BAB 3
ANALISIS DAN PERANCANGAN

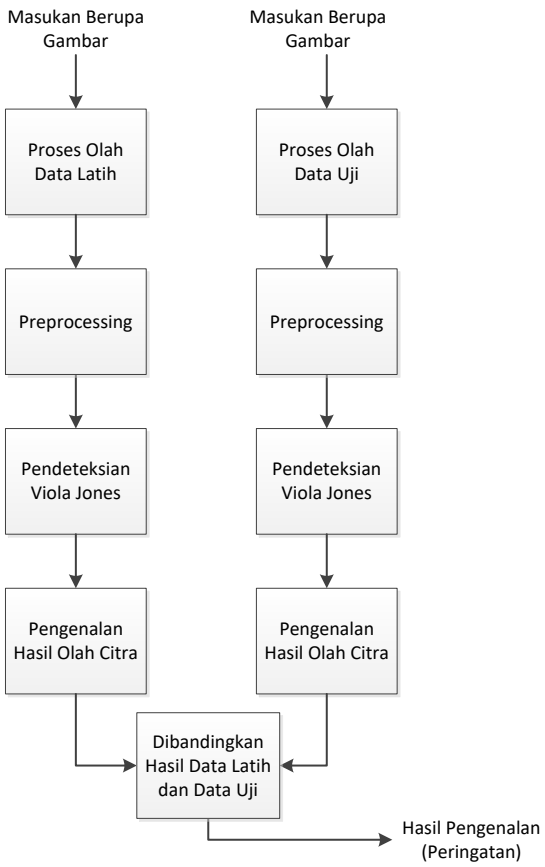
3.1 Analisis Masalah

Langkah ini diperlukan untuk mengetahui permasalahan apa saja yang terjadi didalam sistem yang sedang berjalan. Setelah penelitian terdapat masalah yang timbul yaitu dari penelitian "Tessa Simanjuntak" dengan judul "deteksi pelanggaran marka jalan berbasis kolerasi citra"[4] menyatakan akurasi sistem penelitian metode tersebut bernilai 82.10% keakuratannya, oleh karena itu pendeteksian pelanggaran marka jalan masih membutuhkan sebuah sistem dengan metode baru untuk menemukan hasil deteksi yang lebih akurat dibanding metode metode sebelumnya.

Pada penelitian ini dipilih suatu metode yaitu *Viola Jones.* Metode ini digunakan karena memiliki beberapa ciri fitur *haar* yang sama pada fitur kendaraan atau marka, dan terdapat beberapa fitur yang dapat digunakan pada deteksi pelanggaran yaitu fitur *haar* edge. Oleh karena itu suatu objek akan dapat dikenali oleh sistem melalui karakteristik atau ciri–ciri yang dihasilkan oleh perhitungan algoritma *Viola Jones* terhadap objek tersebut. Pada penelitian ini tentunya selain menggunakan metode *Viola Jones* tentunya penelitian ini tidak luput dari metode lain yang membantu dalam pengolahan data sehingga mendapat hasil yang diinginkan. Diantaranya metode yang digunakan pada tahap *preprocessing* yaitu *cropping*, *grayscaling* dan *thresholding*.

3.2 Analisis Sistem

Pada bagian ini menjelaskan tentang metode yang akan diterapkan pada sistem. Terutama penjelasan untuk bagian-bagian pada algoritma pengolahan citra yang akan diterapkan pada sistem pendeteksian pelanggaran pada lampu merah berdasarkan marka jalan.



Gambar 3. 1 Proses Utama Sistem

1. Citra masukan

   Merupakan langkah pengambilan data citra untuk masukan kedalam sistem

2. Proses olah data latih dan data uji

   Merupakan proses yang dilakukan dalam menentukan data latih dan data uji pada sistem sehingga menentukan objek yang akan dilatih dan diuji

3. Preprocessing

   Merupakan proses sebelum melakukan proses utama sehingga dapat berfungsi pada tahap selanjutnya

4. Pendeteksian menggunkan *Viola Jones*

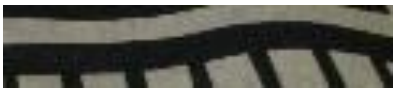
   Merupakan proses pendeteksian sistem. Pada tahap ini citra akan diolah menggunakan metode pada algoritma *viola-jones*

5. Pengenalan Hasil Pendeteksian

   Merupakan proses pengenalan/pencocokan dari pendeteksian sistem yang telah disimpan didalam sistem dengan pendeteksian yang diujikan.

Pada tahap analisis tentunya membutuhkan suatu citra masukan yang akan diproses, citra masukan berupa citra latih, citra latih merupakan citra yang akan digunakan dalam proses latih sebuah metode sehingga menghasilkan data ang dimaksud. Adapun contoh citra masukan yang akan diproses sebagai berikut.





Citra 1 Citra 2

Gambar 3. 2 Citra Masukan 340x240

3.3 Proses Olah Data Latih

Proses pengolahan pada citra dengan metode yang telah ditentukan sebelum dilakukan proses selanjutnya terhadap citra 1.

3.4 Tahap Preprocessing

Proses preprocessing dilakukan dengan tujuan untuk membantu proses utama sehingga suatu sistem bisa berjalan. Pada analisis ini *preprocessing* terdapat metode yaitu *grayscale, thresholding*

3.4.1 Tahap *Grayscale*

Proses pengubahan nilai warna RGB pada setiap pixel citra kedalam derajat keabuan. Untuk mengubah citra kebentuk *grayscale* menggunakan rumus 2.1

Contoh untuk menentukan nilai *grayscale* diilustrasikan dengan nilai-nilai R = 24, G = 23, dan B = 21 pada gambar 3.4, untuk konversi warna menjadi *grayscale* digunakan persamaan 2.1. Misalkan untuk pixel di koordinat (0, 0) perhitungannya sebagai berikut:

Grayscale = (0.299 x R) + (0.587 x G) + (0.114 x B) [2.1]

= (0.299 x 24) + (0.587 x 23) + (0.114 x 21)

= 23.071

Lalu hasil tersebut dilakukan proses pembulatan, maka nilai nya

= 23

Dengan contoh dari rumus diatas lalu diterapkan ke setiap pixel yang terdapat pada citra. Untuk selengkapnya bisa dilihat dilampiran. Setelah

mendapatkan nilai RGB Maka nilai *Grayscale* yang diperoleh setelah memasukan rumus (2.1) yaitu. Selengkapnya terdapat pada lampiran

	0	1	2	3	4	5	6	7	8	9
0	23	36	135	125	122	118	127	132	122	117
1	24	42	132	125	127	123	122	130	127	120
2	15	32	132	127	137	123	119	114	97	84
3	23	20	21	22	23	22	14	17	16	15
4	28	16	21	23	10	21	21	25	24	21
5	30	22	28	22	26	19	21	21	20	17
6	24	28	17	16	29	20	21	19	19	16
7	22	20	25	26	18	21	19	19	23	18
8	25	21	23	21	19	15	18	20	24	16
9	22	26	22	22	26	19	21	21	25	16

Gambar 3. 3 Matrik *Grayscale*

Setelah proses *grayscale* dilakukan lalu dapat dilakukan proses perhitungan selanjutnya. Untuk perhitungan *grayscale* lebih jelas terdapat pada lampiran.



Gambar 3. 4 Hasil Citra *Grayscale*

3.4.2 Tahap *Thresholding*

Selanjutnya adalah tahap *thresholding* yang digunakan untuk mengubah citra hasil *grayscale* menjadi citra biner. Dari hasil *grayscale* pada koordinat (0.0) dengan nilai 23 kemudian dibandingkan dengan nilai T ambang batas, ambang batas yang diambil bernilai 85 karena dengan nilai tersebut kondisi dari citra hitam putih pada *threshold* terlihat jelas.

$$g(x,y) = \begin{cases} 1 & if \ \ f(x,y) > T \\ 0 & if \ \ f(x,y) \leq T \end{cases} \qquad [2.2]$$

Karena pada koordinat (0.0) dengan nilai $23 \leq 83$ maka nilai *threshold* pada koordinat (0.0) tersebut menjadi 0 kemudian dilakukan proses yang sama terhadap keseluruhan nilai *grayscale* sehingga menjadi biner. Berikut ini adalah hasil pengubahan citra *grayscale* menjadi citra biner, untuk selengkapnya bisa di liaht dalam lampiran.

	0	1	2	3	4	5	6	7	8	9
0	0	0	1	1	1	1	1	1	1	1
1	0	0	1	1	1	1	1	1	1	1
2	0	0	1	1	1	1	1	1	1	0
3	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0

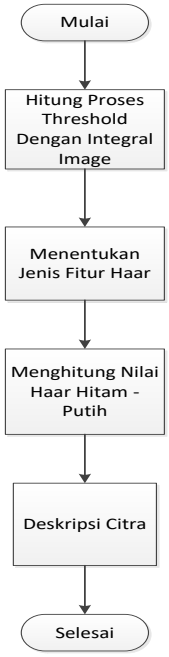
Gambar 3. 5 Proses *Grayscale* ke Biner



Gambar 3. 6 Hasil Threshold

3.5 Tahap *Viola Jones*

Data masukan yang digunakan berupa citra hasil *threshold* yang telah diselesaikan pada tahap *preprocessing*.



Gambar 3. 7 Alur *Viola Jones*

3.5.1 *Integral image*s

Setelah tahap nilai *threshold* dilakukan selanjutnya menghitung nilai *integral image* yang datanya diambil dari nilai *threshold* kemudian dilakukan proses perhitungan dengan persamaan.

$$s(x,y) = s(x,y)+s(x-1,y)+s(x,y-1)-s(x-1,y-1) \quad [2.4]$$

Kemudian sebagai contoh pada koordinat (1.2)

$$s(1,2) = s(1,2) + s(0,2) + s(1,1) - s(0,2) = 2$$

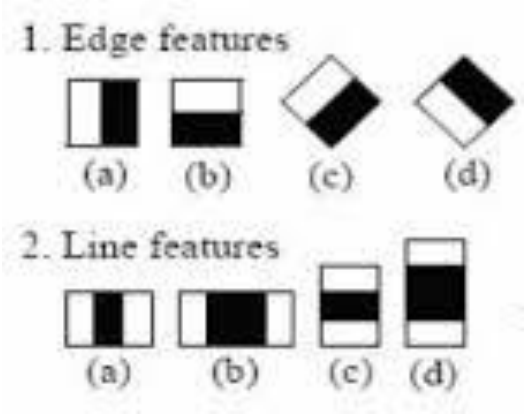
Berikut adalah matrik 130, 30 namun sebagai contoh yang dihitung dengan ukuran 10x10 selengkapnya terdapat pada lampiran.

	-	0	1	2	3	4	5	6	7	8	9
-	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	2	3	4	5	6	7	8
1	0	0	0	2	4	6	8	10	12	14	16
2	0	0	0	3	6	9	12	15	18	21	23
3	0	0	0	3	6	9	12	15	18	21	23
4	0	0	0	3	6	9	12	15	18	21	23
5	0	0	0	3	6	9	12	15	18	21	23
6	0	0	0	3	6	9	12	15	18	21	23
7	0	0	0	3	6	9	12	15	18	21	23
8	0	0	0	3	6	9	12	15	18	21	23
9	0	0	0	3	6	9	12	15	18	21	23

Gambar 3. 8 Matrik Biner 130x30

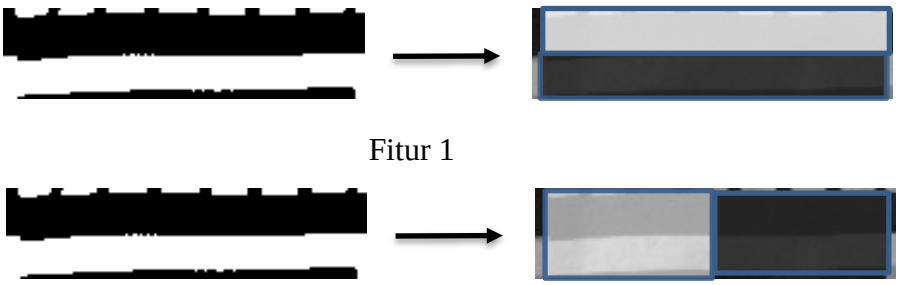
3.5.2 *Haar* Like Feature

Algoritma *Haar* menggunakan metode statistikal dalam melakukan pendeteksian objek. Tiap feature dari *Haar*-like feature didefinisikan pada bentuk dari feature, diantaranya koordinat dari feature dan juga ukuran dari feature tersebut. Feature yang akan di training adalah feature yang didapat dari proses cropping. Untuk feature yang digunakan pada penelitian ini menggunakan jenis edge feature.



Gambar 3. 9 Edge Feature

Selanjutna menentukan jenis *haar* yang akan diterapkan pada hasil perhitungan sebelumnya. Hal ini dilakukan dengan mengevaluasi setiap fitur terhadap data latih dengan menggunakan nilai dari fitur tersebut. Fitur yang memiliki batas terbesar antara pelanggaran dan bukan pelanggaran dianggap sebagai fitur terbaik.



Fitur 1

Fitur 2

Gambar 3. 10 Menentukan Jenis Haar

Setelah menentukan fitur *haar* selanjutnya dilakukan perhitungan dari hasil *integral image* ang telah dilakukan, sehingga dapat diketahui nilai *threshold*nya. Berikut perhitungan nilai latih pada fitur 1. Nilai yang didapat pada fitur hitam A_{hitam} = 0, B_{hitam} = 157, C_{hitam} = 12, D_{hitam} = 1703. Sedangkan pada fitur putih A_{putih} = 0, B_{putih} = 92, C_{putih} = 0, D_{putih} = 157. Data dapat dilihat pada lampiran.

f(x) =*Sumblack rectangle (pixel gray level) – Sumwhite rectangle (pixel gray level)*

$= (D_{hitam} + A_{hitam} - (B_{hitam} + C_{hitam})) - (D_{putih} + A_{putih} - (B_{putih} + C_{putih}))$

= (1703+0 – (157+12)) – (157 + 0 – (92+0))

= 1534 - 65

= 1469

Berikut perhitungan nilai latih pada fitur 2. Nilai yang didapat pada fitur hitam A = 47, B = 92, C = 842, D = 1703. Sedangkan nilai fitur putih A = 0, B = 46, C = 12, D = 829. Data dapat dilihat pada lampiran L. 3.6

f(x) =*Sumblack rectangle (pixel gray level) – Sumwhite rectangle (pixel gray level)*

$= (D_{hitam} + A_{hitam} - (B_{hitam} + C_{hitam})) - (D_{putih} + A_{putih} - (B_{putih} + C_{putih}))$

= (1703+47 – (92+842) – (829+0 – (46+12)

= 816 - 771

= 45

3.6 Proses Olah Data Uji

Pada proses ini akan dilakukan langkah-langkah yang sama yang dilakukan pada proses olah data latih. Sehingga menghasilkan hasil yang sama pada hasil akhir perhitungan. Tetapi terdapat citra uji 320x240 sebagai masukan dalam pendeteksian. Berikut citra uji yang akan diolah:



Gambar 3. 11 Citra Masukan Uji

3.7 Tahap Preprocessing Uji

Proses preprocessing dilakukan dengan tujuan untuk membantu proses utama sehingga suatu sistem bisa berjalan. Langkah yang digunakan sama dengan data latih aitu *grayscale, thresholding*

3.7.1 Tahap *Grayscale* Uji

Proses pengubahan nilai warna RGB pada setiap pixel citra kedalam derajat keabuan. Untuk mengubah citra kebentuk *grayscale* menggunakan rumus 2.1. Untuk menentukan nilai *grayscale* diilustrasikan dengan nilai-nilai R = 85, G = 84, dan B = 80 pada gambar 3.14, untuk konversi warna menjadi *grayscale* digunakan persamaan 2.1. selengkapnya pada lampiran L 3.8 .Misalkan untuk pixel di koordinat (0, 0) perhitungannya sebagai berikut

Grayscale = (0.299 x R) + (0.587 x G) + (0.114 x B) [2.1]

= (0.299 x 85) + (0.587 x 84) + (0.114 x 80)

= 83.73

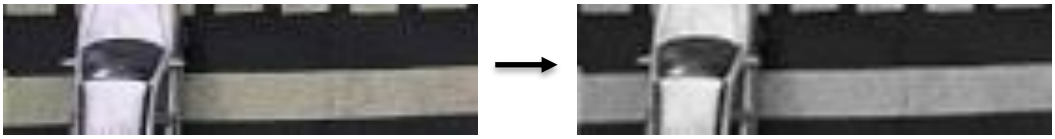
Lalu hasil tersebut dilakukan proses pembulatan, maka nilai nya

= 84

	0	1	2	3	4	5	6	7	8	9
0	84	145	147	153	143	145	143	152	148	145
1	57	108	112	118	122	112	122	116	102	104
2	35	19	28	28	34	24	21	31	23	31
3	35	33	36	28	37	32	41	36	35	37
4	39	40	35	34	36	41	36	37	34	39
5	35	34	40	35	37	37	37	36	36	41
6	33	31	34	34	37	36	34	35	33	42
7	37	37	35	38	40	41	39	36	42	41
8	36	39	33	39	37	40	42	38	38	44
9	34	34	31	38	36	38	41	32	41	34

Gambar 3. 12 Matrik *Grayscale* Uji

Setelah proses *grayscale* dilakukan lalu dapat dilakukan proses perhitungan selanjutnya. Untuk perhitungan *grayscale* lebih jelas terdapat pada lampiran.



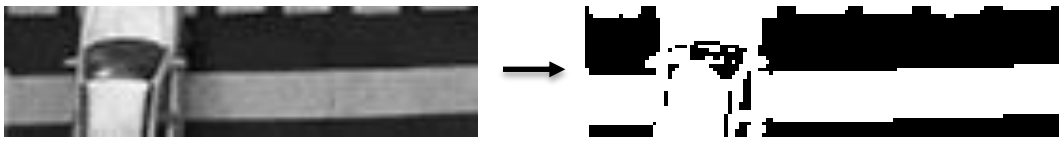
Gambar 3. 13 Hasil *Grayscale* Uji

3.7.2 Tahap *Threshold* Uji

Dari hasil *grayscale* uji dilakukan *threshold* uji dengan cara yang sama pada citra dengan menggunakan rumus 2.3. untuk data lengakp nya terdapat pada lampiran. Maka hasil dari *threshold* tersebut yaitu:

	0	1	2	3	4	5	6	7	8	9
0	0	1	1	1	1	1	1	1	1	1
1	0	1	1	1	1	1	1	1	1	1
2	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0

Gambar 3. 15 *Threshold* Uji



Gambar 3. 14 Hasil Threshold Uji

3.8 Tahap *Viola Jones* Uji

Dari hasil *threshold* uji yang telah didapatkan lalu dilakukan proses *integral image* dan *haar* Dengan cara yang sama yang telah dilakukan pada proses data latih.

3.8.1 *Integral image*

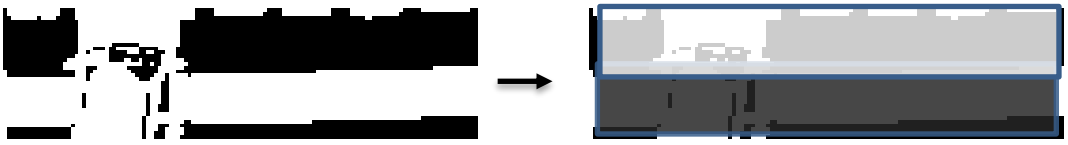
Setelah tahap nilai *threshold* dilakukan selanjutnya menghitung nilai *integral image* yang datanya diambil dari nilai *threshold* yang terdapat pada citra uji, berikut adalah matrik 130, 30 namun sebagai contoh yang dihitung dengan ukuran 10x10 selengkapnya dilampiran.

	-	0	1	2	3	4	5	6	7	8	9
-	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	2	3	4	5	6	7	8	9
1	0	0	2	4	6	8	10	12	14	16	18
2	0	0	2	4	6	8	10	12	14	16	18
3	0	0	2	4	6	8	10	12	14	16	18
4	0	0	2	4	6	8	10	12	14	16	18
5	0	0	2	4	6	8	10	12	14	16	18
6	0	0	2	4	6	8	10	12	14	16	18
7	0	0	2	4	6	8	10	12	14	16	18
8	0	0	2	4	6	8	10	12	14	16	18
9	0	0	2	4	6	8	10	12	14	16	18

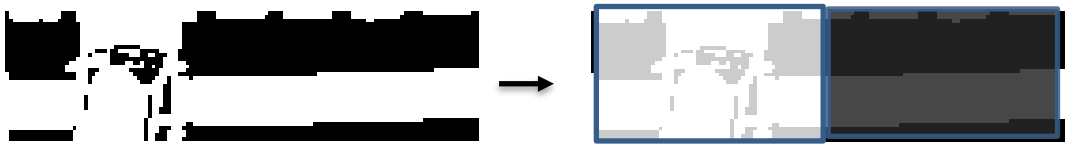
Gambar 3. 16 *Integral image* Uji

3.8.2 *Haar* Like Feature Uji

Selanjutna menentukan jenis *haar* yang akan diterapkan pada hasil perhitungan sebelumnya. Hal ini dilakukan dengan mengevaluasi setiap fitur terhadap data latih dengan menggunakan nilai dari fitur tersebut. Fitur yang memiliki batas terbesar antara pelanggaran dan bukan pelanggaran dianggap sebagai fitur terbaik.



Fitur 1



Fitur 2

Gambar 3. 17 Menentukan Fitur Uji

Setelah menentukan fitur *haar* selanjutnya dilakukan perhitungan dari hasil *integral image* ang telah dilakukan, sehingga dapat diketahui nilai *threshold*nya. Berikut perhitungan nilai latih pada fitur 1 terhadap citra uji. Nilai yang didapat fitur hitam A_{hitam} = 0, B_{hitam} = 405, C_{hitam} = 14, D_{hitam} = 1937. Sedangkan nilai fitur putih A_{putih} = 0, B_{putih} = 84, C_{putih} = 0, D_{putih} = 376 Data dapat dilihat pada lampiran.

f(x) =*Sumblack rectangle (pixel gray level) – Sumwhite rectangle (pixel gray level)*

$=(D_{hitam} + A_{hitam} - (B_{hitam} + C_{hitam})) - (D_{putih} + A_{putih} - (B_{putih} + C_{putih}))$ = (1937+0 – (405+14)) – (376+0-(84+0))

= 1226

Berikut perhitungan nilai latih pada fitur 2 terhadap citra uji. Nilai yang didapat fitur hitam A_{hitam} = 55, B_{hitam} = 84, C_{hitam} = 1091, D_{hitam} = 1937. Sedangkan nilai fitur putih A_{putih} = 0, B_{putih} = 54, C_{putih} = 14, D_{putih} = 1078. Data dapat dilihat pada lampiran.

f(x) =*Sumblack rectangle (pixel gray level) – Sumwhite rectangle (pixel gray level)*

$= (D_{hitam} + A_{hitam} - (B_{hitam} + C_{hitam})) - (D_{putih} + A_{putih} - (B_{putih} + C_{putih}))$

= (1937+55-(84+1091)) – (1078+0-(54+14)

= -193

Maka setelah didapatkannya nilai *threshold* dari citra uji, maka nilai dari hasil citra latih dan citra uji memiliki perbedaan hasil itu disebabkan terjadinya perubahan nilai *threshold* dikarenakan didalam citra uji memiliki adanya sebuah perbedaan citra dimana kondisi pelanggaran terjadi.

Pada citra latih dengan fitur 1 menghasilkan 1469 dan fitur 2 meghasilkan nilai 45, sedangkan pada citra uji dengan fitur 1 memiliki nilai 1226 dan fitur 2 menghasilkan nilai -193. Maka dari itu dapat disimpulkan hal tersebut sebuah pelanggaran dikarenakan terjadinya perbedaan nilai *threshold* pada citra latih dan citra uji.

3.9 Analisis Kebutuhan

Spesifikasi kebututhan perangkat lunak bertujuan sebagai acuan atau panduan bagi pembangun dan pengguna perangkat lunak selama perangkat lunak dibangun. Tahap ini sebagai langkah untuk membuat analisis kebutuhan fungsional dan non fungsional. Berikut rancangan spesifikasi kebutuhan perangkat lunak fungsional pada tabel 3.4

Tabel 3. 1 SKPL Fungsional

No	Kode	Keterangan
1	SKPL – F A – 01	Sistem dapat melakukan proses cropping
2	SKPL – F A – 02	Sistem dapat melakukan proses *grayscaling*
3	SKPL – F A – 03	Sistem dapat melakukan proses *thresholding*
4	SKPL – F A – 04	Sistem dapat melakukan proses *integral image*
5	SKPL – F A – 05	Sistem dapat melakukan proses fitur *haar*
6	SKPL – F A – 06	Sistem dapat melakukan proses deteksi citra

Berikut rancangan spesifikasi kebutuhan perangkat lunak non fungsional pada tabel 3.2.

Tabel 3. 2 SKPL Non Fungsional

No	Kode	Keterangan
2	SKPL - NF – 01	Pengguna dari sistem ini adalah pelaku uji
3	SKPL - NF – 02	Aplikasi ini berbasis dekstop dan dibangun dengan menggunakan bahasa pemograman C#

3.10 Analisis Kebutuhan non Fungsional

Analisis kebutuhan non fungsional adalah langkah dimana seorang pembangun perangkat lunak (software developer) menganalisis sumber daya yang akan digunakan dan menggunakan perangkat lunak yang dibangun. Perangkat keras dan perangkat lunak yang dimiliki harus sesuai dengan kebutuhan atau permintaan, sehingga dapat ditentukan kompabilitas aplikasi yang dibangun terhadap sumber daya yang ada.

Analisis kebutuhan non fungsional yang dilakukan dibagi dalam tiga tahap, yaitu:

1. Analisis perangkat keras
2. Analisis perangkat lunak
3. Analisis perangkat pikir / pengguna

3.10.1 Analisis Perangkat Keras

Spesifikasi perangkat keras minimum yang dibutuhkan untuk menjalankan program pada citra, dapat dilihat pada tabel 3.3.

Tabel 3. 3 Analisis Perangkat Keras

No.	Analisis Spesifikasi Minimum Hardware
1	Processor Intel Pentium 2.3 GHz
2	RAM 1024 MB
3	Hard disk 160 GB
4	Monitor 14 Inc
5	Keyboard dan Mouse standar

3.10.2 Analisis Perangkat Lunak

Spesifikasi perangkat lunak minimum yang dibutuhkan untuk menjalankan program pada citra, dapat dilihat pada tabel 3.4

Tabel 3. 4 Analisis Perangkat Lunak

No.	Analisis Spesifikasi Software
1	Sistem operasi Windows 7
2	Visual Studio 2013
3	.NetFramework 3.5
4	OpenCV

3.10.3 Analisis Perangkat Pikir / Pengguna

Analisis kebutuhan perangkat pikir merupakan uraian mengenai siapa saja yang akan menggunakan sistem dan terlibat dalam pengolahan data beserta karakteristiknya sehingga dapat diketahui tingkat pengalaman dan pemahaman pengguna terhadap sistem. Adapun karakteristik dari pengguna adalah sebagai berikut:

Tabel 3. 5 Analisis Pengguna

Spesifikasi	Hak Akses	Tingkat Keterampilan	Jenis Pelatihan
Pengguna	1 Memilih citra 2 Melakukan cropping 3 Melakukan *grayscaling* 4 Melakukan *thresholding* 5 Melakukan proses *integral image* 6 Melakukan proses fitur *haar* 7 Melakukan deteksi marka	Dapat mengoperasikan Komputer dan	Tidak diperlukan pelatihan khusus.

3.11 Analisis Kebutuhan Fungsional

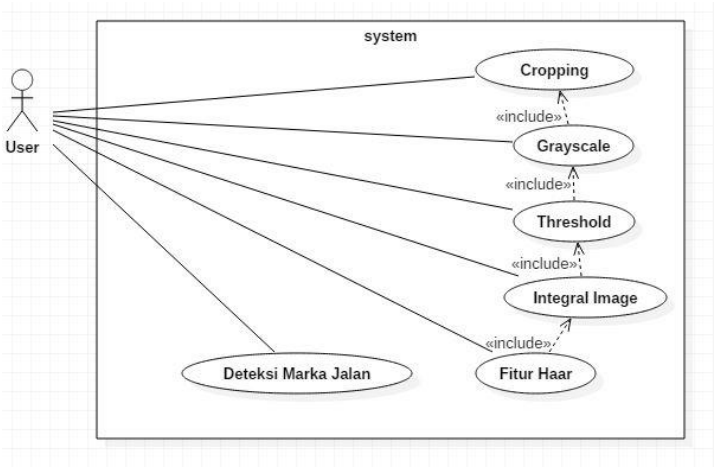
Analisis kebutuhan fungsional adalah spesifikasi tentang hal-hal yang akandilakukan sistem ketika diimplementasikan. Analisis kebutuhan ini diperlukan untuk menentukan keluaran yang akan dihasilkan sistem, masukan yang diperlukan sistem, lingkup proses yang digunakan untuk mengolah masukan menjadi keluaran.

3.11.1 Use Case

Use case merupakan gambaran umum dari perancangan sistem yang akan dibangun. Terdapat tiga buah poin yang merincikan *use case* diagram ini, yaitu.

1. Identifikasi *use case*
2. Identifikasi aktor
3. Identifikasi skenario *use case*

Adapun rancangan dari *use case* diagram *Viola Jones* pada video, dapat dilihat pada gambar berikut



Gambar 3. 18 Usecase

3.11.2 Identifikasi Use Case

Identifikasi *Use Case* berfungsi untuk menjelaskan fungsi *Use Case* yang terdapat pada *Use Case Diagram*. Definisi *Use Case* diterangkan pada tabel berikut.

1. *Actor* adalah abstraksi dari orang dan sistem yang lain yang mengaktifkan fungsi dari target sistem. Berikut adalah faktor yang berperan dalam menjalankan aplikasi yang dibangun

Tabel 3. 6 Identifikasi Usecase Aktor

No	Aktor	Deskripsi
A-01	User	Merupakan aktor dari aplikasi yang dibangun atau pengguna aplikasi.

2. Berikut adalah identifikasi usecase pada aplikasi:

Tabel 3. 7 Identifikasi Usecase

No	Aktor	Deskripsi
UC-01	Cropping	Merupakan aksi untuk memperkecil ukuran citra menjadi 130x30
UC-02	*Grayscale*	Merupakan aksi untuk melakukan proses merubah citra RGB ke citra abu

UC-03	*Threshold*	Merupakan aksi untuk melakukan proses merubah citra abu ke citra hitam putih
UC-04	*Integral image*	Merupakan aksi untuk melakukan proses pengintegralan nilai *threshold*
UC-05	Fitur *Haar*	Merupakan aksi untuk melakukan proses mendeteksi marka
UC-06	Deteksi Marka	Merupakan aksi untuk melakukan proses deteksi marka dengan pernyataan melanggar atau tidak melanggar

3.11.3 Skenario Use Case

Skenario setiap bagian pada *use case* menunjukkan proses apa yang terjadi pada setiap bagian didalam *use case* tersebut, dimana user memberikan perintah pada setiap bagian dan respon apa yang diberikan oleh sistem kepada user setelah user memberikan perintah pada setiap bagian-bagian use case.

3.11.3.1 Skenario Pilih Citra

Scenario uscecase untuk pilih cira terdapat pada tabel 3.8:

Tabel 3. 8 Skenario usecase pilih citra

Identifikasi	
Nama *Use Case*	Pilih Citra
Aktor	User
Tujuan	Digunakan untuk memilih *file* citra yang akan digunakan sebagai data yang akan di uji.
Kondisi Awal	Sistem menampilkan menu pemilihan citra uji
Skenario Utama	
Aktor	**Reaksi Sistem**
1. User menekan tombol pilih citra	
	2. Sistem menampilkan jumlah objek yang terdeteksi
3. User memilih citra target yang akan digunakan, kemudian akan muncul citra targetnya.	

	4. Sistem menampilkan informasi Citra

3.11.3.2 Skenario Usecase deteksi pelanggaran

Skenario usecase untuk deteksi marka jalan terdapat pada tabel 3.9

Tabel 3. 9 Skenario Usecase Deteksi Marka Jalan

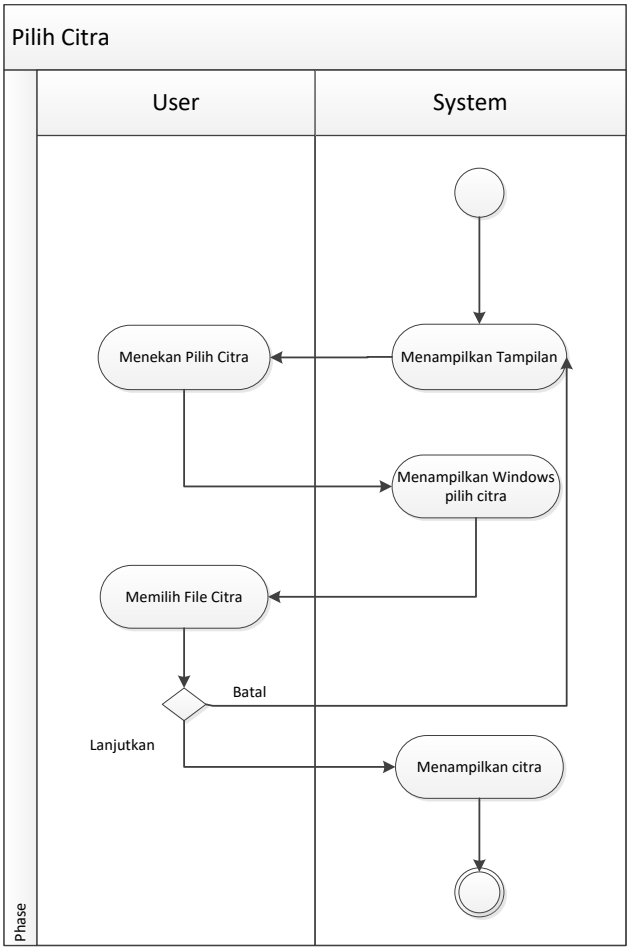
Identifikasi	
Nama *Use Case*	*Deteksi Marka*
Aktor	User
Tujuan	Digunakan untuk melakukan proses pencarian objek yang dipilih
Kondisi Awal	Sistem menampilkan menu *object detection*
Skenario Utama	
Aktor	**Reaksi Sistem**
1. User menekan tombol *deteksi*	
	2. Sistem melakukan cropping ke ukuran 130x30
	3. Sistem menganalisis citra RGB
	4. Sistem mengubah citra RGB ke *Grayscale*
	5. Sistem melakukan proses *thresholding*
	6. Sistem melakukan proses integral pada citra
	7. Sistem melakukan proses *haar*
	8. Sistem menampilkan hasil deteksi

3.12 Activity Diagram

Diagram aktivitas atau *Activity Diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Penggambaran *activity* diagram memiliki kemiripan dengan *flowchart diagram. Activity diagram* memodelkan *event-event* yang terjadi pada *Use Case* dan digunakan untuk pemodelan aspek dinamis dari sistem.

3.12.1 Activity Pilih Citra

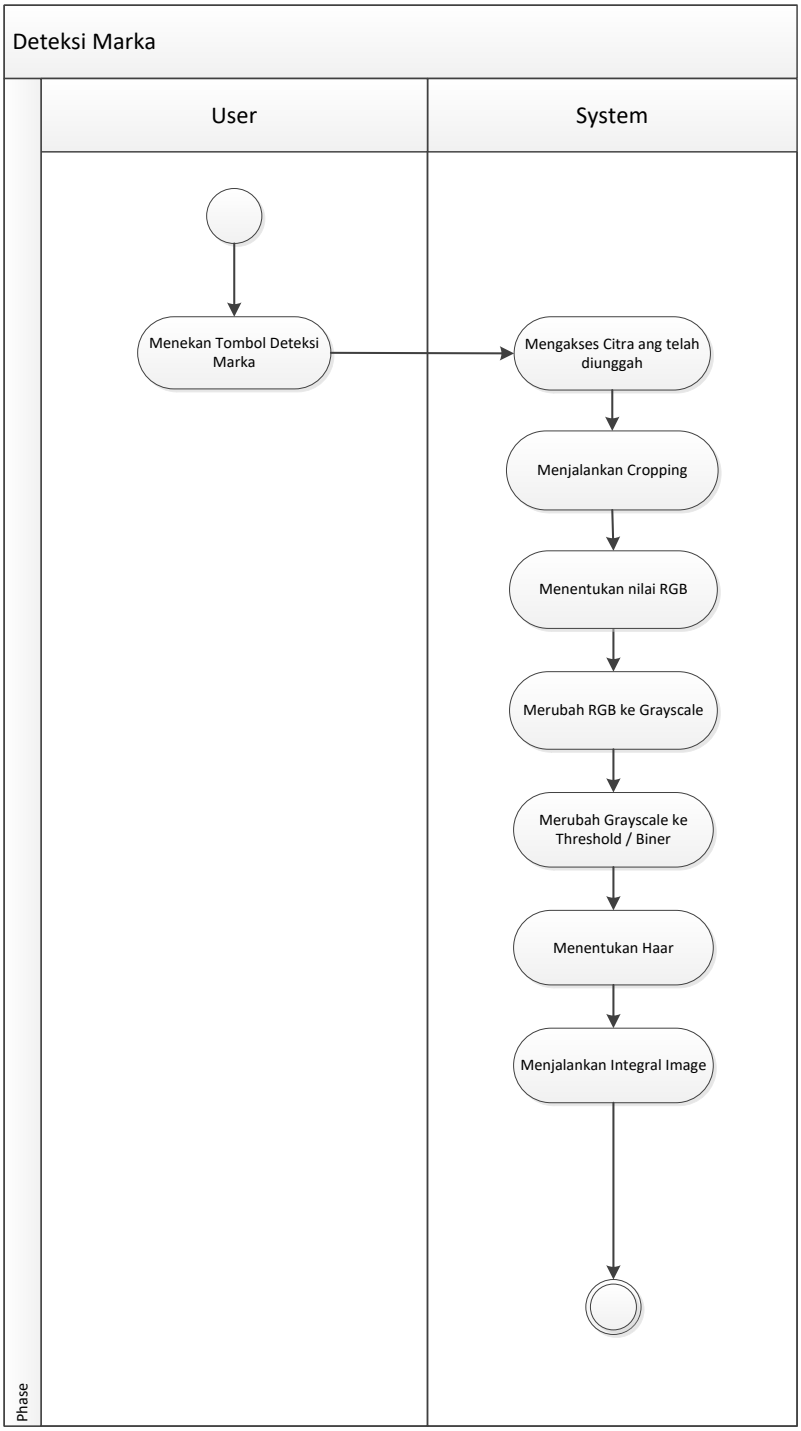
Adapun *activity diagram* dalam melakukan pilih citra pada gambar berikut.



Gambar 3. 19 Activiy Pilih Citra

3.12.2 Activity Deteksi

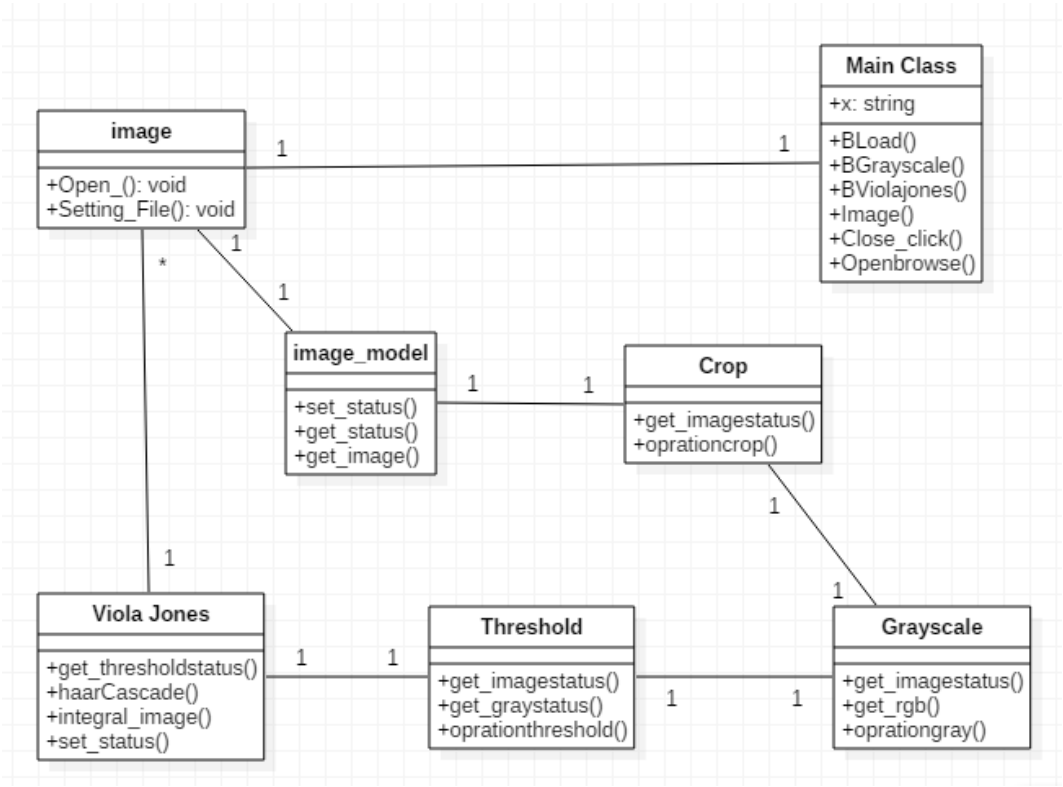
Adapun *activity diagram* dalam melakukan proses deteksi marka pada gambar berikut:



Gambar 3. 20 Activity Deteksi Marka

3.13 Class Diagram

Class Diagram digunakan untuk menggambarkan secara abstrak struktur dari aplikasi yang akan dibangun, *class-class* yang terlibat, serta hubungan antar *class* untuk saling berkomunikasi satu sama lain. Berikut adalah kelas diagram pada prototype yang akan dibangun:

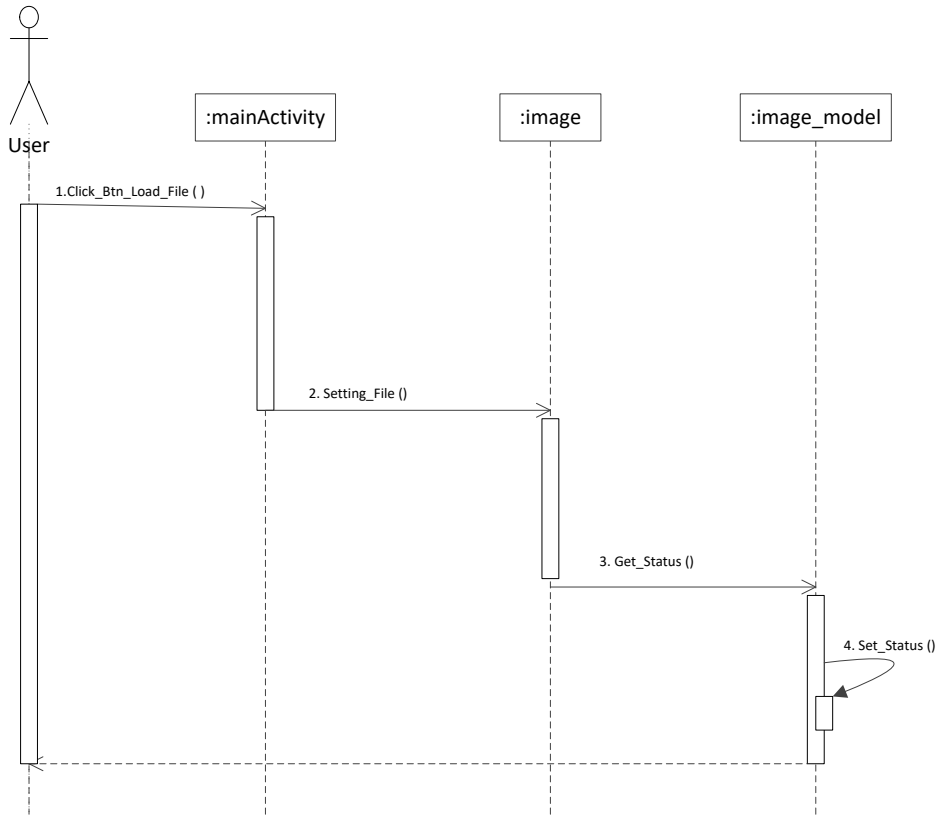


Gambar 3. 21 Class Diagram

3.14 Sequence Diagram

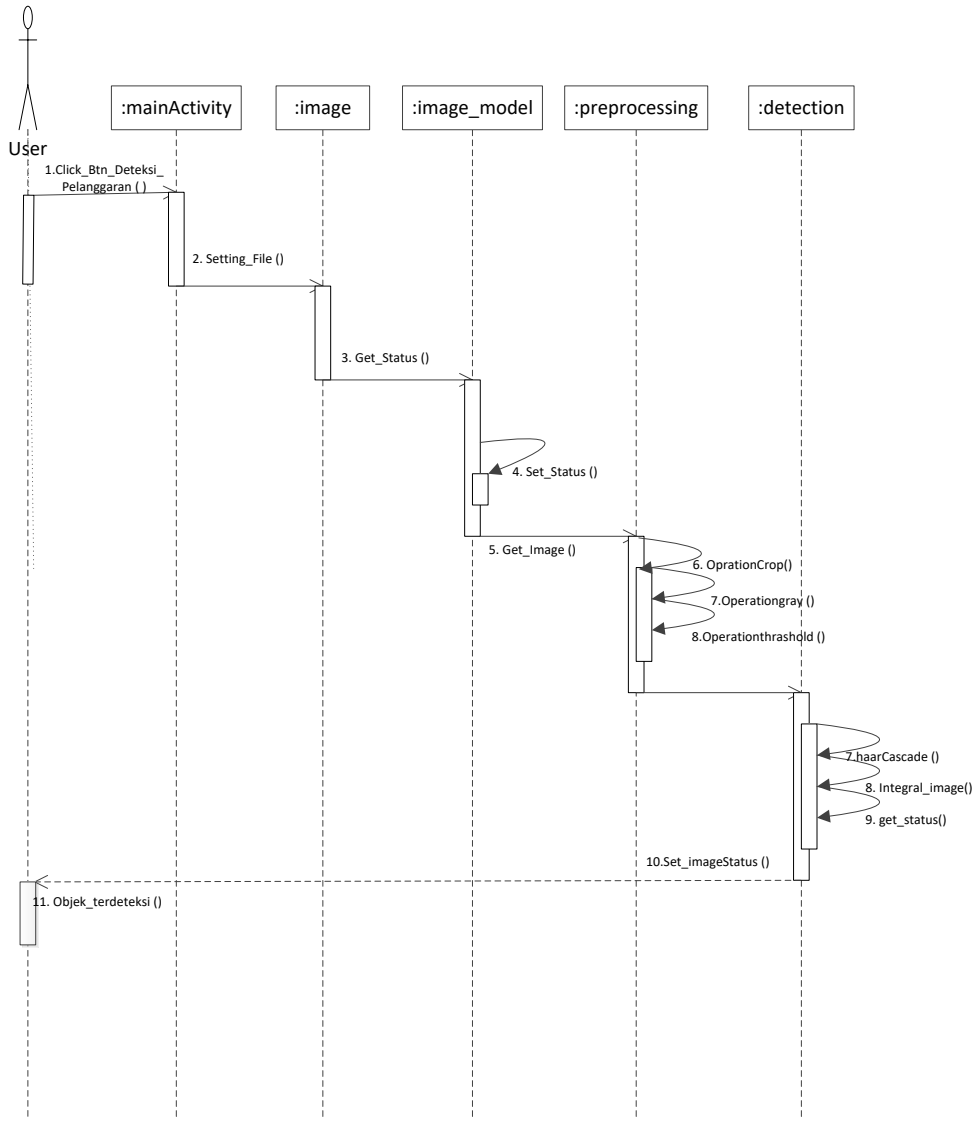
Sequence diagram menggambarkan interaksi antar objek di dalam dan disekitar sistem (termasuk pengguna, display, dan sebagainya) berupa informasi yang digambarkan terhadap waktu. Sequence diagram terdiri atas dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait). Berikut sequence diagram pada sistem.

3.14.1 Sequence Browse Citra



Gambar 3. 22 Sequence Load File

3.14.2 Sequence *Viola Jones*



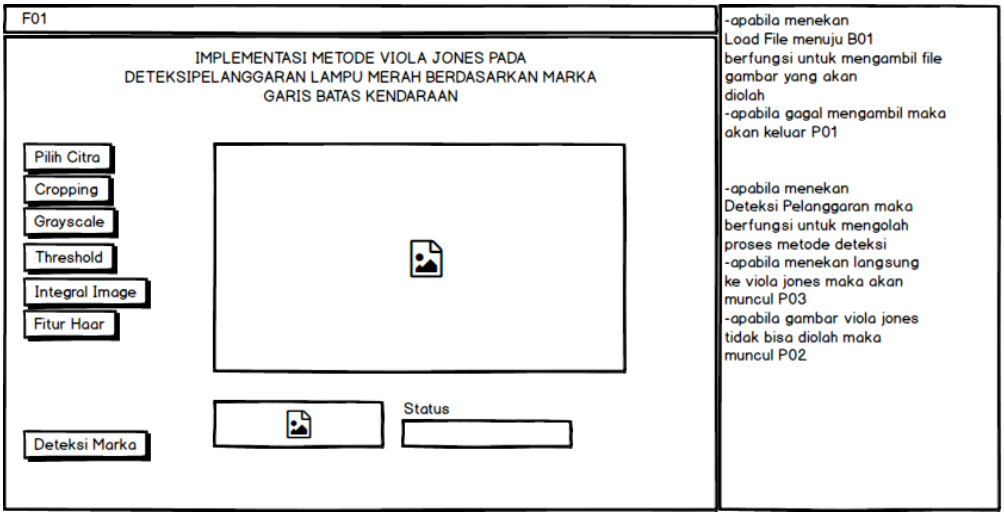
Gambar 3. 23 Sequence Deteksi Marka

3.15 Perancangan Antar Muka

Perancangan antarmuka mendeskripsikan rencana tampilan dari setiap tampilan yang akan digunakan. Berikut adalah perancangan yang terdapat pada sistem.

3.15.1 Tampilan Menu Utama

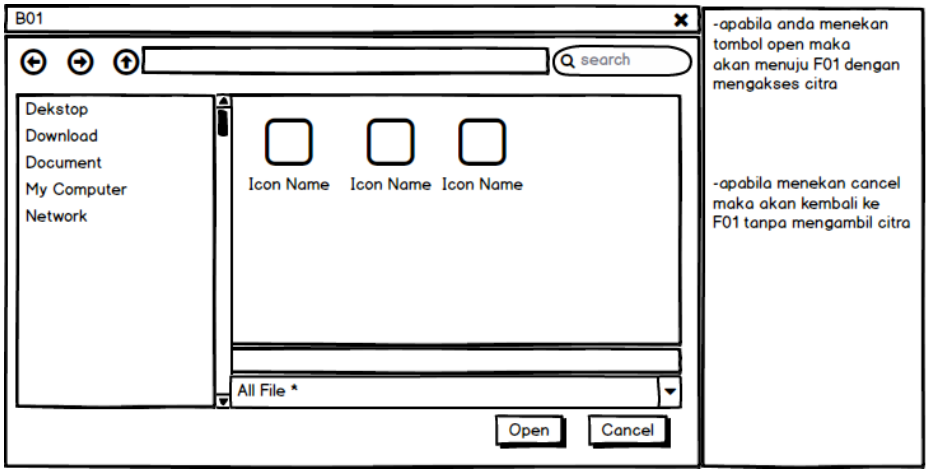
Rancangan pada menu utama disimbolkan dengan F01 dengan fungsi untuk mengakses sistem dapat dilihat pada gambar 3.28



Gambar 3. 24 Menu Utama

3.15.2 Tampilan Menu Pilih Citra / Browse

Menu pilih citra yang sudah ada dalam os tampilannya dengan fungsi untuk memilih citra yan akan diolah. Bias dilihat pada gambar 3.29



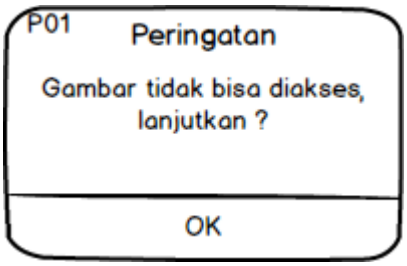
Gambar 3. 25 Menu Pilih

3.16 Perancangan Pesan

Perancangan antarmuka mendeskripsikan rencana tampilan dari pesan pada sistem untuk memberikan informasi pada user.

3.16.1 Pesan P01

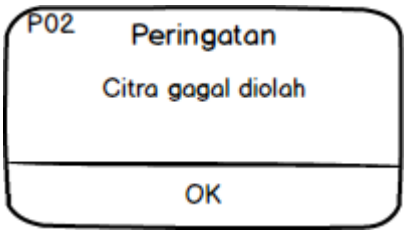
Pesan ini memberitahukan bahwa citra yang dipili tidak dapat diakses. Bisa dilihat pada gambar 3.30



Gambar 3. 26 Perancangan Pesan P01

3.16.2 Pesan P02

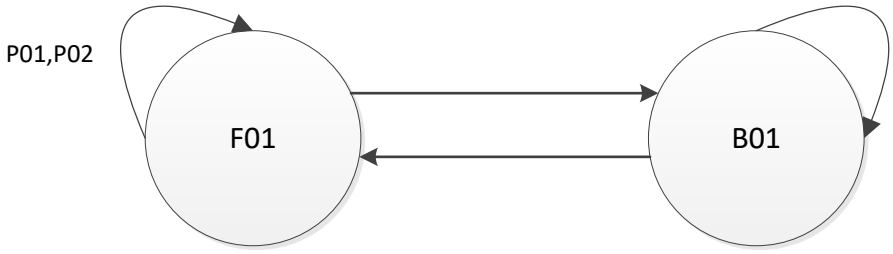
Pesan ini memberitahukan bahwa citra yang dipili tidak dapat diolah. Bisa dilihat pada gambar 3.31



Gambar 3. 27 Perancangan Pesan P02

3.17 Jaringan Semantik

Berikut ini adalah gambar jaringan semantik yang menggambarkan hubungan antar modul.



Gambar 3. 28 Jaringan Semantik