BAB 3

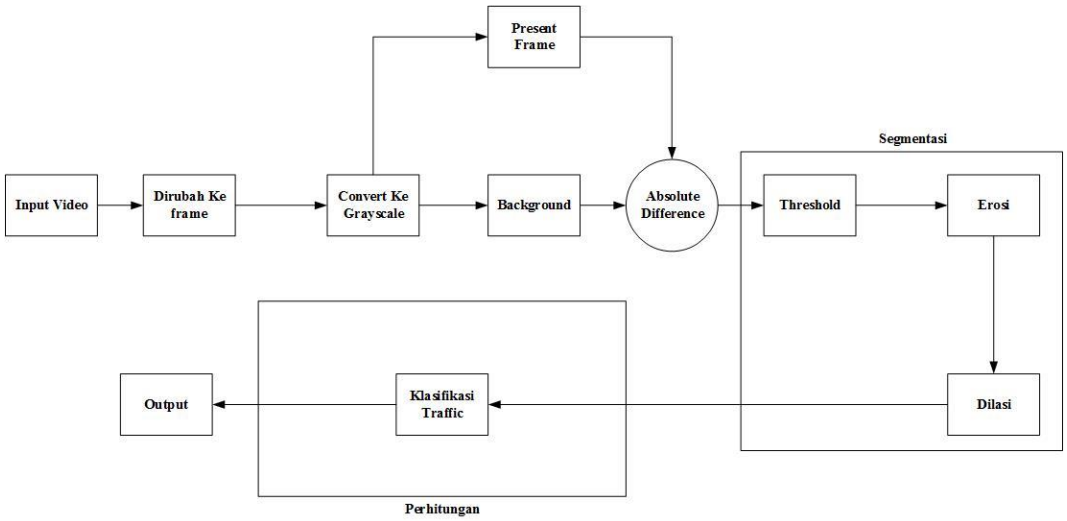
ANALISIS DAN PERANCANGAN

3.1. Analisis Masalah

Analisis masalah pada penelitian ini adalah sulitnya mengimplementasikan perhitungan kendaraan. Hal ini disebabkan karena sulitnya untuk mendeteksi kontur kendaraan dan juga mengukur zona lebar jalan yang sedang di tempati objek kendaraan.

3.2. Analisis Proses

Analisis proses yang akan dilakukan pada penelitian kali ini dibagi menjadi 3 tahapan Bagian pertama adalah *pre-processing*, bagian kedua *Segmentasi* dan bagian ketiga adalah perhitungan atau *counting*. Setelah itu output dari proses perhitungan berupa data statistik lalu-lintas dikirim ke modul integrasi.



Gambar 3.1 Diagram Analisis Proses

3.3. Analisis Data Masukan

Pada penelitian ini yang akan dianalisis adalah data masukan. Data yang dimasukan berupa video hasil rekaman kamera,kemudian video akan dirubah menjadi frame-frame untuk proses *convert* ke *grayscale* setelah itu akan masuk

ketahapan *preprocessing*, setelah *preprocessing* ke *segmentasi* dan terakhir tahap *counting* / perhitungan kendaraan.

3.3.1. Preprocessing

Preprocessing merupakan tahap persiapan untuk proses awal. Input yang berupa video diproses per frame.*Preprocessing* meliputi 2 tahap, yaitu merubah video menjadi frame dan konversi dari RGB ke *grayscale*. Pada tahap pertama inisialisasi, inisialisasi diperlukan untuk memilih background, mendefinisikan background yang akan ditetapkan sebagai acuan proses analisis. lalu akan digunakan sebagai referensi untuk menentukan lalu lintas tersebut lancar dan padat, yaitu daerah pada citra dimana kendaraan akan dihitung ketika melewatinya Tahap berikutnya adalah konversi dari rgb ke *grayscale* .

3.3.1.1. Video Dijadikan Frame

Video merupakan gabungan gambar-gambar mati yang dibaca berurutan dalam suatu waktu dengan kecepatan tertentu. Gambar-gambar yang digabung tersebut dinamakan frame dan kecepatan pembacaan gambar disebut dengan frame rate, dengan satuan fps (frame per second). Karena dimainkan dalam kecepatan yang tinggi maka tercipta ilusi gerak yang halus, semakin besar nilai frame rate maka akan semakin halus pergerakan yang ditampilkan. Tahapan awal yaitu mengetahui jumlah frame yang ada pada video.

Pada perekaman melalui kamera akan menghasilkan video yang berukuran tinggi dan mempunyai frame rate yang tinggi juga. Untuk mempermudah proses analisa, maka perlu pengkompresian untuk memudahkan prosesnya Untuk mengetahui informasi – informasi yang terdapat pada video pertama, dapat kita lihat pada properties Gomplayer dan VLC media player. Berikut ini adalah tabel informasi dari video original.

Tabel 3.1 Informasi Video Sampel

Parameter	Satuan
Format	Avi
File size	5,93MB

Duration	3m, 26s
Overall Bite	241Kbps
Format Video	Audio Video Interleave
Code Id	Dx50
Codec ID/Info	Dx50
Width	178 pixel
Height	144 pixel
Display Aspect Ratio	5:3
Frame Rate	30.000 fps
Bits(pixel*frame)	0.306
Stream size	5.78MiB

Untuk mengetahui jumlah frame dari video sampel 1 tersebut maka dapat menggunakan rumus

```
fps * duration = jumlah frame
```

Dari informasi tabel tersebut dapat diketahui bahwa jumlah frame tersebut adalah sebagai berikut:

Jumlah frame = 30.000 frame/s * 3 minutes, 26second

= 6180 frame

Jumlah frame yang tersedia pada video dipengaruhi oleh *frame rate* dan durasi. Semakin besar *frame rate* atau durasi dalam menampilkan gambar, maka semakin banyak pula yang diperoleh.

3.3.1.2. Convert Grayscale

Pada tahap ini merubah inputan yang berupa image frame yang masih RGB menjadi grayscale agar lebih mudah untuk diproses. Grayscale adalah warna-warna piksel yang berada dalam rentang gradasi warna hitam dan putih. Grayscale didapat dari konversi RGB image dengan rumusan sebagai berikut :

```
Y=0,2*R+0,5*G+0,2*B
```

Dimana :

Y = derajat keabuan (uint8 atau uint16).

R = nilai pixel chanel Red

G = nilai pixel chanel Green

B = nilai pixel chanel Blue

Analisis perhitungan

Agar dapat melakukan tahap perhitungan *grayscale* maka dibutuhkan citra masukan. Berikut ini merupakan citra masukan data 1, yaitu citra image_frame_1.



Gambar 3.2 Citra Image frame 1

Tabel 3.2 Citra RGB Image frame 1 dengan ukuran 3x3

R=118 G=114 B=138	R=110 G=106 B=130	R=107 G=103 B=123
R=110 G=106 B=126	R=227 G=223 B=247	R=220 G=216 B=240
R=215 G=211 B=231	R=220 G=216 B=236	R=150 G=116 B=200

Perhitungan nilai matrix (0.0)

```
Y=0,2*118+0,5*114+0,2*138
Y=23,6+57+27,6
Y=108,2=108
```

Perhitungan nilai matrix (0.1)

```
Y=0,2*110+0,5*106+0,2*130=101
Y=22+53+26
Y=100,2=101
```

Perhitungan nilai matrix (0.2)

```
Y=0,2*107+0,5*103+0,2*123=97
Y=21,4+51,5+24,6
Y=97,5=97
```

Dengan menggunakan rumus yang sama pada semua piksel maka akan didapatkan hasil matrix sebagai berikut:

Tabel 3.3 Matrix Grayscale 3x3

108	101	97
100	206	200
194	199	128

Maka dari perhitungan rumus diatas didapatkan cittra *grayscale* sebagai berikut:



Gambar 3.3 Citra image frame 1

3.3.2. Segementasi

Segmentasi dari program ini adalah mendapatkan obyek dengan cara background substract. Background substract bekerja dengan cara mengurangkan citra referensi dengan *current frame* saat ini. akan tetapi hasil dari background subtract masih belum bisa untuk dihitung. Tahapan selanjutnya *absolute difference* hasil dari background substract setelah itu akan di-*threshold* setelah *threshold* melakukan *erotion* dan *dilation* untuk mengurangi *noise*. Setelah noise berkurang, setiap obyek akan dipisahkan menggunakan Connected component.

3.3.2.1. Absolut Difference

Pada proses pengurangan background dengan *current frame*. Fungsi *absolute different* digunakan untuk menghitung nilai absolut selisih dari dua array atau citra. *Absolute* digunakan untuk menghindari adanya nilai negatif. Rumus untuk menentukan nilai absolut bisa didapatkan dari persamaan sebagai berikut:

(a) $Za = A - B$

(b)Zb=B-A

(c)Zc=|A-B|

Dimana Z=array

a=image_frame_1(X)

b=image_frame_2(Y)

c=hasil(A+B)

agar dapat melakukan perhitungan *absolute difference* maka dibutuhkan 2 citra masukan yang sudah diproses ke *grayscale.* berikut ini merupakan citra masukan data 1 yang sudah di *convert* ke *grayscale,* yaitu image_frame_1 dan yang kedua image_frame_2



A



B

Gambar 3.3 (A)citra image_frame_1 dan (B) citra image_frame_2

Matrix (A)

108	101	97
100	206	0
194	199	128

Matrix (B)

114	122	105
86	115	81
100	81	100

Tabel Matrix 3.4 (A) citra image_frame_1 (B) citra image_frame_2

Perhitungan untuk mencari Za

Perhitungan Matrix (0,0)

Z=A-B

Z=108-114

Z=0

Perhitungan Matrix (0,1)

Z=A-B

Z=101-122

Z=0

Perhitungan matrix (0,2)

Z=A-B

Z=97-105

Z=0

Dengan menggunakan rumus yang sama pada semua piksel maka akan didapatkan hasil matrix sebagai berikut:

Tabel 3.5 tabel Z (a)

0	0	0
14	91	0
94	118	28

Perhitungan untuk mencari Zb

Perhitungan Matrix (0,0)

Z=B-A

Z=114-108

Z=6

Perhitungan Matrix (0,1)

Z=B-A

Z=122-101

Z=21

Perhitungan matrix (0,2)

Z=B-A

Z=105-97

Z=8

Dengan menggunakan rumus Z=B-A maka akan di dapatkan pada semua piksel maka akan dihasilkan matrix sebagai berikut:

Tabel 3.6 Z (b)

6	21	8
0	0	0
0	0	0

Perhitungan untuk mencari Zc

Perhitungan Matrix (0,0)

Z= |A+B|

Z=0+6

Z=6

Perhitungan Matrix (0,1)

Z=|A+B|

Z=0+21

Z=21

Perhitungan matrix (0,2)

Z=|A+B|

Z=0+8

Z=8

Dengan menggunakan rumus Zc=|A+B| maka akan di dapatkan pada semua piksel tidak ada nilai negatif.

Tabel 3.7 Z (c)

6	21	8
14	91	119
94	118	28

3.3.2.2. Thresholding

Thresholding citra adalah suatu metode yang digunakan untuk memisahkan antara obyek dan backgroundnya. Thresholding merupakan teknik yang sederhana dan efektif untuk segmentasi citra. Proses thresholding sering disebut dengan proses binerisasi. Pada beberapa aplikasi pengolahan citra, terlebih dahulu dilakukan threshold terhadap citra gray level untuk dapat menjadi citra biner (citra yang memiliki nilai level keabuan 0 atau 255). Sebuah citra hasil proses thresholding dapat disajikan dalam histogram citra untuk mengetahui penyebaran nilai-nilai intensitas piksel pada suatu citra/bagian tertentu dalam citra sehingga untuk citra bimodal, histogram dapat dipartisi dengan baik (segmentasi objek dengan background) dan dapat ditentukan nilai threshold-nya. Rumus untuk menentukan nilai Threshold bisa didapatkan dari rumus sebagai berikut:

```
g(x,y)={1if f(x,y)>=T}
g(x,y)={0if f(x,y)<=T}
T=125
```

Analisis perhitungan:

Agar dapat melakukan tahap perhitungan *threshold* maka dibutuhkan citra masukan. Berikut ini merupakan citra masukan data 1, yaitu citra image_frame_1.



Gambar 3.4 Citra_Image_Frame_1

Tabel 3.8 Image_frame_1

112	123	108
238	196	221
219	229	216

Perhitungan nilai matrix 0,0

G=(x,y)={ 0 if f(x,y) < =T }

G= 112 < 125

G=0

Perhitungan nilai matrix 0,1

G=(x,y)={ 0 if f(x,y) < =T }

G= 123 < 125

G=0

Perhitungan nilai matrix 0,2

G=(x,y)={ 0 if f(x,y) < =T }

G= 108 < 125

G=0

Perhitungan nilai matrix 1,0

G=(x,y)={ 1 if f(x,y) > =T }

G= 238 > 125

G=1

Perhitungan nilai matrix 1,1

G=(x,y)={ 1 if f(x,y) > =T }

G= 196 > 125

G=1

Perhitungan nilai matrix 1,3

G=(x,y)={ 1 if f(x,y) > =T }

G= 221 > 125

G=1

Perhitungan nilai matrix 2,0

G=(x,y)={ 1 if f(x,y) > =T }

G= 219 > 125

G=1

Perhitungan nilai matrix 2,1

G=(x,y)={ 1 if f(x,y) > =T }

G= 229 > 125

G=1

Perhitungan nilai matrix 2,2

G=(x,y)={ 1 if f(x,y) > =T }

G= 216 > 125

G=1

Dengan menggunakan rumus

g(x,y)={1if f(x,y)>=T}

g(x,y)={0if f(x,y)<=T}

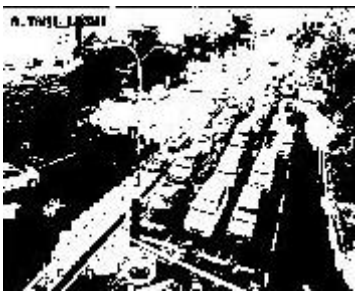
T=125

pada semua piksel maka akan didapatkan hasil nilai matrix *threshold* sebagai berikut:

Tabel 3.9 Image_frame_1

0	0	0
1	1	1
1	1	1

Maka dari perhitungan rumus diatas didapatkan citra *threshold* sebagai berikut:



Gambar 3.5 Image_frame_1_thrshold

3.3.2.3. Dilasi

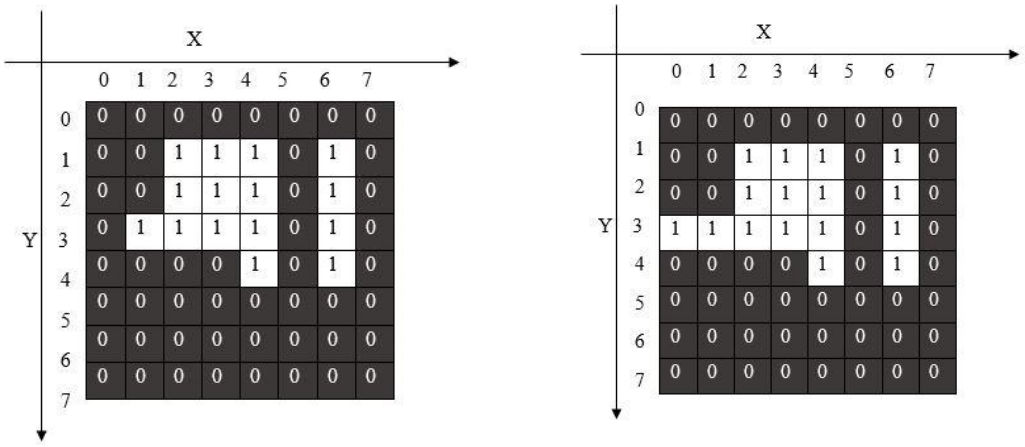
Dilasi merupakan sebuah operasi yang mengembangkan (memperbesar) atau mempertebal objek dalam suatu citra biner. Dilasi dibutuhkan untuk mengembalikan bagian obyek yang terbuang pada saat proses erosi Rumus untuk menentukan nilai Dilasi bisa didapatkan dari rumus sebagai berikut:

$$A \oplus B = \{c | C = a + b, a \in A, b \in B\}$$

Penjelasan rumus, A Dilasi B= c Dimana C adalah a+b dan a himpunan dari matrix A dan b himpunan dari Matrix B

Analisis perhitungan

Agar dapat melakukan tahap perhitungan *dilasi* maka dibutuhkan citra masukan. Berikut ini merupakan citra masukan data 1, yang telah melalui beberapa tahapan menjadi *threshold* dan bernilai citra biner



Citra Image_biner_A **Citra Image_biner_B**

Gambar 3.6(A) adalah data masukan berupa image_frame_1 , (B) adalah image_frame_1 yang digeser 1 kali ke kanan

Perhitungan dilasi dapat dilakukan sebagai berikut:

$C = A \oplus B = [A + \{(0,0)\}] \cup [A + \{(-0,1)\}$

Pertama dapatkan himpunan dari citra_image_biner_A

A=(1,3),(2,1),(2,2),(2,3),(3,1),(3,2),(3,3),(4,1),(4,2),(4,3),(4,4),(6,1),(6,2),(6,3)

Kedua dapatkan himpunan dari citra_image_biner_B

A=(0,3),(1,1),(1,2),(1,3),(2,1),(2,2),(2,3),(3,1),(3,2),(3,3),(3,4),(5,1),(5,2),(5,3)

Maka perhitungan A+(0,0) yaitu matrix yang menetap pada koordinat nya

C=(1,3)+(0,0)=(1,3)

C=(2,1)+(0,0)=(2,1)

C=(2,2)+(0,0)=(2,2)

C=(2,3)+(0,0)=(2,3)

C=(3,1)+(0,0)=(3,1)

C=(3,2)+(0,0)=(3,2)

C=(3,3)+(0,0)=(3,3)

C=(4,1)+(0,0)=(4,1)

C=(4,2)+(0,0)=(4,2)

C=(4,3)+(0,0)=(4,3)

C=(4,4)+(0,0)=(4,4)

C=(6,1)+(0,0)=(6,1)

C=(6,2)+(0,0)=(6,2)

C=(6,3)+(0,0)=(6,3)

Maka perhitungan A+(-1,0) yaitu matrix yang digeser ke kanan 1 kali

C=(1,3)+(-1,0)=(0,3)

C=(2,1)+(-1,0)=(1,1)

C=(2,2)+(-1,0)=(1,2)

C=(2,3)+(-1,0)=(1,3)

C=(3,1)+(-1,0)=(2,1)

C=(3,2)+(-1,0)=(2,2)

C=(3,3)+(-1,0)=(2,3)

C=(4,1)+(-1,0)=(3,1)

C=(4,2)+(-1,0)=(3,2)

C=(4,3)+(-1,0)=(3,3)

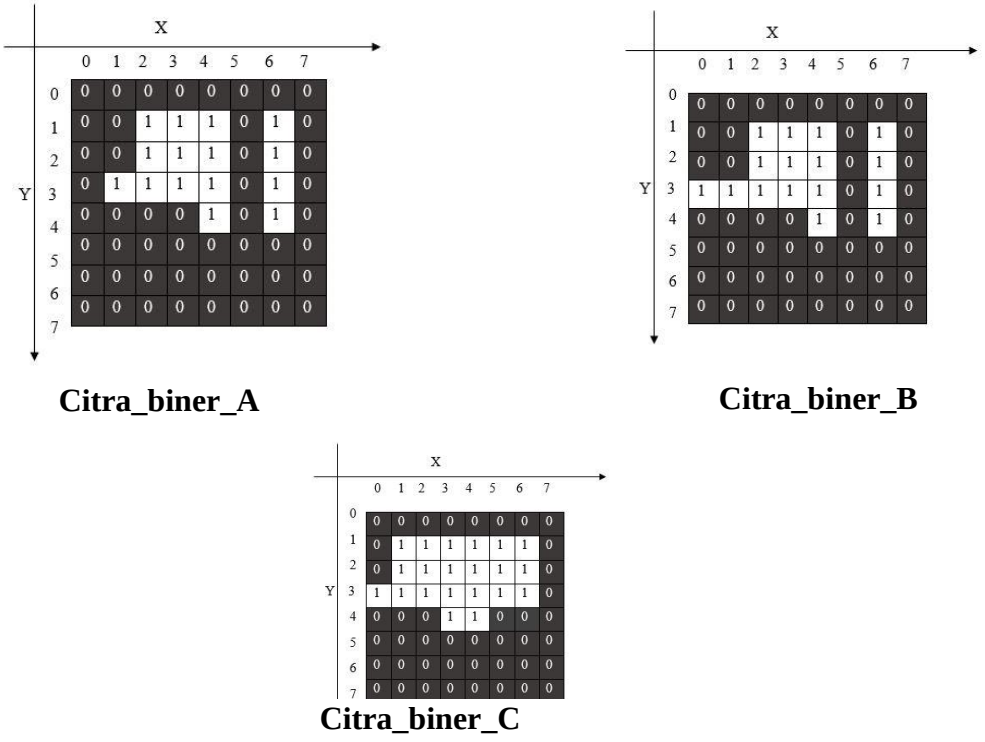
C=(4,4)+(-1,0)=(3,4)

C=(6,1)+(-1,0)=(5,1)

C=(6,2)+(-1,0)=(5,2)

C=(6,3)+(-1,0)=(5,3)

Maka dari hasil perhitungan diatas didapatkan citra_image_biner_C atau citra dilasi yang sudah digabung antara citra A dan B.



Gambar 3.7 (A) adalah image_frame_1 dan (B) adalah image_frame_1 yang digeser ke kanan 1 kali dan (C) adalah dilasi hasil gabungan (A) dan (B)

3.3.2.4. Erosi

Erosi adalah menyusutkan atau menipiskan objek dalam suatu image biner. Digunakan untuk membuang bagian dari citra yang bukan obyek namun dideteksi sebagai obyek pada saat thresholding. Juga untuk menghilangkan white noise dan noise lain yang berukuran kecil pada citra. Rumus untuk menentukan nilai Erosi bisa didapatkan dari rumus sebagai berikut:

$$A\theta B = \{c|(B), \subseteq A\}$$

Penjelasan rumusnya A erosi B = c Dimana C hasil Erosi B yang Ditranslasikan pada matrix A

Sampel yang digunakan adalah sampel hasil dari dilasi maka perhitungannya akan seperti berikut:

B(1,3)={(1,3),(0,3)}

B(2,1)={(2,1),(1,1)}

B(2,2)={(2,2),(1,2)}

B(2,3)={(2,3),(1,3)} ⟶ B(2,3)={(2,3),(1,3)}

B(3,1)={(3,1),(2,1)} ⟶ B(3,1)={(3,1),(2,1)}

B(3,2)={(3,2),(2,2)} ⟶ B(3,2)={(3,2),(2,2)}

B(3,3)={(3,3),(2,3)} ⟶ B(3,3)={(3,3),(2,3)}

B(4,1)={(4,1),(3,1)} ⟶ B(4,1)={(4,1),(3,1)}

B(4,2)={(4,2),(3,2)} ⟶ B(4,2)={(4,2),(3,2)}

B(4,3)={(4,3),(3,3)} ⟶ B(4,3)={(4,3),(3,3)}

B(4,4)={(4,4),(3,4)}

B(6,1)={(6,1),(5,1)}

B(6,2)={(6,2),(5,2)}

B(6,3)={(6,3),(5,3)}

Maka erosi yang dipilih untuk mendapatkan nilai dari suatu erosi adalah mengambil nilai titk koordinat yang ada pada nilai matrix A, menurut jarak dan arah tertentu, karena jarak dan arah nya yaitu koordinat (x,y) pada matrix a maka hasilnya adalah sebagai berikut

B(2,3)={(2,3),(1,3)} ⟶ B(2,3)={(2,3),(1,3)}

B(3,1)={(3,1),(2,1)} ⟶ B(3,1)={(3,1),(2,1)}

B(3,2)={(3,2),(2,2)} ⟶ B(3,2)={(3,2),(2,2)}

B(3,3)={(3,3),(2,3)} ⟶ B(3,3)={(3,3),(2,3)}

B(4,1)={(4,1),(3,1)} ⟶ B(4,1)={(4,1),(3,1)}

B(4,2)={(4,2),(3,2)} ⟶ B(4,2)={(4,2),(3,2)}

B(4,3)={(4,3),(3,3)} ⟶ B(4,3)={(4,3),(3,3)}

Maka hasil erosi citra biner_C adalah sebagai berikut:



Citra_biner_dilasi_A **Citra_biner_erosi_B**

Gambar 3.8 (A) adalah citra biner dilasi, (B)adalah citra biner setelah di Erosi

Maka terdapat perbedaan dari suatu objek yang telah di dilasi dan di erosi pada citra yang akan di analsis

3.4. Analisis Non Fungsional

Kebutuhan non fungsional yang dibutuhkan untuk membangun sistem ini mencakup 2 hal yaitu kebutuhan perangkat keras dan perangkat lunak

3.4.1. Analisis Kebutuhan Perangkat Keras

Adapun kebutuhan perangkat keras yang diperlukan untuk menjalankan sistem ini adalah:

1. Processor minimal core 2 duo 2,13 Ghz
2. Memory 1Gb atau lebih
3. *Webcam* minimal 2Mp atau lebih
4. Hardisk minimal 320 MB untuk Net Framework

3.4.2. Analisis Kebutuhan Perangkat Lunak

Perangkat lunak yang dibutuhkan untuk menjalankan aplikasi ini adalah:

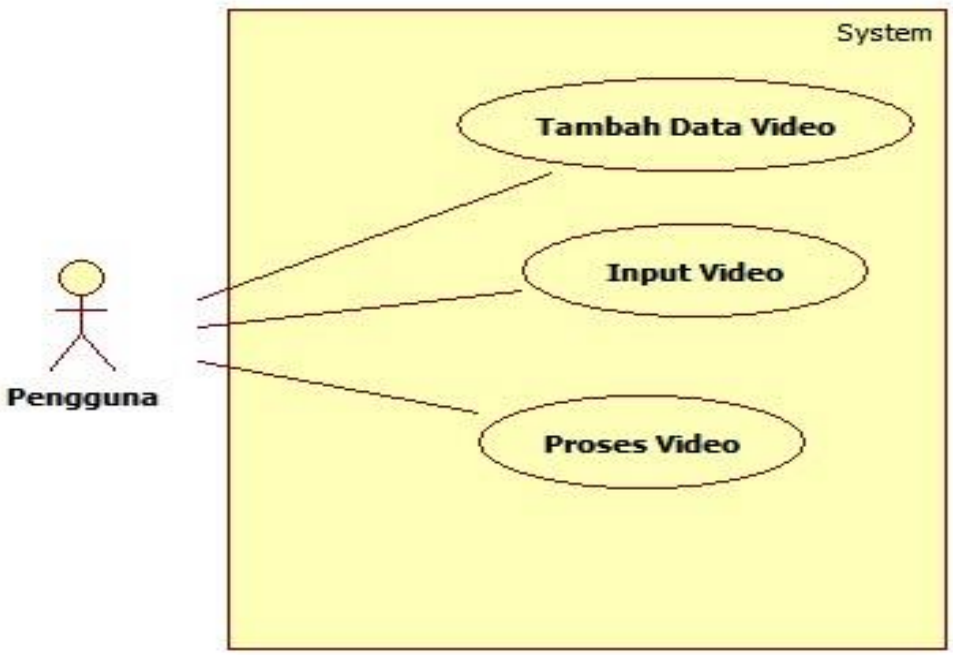
1. Sistem Operasi Window 7 atau lebih
2. Net 4 Framework
3. Library OpenCV 2.3.1 atau lebih

3.4.3. Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional bertujuan untuk menguraikan proses kegiatan dan kebutuhan yang akan diterapkan pada sistem yang akan dibangun. Kebutuhan fungsionalitas pada aplikasi *data mart* yang akan dibangun terdiri dari *use case diagram*, *activity diagram*, *sequence diagram*, dan *class diagram*.

3.4.3.1. Use Case Diagram

Use case diagram merupakan gambaran fungsionalitas atau perilaku dari sistem. Berikut *use case diagram* pada aplikasi dapat dilihat pada gambar 3.9



Gambar 3.9 Use Case Diagram

Deskripsi pendifinisian aktor dan deskripsi pendefinisian *use case* pada sistem dapat dilihat pada Tabel 3.10 dan Tabel 3.11

Tabel 3.10 Definisi use case

No	Aktor	Deskripsi
1	Pengguna	Orang yang bertugas untuk melakukan menganalisis data video

Tabel 3.11 Video skenario use case

No	Use Case	Deskripsi
1	Tambah Data Video	Fungsionalitas untuk masuk kedalam sistem
2	Input Video	Fungsionalitas untuk menambahkan data video ke dalam sistem
3	Proses Video	Fungsionalitas untuk proses video yang sudah di inputkan kedalam sistem
4	Perhitungan data	Fungsionalitas untuk proses perhitungan data dari video yang diinputkan
5	Ektraksi ciri	Fungsionalitas untuk mendaptakan nilai ektraksi ciri dari video yang diinputkan
4	Perhitungan C-means	Fungsionalitas untuk proses perhitungan data dari ektrasi ciri suatu video untuk Untuk mendapatkan klasifikasi kendaraan

3.4.3.2. Skenario Use Case

Skenario *use case* mendeskripsikan langkah-langkah dalam proses sistem, baik dilakukan aktor terhadap sistem maupun yang dilakukan oleh sistem terhadap aktor. Penjelasan skenario *use case* adalah sebagai berikut :

1. Skenario Use Case Tambah Data Video

Skenario *use case* untuk tambah data video dapat dilihat pada tabel 3.12.

Tabel 3.12 Skenario Use Case Diagram Tambah Data Video

Use Case Name	Tambah Data Video	
Goal In Context	Pengguna masuk kedalam sistem	
Description	Fungsionalitas ini digunakan oleh pengguna agar dapat masuk kedalam sistem	
Related Use Case	Input Video, Proses Video, Perhitungan Data	
Successfull End Condition	Pengguna dapat masuk ke dalam sistem.	
Failed End Condition	Pengguna gagal masuk ke dalam sistem.	
Actor	Pengguna	
Trigger	Pengguna menekan tombol Tambah Data Video	
Main Flow	Step	Action
	1	Sistem menampilkan halaman tambah data video
Ektension	Step	Branching Action
	1.1	Sistem menampilkan pesan kesalahan

2. Skenario Use Case Proses Video

Skenario *use case* untuk proses video dapat dilihat pada tabel 3.13

Tabel 3.13 Skenario Use Case Proses Video

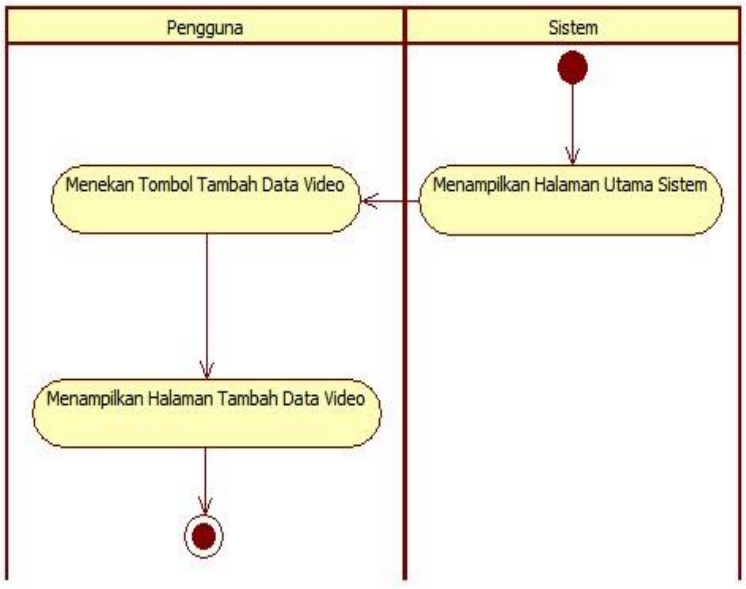
Use Case Name	Proses Video	
Goal In Context	Sistem dapat memproses video	
Description	Fungsionalitas ini digunakan oleh sistem untuk memproses video original	
Related Use Case	-	
Successfull End Condition	Sistem dapat memproses video original	
Failed End Condition	Sistem gagal memproses video original	
Actor	Pengguna	
Trigger	Pengguna menekan tombol proses video	
Main Flow	Step	Action
	1	Sistem menampilkan proses video
Ektension	Step	Branching Action
	1.1	Proses menampilkan proses video gagal

2.1.1.1. Activity Diagram

Activity diagram memodelkan aliran kerja dari urutan aktivitas dalam suatu proses yang mengacu pada *use case diagram* yang ada. Penjelasan masing-masing *activity diagram* adalah sebagai berikut :

1. Activity diagram tambah data video

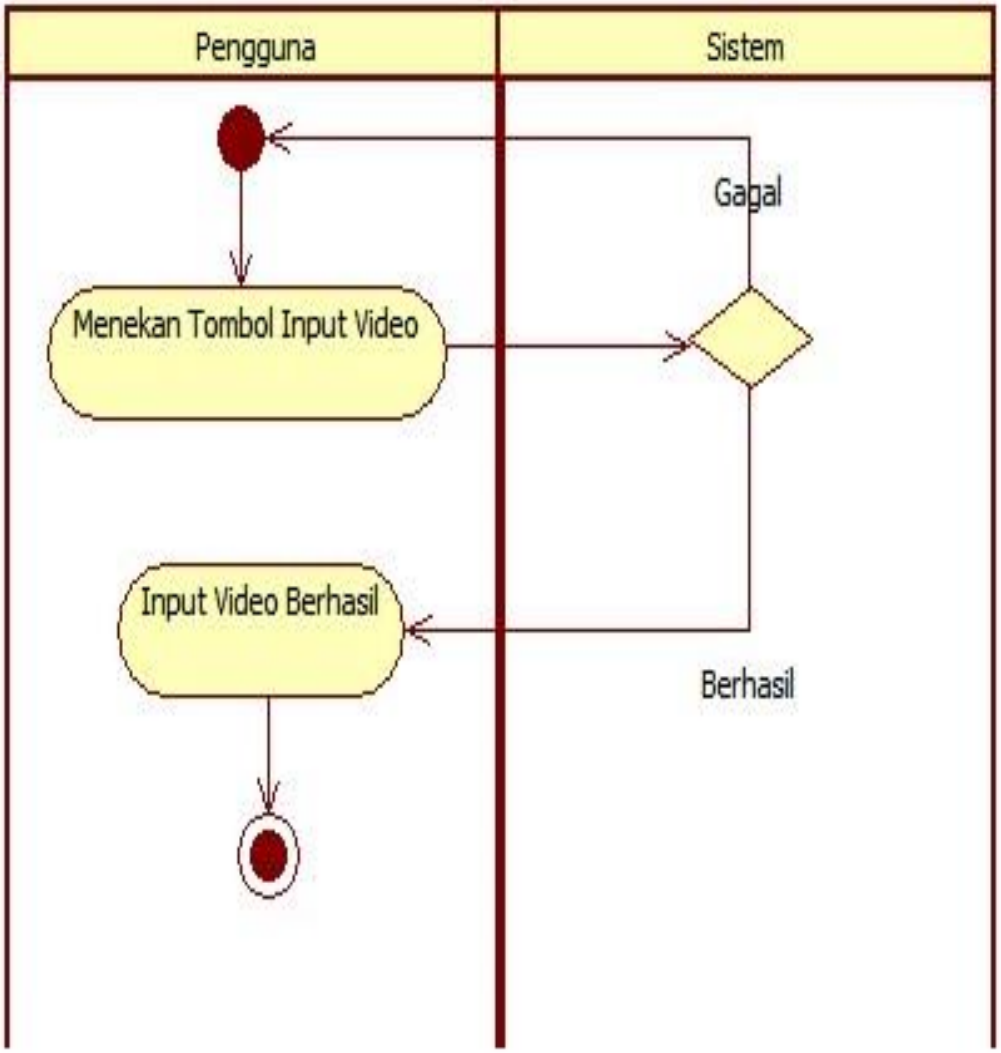
   Activity diagram Tambah Data Video merupakan aktifitas yang dilakukan oleh *pengguna* untuk dapat masuk ke dalam sistem. *Activity diagram* dari tambah data video dapat dilihat pada gambar 3.10



Gambar 3.10 Activity Diagram Tambah Data Video

2. Activity Diagram Input Video

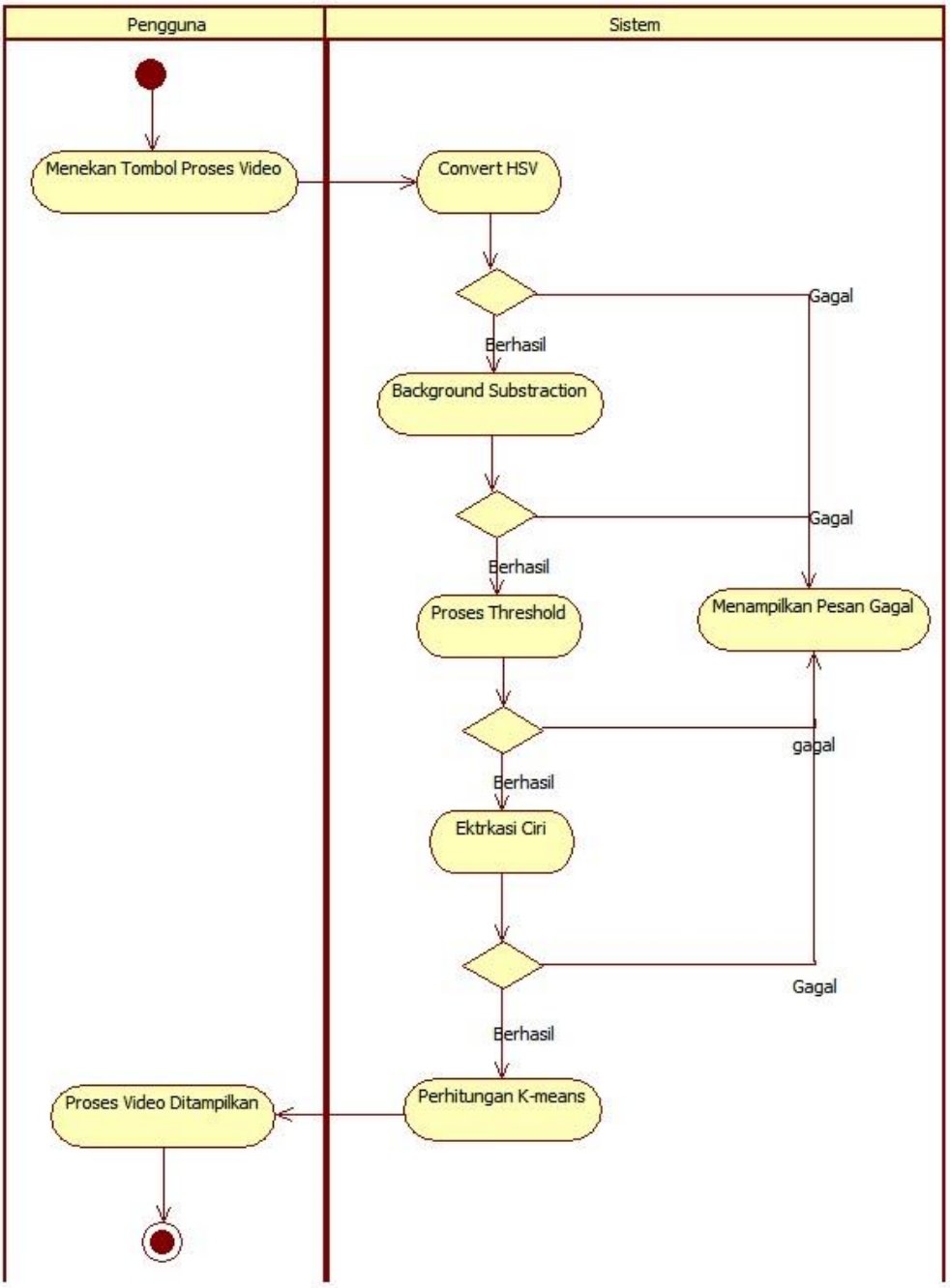
Activity Diagram Input video merupakan aktifitas yang dilakukan oleh pengguna untuk menginputkan video kedalam sistem. *Activity diagram* input video dapat dilihat pada gambar 3.11



Gambar 3.11 Activity Diagram Input Video

3. Activity Diagram Proses Video

Activity diagram proses video merupakan aktifitas yang dilakukan oleh pengguna untuk memproses suatu video. *Activiy diagram* proses video dapat dilihat pada gambar 3.12



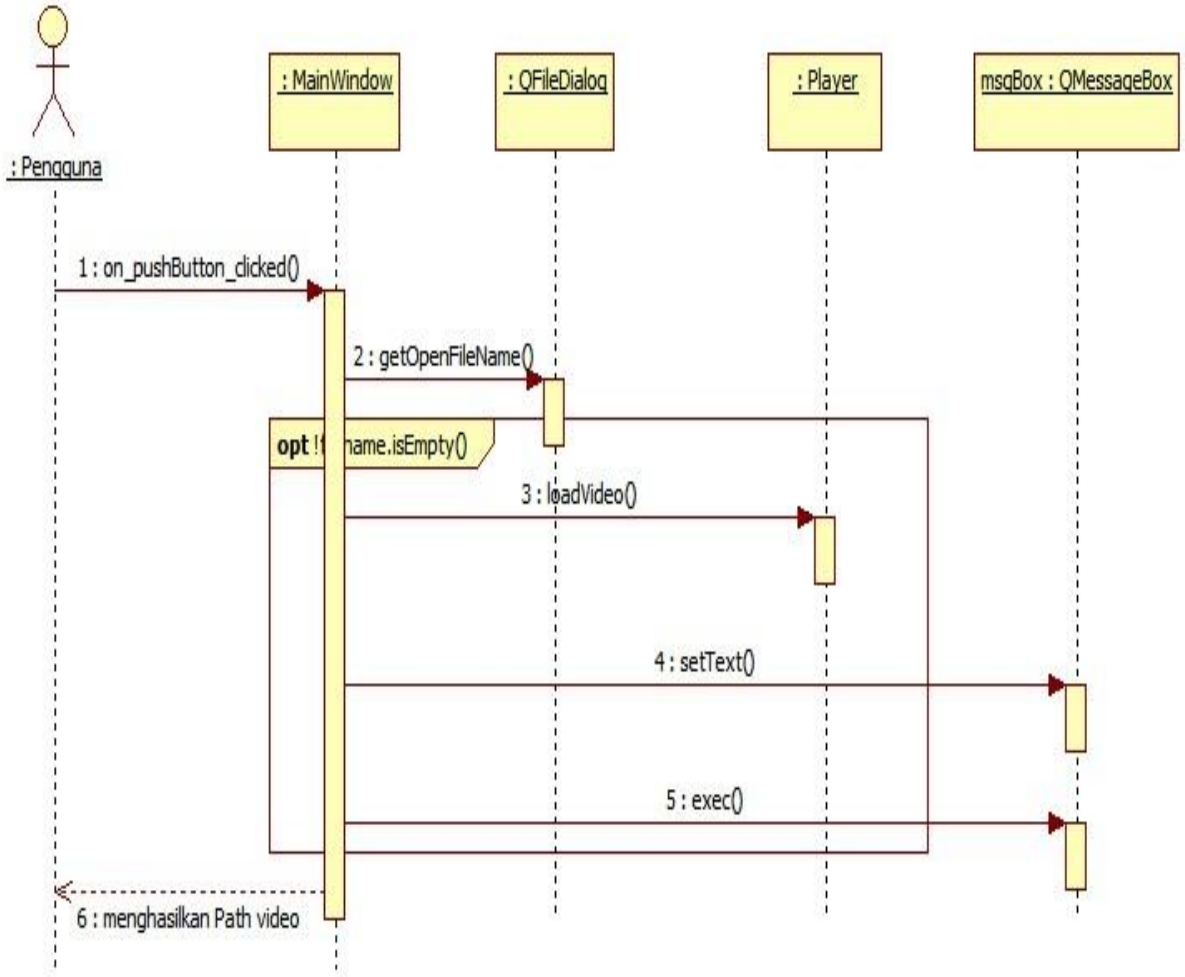
Gambar 3.12 Activity Diagram Proses Video

2.1.1.2. Sequence Diagram

Sequence diagram menggambarkan interaksi antar masing-masing objek pada setiap *use case* dalam urutan waktu. Interaksi ini berupa pengiriman serangkaian data antar objek yang saling berinteraksi. Berikut penjelasan dari masing-masing *sequence diagram*.

1. Sequence Diagram Input Video

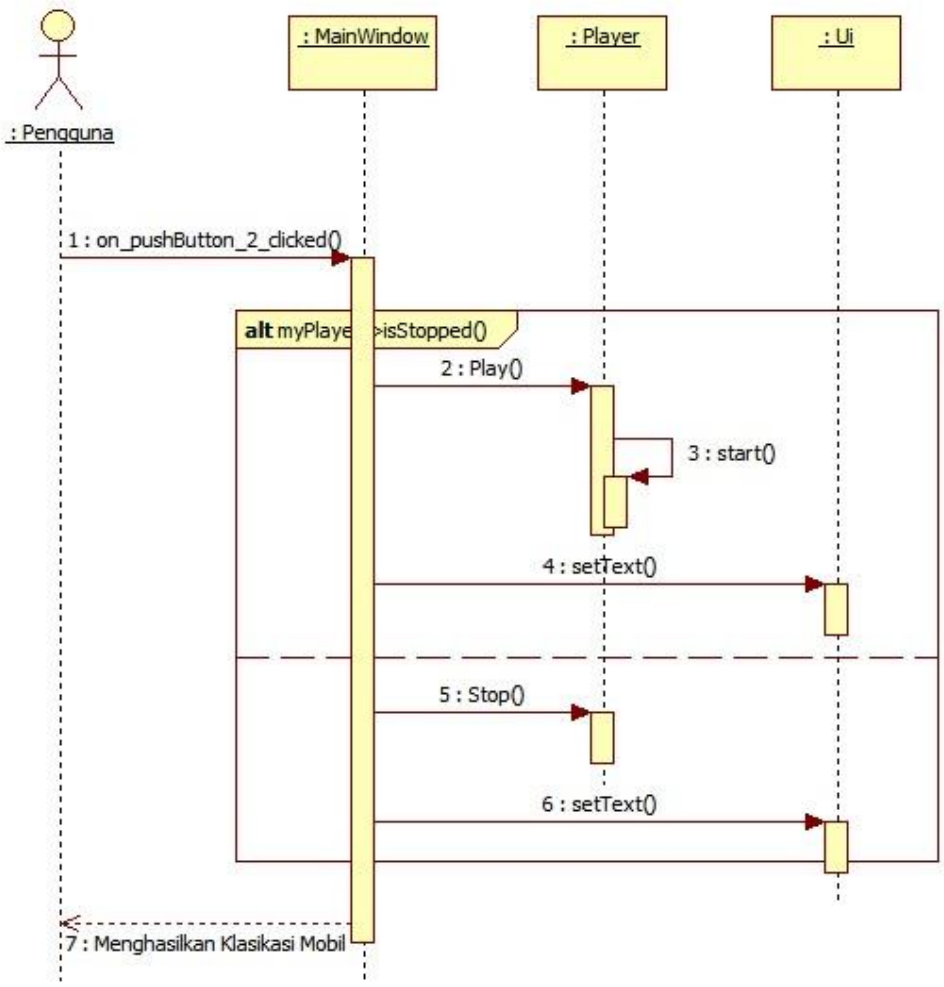
   Sequence diagram input video dapat dilihat pada gambar 3.13



Gambar 3.13 Sequence Diagram Input Video

2. Sequence Diagram Proses Video

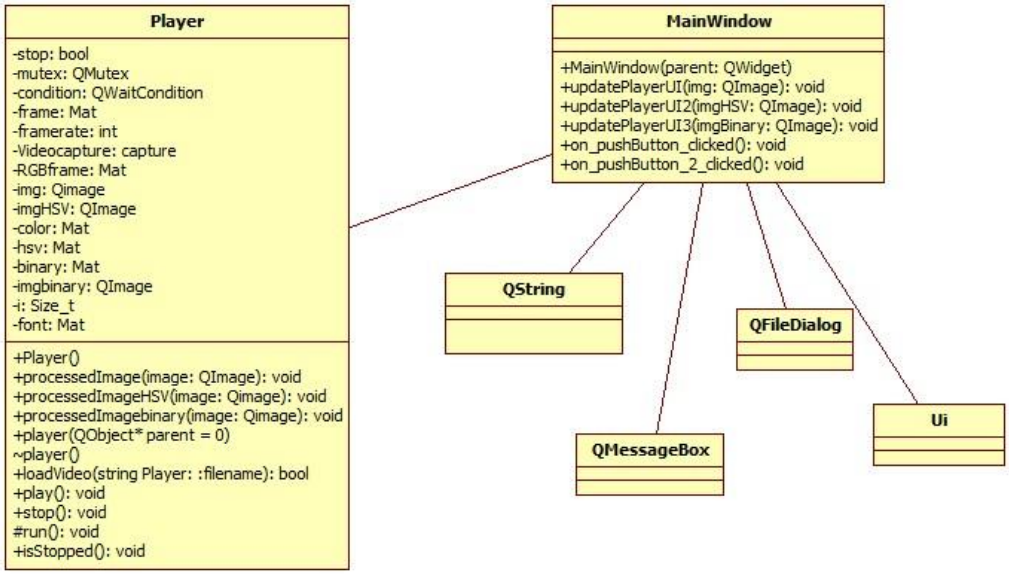
   sequence diagram proses video dapat dilihat pada gambar 3.14



Gambar 3.14 Sequence Diagram Proses Video

2.1.1.3. Class Diagram

Class diagram menggambarkan struktur dan hubungan antar objek-objek yang ada pada sistem. Struktur ini meliputi atribut-atribut dan metode-metode yang ada pada masing-masing kelas. *Class player* dan *Class MainWindow* yang akan dibangun dapat dilihat pada Gambar 3.15



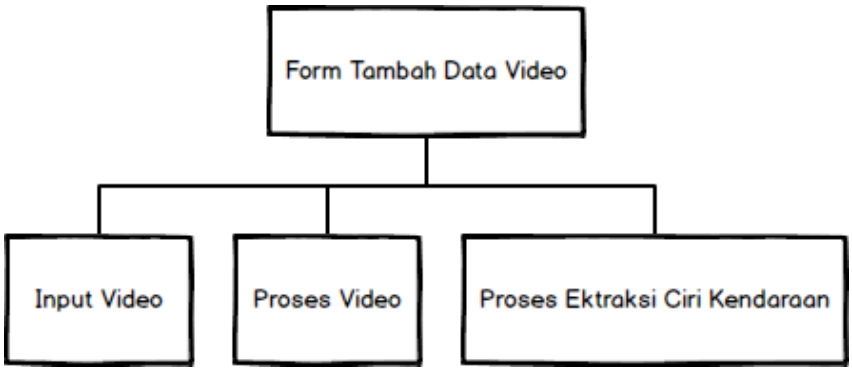
Gambar 3.15 Class Diagram

3.5. Perancangan Sistem

Pada sub bab ini akan membahas tentang perancangan sistem untuk Aplikasi Penghitung Kendaraan Menggunakan OpenCv. Sub bab ini terdiri dari perancangan struktur menu, perancangan antar muka, perancangan pesan, perancangan jaringan semantik, dan perancangan method.

3.5.1. Perancangan Struktur Menu

Berkut ini adalah perancangan struktur menu dari aplikasi perhitungan kendaraan yang akan dibangun dapat dilihat pada Gambar 3.16

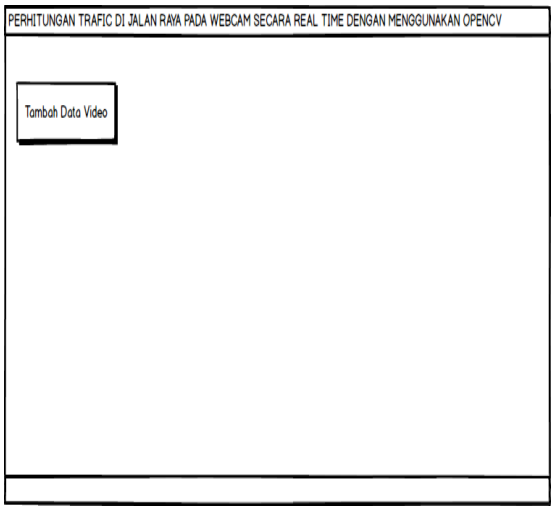


Gambar 3.16 Struktur Menu

3.5.2. Perancangan Antar Muka

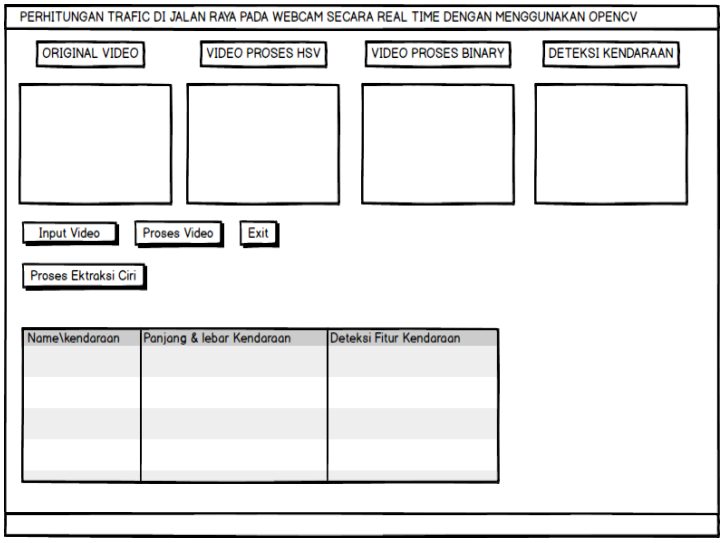
Perancangan antar muka merupakan desain tampilan yang akan dilihat oleh pengguna dari sistem yang akan dibuat. Berikut perancangan antar muka pada aplikasi perhitungan kendaraan ini diantaranya :

1. Perancangan halaman utama tambah data video Halaman Utama dapat dilihat pada gambar 3.17

AO1  	- Klik tombol tambah data video untuk masuk A02 yaitu Input Video

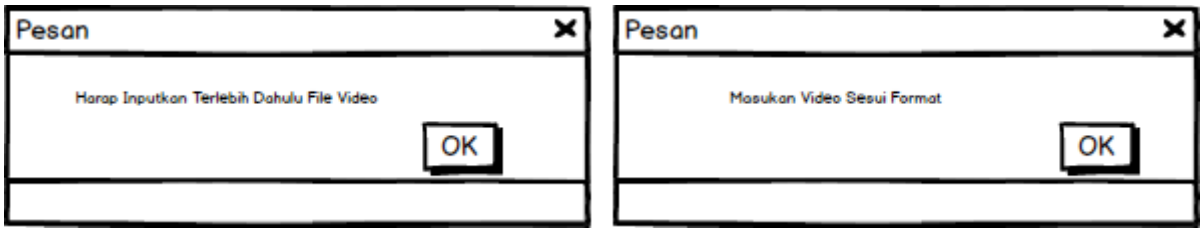
Gambar 3.17 Perancangan Antar Muka Tambah Data Video

2. Halaman input video dapat dilihat pada gambar 3.18

AO2  	- Klik Tombol Input Video Untuk memanggil file video yang akan diproses. - Klik Tombol Proses untuk memproses file video yang sudah dinputkan dan ditampilkan di original video, video proses hsv, video proses binary, dan deteksi kendaraan - Klik tombol proses ektraksi ciri untuk mendapatkan klasifikasi kendaraan, - Panjang dan lebar kendaraan deteksi fitur kendaraan - Klik tombol exit untuk keluar dari form halaman input video

Gambar 3.18 Perancangan Halaman Input Video Perancangan Pesan

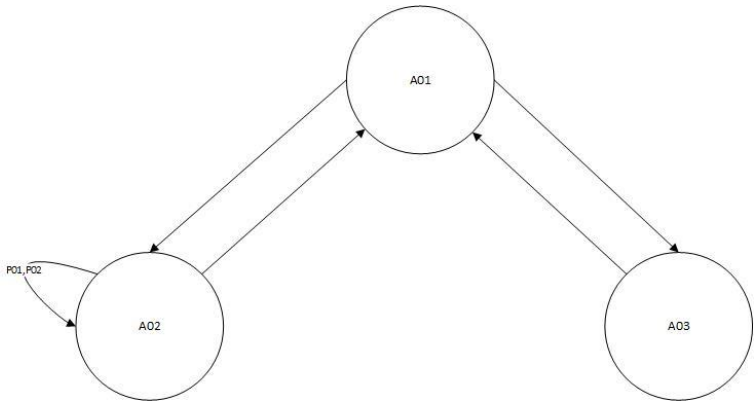
Perancangan pesan yang terdapat pada aplikasi perhitungan kendaraan yang akan dibangun dapat dilihat pada gambar 3.19



Gambar 3.19 Perancangan Pesan

3.5.3. Perancangan Jaringan Semantik

Perancangan jaringan semantik merupakan sebuah gambaran grafis yang berupa node-node yang menggambarkan aliran menu-menu yang ada di aplikasi perhitungan kendaraan perancangan jaringan semantik dapat dilihat pada Gambar 3.20



Gambar 3.20 Perancangan Jaringan Semantik

Keterangan

jaringan semantik :

A01: Tambah Data Video

A02: Input Video

A03: Proses Video

P01: Pesan kesalahan input video file

P02: Pesan kesalahan input video belum dipilih