BAB 3
ANALISIS DAN PERENCANAAN

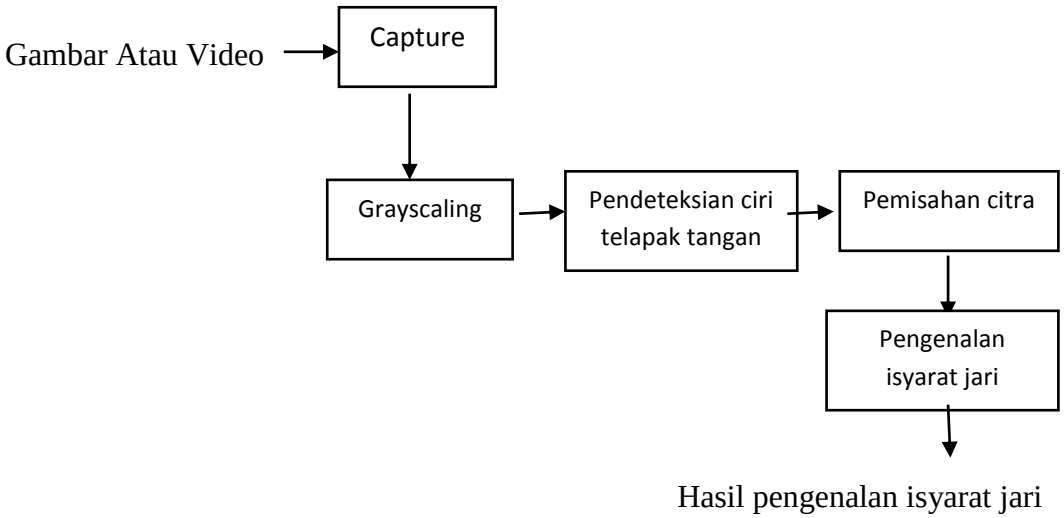
3.1 Analisis Masalah

Analisis masalah merupakan langkah awal dari suatu analisis sistem. Langkah ini diperlukan untuk mengetahui permasalahan apa saja yang terjadi di dalam sistem yang sedang berjalan. Setelah melakukan penelitian maka terdapat beberapa permasalahan yang timbul dalam implementasi metode viola and jones untuk petunjuk navigasi pada perintah play, pause, next, previous dan stop, yakni masih terdapat kesalahan dalam mengenali isyarat jari yang disebabkan oleh terdapatnya citra uji yang memiliki resolusi rendah sehingga gambar terlihat kurang halus serta faktor pencahayaan yang gelap pada citra gambar sehingga akan mempersulit sistem baik untuk pendeteksian maupun pengenalan isyarat jari pada citra gambar.

Dari permasalahan tersebut, maka perlu diterapkan suatu algoritma pendukung yang dapat menormalisasikan citra gambar yang diproses oleh sistem sehingga dapat memperkecil tingkat kesulitan yang terjadi pada sistem. Maka sebagai solusi untuk hal tersebut, algoritma *haar-cascade* akan dicoba untuk diterapkan pada sistem. Sehingga akan didapatkan hasil berupa seberapa besarkah dampak dari penerapan algoritma ini dalam menangani kesulitan yang terjadi pada sistem

3.1.1 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. Dalam sistem pengenalan Isyarat jari terdapat beberapa tahapan. Adapun tahapan sistem tersebut dapat dilihat pada gambar 3.1



Gambar 3.1 Proses utama sistem

1. Penangkapan/*capture* pada citra masukan

Merupakan langkah pengambilan data citra pada media masukan kedalam sistem

2. *Grayscaling* citra

Merupakan proses perubahan warna pada citra kedalam derajat keabuan

3. Pendeteksian Ciri jari menggunakan *metode viola jones*

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

4. Pemisahan citra

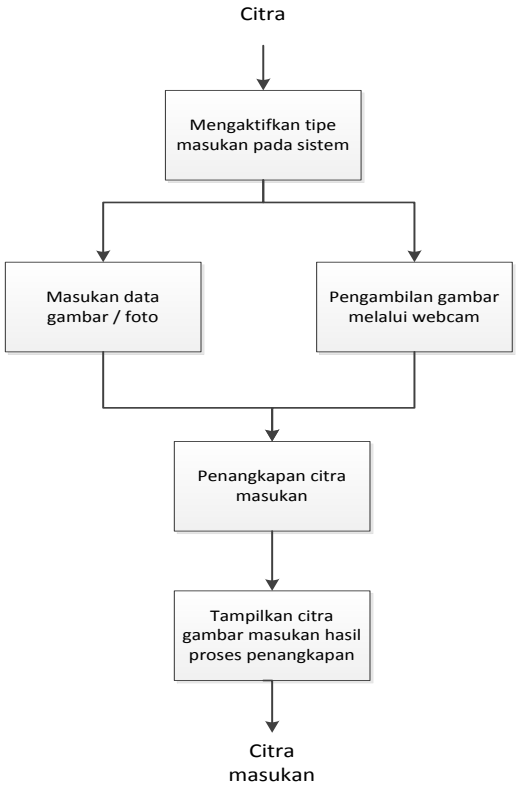
Merupakan proses pemotongan/pemisahan objek pada citra

5. Pengenalan isyarat jari

Merupakan proses pengenalan/pencocokan dari isyarat jari yang telah disimpan didalam sistem dengan isyarat jari yang diujikaN

3.1.1.1 Penangkapan/*capture* Pada Citra Masukan

Penangkapan citra masukan yang dilakukan oleh sistem terhadap tipe citra yang dimasukan yaitu dapat berupa file gambar dengan menggunakan menu *input* atau juga dapat diambil secara langsung melalui kamera *webcam*. Untuk lebih jelasnya dapat dilihat pada gambar berikut :



Gambar 3.2 Alur proses penangkapan citra masukan

3.1.1.2 Grayscaling Citra

Proses pengubahan nilai warna RGB pada setiap *pixel* citra gambar kedalam derajat keabuan. Untuk lebih jelasnya dapat dilihat pada alur diagram berikut :



Gambar 3.3 Alur proses *grayscaling* citra

3.1.1.2 Pendeteksian Ciri jari menggunakan Metode *Viola Jones*

Proses pengolahan pada citra dengan metode yang telah ditentukan sebelum dilakukan proses selanjutnya terhadap citra. Untuk lebih lengkapnya dapat dilihat pada gambar alur berikut :

Citra abu



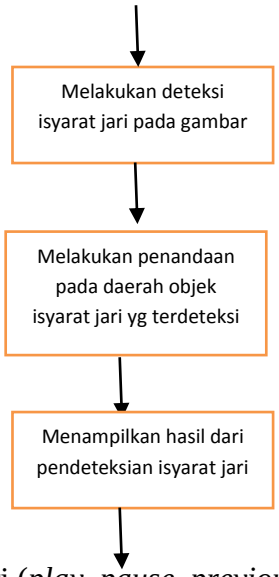
Citra Baru

Gambar 3.4 Alur proses pre-processing citra dengan *Metode Viola Jones*

3.1.1.3 Pendeteksian Isyarat Jari

Proses untuk melakukan pencarian objek isyarat jari pada citra gambar kemudian objek isyarat jari pada gambar akan diberi tanda. Untuk alurnya dapat dilihat pada gambar 3.5.

Citra gambar



Perintah isyarat jari (*play*, *pause*, *previou*, *next* dan *stop*)

Gambar 3.5 Alur proses pendeteksian isyarat jari

3.1.2 Analisis Masukan

Data masukan yang digunakan berupa citra gambar. Citra gambar yang dimasukan bisa berupa file gambar/foto atau juga citra gambar dapat diambil melalui kamera pada komputer yaitu kamera *webcam*. Maka setelah citra gambar dimasukan pada sistem maka akan dilakukan proses dari tahapan-tahapan yang telah diketahui sebelumnya. Contoh citra gambar yang dapat digunakan dapat dilihat pada gambar berikut :



Gambar 3.6 citra masukan

Tahap ini dilakukan untuk mengubah citra warna menjadi citra *greyscale*, hal ini digunakan untuk menyederhanakan model citra. Citra warna terdiri dari 3 layer matrik yaitu R-layer, G-layer, B-layer. Sebagai contoh, misalkan citra berukuran 8x8 *pixel* dengan nilai-nilai R, G, dan B sebagai berikut:

84	82	102	109	120	134	148	167
78	81	89	105	121	134	152	168
86	82	98	112	132	150	159	178
91	95	109	125	140	158	169	186
94	103	120	137	151	165	175	194
107	109	128	144	161	174	188	201
109	115	136	156	170	187	197	205
122	130	146	166	185	197	205	219

Matriks Red

78	76	90	99	110	121	142	167
69	69	74	92	106	119	138	157
77	69	83	97	115	132	141	163
79	78	92	106	119	136	145	165
81	84	101	116	128	139	147	170
94	92	107	123	138	148	159	175
92	97	113	133	145	157	168	178
107	115	124	144	160	168	179	192

Matriks Green

44	40	52	63	74	86	110	132
30	31	31	48	67	80	101	125
38	27	40	54	72	92	105	132
39	34	49	63	76	97	109	134
39	41	58	71	87	102	110	142
52	48	64	80	97	111	127	148
49	51	69	92	105	121	138	149
68	76	83	107	129	136	152	171

Matriks Blue

Gambar 3.7 Matrik (R, G dan B)

Jadi dalam proses ini akan mengubah 3 layer matrik citra berwarna menjadi 1 layer matrik *grayscale* dan akan menghasilkan citra *grayscale*. Dalam citra ini tidak ada lagi warna, yang ada adalah derajat keabuan. Pada

dasarnya proses *grayscaling* dilakukan dengan meratakan nilai *pixel* dari 3 nilai R, G, dan B dengan menggunakan persamaan berikut:

Grayscale = 0.299*R + 0.587*G + 0.114*B .....(3-1)

Contoh untuk menentukan nilai *grayscale* diilustrasikan dengan nilai-nilai R, G, dan B pada gambar 3.3, untuk konversi warna menjadi *grayscale* digunakan persamaan 3-1. Misalkan untuk *pixel* di koordinat (0,0) perhitungannya sebagai berikut:

Grayscale = (0,299*84) + (0,587*78) + (0,114*44)
= 25,12 + 45,79 + 5,02
= 75,93

Didapatkan hasil perhitungan untuk koordinat (0,0) = 75,93. Setelah perhitungan tersebut dilakukan untuk semua koordinat, maka dari hasil perhitungan *grayscale* dilakukan *rounding* (pembulatan) nilai ke paling dekat. Jadi hasil pembulatan dari koordinat untuk (0,0) = 76. Sehingga didapatkan citra *grayscale* sebagai berikut:

(i,j)	0	1	2	3	4	5	6	7
0	76	74	93	97	108	120	140	163
1	73	72	82	68	105	118	136	155
2	75	69	67	96	114	132	141	162
3	78	78	92	106	119	136	146	165
4	80	84	101	116	129	140	149	171
5	93	89	107	123	140	149	171	177
6	92	96	114	134	146	156	170	180
7	106	114	124	144	161	170	181	194

Gambar 3.8 Matriks Grayscale

Citra yang berwarna akan digantikan dengan citra *grayscale* sesuai dengan nilai yang sudah didapatkan melalui perhitungan di atas. Berikut adalah citra *grayscale* yang dihasilkan.



Gambar 3.9 Hasil Konversi Citra RGB Menjadi Citra *Grayscale*

3.1.14 Analisis Metode *Viola Jones* Pada pengenalan isyarat jari

Berikut adalah Alur Algoritma *Viola and Jones* :



Gambar 3.10 Metode Viola Jones

3.1.2.1.1 *Scan* Per *Sub window*

Kemudian setelah proses penormalisasian gambar menggunakan grayscale selesai dilakukan maka tahapan selanjutnya adalah proses pendeteksian objek wajah yang terkandung pada citra gambar masukan dengan

menggunakan library *Haarcascade Frontal hand*, yaitu suatu *library* berupa *file* templet berekstensi *xml*. *Library* ini mengadopsi metode Viola-Jones yang memiliki fungsi untuk mendeteksi objek berupa isyarat jari dengan posisi tegak lurus kedepan. File templet ini memiliki karakteristik sebagai berikut :

1. Menggunakan *sub-window* citra dengan dimensi 20x20
2. Memiliki 47 tahapan (*stage* 0 sampai *stage* 46) *cascade classifier*
3. Memiliki 2133 *weak classifier* (*tree*)

Berikut ini adalah potongan yang diambil dari sebagian kode *file* templet *play*.xml :

```
<?xml version="1.0"?>
<opencv_storage>
<output type_id="opencv-haar-classifier">
 <size>
  24 24</size>
 <stages>
  <_>
   <trees>
    <_>
     <_>
      <feature>
       <rects>
        <_>
         6 8 9 15 -1.</_>
        <_>
         9 13 3 5 9.</_></rects>
       <tilted>0</tilted></feature>
      <threshold>-0.0847022831439972</threshold>
      <left_val>1.</left_val>
      <right_val>-1.0000009536743164</right_val></_></_></trees>
```

```
  <stage_threshold>1.</stage_threshold>
  <parent>-1</parent>
  <next>-1</next></_>
<_>
  <trees>
   <_>
    <_>
     <feature>
      <rects>
       <_>
        10 6 3 14 -1.</_>
       <_>
        10 13 3 7 2.</_></rects>
      <tilted>0</tilted></feature>
     <threshold>-0.0366368703544140</threshold>
     <left_val>0.9285715222358704</left_val>
     <right_val>-1.0000009536743164</right_val></_></_></trees>
  <stage_threshold>0.9285715222358704</stage_threshold>
  <parent>0</parent>
  <next>-1</next></_>
<_>
  <trees>
   <_>
    <_>
     <feature>
      <rects>
       <_>
        11 20 12 2 -1.</_>
       <_>
        14 20 6 2 2.</_></rects>
      <tilted>0</tilted></feature>
```

```
        <threshold>-9.4083501026034355e-003</threshold>
        <left_val>1.</left_val>
        <right_val>-0.9047629833221436</right_val></_></_>
      <_>
       <_>
        <feature>
         <rects>
          <_>
           11 19 12 3 -1.</_>
          <_>
           14 19 6 3 2.</_></rects>
         <tilted>0</tilted></feature>
        <threshold>-5.6348210200667381e-003</threshold>
        <left_val>0.8808284997940064</left_val>
        <right_val>-1.0000009536743164</right_val></_></_></trees>
    <stage_threshold>-0.0239344798028469</stage_threshold>
    <parent>1</parent>
    <next>-1</next></_>
  <_>
    <trees>
     <_>
      <_>
       <feature>
        <rects>
         <_>
          0 11 21 3 -1.</_>
         <_>
          7 12 7 1 9.</_></rects>
        <tilted>0</tilted></feature>
       <threshold>-0.0456034094095230</threshold>
       <left_val>0.9164927005767822</left_val>
```

```
        <right_val>-0.7304499149322510</right_val></_></_>
      <_>
       <_>
        <feature>
         <rects>
          <_>
           0 1 8 1 -1.</_>
          <_>
           2 1 4 1 2.</_></rects>
         <tilted>0</tilted></feature>
        <threshold>7.1512709837406874e-004</threshold>
        <left_val>0.7968924045562744</left_val>
        <right_val>-0.9208245277404785</right_val></_></_></trees>
    <stage_threshold>-4.3318271636962891e-003</stage_threshold>
    <parent>2</parent>
    <next>-1</next></_>
  <_>
    <trees>
     <_>
      <_>
       <feature>
        <rects>
         <_>
          10 14 8 2 -1.</_>
         <_>
          10 14 8 1 2.</_></rects>
        <tilted>1</tilted></feature>
       <threshold>-2.0649400539696217e-003</threshold>
       <left_val>-0.8975331783294678</left_val>
       <right_val>0.8553348183631897</right_val></_></_>
     <_>
```

```
      <_>
        <feature>
          <rects>
            <_>
              14 19 3 2 -1.</_>
            <_>
              14 19 3 1 2.</_></rects>
          <tilted>1</tilted></feature>
        <threshold>-7.1654270868748426e-004</threshold>
        <left_val>-1.</left_val>
        <right_val>0.7942516803741455</right_val></_></_></trees>
    <stage_threshold>-0.1032814979553223</stage_threshold>
    <parent>3</parent>
    <next>-1</next></_>
  <_>
    <trees>
      <_>
        <_>
          <feature>
            <rects>
              <_>
                19 17 1 4 -1.</_>
              <_>
                18 18 1 2 2.</_></rects>
            <tilted>1</tilted></feature>
          <threshold>-4.7855640877969563e-004</threshold>
          <left_val>-1.</left_val>
          <right_val>0.7419356107711792</right_val></_></_></trees>
    <stage_threshold>0.7419356107711792</stage_threshold>
    <parent>4</parent>
    <next>-1</next></_>
```

```
<_>
  <trees>
    <_>
      <_>
        <feature>
          <rects>
            <_>
              14 14 9 9 -1.</_>
            <_>
              17 14 3 9 3.</_></rects>
          <tilted>0</tilted></feature>
        <threshold>-0.0270329192280769</threshold>
        <left_val>0.7419356107711792</left_val>
        <right_val>-1.0000009536743164</right_val></_></_></trees>
  <stage_threshold>0.7419356107711792</stage_threshold>
  <parent>5</parent>
  <next>-1</next></_>
<_>
  <trees>
    <_>
      <_>
        <feature>
          <rects>
            <_>
              14 16 2 1 -1.</_>
            <_>
              15 16 1 1 2.</_></rects>
          <tilted>0</tilted></feature>
        <threshold>2.2036230802768841e-005</threshold>
        <left_val>-1.</left_val>
        <right_val>0.8620691895484924</right_val></_></_></trees>
```

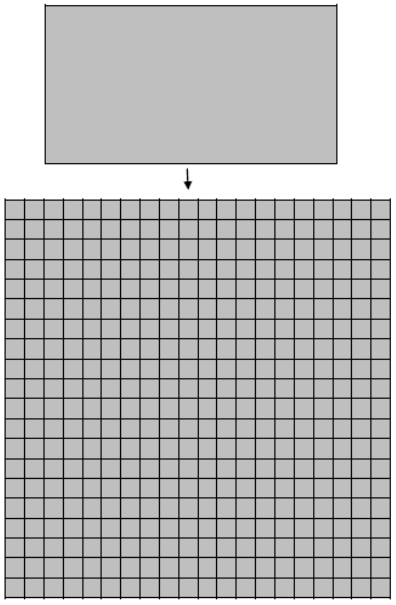
```
    <stage_threshold>0.8620691895484924</stage_threshold>
    <parent>6</parent>
    <next>-1</next></_>
  <_>
    <trees>
      <_>
        <_>
          <feature>
            <rects>
              <_>
                7 14 3 6 -1.</_>
              <_>
                8 16 1 2 9.</_></rects>
            <tilted>0</tilted></feature>
          <threshold>-2.3395009338855743e-003</threshold>
          <left_val>0.7419356107711792</left_val>
          <right_val>-1.0000009536743164</right_val></_></_></trees>
    <stage_threshold>0.7419356107711792</stage_threshold>
    <parent>7</parent>
    <next>-1</next></_>
  <_>
    <trees>
      <_>
        <_>
          <feature>
            <rects>
              <_>
                14 1 3 4 -1.</_>
              <_>
                14 2 3 2 2.</_></rects>
            <tilted>0</tilted></feature>
```

```
        <threshold>-1.2086000060662627e-003</threshold>
        <left_val>0.8479086756706238</left_val>
        <right_val>-0.8050550818443298</right_val></_></_>
      <_>
       <_>
        <feature>
         <rects>
          <_>
           9 7 8 1 -1.</_>
          <_>
           9 7 4 1 2.</_></rects>
         <tilted>1</tilted></feature>
        <threshold>0.0135317100211978</threshold>
        <left_val>-0.8523197770118713</left_val>
        <right_val>0.8387230038642883</right_val></_></_></trees>
    <stage_threshold>-4.4111008755862713e-003</stage_threshold>
    <parent>8</parent>
    <next>-1</next></_>
  <_>
    <trees>
     <_>
      <_>
       <feature>
        <rects>
         <_>
          5 9 2 4 -1.</_>
         <_>
          5 9 2 2 2.</_></rects>
        <tilted>1</tilted></feature>
       <threshold>8.4033068269491196e-003</threshold>
       <left_val>0.6363639235496521</left_val>
```

```
        <right_val>-1.0000009536743164</right_val></_></_></trees>
    <stage_threshold>0.6363639235496521</stage_threshold>
    <parent>9</parent>
    <next>-1</next></_>
  <_>
    <trees>
     <_>
      <_>
       <feature>
        <rects>
         <_>
          11 19 6 4 -1.</_>
         <_>
          13 19 2 4 3.</_></rects>
        <tilted>0</tilted></feature>
       <threshold>-1.0043339570984244e-003</threshold>
       <left_val>-1.</left_val>
       <right_val>0.8000001907348633</right_val></_></_></trees>
    <stage_threshold>0.8000001907348633</stage_threshold>
    <parent>10</parent>
    <next>-1</next></_></stages></output>
</opencv_storage>
```

Gambar 3.11 *Source code library file play.xml*

Pada awal kode, tertera <*size*> yang mendefinisikan ukuran *sub-window* yang digunakan yaitu 20x20. Hal ini menunjukan bahwa setiap citra masukan akan dibentuk menjadi *sub-window* berukuran 20x20 sebelum masuk ke tahapan deteksi isyarat jari. Gambar 3.20 akan menjelaskan pembentukan *sub-window* 20x20.



Gambar 3.12 Pembentukan *sub-window*

Setelah *sub-window* terbentuk tahap selanjutnya adalah tahap klasifikasi yang melalui tingkatan-tingkatan (*stage*) *cascade classifier* dan *tree* sebagai *weak classifier*. Didalam *tree* terdapat satu *node* yaitu *root node,* yang isinya berbeda-beda disetiap *tree*. Didalam *node* ini, terdapat fitur *Haar* (*rects*), nilai ambang (*treshold*) proses deteksi isyarat jari, nilai batas minimum (*left_val*) dan nilai batas maksimum (*right_val*) yang harus dipenuhi agar *sub-window* lolos ke tahap selanjutnya. Untuk dapat melewati suatu tahapan *sub-window* harus berhasil melewati setiap *tree* didalam suatu tahapan. Perhitungan nilai fitur dilakukan juga pada *tree*. Sebagai contoh, berikut ini adalah perhitungan fitur *haar* (*rects*) pada suatu *tree*.

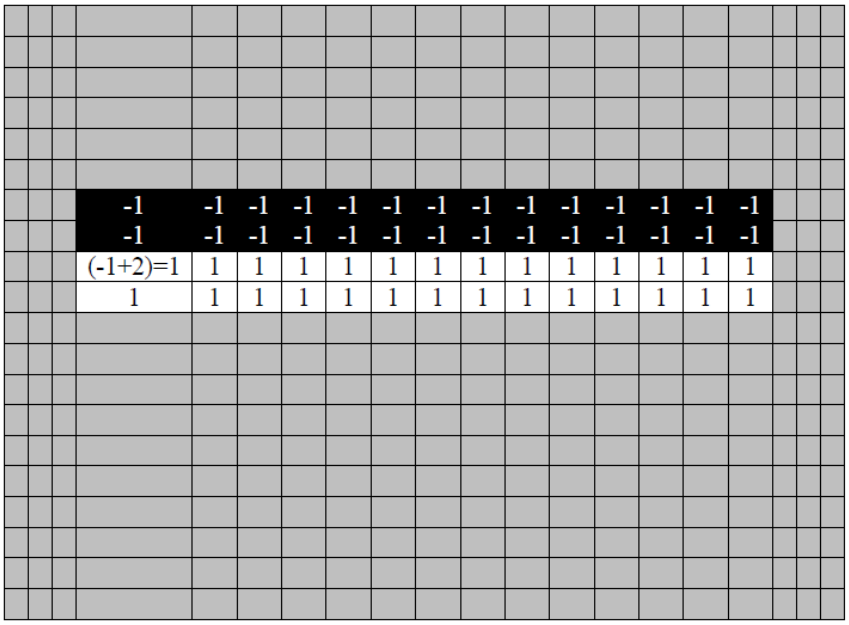
<rects>

<_>

11 19 6 4 -1.</_>

<_>
13 19 2 4 3.</_></rects>

Angka pada kolom pertama dan kedua menunjukan posisi piksel yang akan diklasifikasikan (x,y) pada *sub-window*. Kolom ketiga dan keempat menunjukan lebar dan tinggi fitur. Sedangkan, angka pada kolom terakhir adalah konstanta yang akan dikalikan ke setiap fitur persegi tersebut, dimana angka ini berfungsi untuk menentukan piksel hitam dan piksel putih suatu fitur, dengan ketentuan jika nilai suatu piksel dalam suatu piksel < 0, maka piksel tersebut masuk ke daerah piksel hitam, jika tidak piksel tersebut masuk ke daerah putih.



Gambar 3.13 Contoh fitur *Haar*

Gambar tersebut menjelaskan gambaran fitur dalam *sub-window* pada kasus fitur *Haar* (*rects*) diatas. Sesuai dengan gambar tersebut, pixel hitam adalah piksel dengan nilai fitur = -1, sedangkan piksel putih adalah piksel dengan nilai fitur = (-1+2) = 1. Setelah daerah piksel hitam dan piksel putih ditentukan, langkah selanjutnya adalah menghitung nilai fitur *Haar* keseluruhan caranya dengan mengurangi jumlah nilai fitur pada piksel area hitam dengan jumlah nilai

fitur pada piksel area putih. Hasilnya akan dibandingkan dengan nilai yang tertera pada *treshold* pendeteksian wajah, jika nilainya dibawah *treshold* proses pendeteksian isyarat jari maka fitur dianggap tidak ada, jika diatas *treshold* pendeteksian isyarat jari fitur akan dilanjutkan ke *left_val* dan *right_val*. Apabila, fitur tersebut memiliki nilai diantara *left_val* dan *right_val*, maka *sub-window* dinyatakan berhasil melewati *tree* tersebut.

Untuk dapat melanjutkan ke tingkat *classifier* selanjutnya, seluruh *sub-window* harus dapat melewati seluruh *tree* dalam suatu tingkat, jika tidak maka secara otomatis, *sub-window* akan dieleminasi dan dianggap bukan wajah. Jika, berhasil melalui keseluruhan tingkat *classifier* maka *sub-window* dianggap sebagai isyarat jari.

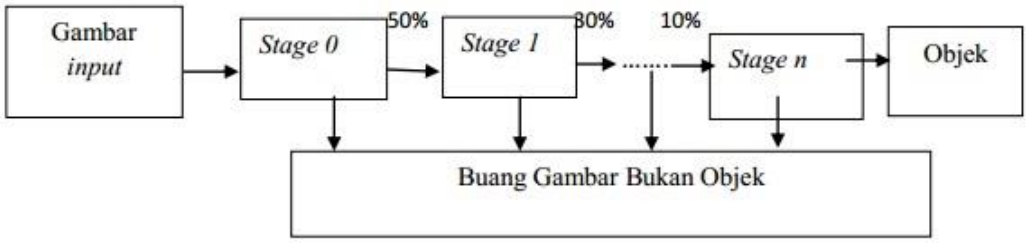
Setelah isyarat jari terdeteksi, maka citra isyarat jari tersebut akan dipisahkan dari keselruhan citra gambar. Setelah itu citra isyarat jari akan diubah ukurannya sesuai dengan ukuran citra isyarat jari yang tersimpan pada sistem kemudian kedua citra isyarat jari tersebut akan dicocokan menggunakan metode Metode *viola Jones*.

3.1.1.2.2 Seleksi Fitur *haar* dengan *AdaBoost*

Metode *machine* learning yang disebut *AdaBoost* digunakan dalam memilih fitur Haar yang spesifik. Hal ini dilakukan dengan cara mengevaluasi setiap fitur terhadap data latih dengan menggunakan nilai dari fitur tersebut. Algoritma *Adaboost* adalah algoritma yang berusaha membangun strong classifier dengan mengkombinasikan sejumlah simple atau *weak classifier* secara linier. *Stage classifier* dibangun dengan menggunakan algoritma *adaptive-boost (AdaBoost)*. Di dalam *stage classifier* terdapat proses pengklasifikasian fitur objek menggunakan algoritma *AdaBoost*. Algoritma *AdaBoost* digunakan untuk meningkatkan performa pengklasifikasian fitur.

Algoritma tersebut mengkombinasikan performance banyak *weak classifier* untuk menghasilkan strong classifier. *Weak classifier* dalam hal ini adalah nilai dari *haar-like feature*. Alur proses pendeteksian pada *Haar cascade*

classifier dapat dilihat pada Gambar 3.33.



Gambar 3.14 Alur proses Haar cascade classifier

Algoritma *AdaBoost* yang dipakai untuk pengklasifikasian adalah sebagai berikut:

Bobot awal $\boldsymbol{w_{1,y_i} = \frac{1}{2m}}$ **,** $\boldsymbol{w_{1,y_i} = \frac{1}{2l}}$ **(3.4)**

Dengan *m* = jumlah gambar negatif dengan $y_i = 0$ untuk gambar negatif

Dengan *l* = jumlah gambar positif dengan $y_i = 1$ untuk gambar positif.

Contoh : *l* = 5, m= 10

$\boldsymbol{w_{1,0} = \frac{1}{2\times10} = 0.05,\ w_{1,y_i} = \frac{1}{2\times5} = 0.1}$ **(3.5)**

Untuk t = 1, 2, 3…T, dimana t adalah iterasi ke t untuk gambar positif.

Untuk j = 1, 2, 3…J, dimana j adalah iterasi ke j untuk gambar positif.

$h_t\ (x)$ merupakan nilai fitur gambar positif.

$h_j\ (x)$ merupakan nilai fitur gambar positif.

Untuk mendapatkan nilai error rate setiap weak classifier. Maka untuk setiap fitur dilakukan proses sebagai berikut:

Untuk gambar positif : $\boldsymbol{\in_i = \left(\sum_t^t w_{t,i}\right)| h_t(x) - y_i |}$ **(3.6)**

Untuk gambar negatif : $\boldsymbol{\in_i = \left(\sum_J^J w_{j,i}\right)| h_j(x) - y_i |}$ **(3.7)**

Diberikan citra positif (8x8) piksel yang memiliki nilai-nilai sebagai berikut:



Gambar 3.15 Contoh citra Positif 1-5

1. Citra positif ke-1

21	20	19	18	18	18	21	22
20	18	21	21	24	25	26	27
23	23	24	25	27	28	29	27
23	22	26	27	28	29	30	28
23	23	26	27	29	30	31	30
23	24	25	25	27	31	32	29
23	24	24	26	28	29	30	28
23	25	24	26	28	28	30	31

Gambar 3.16 Citra positif 1

Nilai positif pada gambar tersebut akan dihitung nilai fiturnya.

Nilai fitur = | (total piksel hitam) – (total piksel biru) |

$h_h(x)$ = | (26+21-23-18)-(31+18-26-22) |

= 5

Maka nilai error rate yang didapatkan adalah sebagai berikut :

h_t= (0,1) | 5 - 1| = 0,4|

2. Citra positif ke-2

95	88	84	76	76	81	91	103
90	80	70	67	63	85	26	27
71	76	24	59	27	28	29	27
89	70	58	60	28	78	60	70
92	80	75	70	61	55	51	50
93	80	79	78	74	71	71	75
96	93	94	86	81	83	82	88
100	103	95	88	92	92	91	94

Gambar 3.17 Citra positif 2

Nilai positif pada gambar tersebut akan dihitung nilai fiturnya.

Nilai fitur = | (total piksel hitam) – (total piksel biru) |

$h_h(x)$ = (88+95-100-76)-(94+76-88-103)

= 28

Maka nilai error rate yang didapatkan adalah sebagai berikut :

h_t=(0,1) | 28 - 1| =2,7

3. Citra positif ke-3

69	88	84	45	76	81	91	32
90	80	70	67	63	85	26	27
71	76	24	59	27	28	29	27
89	70	58	60	28	78	60	70
92	80	75	70	61	55	51	50
93	80	79	78	74	71	71	75
96	93	94	86	81	83	82	88
98	103	95	105	92	92	91	56

Gambar 3.18 Citra Positif 3

Nilai positif pada gambar tersebut akan dihitung nilai fiturnya.

Nilai fitur = | (total piksel hitam) – (total piksel biru) |

$h_h(x)$= (105+69-98-45)-(56+45-105-32) = 67

Maka nilai error rate yang didapatkan adalah sebagai berikut :

h_t=(0,1)|67-1|=6,6

4. Citra positif ke-4

72	88	84	57	76	81	91	58
90	80	70	67	63	85	26	27
71	76	24	59	27	28	29	27
89	70	58	60	28	78	60	70
92	80	75	70	61	55	51	50
93	80	79	78	74	71	71	75
96	93	94	86	81	83	82	88
58	103	95	35	92	92	91	35

Gambar 3.19 Citra Positif 4

Nilai positif pada gambar tersebut akan dihitung nilai fiturnya.

Nilai fitur = | (total piksel hitam) – (total piksel biru) |

$h_h(x)$= | (35+72-58-57)-(35+57-35-58) ||

= 7

Maka nilai error rate yang didapatkan adalah sebagai berikut :

h_t =(0,1) | 7- 1| = 0,6

5. Citra positif ke-5

20	20	19	24	18	18	21	25
20	18	21	21	24	25	26	27
23	23	24	25	27	28	29	27
23	22	26	27	28	29	30	28
23	23	26	27	29	30	31	30
23	24	25	25	27	31	32	29
23	24	24	26	28	29	30	28
18	25	24	20	28	28	30	23

Gambar 3.20 Citra Positif 5

Nilai positif pada gambar tersebut akan dihitung nilai fiturnya.

Nilai fitur = | (total piksel hitam) – (total piksel biru) |

$h_h(x)$= (20+20-18-24)-(23+24-20-25)= 4

Maka h_t=(0,1) | 4 - 1| =0,3

Jadi:

Error rate masing-masing citra adalah 0,4; 2,7; 6,6; dan 0,3. Sehingga untuk memudahkan pemilihan fitur yang terbaik digunakan klasifikasi data training menggunakan error rate terkecil yang telah didapatkan yaitu 0,3 dimana

$\beta_t = \frac{\epsilon_t}{1-\epsilon_t} = \frac{0,3}{1-0,3} = 0,428$

Update bobot : $w_{t+1,i} = w_{t,i}.\beta_t$

Maka bobot setelah iterasi 1 : $w_{2,1}$= 0,1 x 0, 428 = 0, 0428

$$\in_t = (0{,}1 + 0{,}0428)|\,4 - 1| = 0{,}4284$$

$\beta_t = \frac{0{,}4284}{1-0{,}4284}$ = 0, 7494

Maka bobot setelah iterasi 2 : $w_{3,1}$= 0,0428 x 0,7494 = 0, 032

$$\in_t = (0{,}1 + 0{,}0428 + 0{,}032)|\,4 - 1| = 0{,}5244$$

$\beta_t = \frac{0{,}5244}{1-0{,}5244}$ = 1, 1026

Maka bobot setelah iterasi 3 : $w_{4,1}$= 0,032 x 1,1026 = 0, 035

$$\in_t = (0{,}1 + 0{,}0428 + 0{,}032 + 0{,}035)|\,4 - 1| = 0{,}6294$$

$\beta_t = \frac{0{,}6294}{1-0{,}6294}$ = 1,67

Maka bobot setelah iterasi 4 : $w_{5,1}$= 0,035 x 1,67 = 0,058

$$\in_t = (0{,}1 + 0{,}0428 + 0{,}032 + 0{,}035 + 0{,}058)|\,4 - 1| = 0{,}8033$$

$\beta_t = \frac{0{,}803}{1-0{,}803}$ = 4,076

Maka bobot setelah iterasi 5 : $w_{6,1}$= 0,058 x 4,076 = 0,236

$$\in_t = (0{,}1 + 0{,}0428 + 0{,}032 + 0{,}035 + 0{,}058 + 0{,}236)|\,4 - 1| = 1{,}511$$

$\beta_t = \frac{1{,}511}{1-1{,}511}$ = -2,957

Maka bobot setelah iterasi 6 : $w_{7,1}$= 0,236 x (-2,957) = -0,697

Diberikan citra Negatif (8x8) piksel yang memiliki nilai-nilai sebagai berikut:

$$\mathbf{H(x)} = \begin{cases} \mathbf{1} \\ \mathbf{0} \end{cases} \sum_{j=1}^{J} \propto_j \boldsymbol{h_j(x)} \geq \frac{1}{2} \sum_{t}^{J} \propto_j \quad \textbf{(3.8)}$$

Otherwise

Dimana :

$$\propto_j = \boldsymbol{log} \frac{1}{\beta_j} \propto_j = \boldsymbol{log} \frac{1}{\beta_t} \qquad \textbf{(3.9)}$$

Jika posisi H(x) = Ketentuan 0 maka citra tersebut merupakn bukan objek H(x) = Strong Classifier atau kelasifikasi yang menyatakan objek atau bukan αj = Tingkat pembelajaran gambar positif. αt = Tingkat pembelajaran gambar negatif.

βj=Nilai bobot setelah error rate pada gambar negatif

βt= Nilai bobot setelah error rate pada gambar positif

Hj= weak atau basic classifiers (awal dari klasifikasi) gambar negatif.

Ht = weak atau basic classifiers (awal dari klasifikasi) gambar positif.

Sehingga Diberikan citra negatif (8x8) piksel yang memiliki nilai-nilai sebagai berikut:



Gambar 3.21 Contoh citra Negatif 1-2

1. Citra negatif ke-1

119	117	116	115	114	112	110	100
118	117	116	115	113	111	109	104
116	114	112	111	103	110	116	120
114	116	117	124	111	113	119	113
102	103	104	105	134	104	114	114
150	144	111	102	123	103	117	116
160	133	122	109	142	102	112	115
104	104	102	101	99	95	96	93

Gambar 3.22 Citra Negatif 1

Dengan nilai fitur :

$h_j(x)$= (120 + 77 – 162 -25)-(100 + 25 - 120- 52)= 52

Maka $\in_j$=(0,05) | 57 – 0 | =2,8

$\beta_t = \frac{\in j}{1-\in j}$ maka $\beta_t = \frac{2,85}{1-2,85}$= -1,54

Perbaharui bobot : $w_{t+1,i} = w_{t,i}.\ \beta_j$

Maka bobot setelah iterasi 1 : $w_{2,1}$ = 0,05× $(-1,54) = -0,077$

jika bobot setelah iterasi ke n jumlahnya < 0 maka iterasi berhenti. βj yang digunakan untuk mencari H(x) mengunakan $\in_t$ pada citra negatif yang nilai βj<0

$$H_{(x)} = \sum_{j=1}^{J} a_j\, h_j(x) \geq \frac{1}{2}\sum_{j}^{T} a_t = untuk\ nilai\ 1\ bernilai\ benar$$

Untuk salah bernilai = 0

Maka: H(x)= $\log \frac{1}{2,85} \times 57 \geq \frac{1}{2}\log \frac{1}{0,428}$

= -25,99 ≥ 0,184

Karena -25,99 ≥ 0,184 itu bernilai salah jadi citra bukan merupakan objek

*2. C*itra negatif ke-2

119	117	116	115	114	112	110	100
118	117	116	115	113	111	109	104
116	114	112	111	103	110	116	120
114	116	117	124	111	113	119	113
102	103	104	105	134	104	114	114
150	144	111	102	123	103	117	116
160	133	122	109	142	102	112	115
104	104	102	101	99	95	96	93

Gambar 3.23 Citra Negatif 2

Dengan nilai fitur :

$h_j(x)$ = (101+119-104-115)-(93+115-101-100) = 6

Maka $\in_j$=(0,05) | 6 – 0 | =0,3

$\beta_j = \frac{\in j}{1-\in j}$ maka $\beta_j = \frac{0,3}{1-0,3}$= 0,428

Perbaharui bobot : $w_{t+1,i} = w_{t,i}.\ \beta_j$

Maka bobot setelah iterasi 1 : $w_{2,1}$ = 0,05× $(0,428) = 0,021$

$\in_t$= (0,05 + 0,021) | 6 – 0 | =0,426 ; $\beta_t = \frac{0,426}{1-0,42}$= 0,742

Maka bobot setelah iterasi 2 : $w_{3,1}$ = 0,021× $0,742 = 0,0155$

$\in_t$= (0,05 + 0,021+0,0155) | 6 – 0 | =0,519 ; $\beta_t = \frac{0,519}{1-0,519}$= 1,079

Maka bobot setelah iterasi 3 : $w_{4,1}$ = 0,0155× $1,079 = 0,0167$

$\in_t$= (0,05 + 0,021 + 0,0155 + 0,0167) | 6 – 0 | =0,619 ; $\beta_t = \frac{0,619}{1-0,619}$ = 1,624

Maka bobot setelah iterasi 4 : $w_{5,1}$ = 0,0167× $1,624 = 0,0271$

$\in_t$= (0,05 + 0,021 + 0,0155 + 0,0167 + 0,0271) | 6 – 0 | =0,7818 ; $\beta_t = \frac{0,7818}{1-0,7818}$ = 3,582

Maka bobot setelah iterasi 5 : $w_{6,1}$ = 0,058× $4,076 = 0,236$

$\in_t$= (0,05 + 0,021 + 0,0155 + 0,0167 + 0,0271+ 0,0970) | 6 – 0 | =1,364 ; $\beta_t = \frac{1,364}{1-1,364}$ = -3,747

Maka bobot setelah iterasi 6 : $w_{7,1}$ = 0,0971× $(-3,747) = -0,363$

Maka :

$H_{(x)} = \sum_{j=1}^{J} a_j\, h_j(x) \geq \frac{1}{2}\sum_t^T a_t$.....(3.10)

$$H(x)=(\log \frac{1}{0,428} + \log \frac{1}{0,742} + \log \frac{1}{1,079} + \log \frac{1}{1,624} + \log \frac{1}{3,582}) \times 6 \geq$$

$$\frac{1}{2}(\log \frac{1}{0,428} + \log \frac{1}{0,749} + \log \frac{1}{1,1026} + \log \frac{1}{1,67} + \log \frac{1}{4,076})$$

$$= 0,368 + 0,129 – 0,033 – 0,210 – 0,554$$

$$\geq \frac{1}{2}(0,368 + 0,125 – 0,042 – 0,223 – 0,610)$$

$$= -0,3 \geq -0,191$$

Maka citra tersebut bukan berupa gambar positif.

Jadi semakain banyak interasi yang bisa dilakukan oleh sebuah gambar positif dan negatif maka semakain besak kemungkinan untuk objek ditemukan. Dari analisa diatas dapat diambil kesimpulan, bahwa tiga faktor yang mempegaruhi kecepatan dan tingkat akurasi dalam pendeteksian, faktor - faktor tersebut antara lain:

1. jumlah dari tahapan classifier.
2. Jumlah semakin banyak gambar positif dimasukan semakin akurat pendeteksianya.
3. Jumlah dari data latih yang dimasukkan

3.2 Analisis Kebutuhan Non-Fungsional

Adapun kebutuhan non-fungsional untuk menjalankan aplikasi ini meliputi kebutuhan perangkat keras, kebutuhan perangkat lunak dan pengguna sistem yang akan memakai aplikasi. Analisis kebutuhan non-fungsional bertujuan agar aplikasi yang dibangun dapat digunakan sesuai dengan kebutuhan pengguna aplikasi dalam mencari informasi yang dibutuhkan.

3.2.1 Analisis Kebutuhan Perangkat keras

Spesifikasi kebutuhan perangkat keras komputer yang dibutuhkan untuk membangun aplikasi dapat dilihat pada tabel 3.3

Tabel 3.1 Spesifikasi Kebutuhan Perangkat Keras Komputer

No	Perangkat Keras	Spesifikasi
1	Processor	Kecepatan minimun 2.2 GHz
2	VGA	500 MB
3	Memori	RAM 2 GB DDR3
4	Keyboard	Standart Port USB
5	Mouse	Optical Mouse USB
6	Monitor	LED
7	Webcam	Standart Port USB
8	Harddisk	320

3.2.2 Analisis Kebutuhan Perangkat Lunak

Spesifikasi perangkat lunak komputer yang dibutuhkan untuk membangun aplikasi dapat dilihat pada tabel 3.4

Tabel 3.2 Spesifikasi Kebutuhan Perangkat Lunak Komputer

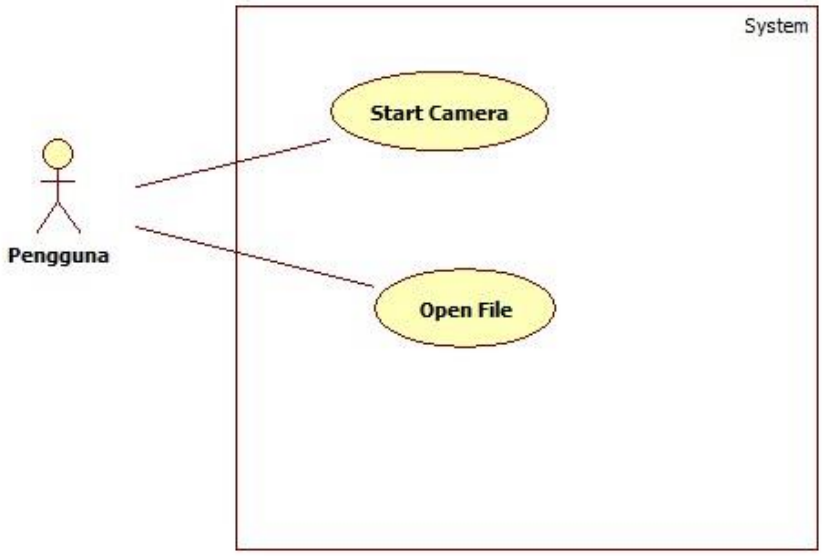
No	Perangkat Lunak	Spesifikasi
1	Sistem Operasi	Microsoft Windows 7 atau 8 32bit
2	Program	a. Mirosoft Visual Studio 2010 b. Microsoft Visio 2010 c. Library OpenCV

3.3 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional bertujuan untuk menganalisis proses yang akan diterapkan dalam sistem dan menjelaskan kebutuhan yang diprlukan agar aplikasi dapat berjalan dengan baik. Pemodelan yang akan digunakan akan dibuat mengunakan Microsoft Visio 2010.

3.3.1 Use case

Use case diagram merupakan pemodelan yang menunjukan interaksi antara sistem dan lingkungannya. *Use case diagram* memberikan gambaran yang cukup sederhana dari interaksi-interaksi yang terlibat. Berikut adalah perancangan proses-proses pada sistem pengenalan isyarat jari pada aplikasi pemutar musik yang digambarkan dengan *use case diagram* :



Gambar 3.24 Gambar Use Case

Use case terdiri dari tiga bagian yaitu definisi *actor*, definisi *use case*, dan skenario *use case*.

3.3.1.1 Identifikasi Actor

Actor yaitu pihak yang mengakses *use case* dengan berperan sebagai pengguna yang akan menggunakan sistem.

Tabel 3.3 Aktor Use Case

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

3.3.1.2 Identifikasi Use Case

Berikut tabel Identifikasi *Use Case* yang terdapat pada aplikasi:

Tabel 3.4 Use Case Aktor

Use Case Aplikasi User		
No	**Use Case**	**Deskripsi**
UC-01	Start Camera	Fungsional untuk mengaktifkan kamera
UC-02	Open File	Fungsional untuk memungkinkan pengguna memilih lagu yang sudah tersimpan di file directory

3.3.2 Skenario Use Case

Skenario *use case* mendeskripsikan urutan langkah-langkah dalam proses bisnis, baik dilakukan aktor terhadap sistem maupun yang dilakukan oleh sistem terhadap aktor.

3.3.2.1 Skenario Use start camera

Skenario *use case start camera* gambar dapat dilihat pada tabel 3.5.

Tabel 3.5 Skenario Use Case Start Camera

Identifikasi	
Nama Use Case User	Start Camera
Aktor	*User*
Tujuan	Digunakan untuk menyiapkan kamera webcam
Keadaan Awal	Sistem menampilkan tampilan utama
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
	1. Menampilkan Menu Utama
2. Pilih button start pada	

aplikasi	
	3. Menyimpan gambar

3.3.2.2 Skenario Use Case Open file musik

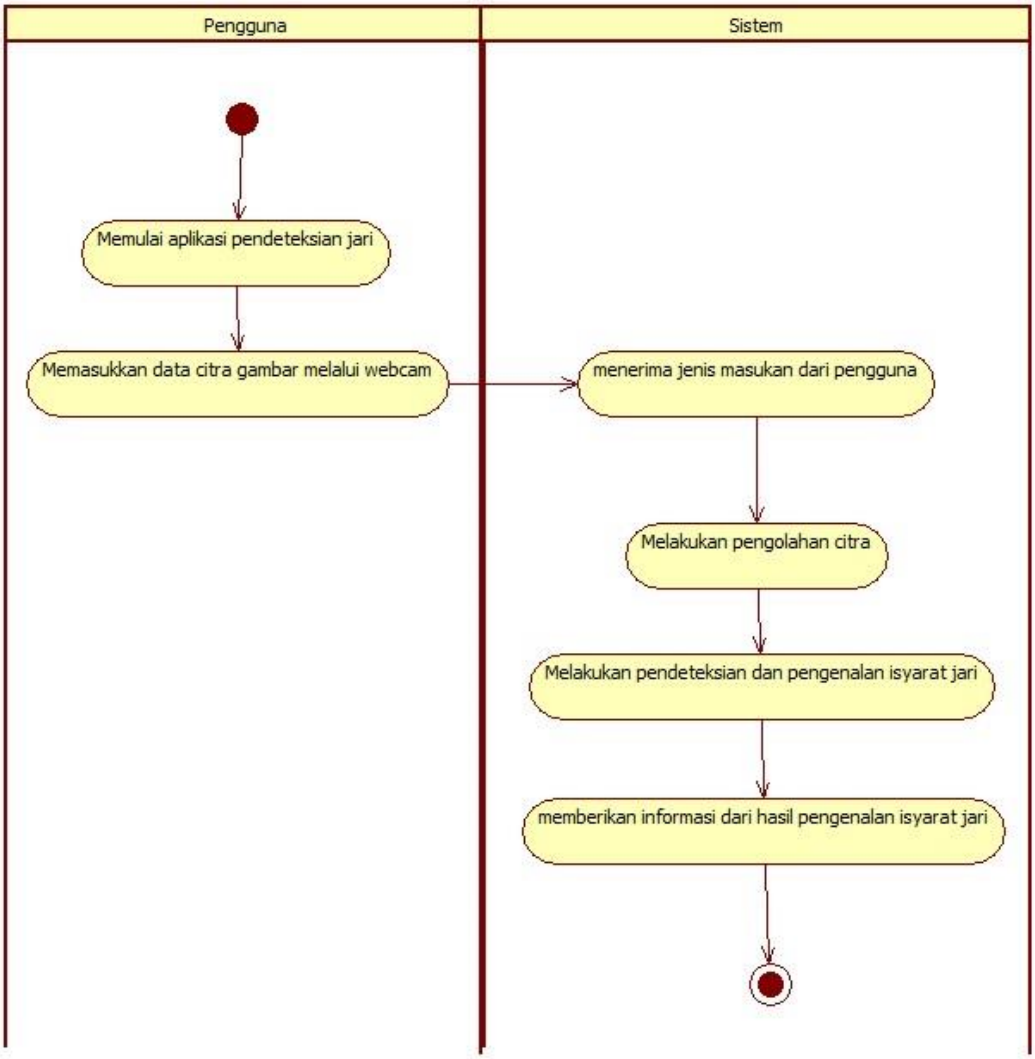
Skenario *use case open file musik* gambar dapat dilihat pada tabel 3.6.

Tabel 3.6 Skenario Use Case open file musik

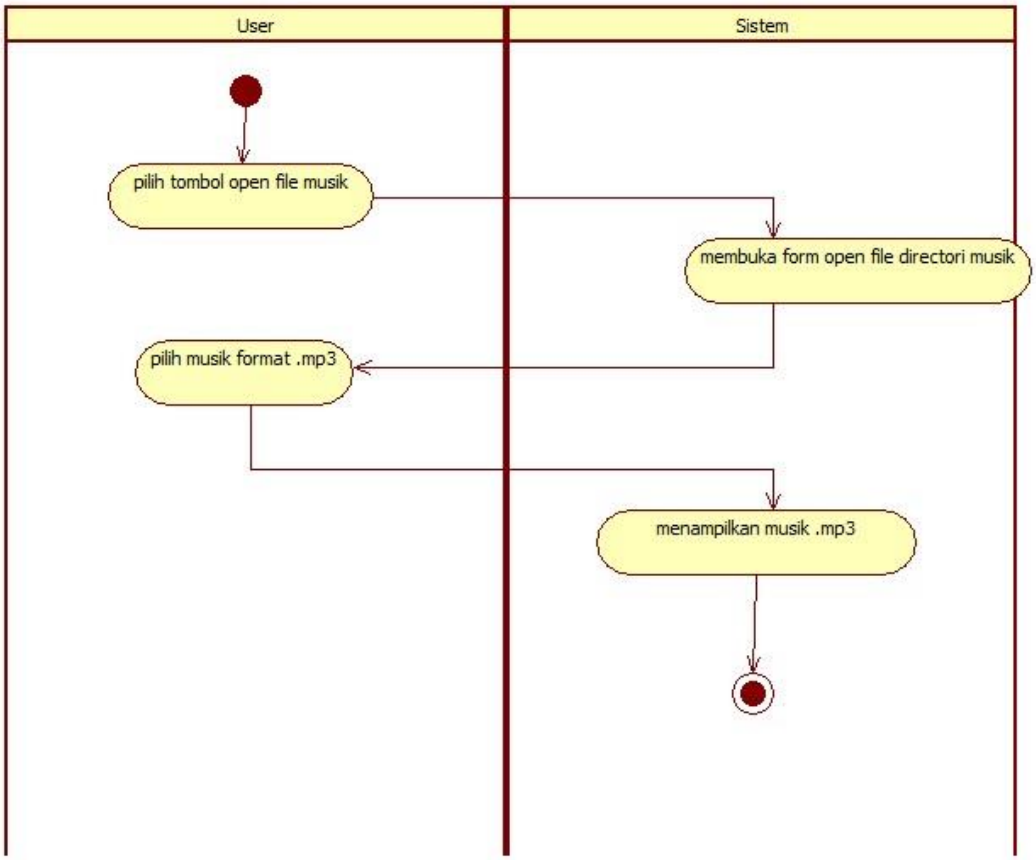
Identifikasi	
Nama Use Case User	Open File musik
Aktor	*User*
Tujuan	Proses untuk memilih lagu di directori
Keadaan Awal	Sistem menampilkan directori file
Skenario Utama	
Aksi Aktor	**Reaksi Sistem**
	1. Menampilkan directori file

3.3.3 Activity Diagram

Activity diagram digunakan untuk menggambarkan alur rangkaian proses yang terjadi di dalam sistem aplikasi serta menjelaskan bagaimana sebuah *use case* terjadi atau dipicu dari *use case* yang lain.



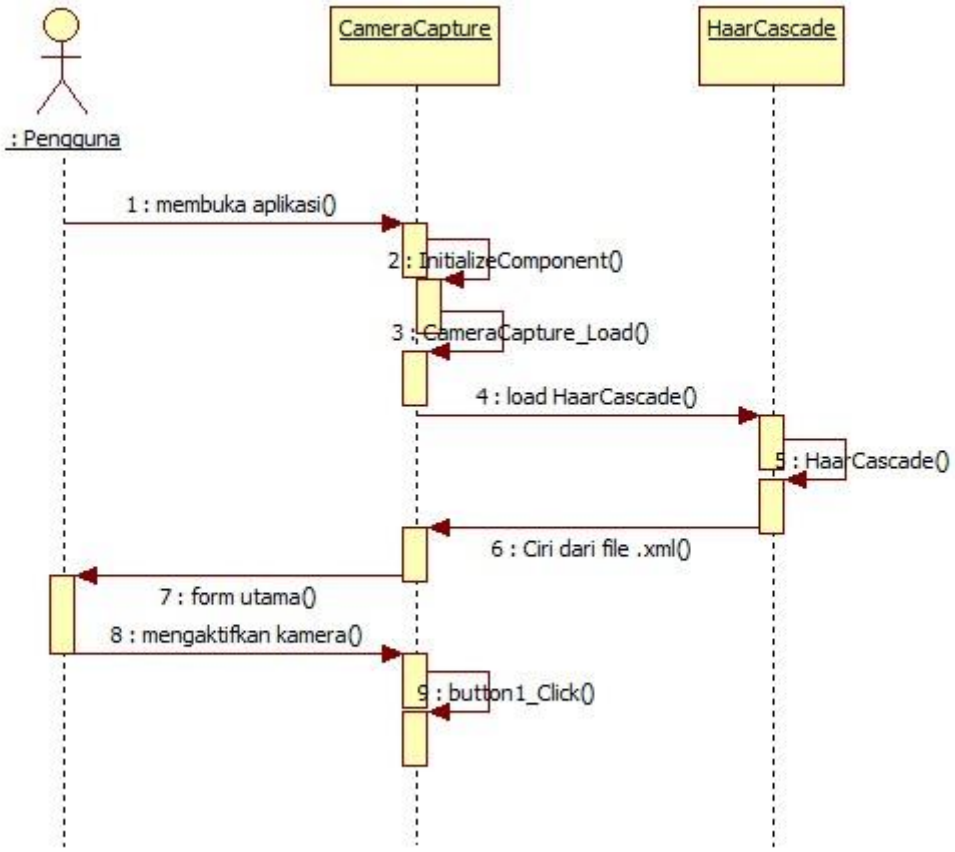
Gambar 3.25 *Activity diagram* pengenalan isyarat jari



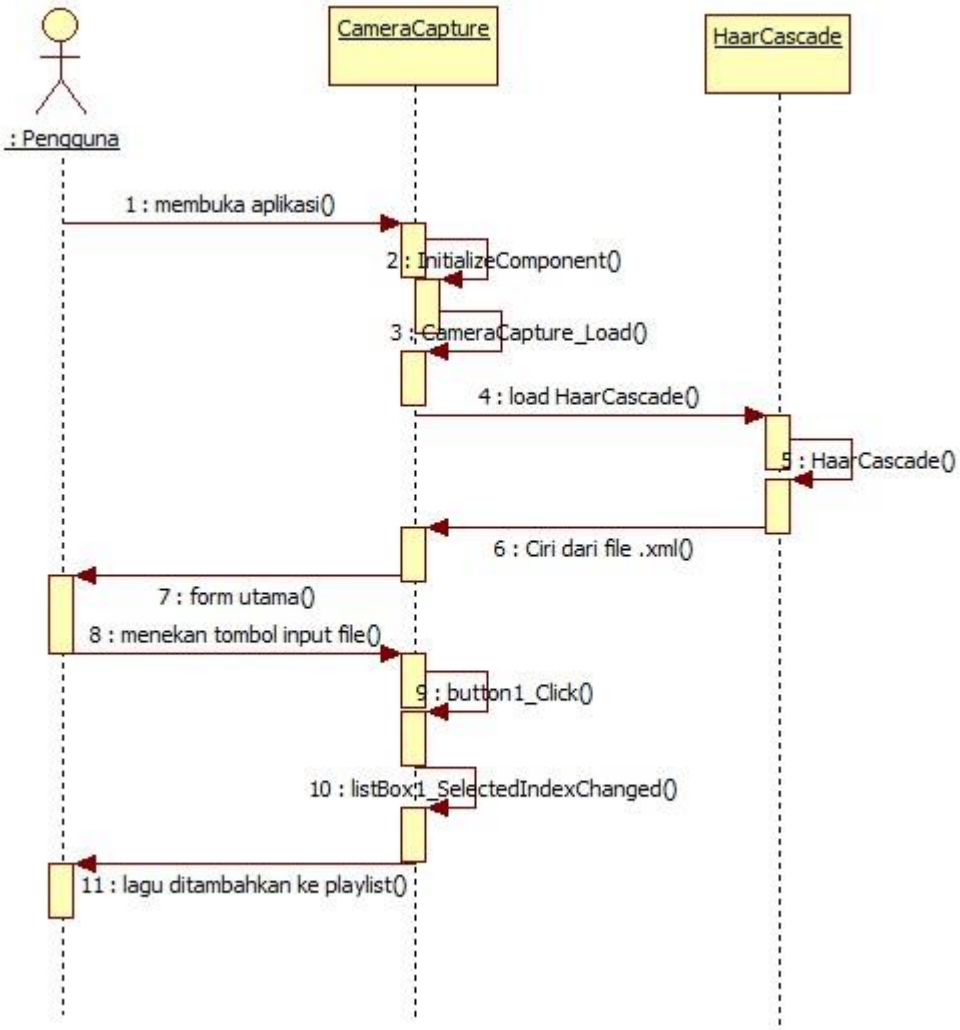
Gambar 3.26 Activity Diagram Open file musik

3.3.4 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 :



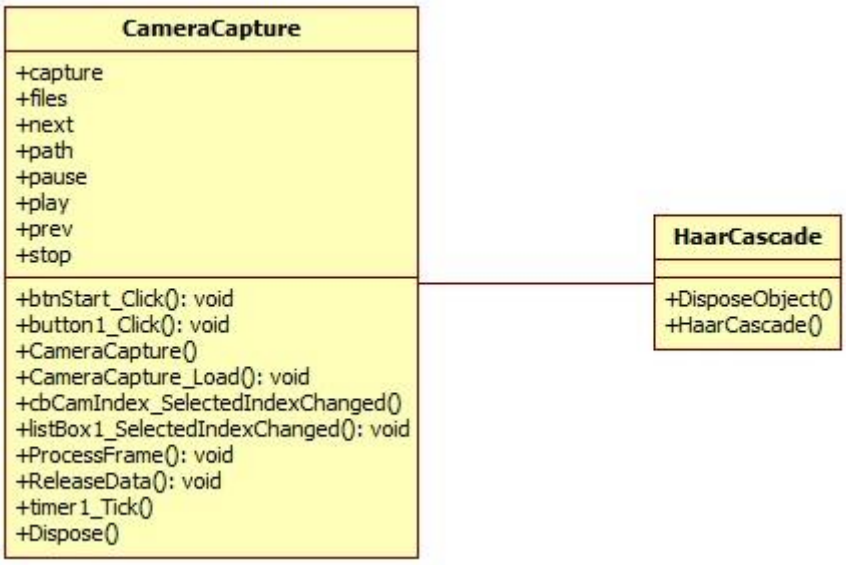
Gambar 3.27 Sequence Diagram Start Camera



Gambar 3.28 Sequence Diagram Alir Open file musik

3.3.5 *Class Diagram*

Class Diagram menggambarkan struktur dan hubungan antar objek- objek yang ada pada sistem. Struktur itu meliputi atribut-atribut dan metode-metode yang ada pada masing-masing *class*. Hubungannya dari masing-masing class yang ada tersebut digambarkan seperti pada gambar dibawah.



Gambar 3.29 Class Diagram

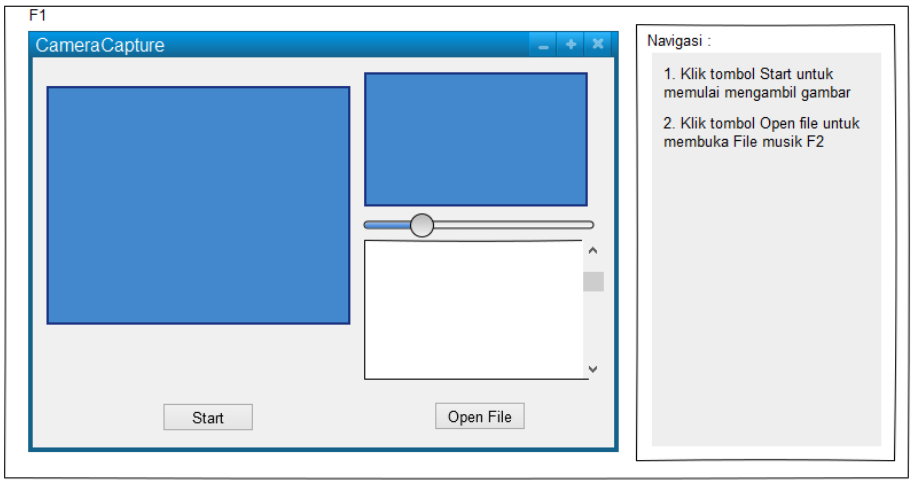
3.4 Perancangan Sistem

Perancangan sistem merupakan penggambaran, perencanaan, dan pembuatan sketsa atau pengaturan dari beberapa elemen yang terpisah ke dalam suatu kesatuan yang utuh. 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.

3.4.1 Perancangan Antarmuka Sistem

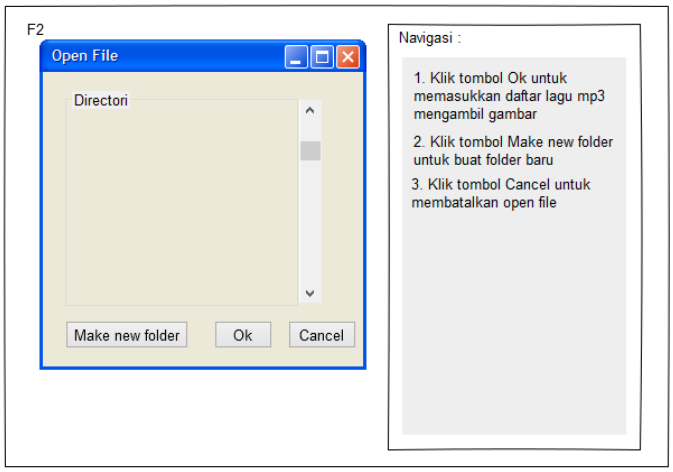
Tahap perancangan antarmuka bertujuan untuk memberikan gambaran tentang aplikasi yang akan dibangun, sehingga akan mempermudah dalam mengimplementasikan aplikasi serta akan memudahkan pembuatan aplikasi. Berikut ini gambaran perancangan antarmuka yang ada dari aplikasi yang akan dibangun yaitu:

3.4.1.1 Perancangan Antar Muka Tampilan Utama



Gambar 3.30 Perncangan Antar Muka Tampilan Utama

3.4.1.2 Perancangan Antar Muka Open File



Gambar 3.31 Perancangan Antar Muka *Open File*

3.5 Perancangan Jaringan Semantik

Jaringan semantik merupakan gambaran grafis yang menunjukkan antar objek (lingkaran) dan garis yang menggambarkan formasi antarobjek tersebut. Adapun jaringan semantik untuk aplikasi pengecekan umur kulit dapat dilihat pada gambar dibawah:



Gambar 3.32 Jaringan Semantik

Keterangan:

F1 : Tampilan Utama

F2 : Tampilan Open File