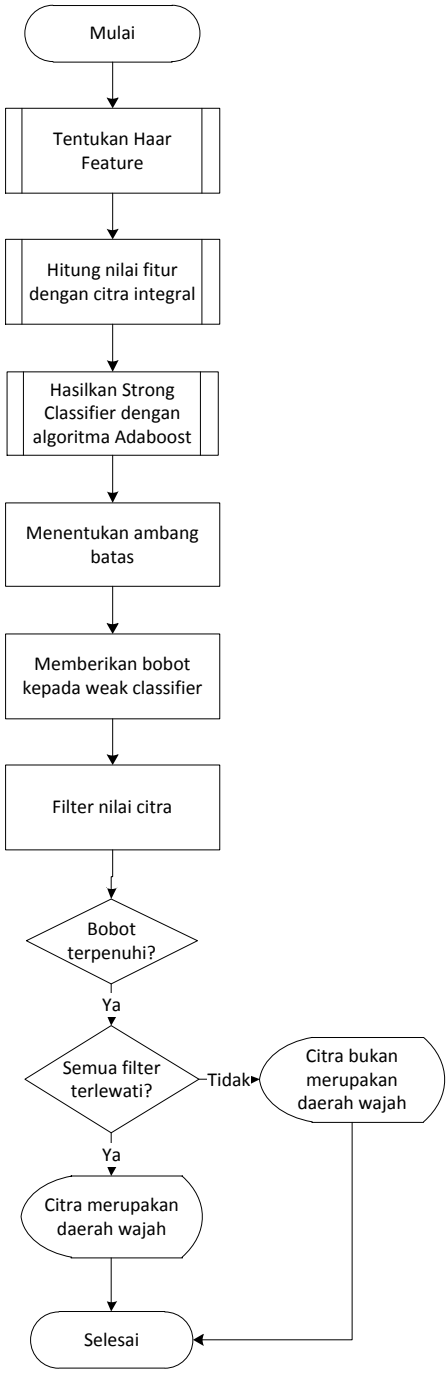
Dalam sistem ini tahap analisis dikhususkan dalam proses pendeteksian wajah, bagan pengenalan secara utuh dapat dilihat dalam Gambar III-1.



Gambar III-1 Gambaran Umum Sistem Pengenalan Wajah

1. Metode *Viola Jones*

Metode *Viola Jones* mempunyai tahapan-tahapan dalam proses pendeteksian wajah, seperti yang dapat dilihat pada Gambar III-2.



Gambar III-2 Flowchart Algoritma AdaBoost

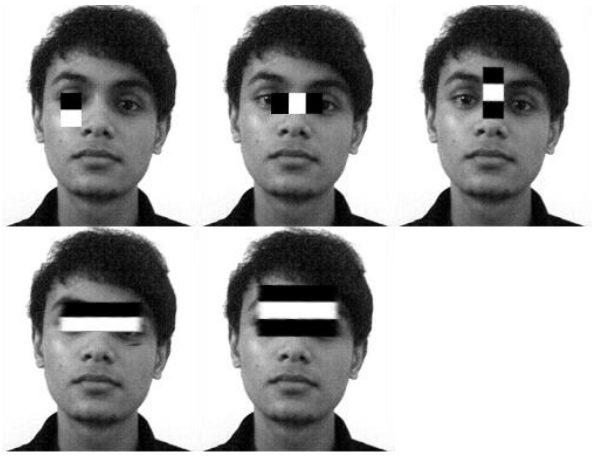
Berikut ini merupakan penjelasan dari tahapan-tahapan yang ada dalam proses pendeteksian wajah pada metode viola jones:

a. Menentukan fitur wajah

Metode *Viola Jones* menggunakan data latih dari citra-citra yang kurang tajam sebagai bagian dari proses pengklasifikasian citra. Klasifikasi citra dilakukan berdasarkan nilai dari sebuah fitur.

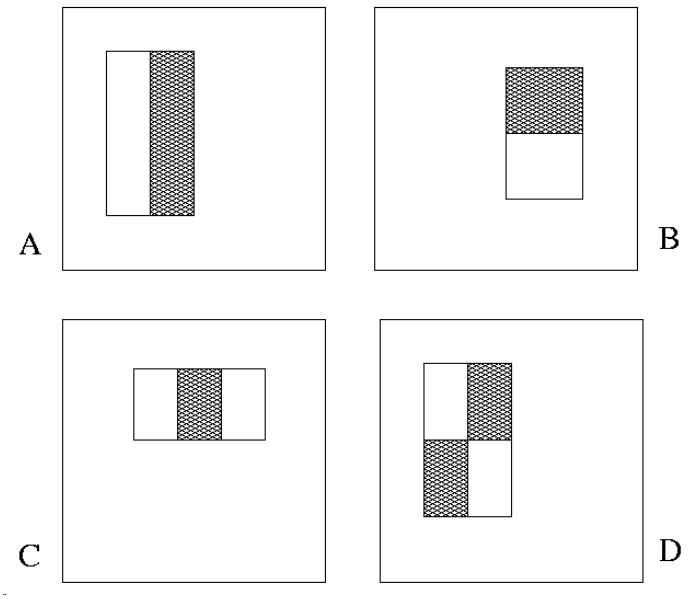
Keberadaan ada atau tidaknya fitur wajah ditentukan dengan mengurangi nilai pixel di wilayah gelap dengan nilai pixel di wilayah terang. Jadi Setiap gambar dirubah kedalam warna hitam dan putih. Jika nilai dari hasil perbedaanya di atas dari ambang batas selama masa pembelajaran citra maka fitur tersebut dapat dikatakan ada.

Fitur-fitur ini merupakan gambaran dari wajah manusia yang dikelompokkan berdasarkan sisi yang terang dan sisi yang gelap seperti yang dapat dilihat pada Gambar III-3. Daerah mata memiliki sisi yang lebih gelap daripada bagian di antara dua mata.



Gambar III-3 Fitur Wajah Metode Viola Jones

Terdapat tiga jenis fitur berdasarkan jumlah persegi panjang yang terdapat di dalamnya [1], seperti dalam Gambar III-4. Fitur-fitur inilah yang biasa disebut sebagai *Haar-like features. Haar-like features* ini dibenamkan dalam library yang bernama opencv.



Gambar III-4 Jenis fitur Viola Jones

1. Pada gambar di atas dapat dilihat bahwa fitur a dan b mempunyai dua persegi panjang. Fitur yang memiliki dua persegi panjang merupakan perbedaan antara jumlah piksel dalam dua daerah persegi panjang. Daerah persegi panjang memiliki ukuran dan bentuk yang sama dan posisi horizontal atau vertikal yang saling berdekatan.
2. Fitur yang memiliki tiga persegi panjang adalah jumlah dari dua persegi panjang luar dikurangi dengan jumlah dari persegi panjang pusat.
3. Fitur yang memiliki empat persegi panjang adalah nilai dari perbedaan antara pasangan diagonal dari persegi panjang.

b. Citra Integral

Untuk memudahkan proses perhitungan nilai dari setiap fitur *Haar* pada setiap lokasi gambar digunakan teknik yang disebut citra integral. Secara umum integral mempunyai makna menambahkan bobot, bobot merupakan nilai-nilai piksel yang akan ditambahkan ke dalam gambar asli.

Nilai integral dari setiap piksel merupakan jumlah dari semua piksel sebelah atasnya dan di sebelah kirinya. Keseluruhan gambar dapat diintegrasikan dengan operasi bilangan bulat per piksel.

1	2	3
4	5	6
7	8	9

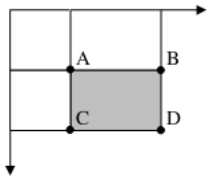
Citra Masukan

1	3	6
5	12	21
12	27	45

CItra Integral

Gambar III-5 Contoh Citra Integral

Dengan mendapatkan nilai dari citra integral maka jumlah dari seluruh piksel yang ada dalam setiap persegi panjang dapat dihitung dengan empat nilai. Nilai-nilai ini merupakan piksel pada citra integral yang bertepatan dengan sudut-sudut persegi panjang yang ada pada citra masukkan.



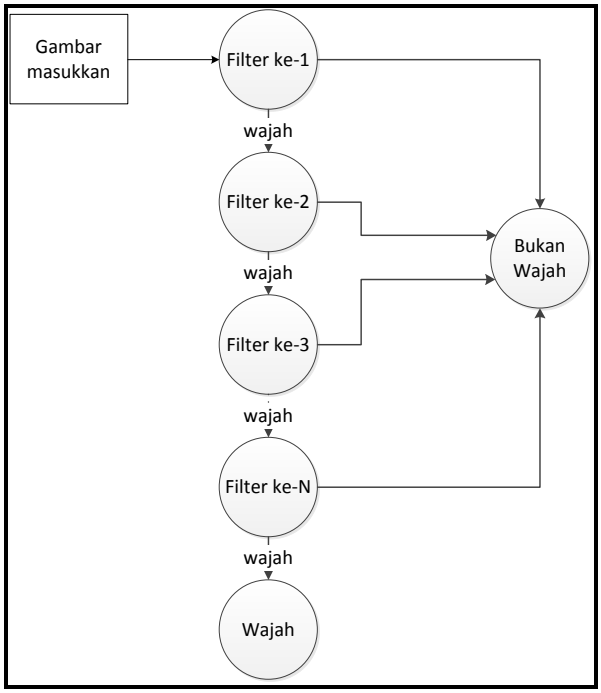
$$\mathbf{ABCD} = \boldsymbol{D} - (\boldsymbol{B} + \boldsymbol{C}) + \boldsymbol{A}$$

Gambar III-6 Jumlah dari seluruh piksel yang ada pada setiap persegi panjang

Untuk memilih fitur *Haar* yang khusus untuk digunakan dalam proses pendeteksian wajah dan untuk menetapkan ambang batas maka digunakan teknik pembelajaran yang disebut sebagai algoritma *AdaBoost*.

c. Algoritma *AdaBoost*

Algoritma *AdaBoost* mengkombinasikan banyak citra-citra yang kurang tajam (*weak classifiers)* untuk menjadi citra-citra yang lebih tajam (*strong classifiers)* dengan memberi bobot kepada citra *weak classifiers.* Secara tidak langsung algoritma *AdaBoost* ini dapat dikatakan sebagai rantai filter, seperti yang ditunjukkan pada Gambar III-7. Setiap filter merupakan *classifier AdaBoost* terpisah dengan jumlah *weak classifiers* yang relatif kecil.



Gambar III-7 Algoritma Adaboost

Nilai ambang batas pada setiap filter diatur setiap saat proses filter terjadi, sehingga nilai dari ambang batas bersifat dinamis. Filter pada setiap tingkat telah dilatih untuk mengklasifikasi citra yang telah melalui tahap sebelumnya. Saat proses pengklasifikasian, jika salah satu filter gagal terlewati maka citra dapat dikatakan sebagai daerah yang bukan wajah. Jika citra belum cukup untuk dikatakan sebagai *strong classifier* maka proses diulang sampai bobot terpenuhi dengan menaikkan nilai dari ambang batas. Ketika citra dapat melewati setiap filter yang ada di dalam rantai, maka dapat dikatakan daerah tersebut merupakan wajah.

Tabel dibawah ini merupakan penggalan *pseudo code* dari algoritma *AdaBoost.*

Tabel III-1 Pseudocode Algoritma AdaBoost

No	Algoritma
1	function adaboost (input sc : StrongClassifier) ← array of record
2	{I.S : nilai variabel sc telah dihasilkan}
3	{F.S : menghasilkan classifier terbaik}
4	
5	Kamus :
6	i,j,T : integer
7	**x : integer
8	min_idx, min_error : integer
9	w_sum : integer
10	beta : real
11	*alfa : double
12	fxidx : double
13	values : real
14	
15	Algoritma :
16	t ← 20
17	if (t > maxstage) then

```
18      return -1
19    endif
20    img_count ←pos_count + neg_count
21    for i← 0 to pos_count do
22      labels[i] ←1
23    endfor
24    for i←pos_count to i<img_count do
25      labels[i] ←-1
26    endfor
27    for i←0 do i<img_count to
28      if (labels[i]) then
29        w[i] ←0.5*1/pos_count;
30      else
31        w[i] ←0.5*1/neg_count;
32      endif
33    endfor
34    for t←0 to t←T do
35      min_error ←0
36      min_idx ←0
37      w_sum ←sum(w, img_count)
38      for i←0 to i<img_count do
39        w[i] ←w_sum
40      endfor
41      weakClassifier cls[feat_count]
42      for j←0 to j<feat_count do
43        if (used[j]) then
44          continue
45        endif
46        values[img_count]
47        for i←0 to i<img_count do
48          values[i] ←applyfeature(features[j], x[i])
49        endfor
50        sort(values, fxidx)
51        cls[j] = findweakclassifier(features[j], fxidx)
52        if (cls[j].error < min_error) then
53          min_error ←cls[j].error
54          min_idx ←j
55        endif
56      endfor
57      addclassifier (sc, cls[min_idx])
58      used[min_idx] ←1
```

59	alfa[t] ←log(1/beta)
60	for i←0 to i<img_count to
61	w[i] ←w[i]*pow(beta, 1-abs(labels[i] - ApplyClassifier(cls[min_idx],images[i])))
62	endfor
63	sc.theta ←sc.theta + 0.5*alfa[t]
64	endfor
65	endfunction

Untuk mengetahui kompleksitas dari algoritma adaboost maka digunakan Big-O sebagai alat untuk mengukur kompleksitas dari algoritma adaboost.

Tabel III-2 Perhitungan Big O Algoritma AdaBoost

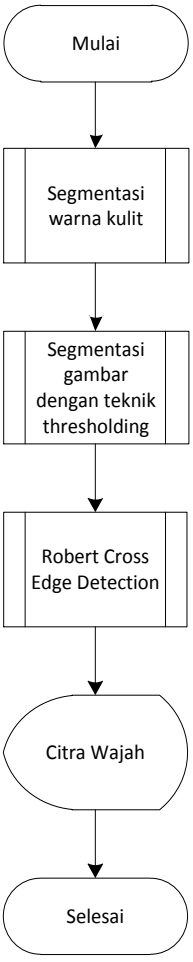
No	Algoritma	Nilai Big O
1	t ← 20	O(1)
2	if (t > maxstage) then	O(1)
3	return -1	O(1)
4	endif	
5	img_count ←pos_count + neg_count	
6	for i← 0 to pos_count do	O(n)
7	labels[i] ←1	O(1)
8	endfor	
9	for i←pos_count to i<img_count do	O(n)
10	labels[i] ←-1	O(1)
11	endfor	
12	for i←0 do i<img_count to	O(n)
13	if (labels[i]) then	O(1)
14	w[i] ←0.5*1/pos_count;	O(1)
15	else	
16	w[i] ←0.5*1/neg_count;	O(1)
17	endif	
18	endfor	
19	for t←0 to t←T do	O(n)
20	min_error ←0	O(1)
21	min_idx ←0	O(1)
22	w_sum ←sum(w, img_count)	O(1)
23	for i←0 to i<img_count do	O(n)
24	w[i] ←w_sum	O(1)
25	endfor	

26	weakClassifier cls[feat_count]	O(1)
27	for j←0 to j<feat_count do	O(n)
28	if (used[j]) then	O(1)
29	continue	
30	endif	
31	values[img_count]	O(1)
32	for i←0 to i<img_count do	O(n)
33	values[i] ←applyfeature(features[j], x[i])	O(1)
34	endfor	
35	sort(values, fxidx)	O(1)
36	cls[j] = findweakclassifier(features[j], fxidx)	O(1)
37	if (cls[j].error < min_error) then	O(1)
38	min_error ←cls[j].error	O(1)
39	min_idx ←j	O(1)
40	endif	
41	endfor	
42	addclassifier (sc, cls[min_idx])	O(1)
43	used[min_idx] ←1	O(1)
44	alfa[t] ←log(1/beta)	O(1)
45	for i←0 to i<img_count to	O(n)
46	w[i] ←w[i]*pow(beta, 1-abs(labels[i] - ApplyClassifier(cls[min_idx],images[i])))	O(1)
47	endfor	
48	sc.theta ←sc.theta + 0.5*alfa[t]	O(1)
49	endfor	
Jumlah		O(n)

Nilai dari kompleksitas algoritma Adaboost yang dihasilkan dari perhitungan Big O adalah O(n).

2. Metode *Roberts Cross*

Metode *Roberts Cross* menggunakan teknik segmentasi warna, segmentasi citra dan algoritma *Roberts Cross Edge detectio*n sebagai cara untuk mendeteksi wajah. Gambar III-8 merupakan gambaran umum dari metode Roberts Cross.



Gambar III-8 FlowChart Metode RobertCross

a. Segmentasi warna kulit

Dalam proses pendeteksian warna kulit, setiap piksel digolongkan sebagai “warna kulit” maupun “bukan warna kulit” dengan membuang sebanyak mungkin citra yang diindikasikan sebagai wilayah yang bukan wajah. Segmentasi warna kulit ini dilakukan dengan mengubah citra RGB ke ruang YCbCr dengan rumus :

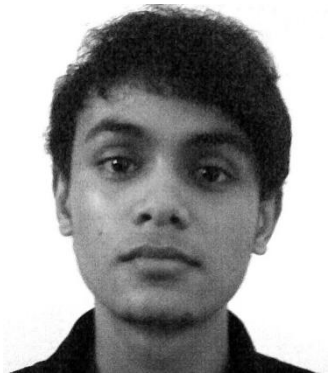
$$Y = 0.299R + 0.587G + 0.114B$$

$$Cb = -0.169R - 0.332G + 0.500B$$

$$Cr = 0.500R - 0.419G - 0.081B$$

Perlu diingat bahwa daerah yang terdeteksi sebagai warna kulit tidak selalu merupakan kulit. Hanya perlu untuk diketahui bahwa daerah tersebut mungkin mempunyai warna yang serupa dengan warna kulit. Hal yang perlu diperhatikan dalam segmentasi kulit ialah bahwa wilayah yang tidak mempunyai warna menyerupai warna kulit akan diabaikan dalam proses penentuan wajah.

Nilai Cr dan Cb pada masing–masing contoh kulit akan didapat dari persamaan di atas. Kemudian menghitung nilai rata–rata dari seluruh nilai Cr dan Cb dari contoh kulit. Nilai rata–rata ini akan digunakan sebagai acuan perhitungan jarak Euclidean antara nilai Cr dan Cb pada citra input dengan nilai Cr dan Cb pada rata – rata contoh kulit. Berikut hasil dari segmentasi warna kulit.



Gambar III-9 Segmentasi warna kulit

Seperti yang dapat dilihat dari gambar di atas daerah yang bukan merupakan warna kulit tidak dideteksi dan daerah yang berwarna kulit terdeteksi dengan jelas.

b. Segmentasi gambar

Proses segmentasi merupakan proses identifikasi dan pemisahan gambar wajah yang merupakan gambar abu-abu dengan gambar latar. Untuk memisahkan gambar wajah dengan gambar latar digunakan teknik *thresholding.*

Teknik *thresholding* digunakan untuk memecah bagian-bagian gambar yang berupa karakter abjad dengan karakter grafis. Proses ini biasa disebut sebagai teknik binerisasi yang mengubah gambar abu-abu menjadi gambar hitam-putih (biner) yang diwakili oleh 0 dan 1, dimana 0 mewakili warna hitam dan 1 mewakili warna putih.

Dalam perangkat lunak ini nilai dari ambang batas ditentukan dengan menghitung nilai rata-rata antara nilai minimum dan maksimum dari gambar yang telah di ubah menjadi warna abu-abu (*grey scale level*). Jika

nilai dari *grey scale level* lebih besar dari nilai ambang batas maka bagian piksel gambar di ubah menjadi warna terang (putih), jika nilai dari *grey scale level* lebih besar dari nilai ambang batas maka bagian piksel gambar di ubah menjadi warna gelap (hitam). Seperti yang terlihat pada Gambar III-10.



Gambar III-10 Segmentasi Gambar

c. *Robert Cross Edge Detection*

Dalam teknik deteksi tepi, tepi merupakan piksel yang memiliki nilai gradien yang tinggi. Untuk menghitung gradien ini, setiap gambar masukkan diubah menjadi gambar dua dimensi dengan menggunakan dua matriks dua dimensi *Robert Cross* yang berukuran 2 x 2.

$$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \qquad \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$

Gx Gy

Gx merupakan matriks *Robert Cross* dari gradien yang dihitung dari sumbu x dan Gy merupakan matriks dari gradien yang dihitung dari sumbu y. Untuk menghitung gradien dari gambar secara keseluruhan, yang perlu dilakukan hanya tinggal menambahkan nilai dari gradien pada

sumbu x (Gx) dan nilai dari gradien pada sumbu y (Gy) dan hasilnya dapat dilihat pada Gambar III-11.



Gambar III-11 ***Robert Cross Edge Detection***

Tabel dibawah ini merupakan penggalan *pseudo code* dari algoritma *roberts cross edge detection.*

Tabel III-3 ***Pseudo code*** **algoritma** ***roberts cross edge detection.***

No	Algoritma
1	function robertcross (input x,y : integer,) → integer
2	{i.s : fungsi-fungsi telah didefinisikan}
3	{f.s : menghasilkan nilai dari fungsi SetPixelV,j,i,RGB()}
4	
5	kamus :
6	i, j: integer
7	
8	algoritma :
9	for i ← 1 to y do
10	for j← 1 to x do
11	Kernel←Abs(ImagePixels(i,j)– ImagePixels(i+1,j+1))+Abs(ImagePixels(i,j+1) - ImagePixels(i+1,j))
12	if kernel < 0 then
13	kernel = 0;
14	SetPixelV,j,i,RGB(0, 0, 0);
15	endif
16	if kernel > 255 then
17	kernel = 255;

18	SetPixelV,j,i,RGB(255, 255, 255);
19	endif
20	endfor
21	endfor
22	endfunction

Untuk menghitung besarnya kompleksitas terhadap algoritma Robert Cross *Edge Detection* maka digunakan Big O sebagai alat untuk menghitung kompleksitasnya. Berikut ini adalah perhitungan kompleksitas algoritma Robert Cross *Edge Detection* :

Tabel III-4 Perhitungan nilai Big O Algoritma Robert Cross *Edge Detection*

No	Algoritma	Nilai Big O
1	for i ← 1 to y do	O(n)
2	for j← 1 to x do	O(n)
3	kernel←Abs(ImagePixels(i,j)– ImagePixels(i+1,j+1))+Abs(ImagePixels(i,j+1) - ImagePixels(i+1,j))	O(1)
4	if kernel < 0 then	O(1)
5	kernel = 0;	O(1)
6	SetPixelV,j,i,RGB(0, 0, 0);	O(1)
7	endif	
8	if kernel > 255 then	O(1)
9	kernel = 255;	O(1)
10	SetPixelV,j,i,RGB(255, 255, 255);	O(1)
11	endif	
12	endfor	
13	endfor	
Jumlah		$O(n^2)$

Nilai kompleksitas algoritma Robert Cross *Edge Detection* adalah $O(n^2)$

III.1.3 Spesifikasi Kebutuhan Perangkat Lunak

Spesifikasi kebutuhan perangkat lunak dibagi menjadi 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 pengenalan wajah :

Tabel III-5 Spesifikasi kebutuhan perangkat lunak fungsional

Kode	Kebutuhan
SKPL-F001	Perangkat lunak harus dapat menangkap gambar wajah.
SKPL-F002	Perangkat lunak harus dapat mengenali gambar wajah.
SKPL-F003	Perangkat lunak harus dapat mengukur kecepatan dari perbandingan algoritma pada metode *Roberts Cross* dan pada metode *Viola Jones.*
SKPL-F004	Perangkat lunak harus dapat mengolah data wajah.

Tabel III-6 Spesifikasi kebutuhan perangkat lunak non-fungsional

Kode	Kebutuhan
SKPL-NF001	Pengguna perangkat lunak menggunakan wajah sebagai media untuk dideteksi oleh perangkat lunak.
SKPL-NF002	Kondisi lingkungan pengguna harus dalam keadaan bercahaya.
SKPL-NF003	Perangkat keras yang digunakan adalah kamera dan komputer dengan spesifikasi yang cukup baik dalam proses pengolahan citra.
SKPL-NF004	Perangkat lunak ini menggunakan sistem operasi Windows.

III.1.4 Analisis Kebutuhan Non Fungsional

Analisis kebutuhan non fungsional menjelaskan pendukung sistem yang akan dijalankan. Adapun kebutuhan-kebutuhan yang diperlukan untuk mendukung kinerja dari perangkat lunak yaitu **:**

1. Analisis Kebutuhan Perangkat Keras

   Komputer merupakan gabungan dari perangkat keras dan perangkat lunak yang satu sama lain saling mendukung kinerjanya. Perangkat lunak memerintahkan perangkat keras instruksi untuk melakukan tugas-tugas tertentu.

   Pada perangkat lunak *face recognition* ini digunakan sebuah komputer berbasis desktop. Adapun kriteria dari komputer yang dibutuhkan adalah :

   a. Tahap pada saat pembangunan perangkat lunak

Tabel III-7 Perangkat keras saat pembangunan perangkat lunak

No	Perangkat Keras	Spesifikasi
1	Processor	Dual Core @ 2.2 GHz
2	Monitor	CRT 14"
3	VGA	512 MB
4	Memori	2 GB
5	Kamera	2.0 Megapixel
6	Keyboard dan Mouse	

b. Tahap pada saat menjalankan perangkat lunak

Tabel III-8 Perangkat keras saat pembangunan perangkat lunak

No	Perangkat Keras	Spesifikasi
1	Processor	Dual Core @ 2.2 GHz
2	Monitor	CRT 14"
3	VGA	512 MB
4	Memori	1 GB
5	Kamera	2.0 Megapixel
6	Keyboard dan Mouse	

2. Analisis Kebutuhan Perangkat Lunak

Perangkat lunak digunakan untuk memberikan perintah-perintah kepada perangkat keras untuk melakukan instruksi-instruksi. Berikut kriteria dari perangkat lunak yang dibutuhkan untuk mendukung perangkat lunak *face recognition* ini :

1. Sistem operasi berbasis Windows 7
2. Microsoft Visual Studio 2010

3. Analisis Kebutuhan Perangkat Pikir

Agar perangkat lunak dapat berjalan dengan optimal selain perangkat keras dan perangkat lunak pendukung maka dibutuhkan juga perangkat pikir berupa pengguna. Perangkat pikir yang dibutuhkan adalah pengguna yang mempunyai wajah lengkap dan normal dengan arah cahaya yang menghadap ke wajah.

III.1.5 Analisis File

Dalam perangkat lunak pengenalan wajah ini digunakan basis data berupa file berekstensi .xml. Dalam penggunaannya file ini mempunyai format tersendiri agar dapat digunakan oleh perangkat lunak. Berikut ini merupakan format penulisan dari file xml yang digunakan dalam sistem :

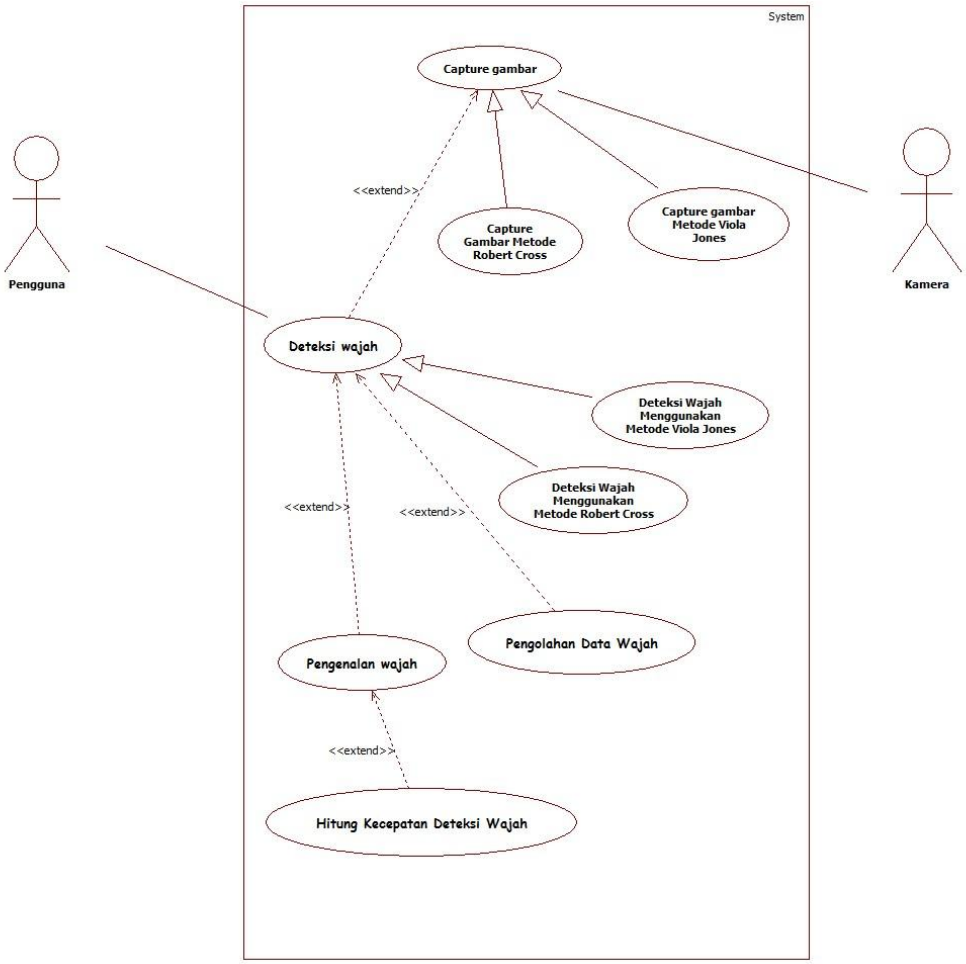
Tabel III-9 Format penulisan file xml

No	Tag XML	Keterangan	Fungsi
1	<?xml version="1.0" encoding="UTF-8"?>	Deklarasi header file xml	Sebagai deklarasi awal file xml
2	<GambarWajah> </GambarWajah>	Deklarasi RootNode	Tag induk yang membungkus elemen-elemen lain
3	<Face></Face>	Deklarasi ChildNode	Tag xml yang membungkus elemen-elemen lain setelah Rootnode
4	<Nama></Nama>	Deklarasi ChildElement	Tag xml yang merupakan nama dari wajah yang dimasukkan oleh pengguna
5	<File></File>	Deklarasi ChildElement	Tag xml yang merupakan nama dari file gambar wajah yang diinputkan oleh pengguna

III.1.6 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional merupakan alur dan pekerjaan dari perangkat lunak yang akan dibangun. Dalam perangkat lunak pengenalan wajah ini digunakan metode pendekatan berorientasi objek.

1. Use Case



Gambar III-12 *Use Case*

Pada gambar di atas terdapat aktor yang merupakan pengguna yang berhubungan langsung dengan perangkat lunak. Sedangkan perangkat lunak sendiri mempunyai beberapa *use case* yang merupakan kegiatan atau pekerjaan yang dilakukan oleh perangkat lunak. Setiap *use case* dijelaskan dalam *use case* skenario.

2. Use case skenario

Nama *use case* : Capture gambar
Aktor : Pengguna dan kamera
Trigger : Pengguna memilih menu deteksi wajah atau pengenalan wajah
Skenario :

No	Aksi aktor	No	Reaksi sistem
Skenario normal			
1	Pengguna memilih menu deteksi wajah dengan metode viola jones		
		2	Menampilkan form deteksi wajah dengan metode viola jones
3	Pengguna memilih menu deteksi wajah dengan metode robert cross		
		4	Menampilkan form deteksi wajah dengan metode robert cross
5	Pengguna memilih menu pengenalan wajah		
		6	Menampilkan form pengenalan wajah

Nama *use case* : Capture gambar menggunakan metode viola jones
Aktor : Pengguna dan kamera
Trigger : Pengguna memilih menu deteksi wajah dengan metode viola jones
Skenario :

No	Aksi aktor	No	Reaksi sistem
Skenario normal			
1	Pengguna menghadapkan wajah kedepan kamera		
		2	Sistem menyalakan kamera
3	Kamera menangkap gambar		
		4	Mengcapture gambar wajah dengan metode viola jones
		5	Menampilkan kotak deteksi wajah

Nama *use case* : Capture gambar menggunakan metode Robert cross
Aktor : Pengguna dan kamera
Trigger : Pengguna memilih menu deteksi wajah dengan metode Robert cross
Skenario :

No	Aksi aktor	No	Reaksi sistem
Skenario normal			
1	Pengguna menghadapkan wajah kedepan kamera		
		2	Sistem menyalakan kamera
3	Kamera menangkap gambar		
		4	Mengcapture gambar wajah dengan metode Robert cross
		5	Menampilkan kotak deteksi wajah

Nama *use case* : Deteksi wajah
Aktor : Pengguna
Trigger : Pengguna memilih menu deteksi wajah
Skenario :

No	Aksi aktor	No	Reaksi sistem
Skenario normal			
1	Pengguna memilih menu deteksi wajah dengan metode viola jones		
		2	Menampilkan form deteksi wajah dengan metode viola jones
3	Pengguna memilih menu deteksi wajah dengan metode robert cross		
		4	Menampilkan form deteksi wajah dengan metode robert cross

Nama *use case* : Deteksi wajah menggunakan metode viola jones
Aktor : Pengguna
Trigger : Pengguna memilih menu deteksi wajah dengan metode viola jones
Skenario :

No	Aksi aktor	No	Reaksi sistem
Skenario normal			
1	Pengguna menghadapkan wajah kedepan kamera		
		2	Mendeteksi wajah dengan metode viola jones
		3	Menampilkan kotak deteksi wajah

Nama *use case* : Deteksi wajah menggunakan metode Robert cross
Aktor : Pengguna
Trigger : Pengguna memilih menu deteksi wajah dengan metode Robert cross
Skenario :

No	Aksi aktor	No	Reaksi sistem
Skenario normal			
1	Pengguna menghadapkan wajah kedepan kamera		
		2	Mendeteksi wajah dengan metode Robert cross
		3	Menampilkan kotak deteksi wajah

Nama use case : Pengolahan data wajah
Aktor : Pengguna dan Kamera
Trigger : Kamera menangkap gambar wajah dan pengguna menginputkan informasi data wajah

Skenario :

No	Aksi aktor	No	Reaksi sistem
Skenario normal			
1	Pengguna menekan tombol input data wajah		
2	Pengguna menghadapkan wajah ke depan kamera		
3	Kamera menangkap gambar		
		4	Mendeteksi wajah
5	Menginputkan nama dari gambar wajah		
		6	Menyimpan data wajah
		7	Menampilkan pesan berhasil disimpan
		8	Mengenali wajah
Skenario ekstension			
1	Pengguna menekan tombol input data wajah		
2	Pengguna menghadapkan wajah ke depan kamera		
3	Kamera menangkap gambar		
		4	Mendeteksi wajah
5	Tidak menginputkan nama dari gambar wajah		
		6	Menampilkan pesan kesalahan
Skenario ekstension			
1	Pengguna menekan tombol input data wajah		
2	Pengguna menghadapkan wajah ke depan kamera		
3	Kamera menangkap gambar		
		4	Mendeteksi wajah
5	Wajah tidak terdeteksi		
		6	Menampilkan pesan kesalahan

Nama *use case* : Pengenalan wajah
Aktor : Pengguna
Trigger : Pengguna menghadapkan wajah ke depan kamera
Skenario :

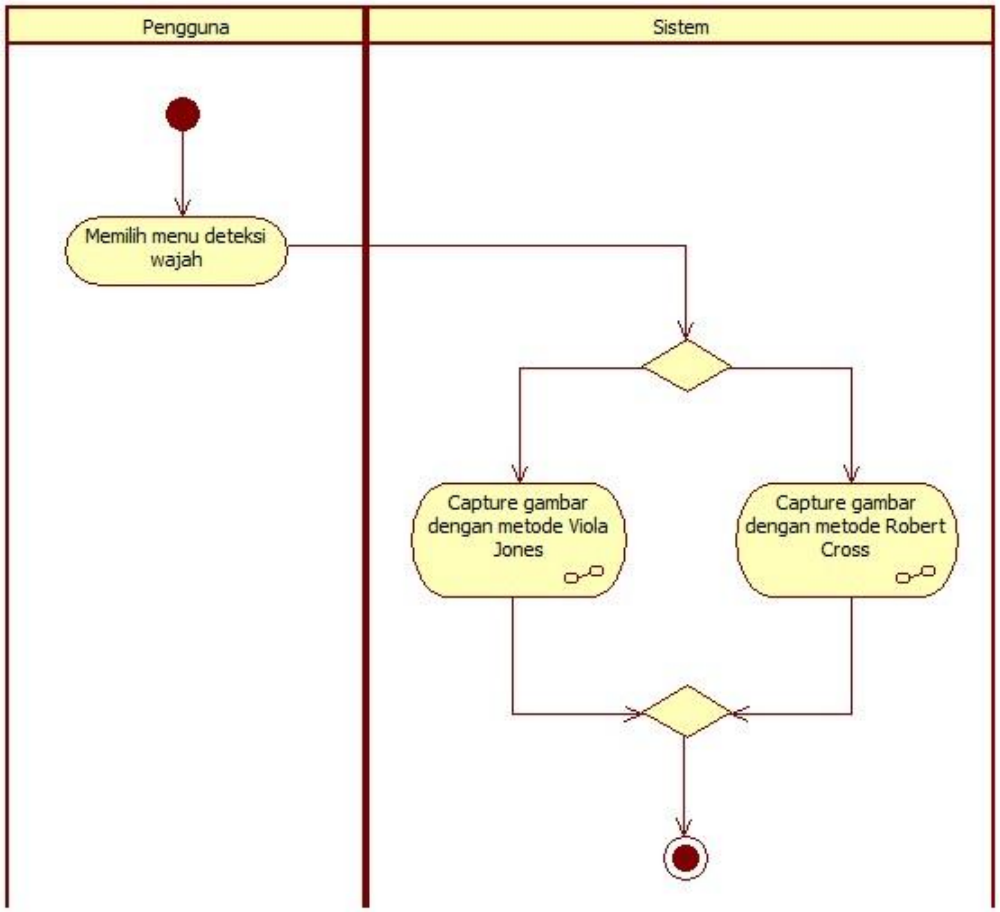
No	Aksi aktor	No	Reaksi sistem
Skenario normal			
1	Pengguna menekan tombol deteksi wajah		
2	Pengguna menghadapkan wajah ke depan kamera		
3	Kamera menangkap gambar		
		4	Mendeteksi wajah
		5	Memeriksa data wajah di basis data
		6	Menampilkan identitas wajah
Skenario ekstension			
1	Pengguna menekan tombol deteksi wajah		
2	Pengguna menghadapkan wajah ke depan kamera		
3	Kamera menangkap gambar		
		4	Mendeteksi wajah
		5	Memeriksa data wajah di basis data
		6	Menampilkan kotak deteksi wajah

Nama *use case* : Hitung kecepatan deteksi wajah
Aktor : Pengguna
Trigger : Pengguna menghadapkan wajah ke depan kamera
Skenario :

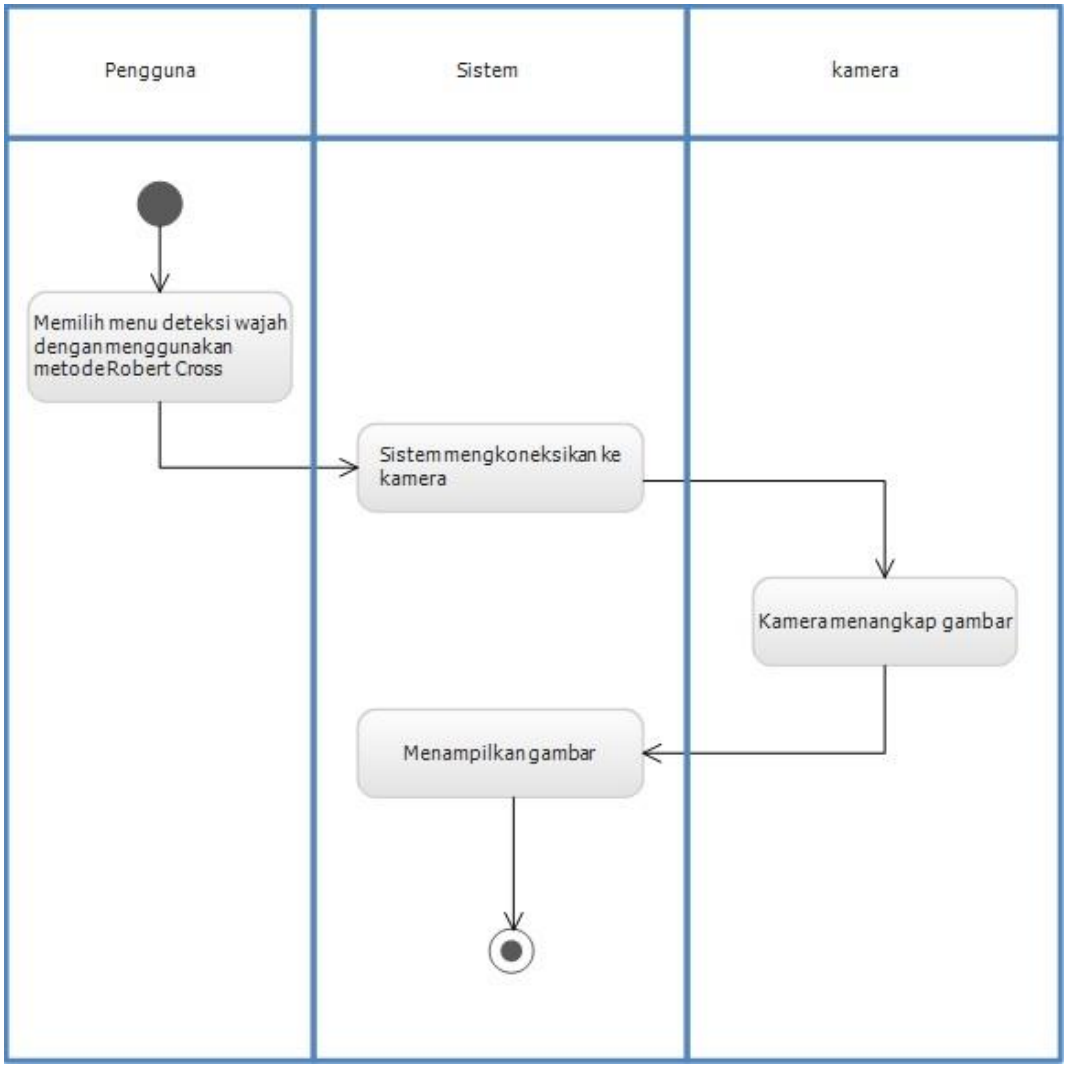
No	Aksi aktor	No	Reaksi sistem
Skenario normal			
1	Pengguna menghadapkan wajah ke depan kamera		
2	Kamera menangkap gambar		
		4	Mendeteksi wajah
		5	Menghitung kecepatan deteksi wajah
		6	Menampilkan informasi kecepatan deteksi wajah

3. Activity Diagram

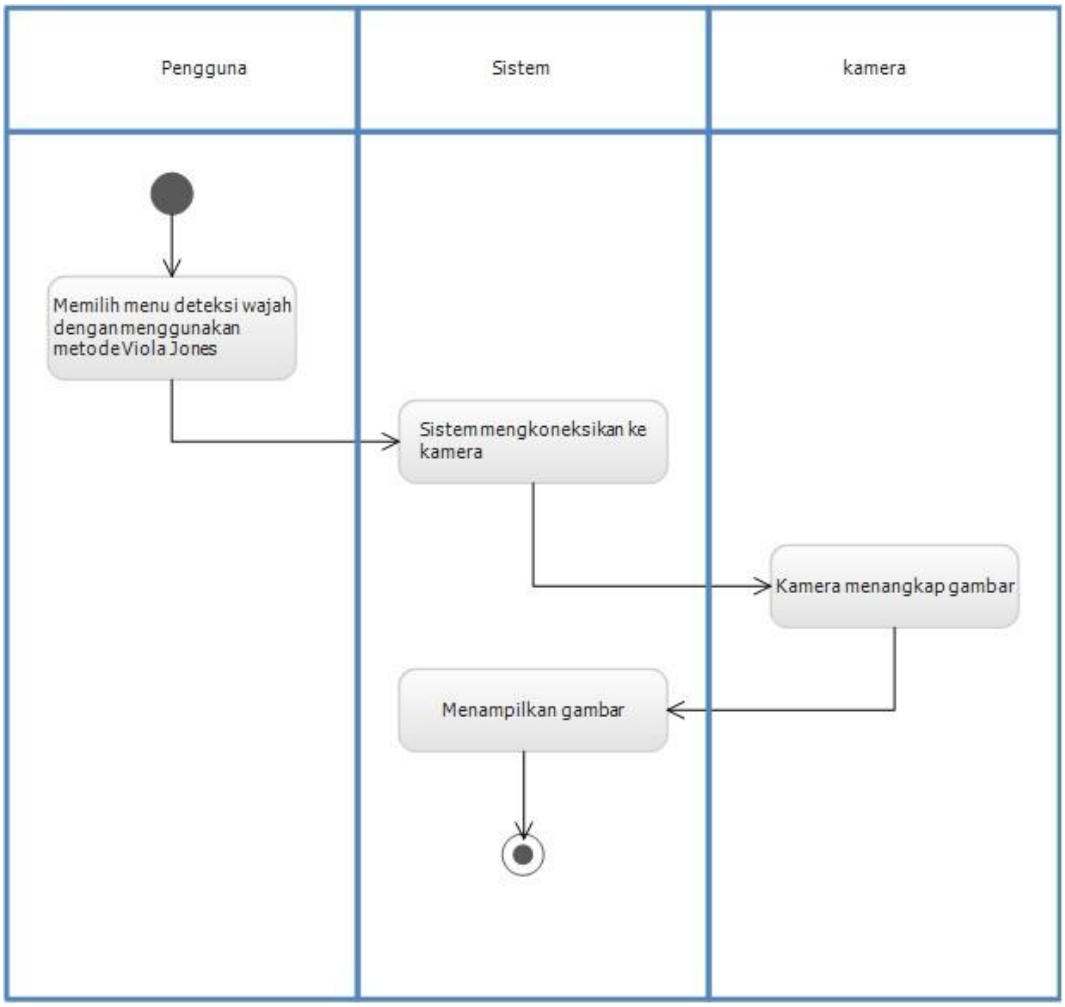
Untuk menjelaskan aktifitas setiap *use case* digunakan tool yang bernama *activity diagram.* Berikut ini merupakan gambaran aktifitas pada setiap *use case* :



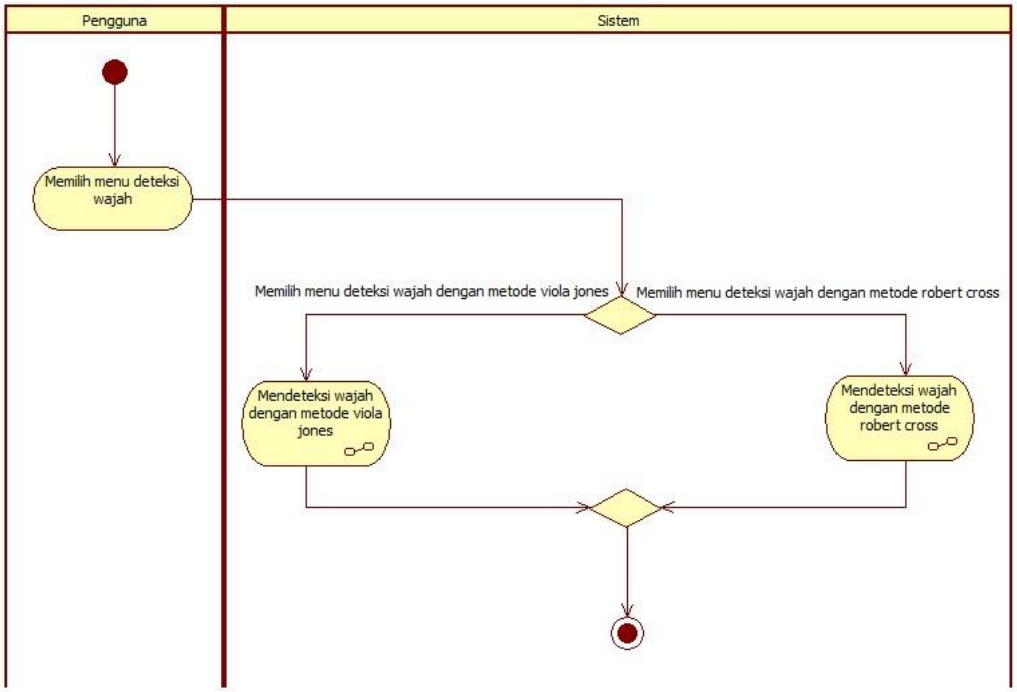
Gambar III-13 Activity Diagram Capture Gambar



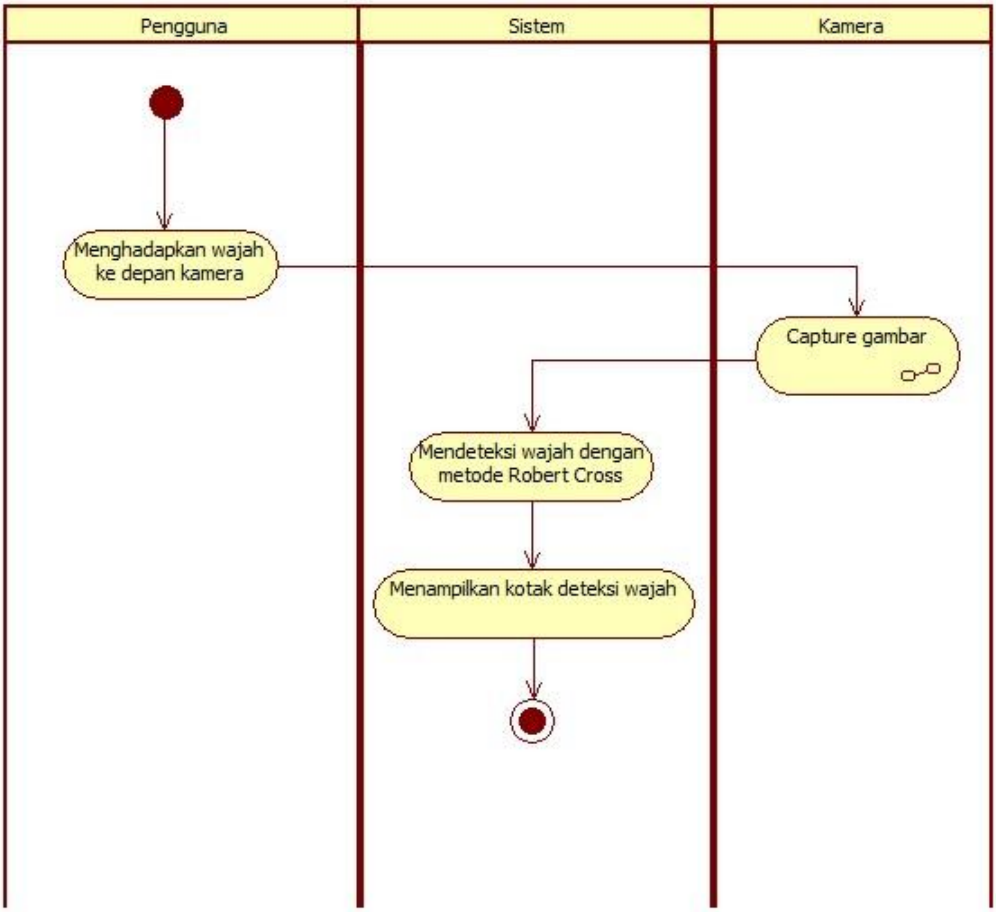
Gambar III-14 Activity Diagram Capture Gambar Dengan Menggunakan Metode Robert Cross



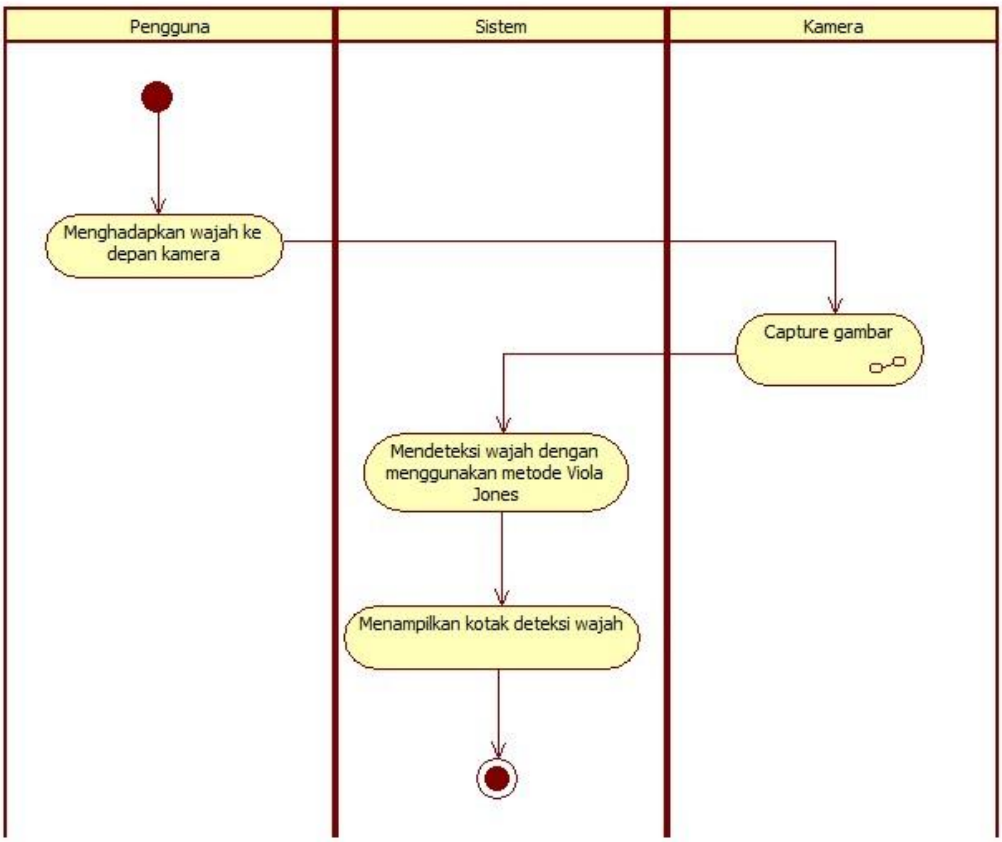
Gambar III-15 Activity Diagram Capture Gambar Dengan Menggunakan Metode Viola Jones



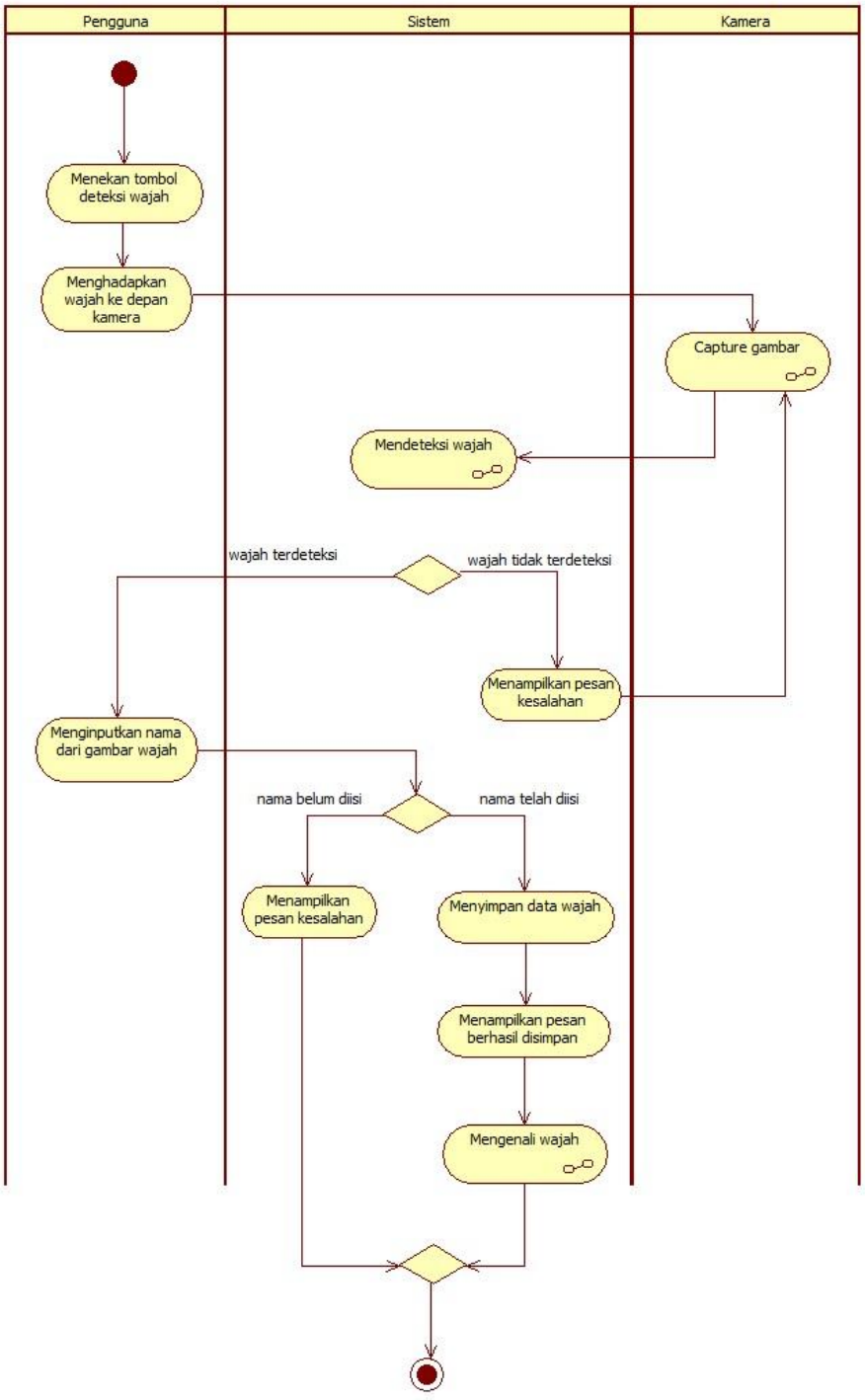
Gambar III-16 Activity Diagram Deteksi Wajah



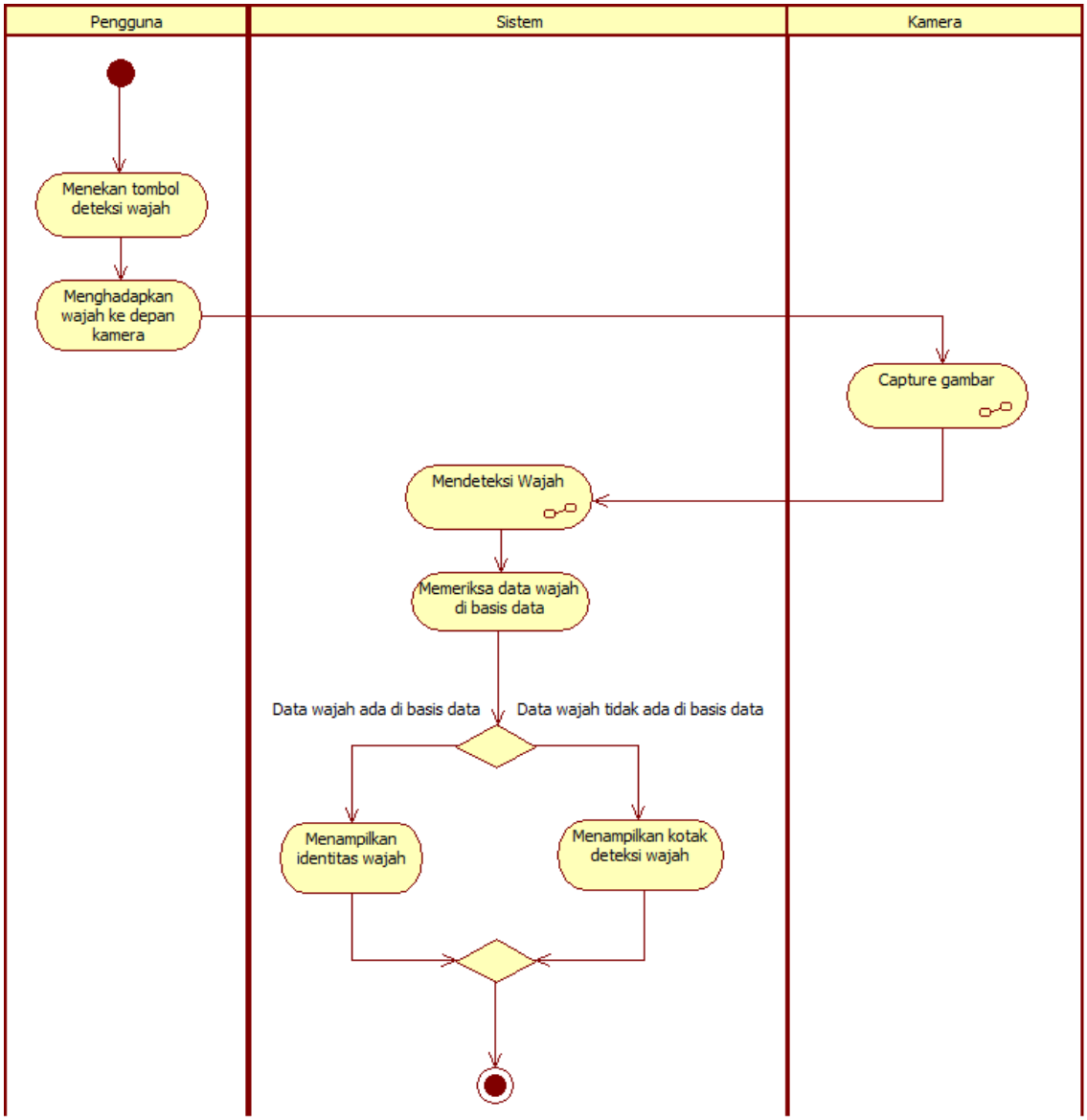
Gambar III-17 Activity Diagram Deteksi Wajah Dengan Menggunakan Metode Robert Cross



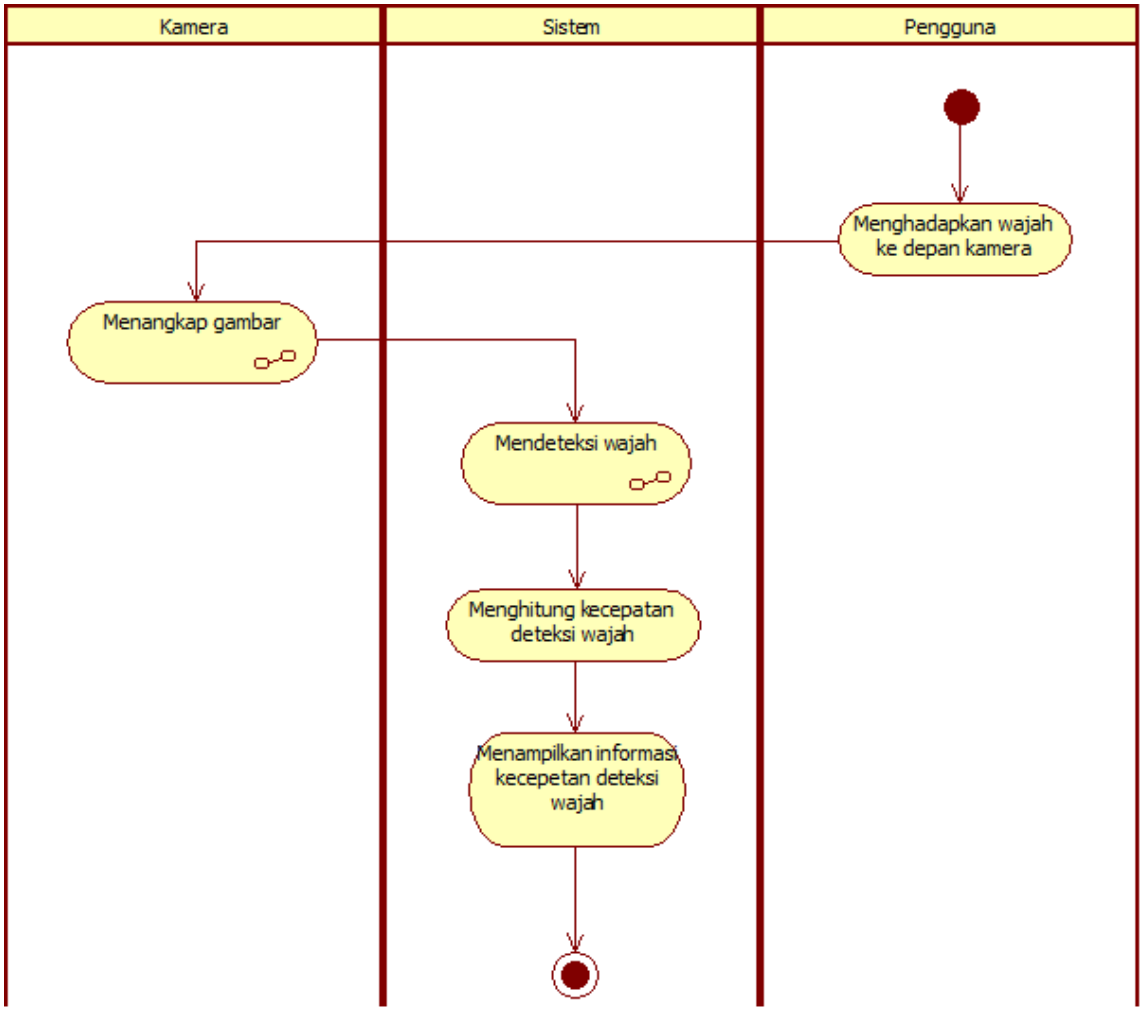
Gambar III-18 Activity Diagram Deteksi Wajah Dengan Menggunakan Metode Viola Jones



Gambar III-19 Activity Diagram Pengolahan Data Wajah

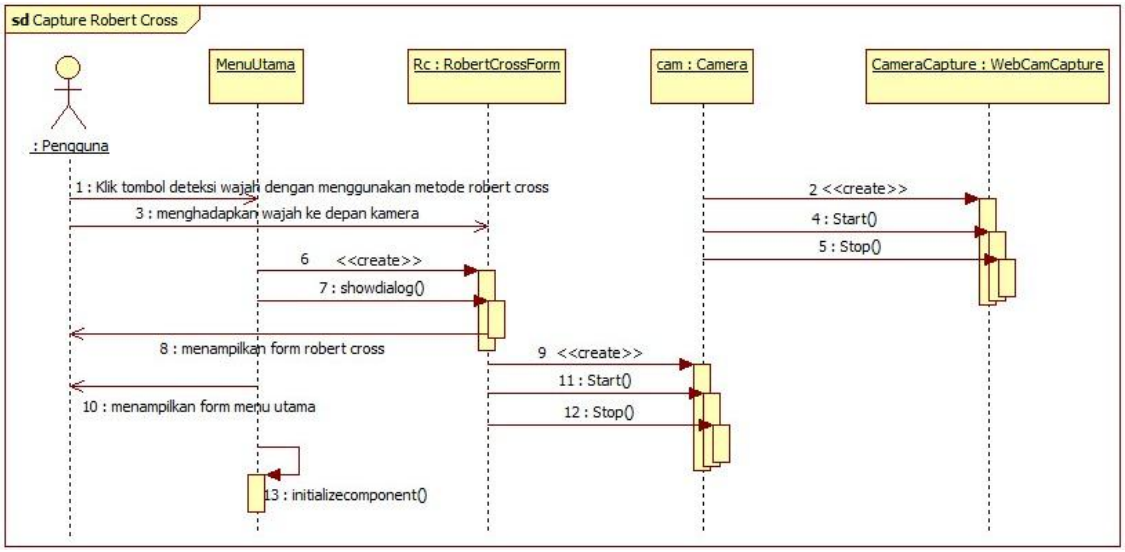


Gambar III-20 Activity Diagram Pengenalan Wajah

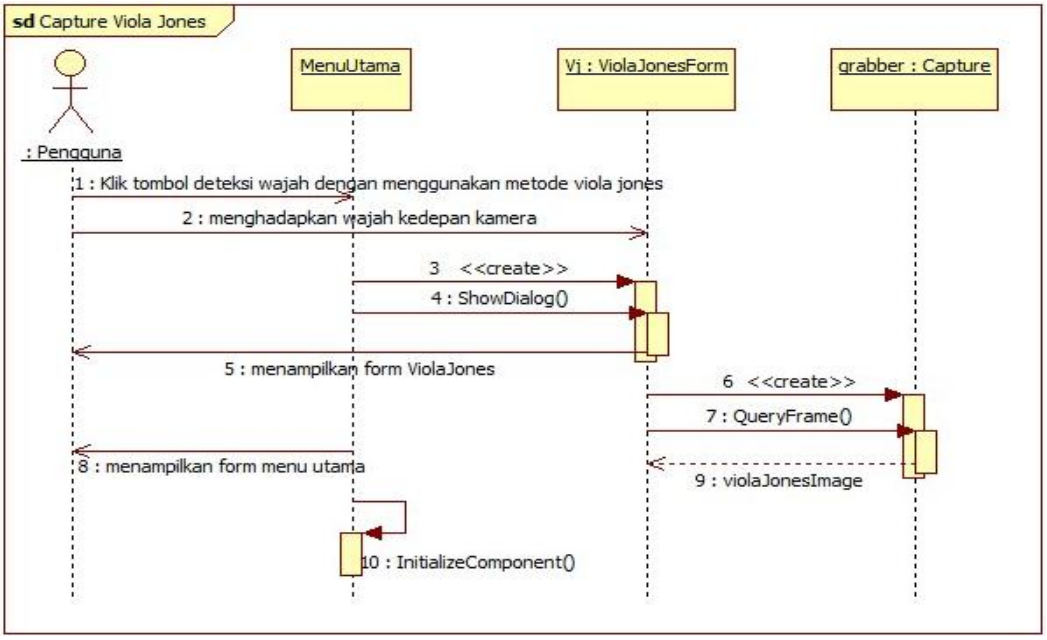


Gambar III-21 Activity Diagram Hitung Kecepatan Deteksi Wajah

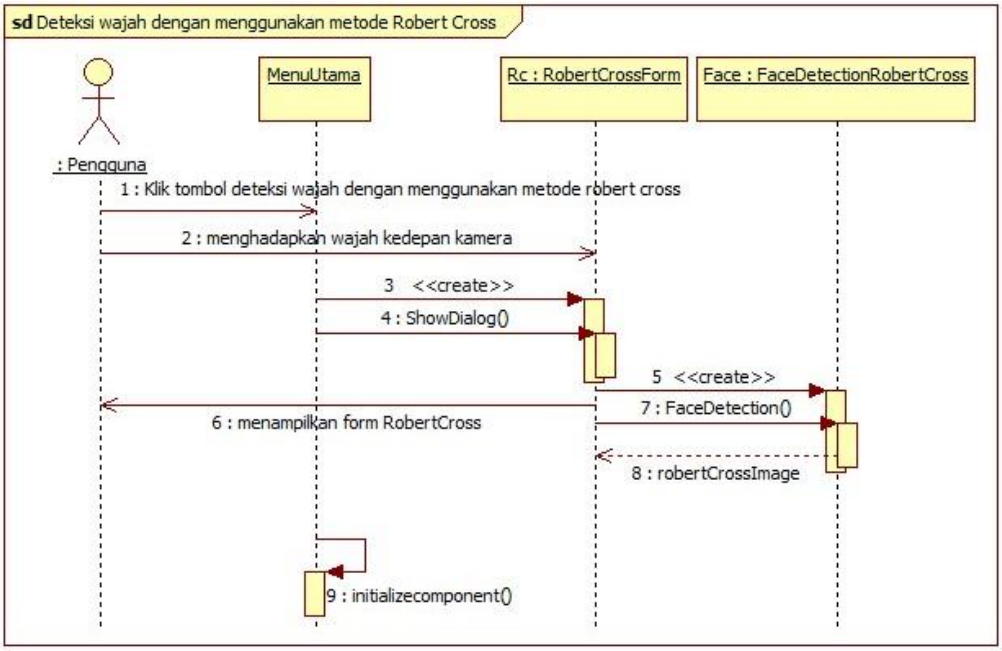
5. Sequence diagram



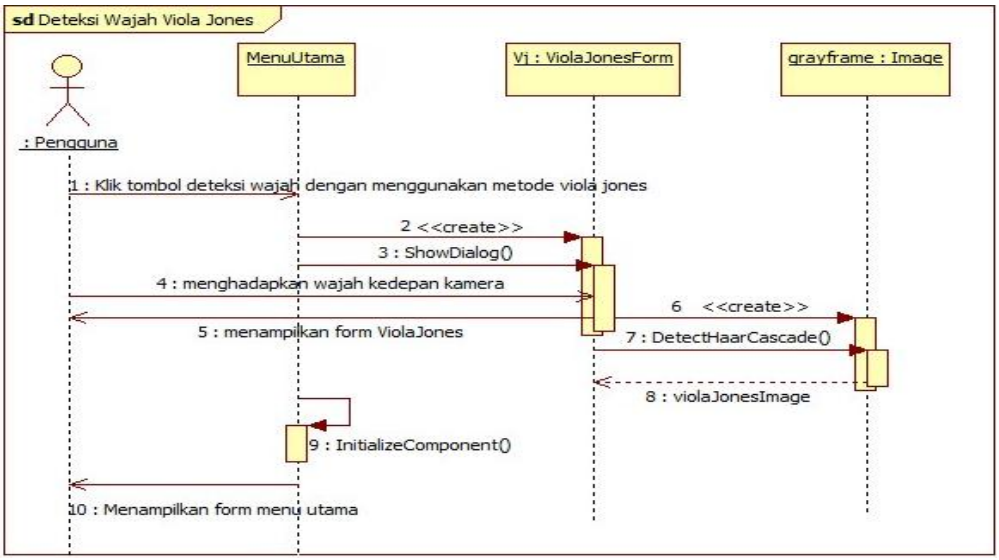
Gambar III-22 Sequence Diagram Capture gambar dengan Metode Robert Cross



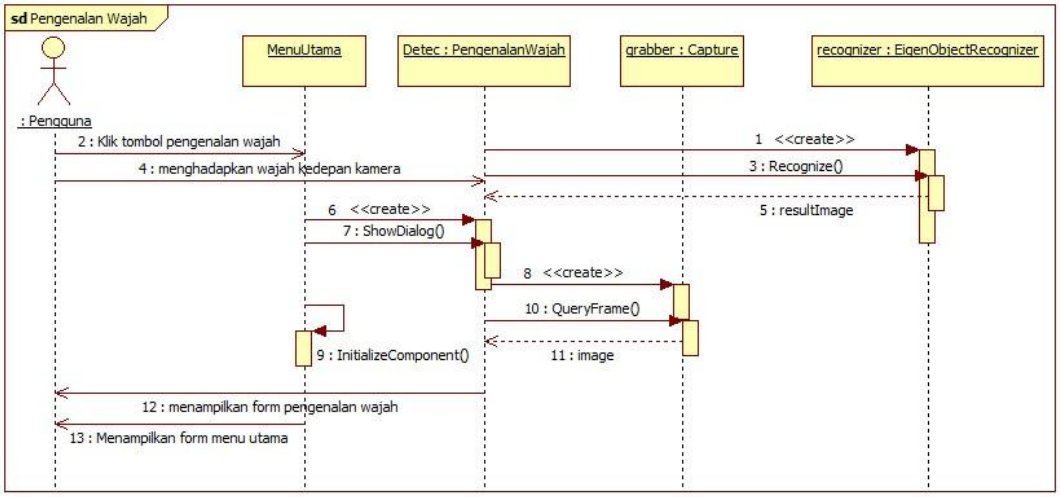
Gambar III-23 Sequence Diagram Capture Gambar dengan Metode Viola Jones



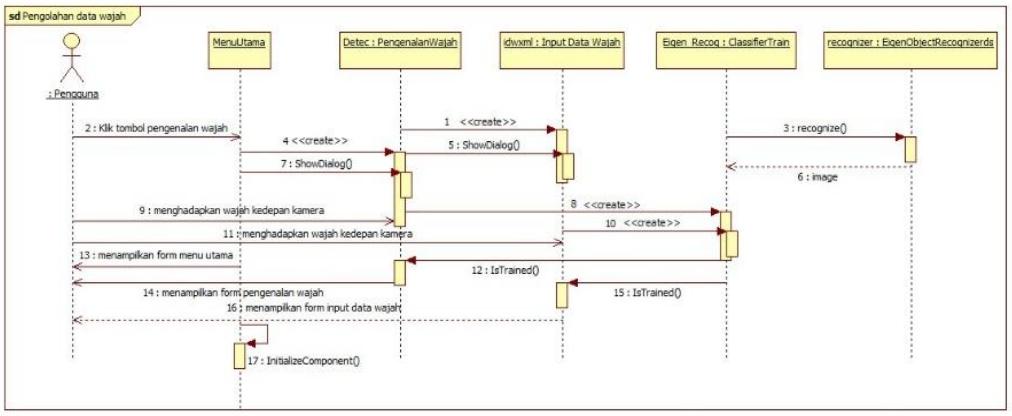
Gambar III-24 Sequence Diagram Deteksi Wajah dengan Menggunakan Metode Robert Cross



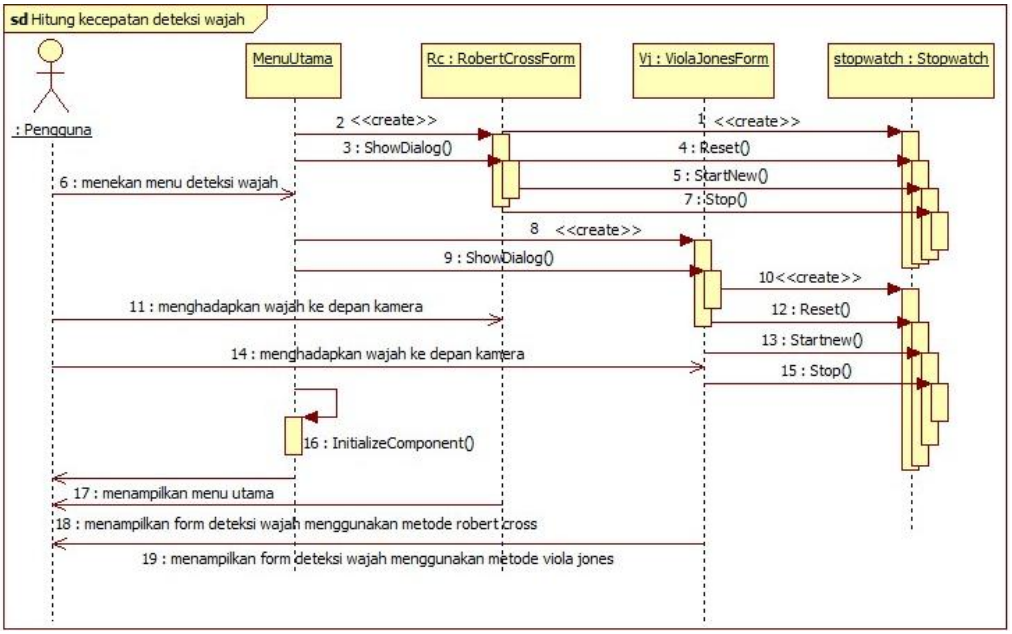
Gambar III-25 Sequence Diagram Deteksi Wajah dengan menggunakan Metode Viola Jones



Gambar III-26 Sequence Diagram Pengenalan Wajah

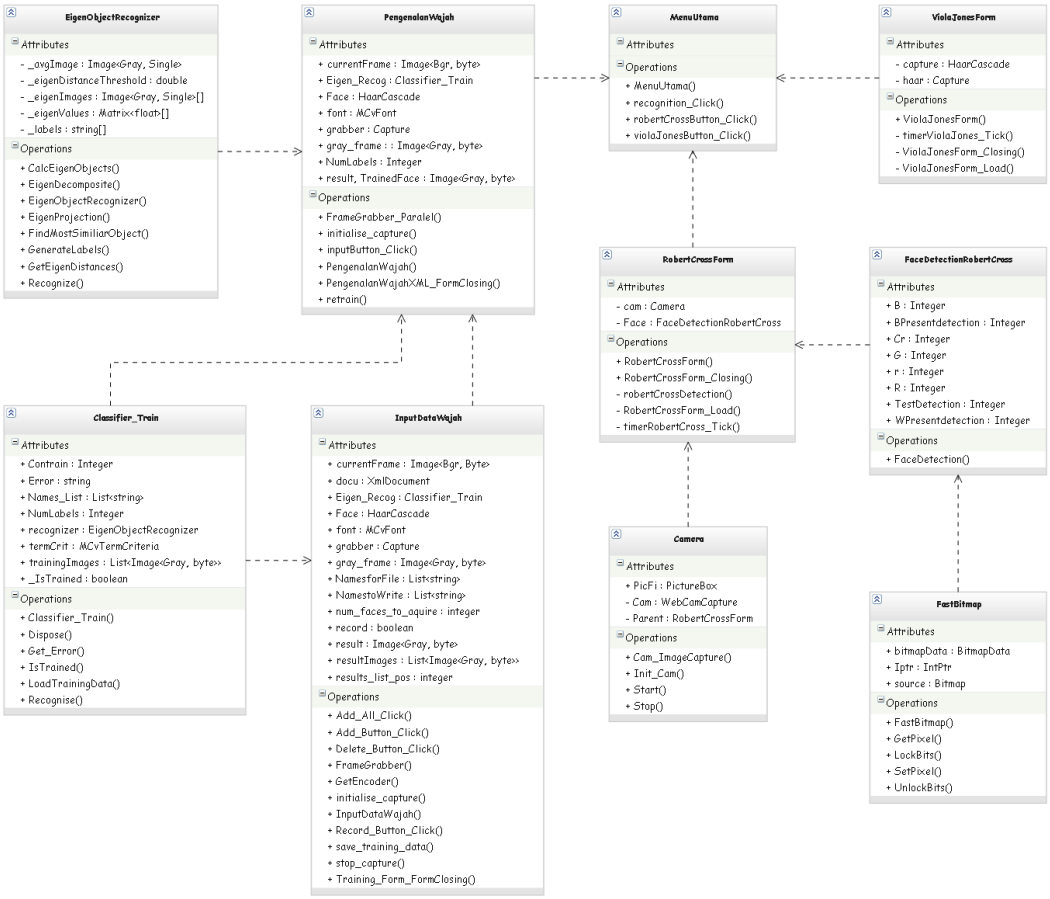


Gambar III-27 Sequence Diagram Pengolahan Data Wajah



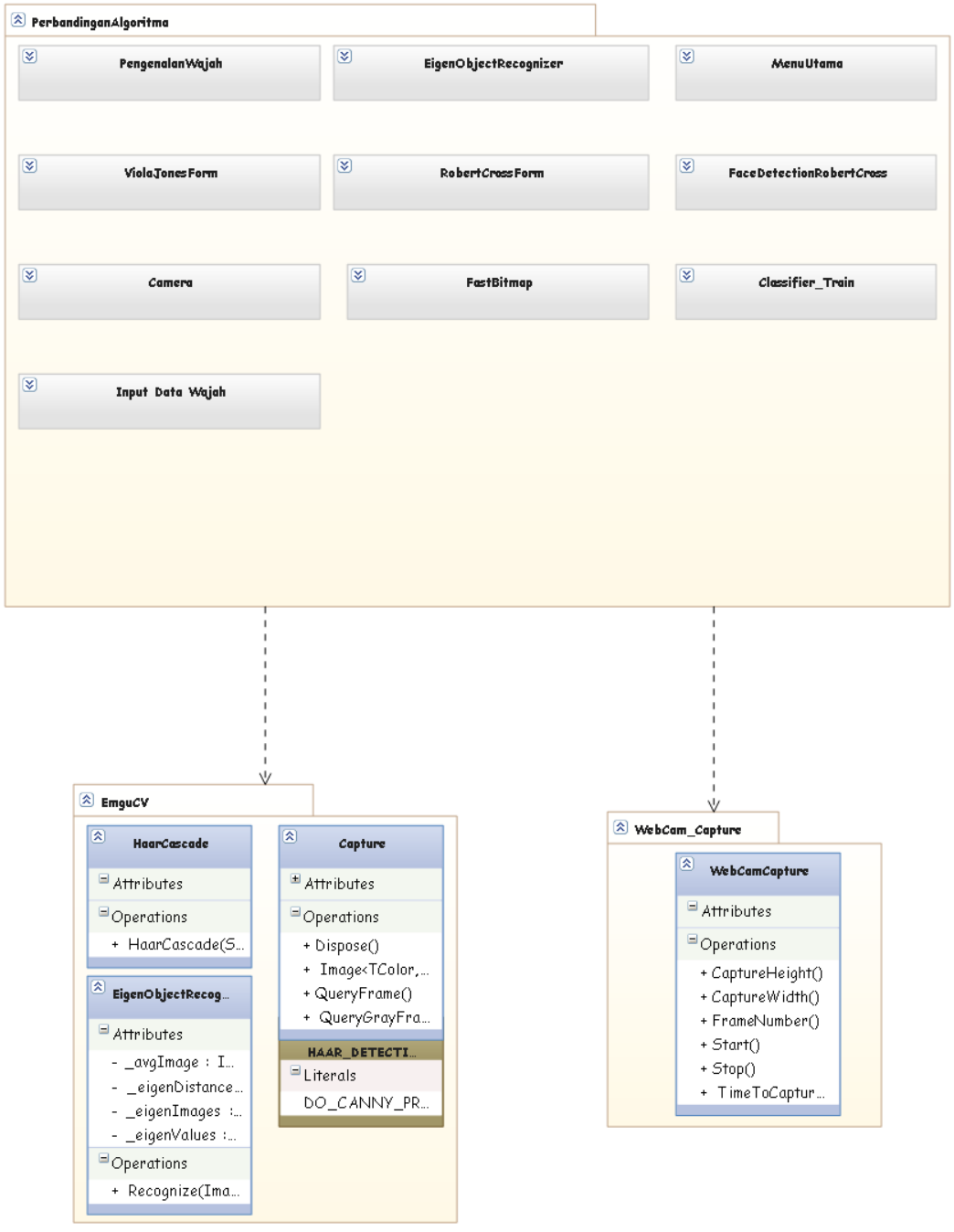
Gambar III-28 Sequence diagram hitung kecepatan deteksi wajah

6. Class Diagram



Gambar III-29 Class Diagram

7. Package Diagram



Gambar III-30 Package Diagram

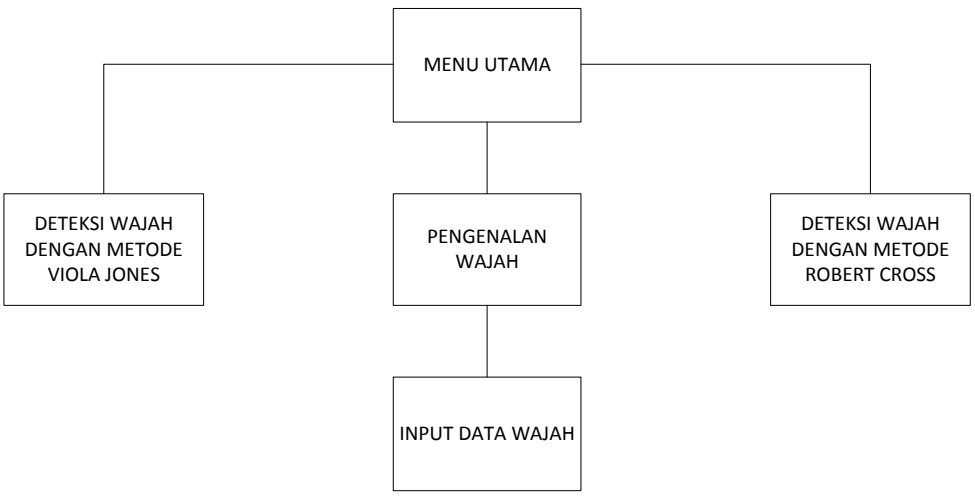
III.2 Perancangan Sistem

Perancangan sistem merupakan penggambaran, perencanaan, dan pembuatan sketsa atau pengaturan dari beberapa elemen yang terpisah ke dalam suatu kesatuan yang utuh. Tahapan ini meliputi mengkonfigurasi komponen-komponen perangkat lunak dan perangkat keras dari suatu sistem. Adapun perancangan sistem dari perangkat lunak yang akan dibuat dijelaskan sebagai berikut.

III.2.1 Perancangan Arsitektur Perangkat Lunak

Perancangan arsitektur merupakan perancangan yang dibuat sebelum perangkat lunak dibuat. Perancangan Arsitektur yang digunakan dalam perangkat lunak ini adalah berbasis desktop. Sedangkan perancangan arsitektur menu dalam perangkat lunak ini adalah *sequential linear*.

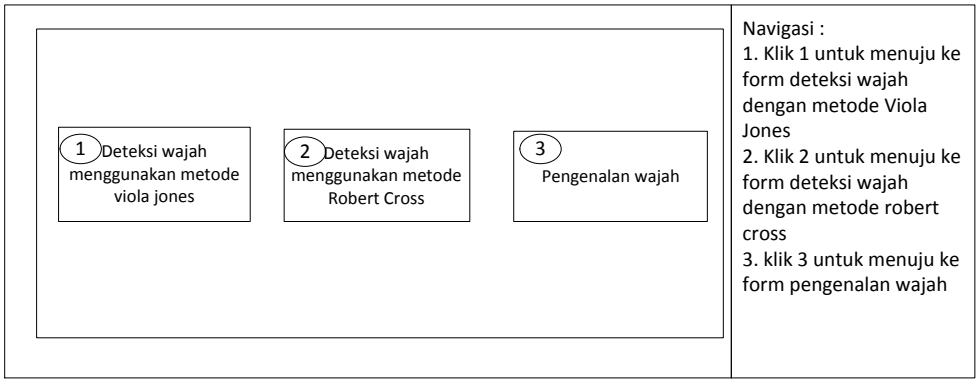
III.2.2 Perancangan Struktur Menu



Gambar III-31 Perancangan Struktur Menu

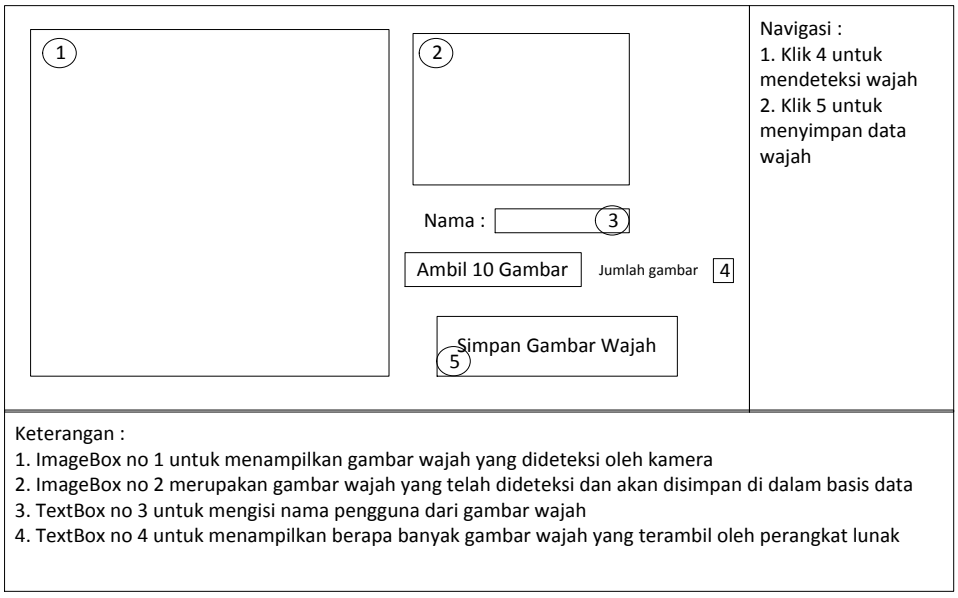
III.2.3 Perancangan Antarmuka Perangkat Lunak

1. Antarmuka menu utama (F01)



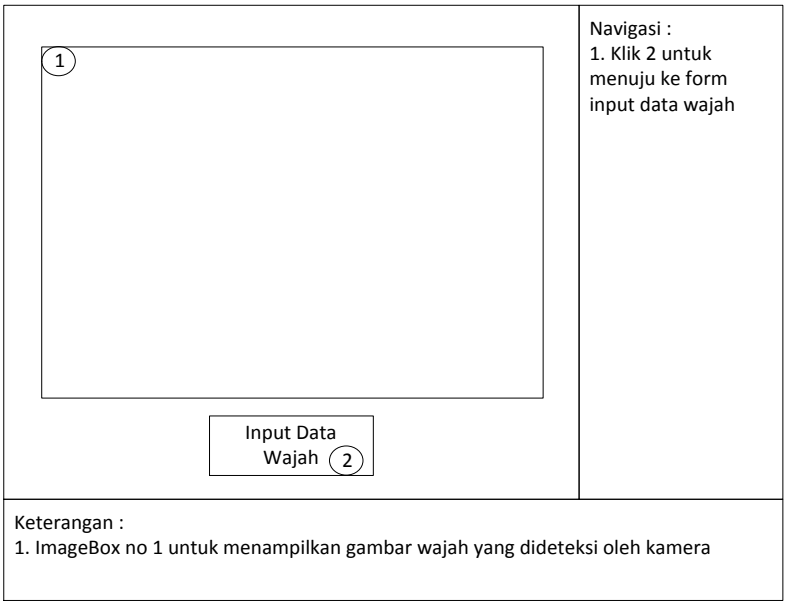
Gambar III-32 Form Menu Utama (F01)

2. Antarmuka pengenalan wajah (F02)



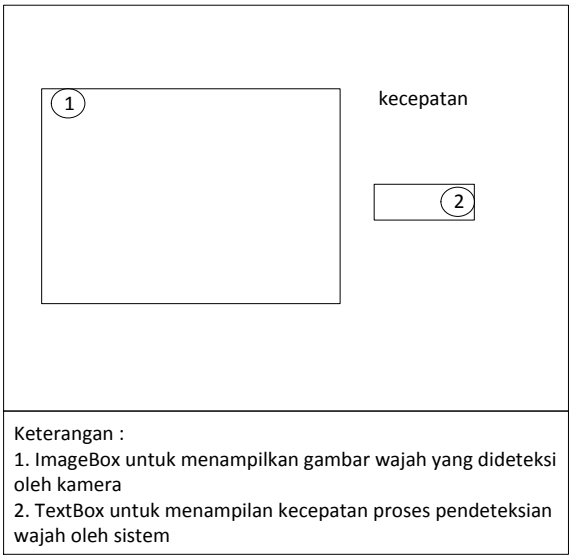
Gambar III-33 Form Pengenalan Wajah (F02)

3. Antarmuka input data wajah (F03)



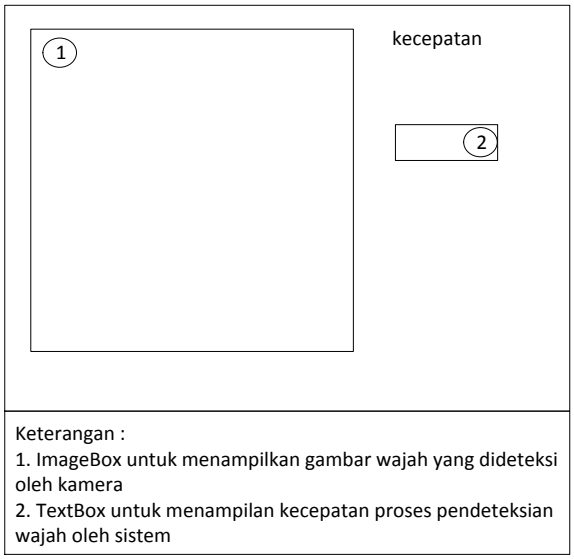
Gambar III-34 Form input Data Wajah

4. Antarmuka deteksi wajah menggunakan metode viola jones (F04)



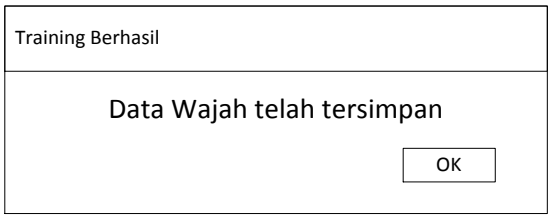
Gambar III-35 Form Deteksi Wajah dengan metode Viola Jones (F04)

5. Antarmuka deteksi wajah menggunakan metode Robert cross (F05)

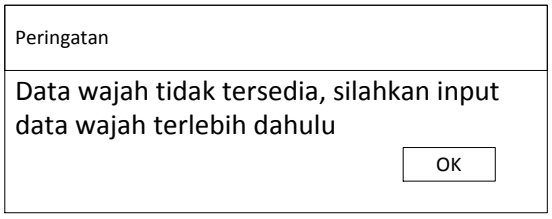


Gambar III-36 Form Deteksi Wajah dengan metode Robert Cross (F05)

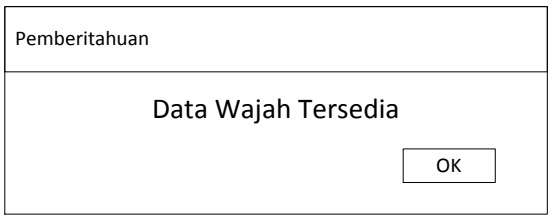
III.2.4 Perancangan Pesan



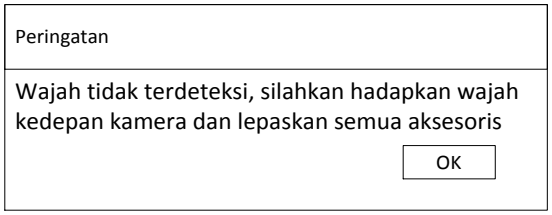
Gambar III-37 Pesan M01



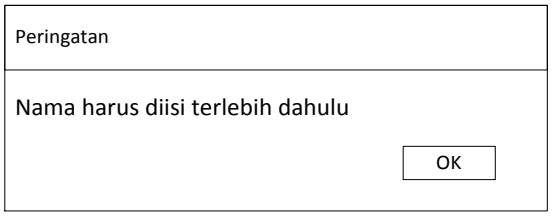
Gambar III-38 Pesan M02



Gambar III-39 Pesan M03

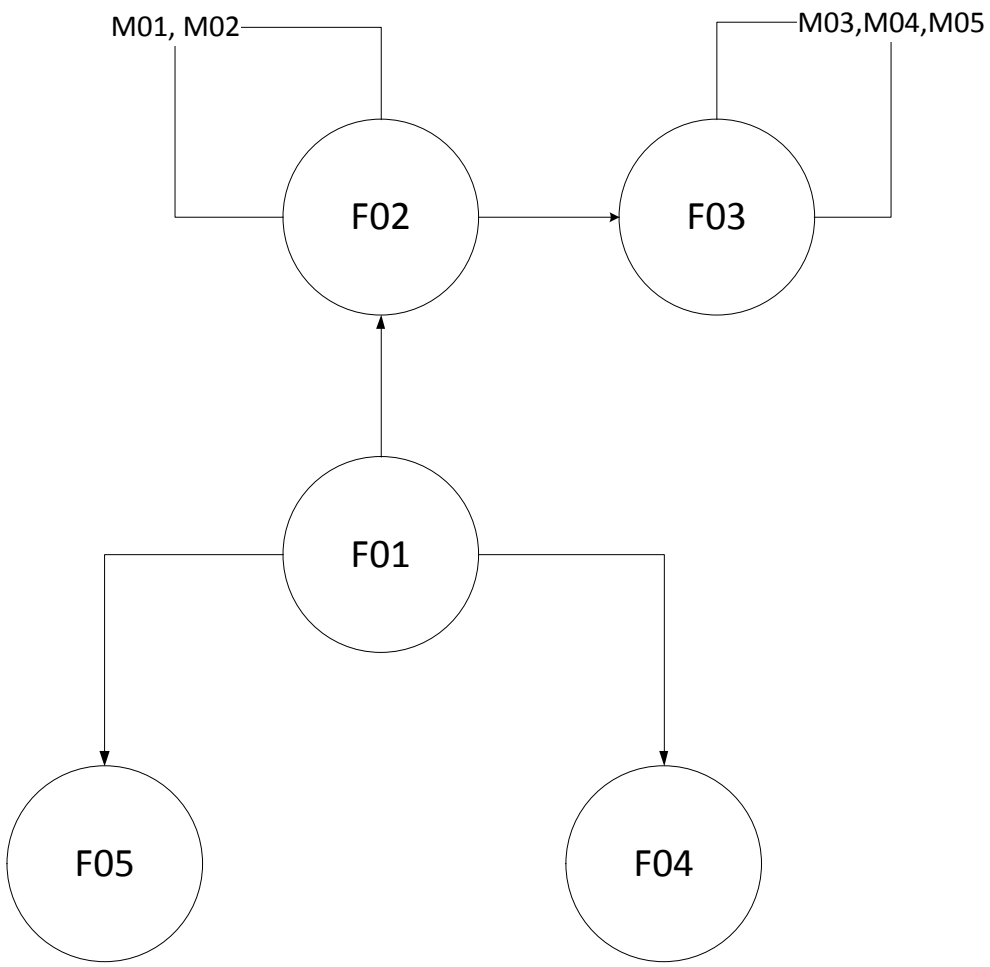


Gambar III-40 Pesan M04



Gambar III-41 Pesan M05

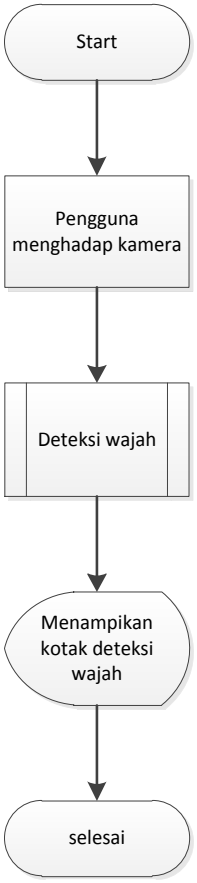
III.2.5 Jaringan Semantik



Gambar III-42 Jaringan Semantik

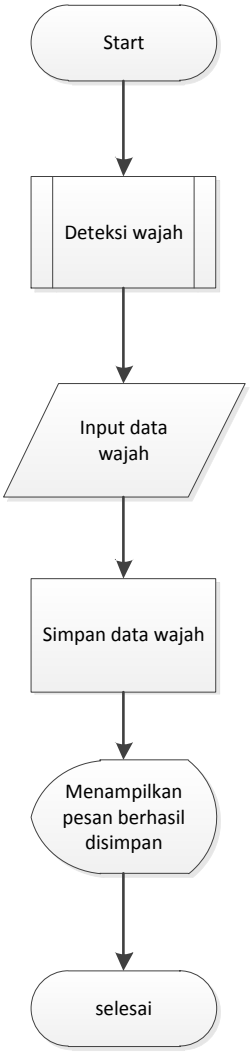
III.2.6 Perancangan prosedural

1. Prosedur deteksi wajah



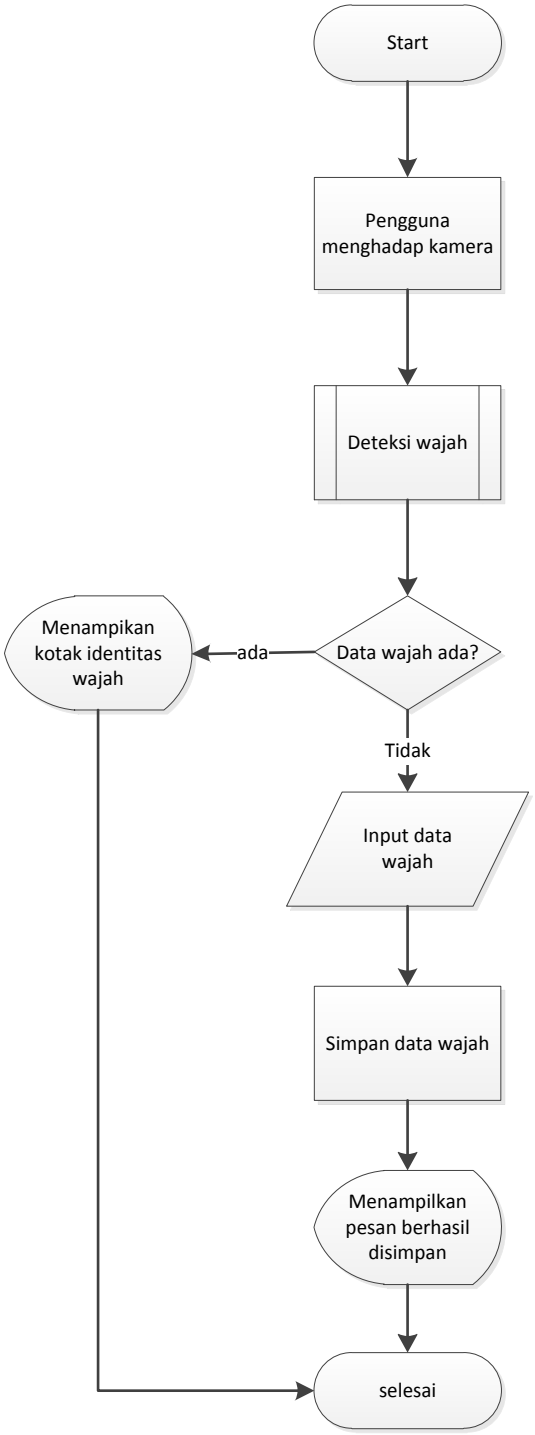
Gambar III-43 Prosedur deteksi wajah

2. Prosedur input data wajah



Gambar III-44 Prosedur input data wajah

3. Prosedur pengenalan wajah



Gambar III-45 Prosedur input data wajah