BAB 3
ANALISIS DAN PERANCANGAN SISTEM

3.1 Analisis Sistem

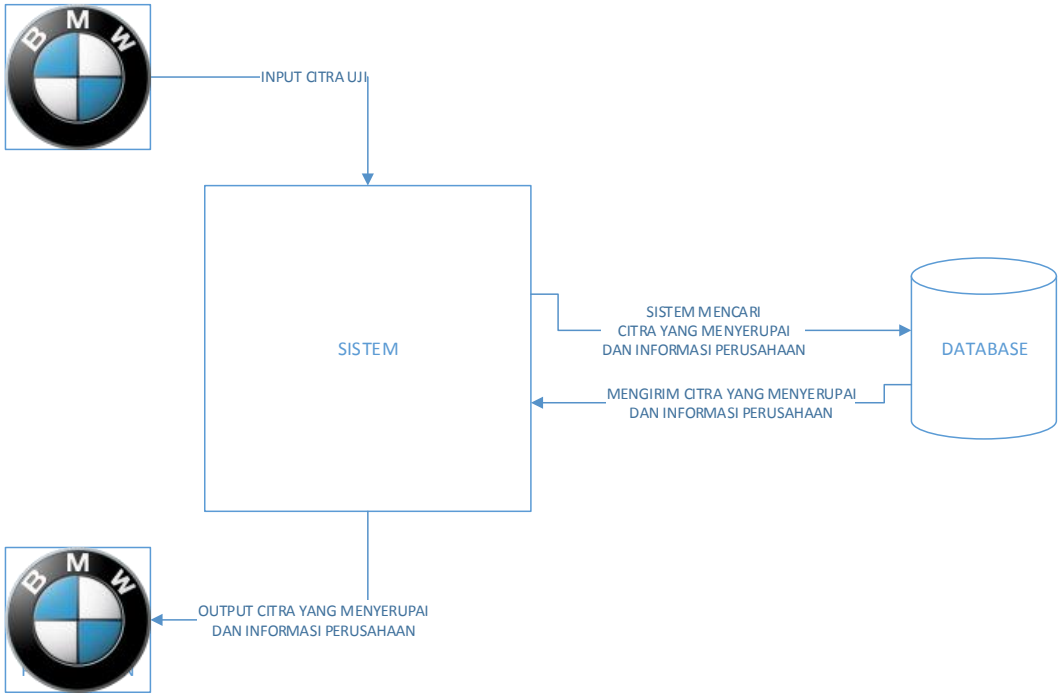
Analisis sistem merupakan suatu proses penguraian konsep kedalam bagian-bagian yang lebih sederhana dengan maksud untuk mengetahui, mengidentifikasi, dan mengevaluasi masalah-masalah yang ada, hambatan-hambatan, kemungkinan-kemungkinan dari setiap solusi dan kebutuhan dari sistem atau aplikasi yang akan dibuat. Analisis sistem terdiri dari beberapa tahapan, diantaranya analisis masalah, analisis metode, analisis kebutuhan non-fungsional, dan analisis kebutuhan fungsional.

3.1.1 Analisis Masalah

Analisis masalah dilakukan untuk mengetahui masalah-masalah apa saja yang terjadi sehingga dilakukannya penelitian kali ini. Masalah yang terjadi saat ini ketika melakukan aktifitas seperti membeli sesuatu atau menggunakan jasa sebuah perusahaan adalah seringnya masyarakat kebingungan atau bahkan tidak tahu tentang informasi kebanyakan perusahaan yang sudah ada, bahkan jika perusahaan tersebut sebenarnya telah lama berdiri.

3.1.1.1 Arsitektur Sistem

Arsitektur sistem adalah gambaran proses dari sistem yang akan dibangun dalam penelitian ini. Adapun arsitektur sistemnya dapat dilihat pada gambar 3.1.



Gambar 3.1 Arsitektur Sistem

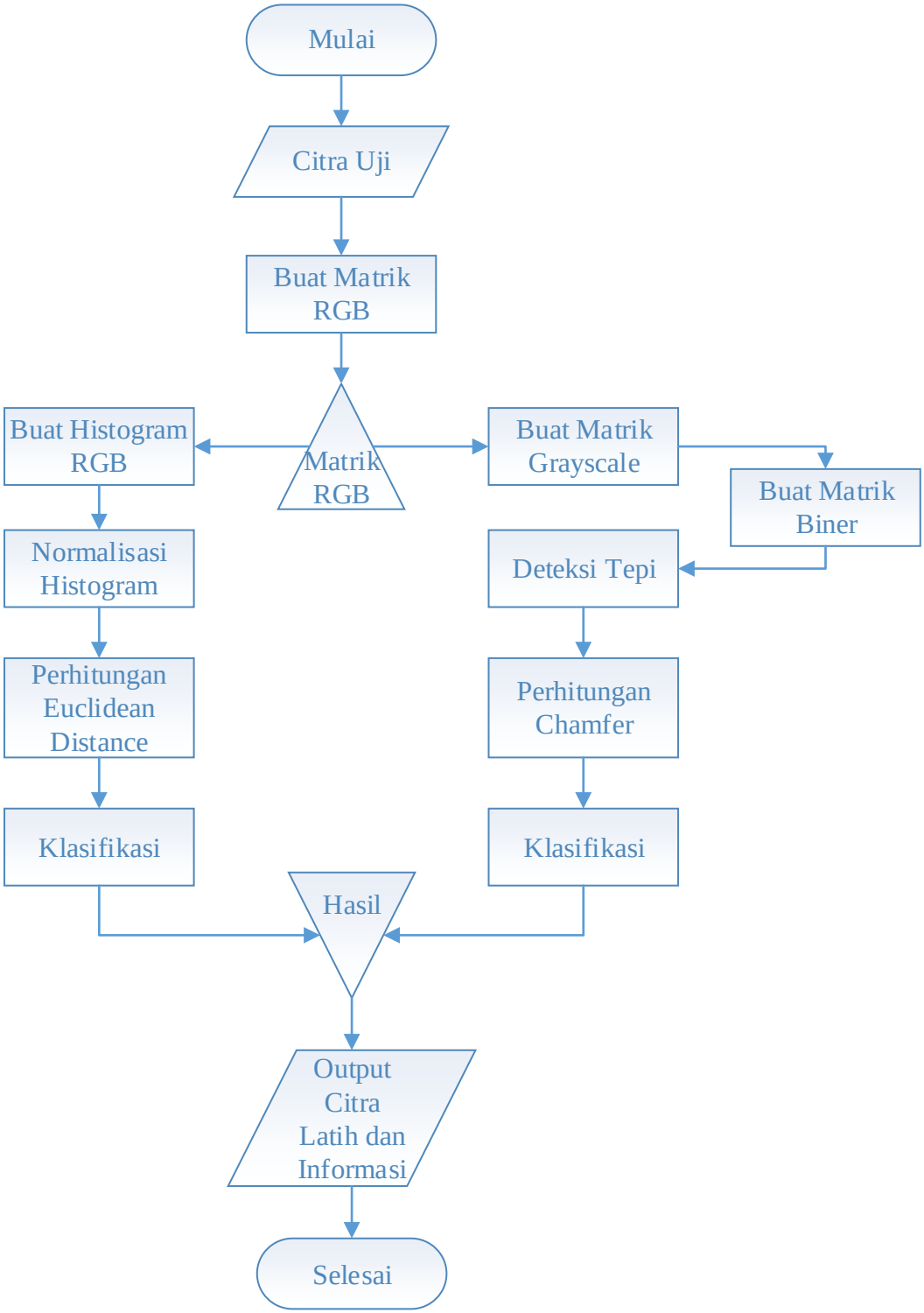
1. Pada tahap awal, citra yang akan dicari informasinya di-*input*-kan kedalam sistem.
2. Selanjutnya sistem akan melakukan proses dan mencari citra yang menyerupai yang ada didalam *database* kemudian mengambil informasi yang menyangkut dengan citra tersebut.
3. Kemudian sistem akan mengambil citra dan informasi yang telah didapatkan.
4. Selanjutnya sistem akan menampilkan *output* citra yang menyerupai dan informasi perusahaan mengenai citra tersebut.

3.1.2 Analisis Metode

Analisis metode digunakan untuk mengetahui alur proses dari sebuah metode yang digunakan dapat diterapkan ke dalam aplikasi yang dibangun dan contoh penerapannya pada beberapa gambar sebagai berikut :

3.1.2.1 Tahapan Metode

Pada bagian ini akan dijelaskan tahapan-tahapan yang akan dilakukan dalam melakukan proses pengidentifikasian citra oleh aplikasi yang dibangun. Tahapan metode aplikasi ini dapat dilihat pada gambar 3.2.



Gambar 3.2 Tahapan Metode

Citra yang akan diuji pada gambar 3.3 :



Gambar 3.3 Citra Uji

Dan gambar 3.4 adalah beberapa citra latihnya :

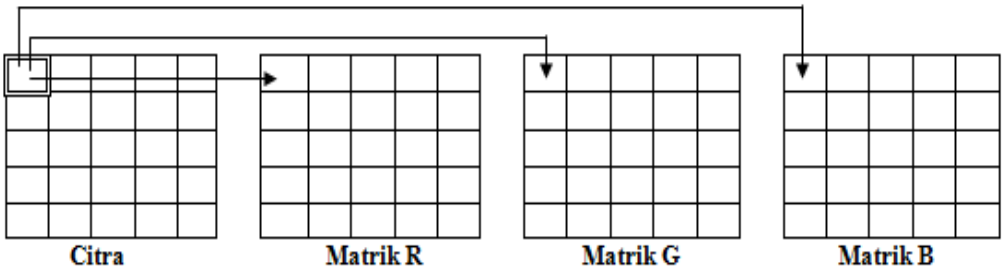


Gambar 3.4 Citra Latih

3.1.2.2 Analisis Histogram Warna RGB (Red, Green, Blue)

Pada tahap ini, suatu citra akan dicari nilai histogram warnanya. Histogram warna adalah representasi dari distribusi warna dari citra digital. Histogram warna merupakan fitur yang paling banyak digunakan untuk merepresentasikan ciri warna suatu citra. Citra pada umumnya dikonversi kedalam suatu ruang warna tertentu, kemudian setiap komponen ruang warna dibuat histogramnya. Didalam histogram ini terdapat *bins* atau intensitas kemunculan nilai warna pada suatu citra.

Pada kasus ini, suatu citra terlebih dahulu dibagi kedalam 3 *channel* matrik, menjadi *channel* R, *channel* G, dan *channel* B, kemudian dari matrik tersebut akan dibuat nilai histogramnya. Contohnya pada gambar 3.5 apabila citra berukuran 5x5.



Gambar 3.5 Matrik Citra Ke Matrik R, G, Dan B

Nilai dari 1 piksel yang ada dicitra akan di konversi ke nilai piksel ke matrik R, matrik G, dan matrik B. Jumlah kolom dan baris harus sama dan apabila pada posisi (0,0) pada image maka pada matrik R, G, dan B pun harus pada posisi (0,0). Setelah suatu citra telah dibagi kedalam masing-masing nilai R, G, dan B, maka akan dibuat histogram dari tiap-tiap *channel* dari matrik citra tersebut.

Penerapan metodenya adalah sebagai berikut:

Pada citra baik yang akan diuji maupun citra latihnya, yang akan diambil adalah posisi tengah dari citra tersebut, maka nilai matrik RGB yang akan didapatkan dari 6x5 pixel dari citra tersebut dapat dilihat pada gambar 3.6, gambar 3.7, gambar 3.8, dan gambar 3.9 :

Matrik RGB citra uji :

98	76	5	3	164	202
16	10	5	5	27	35
0	9	10	11	1	0
31	21	0	0	28	39
175	118	0	0	179	253

Matrik R

124	79	0	0	164	202
16	6	0	0	26	36
0	5	4	5	0	0
34	24	1	0	26	37
178	121	0	0	177	250

Matrik G

151	98	9	7	172	210
14	5	0	0	24	38
0	4	4	5	0	2
39	33	14	12	37	48
183	130	12	13	188	255

Matrik B

Gambar 3.6 Matrik R, G, B Dari Citra Uji

Matrik RGB citra latih 1 :

98	54	0	1	99	191
50	29	10	4	17	97
0	4	4	3	0	2
2	3	9	2	0	0
90	42	4	0	22	74

Matrik R

125	62	0	0	97	194
60	30	11	8	41	99
8	5	5	7	1	4
3	1	7	11	2	2
91	40	2	2	31	82

Matrik G

144	65	5	4	102	203
69	34	15	11	44	98
17	9	9	10	4	3
5	4	10	16	7	4
93	43	5	7	36	84

Matrik B

Gambar 3.7 Matrik R, G, B Dari Citra Latih 1

Matrik RGB citra latih 2 :

15	9	60	241	249	255
8	37	245	255	255	252
5	153	247	255	255	250
77	252	255	255	255	255
204	252	255	255	255	255

Matrik R

80	73	120	255	251	252
74	65	253	254	255	255
71	181	255	254	255	254
117	181	255	251	250	252
244	255	255	251	250	252

Matrik G

180	171	216	255	255	255
186	115	255	255	255	255
183	231	255	255	255	253
178	231	250	255	255	255
255	250	250	255	255	255

Matrik B

Gambar 3.8 Matrik R, G, B Dari Citra Latih 2

Matrik RGB citra latih 3 :

19	19	185	244	255	255
25	14	81	252	255	255
16	21	13	113	244	251
16	18	24	16	72	224
14	8	4	10	13	23

Matrik R

27	35	200	247	253	249
33	30	96	255	250	252
35	36	20	125	253	254
35	33	31	28	81	227
33	46	43	37	32	38

Matrik G

164	130	255	251	255	255
170	125	155	255	255	255
164	165	152	201	255	255
164	162	163	104	122	234
171	169	162	154	152	157

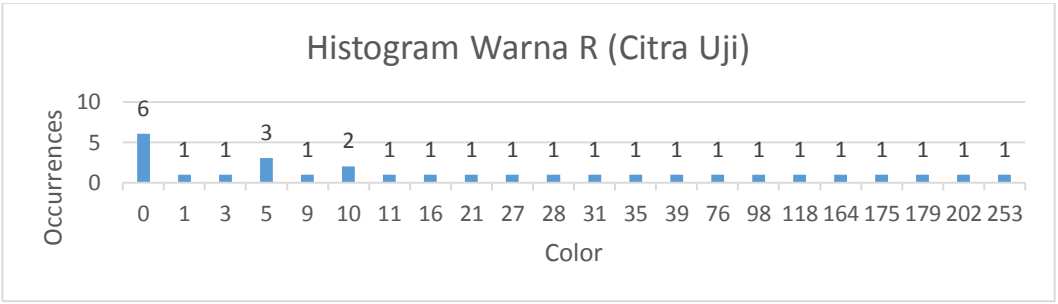
Matrik B

Gambar 3.9 Matrik R, G, B Dari Citra Latih 3

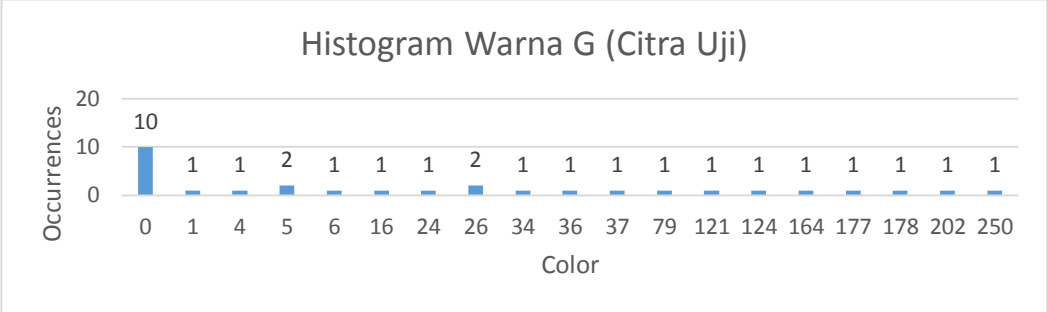
Setelah citra dibagi kedalam 3 buah *channel* matrik, dari tiap-tiap *channel* tersebut akan dibuat histogramnya, yaitu histogram warna R, histogram warna G, dan histogram warna B.

Contoh penerapannya adalah sebagai berikut :

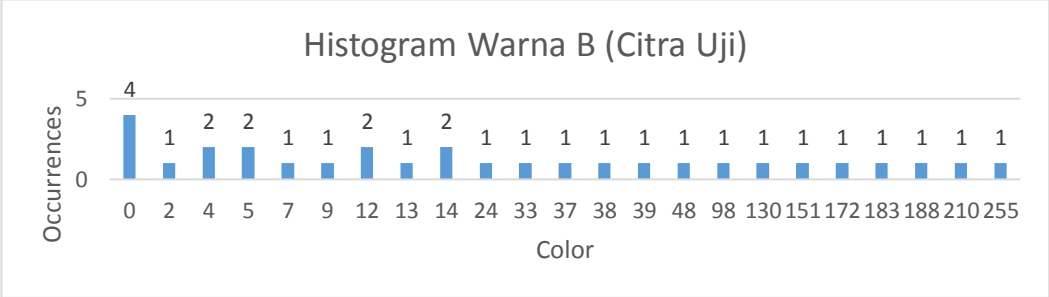
1. Histogram warna RGB dari citra uji dapat dilihat pada gambar 3.10, gambar 3.11, dan gambar 3.12.



Gambar 3.10 Histogram Warna R (Citra Uji)

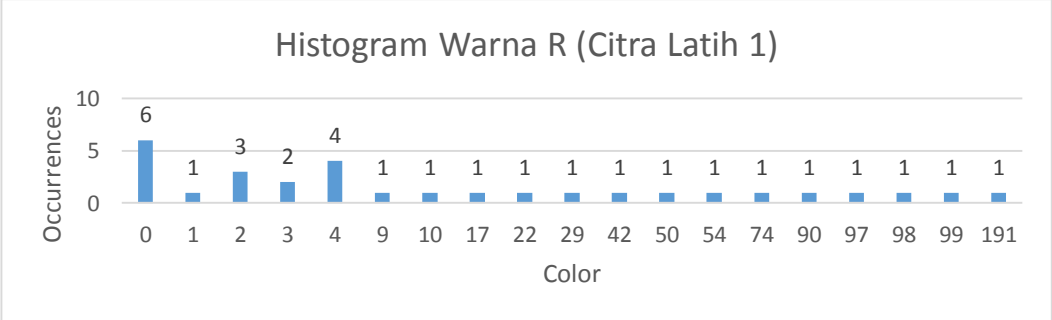


Gambar 3.11 Histogram Warna G (Citra Uji)

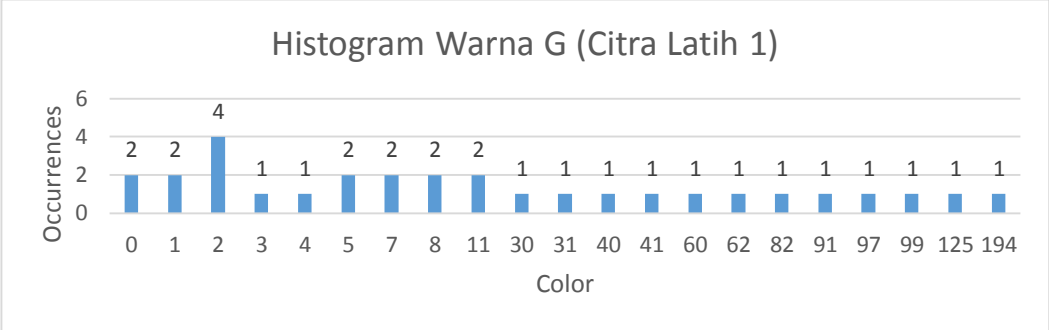


Gambar 3.12 Histogram Warna B (Citra Uji)

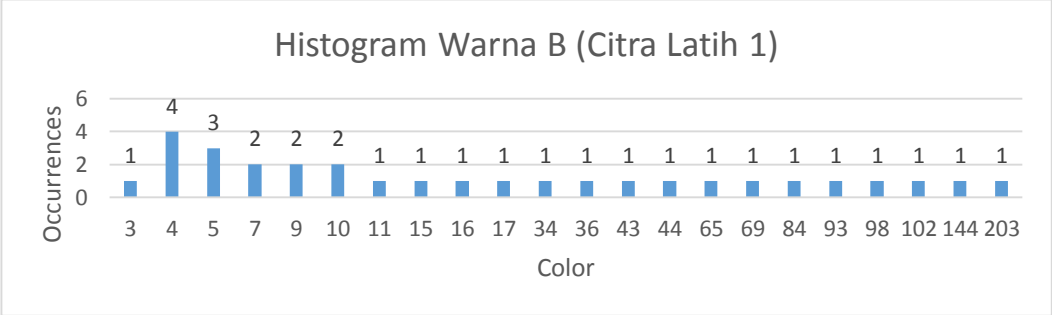
2. Histogram warna RGB dari citra latih 1 dapat dilihat pada gambar 3.13, gambar 3.14, dan gambar 3.15.



Gambar 3.13 Histogram Warna R (Citra Latih 1)

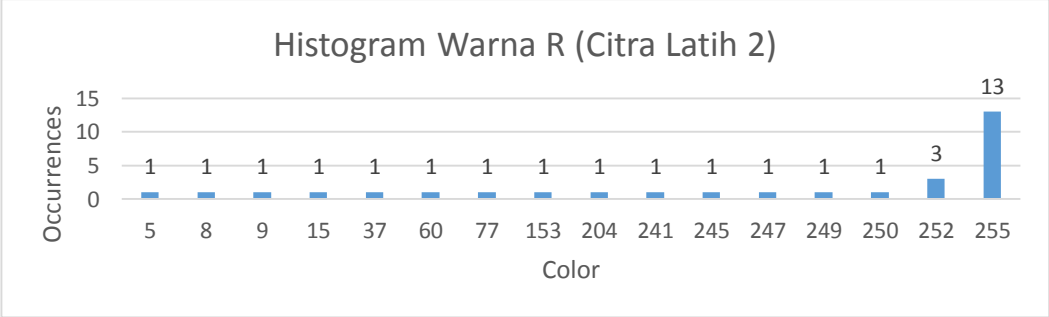


Gambar 3.14 Histogram Warna G (Citra Latih 1)

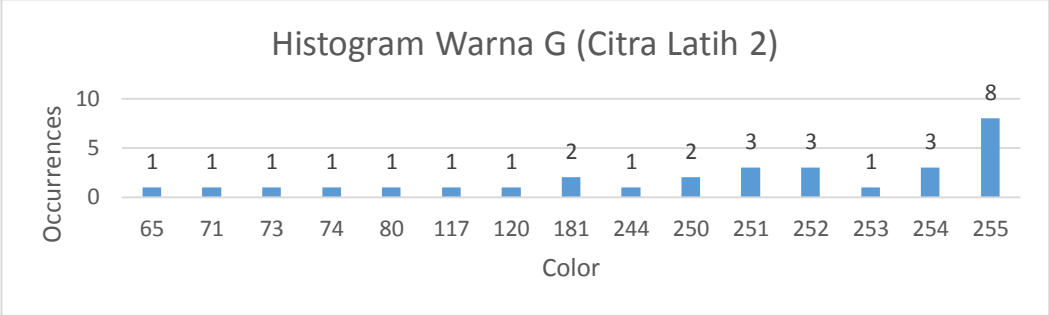


Gambar 3.15 Histogram Warna B (Citra Latih 1)

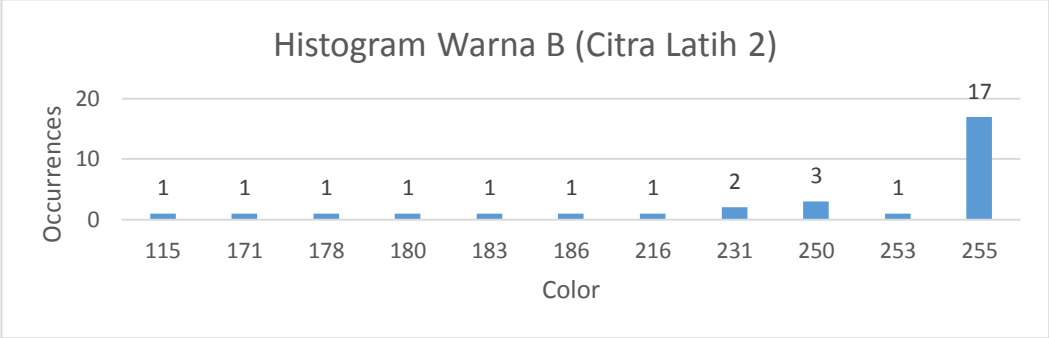
3. Histogram warna RGB dari citra uji dapat dilihat pada gambar 3.16, gambar 3.17, dan gambar 3.18.



Gambar 3.16 Histogram Warna R (Citra Latih 2)

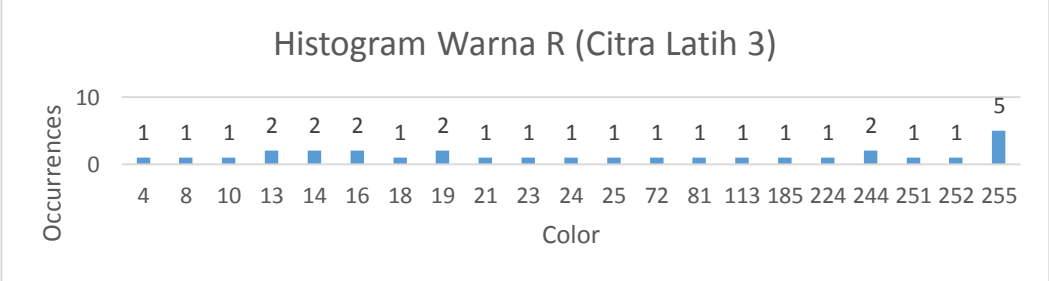


Gambar 3.17 Histogram Warna G (Citra Latih 2)

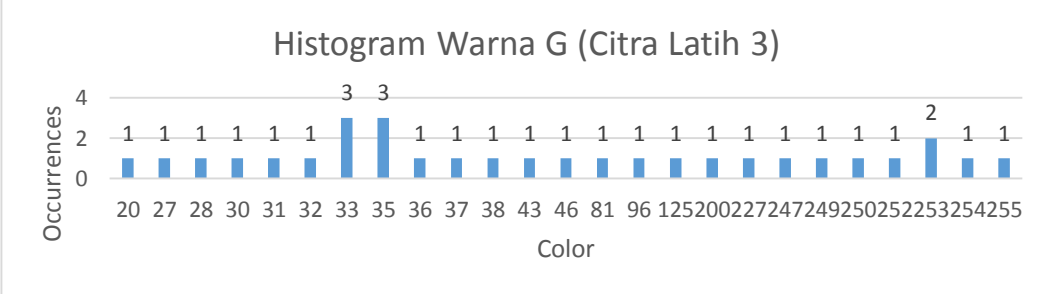


Gambar 3.18 Histogram Warna B (Citra Latih 2)

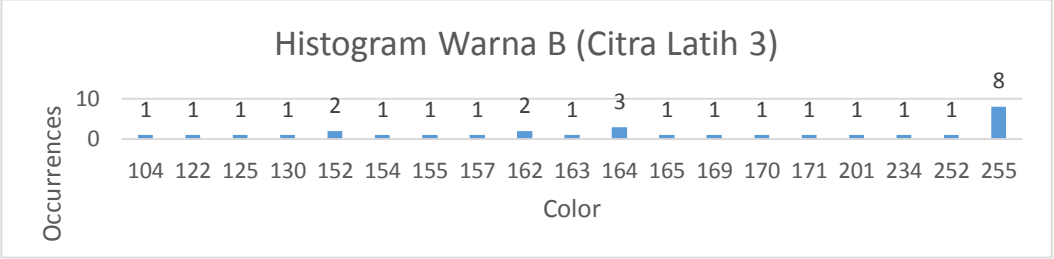
4. Histogram warna RGB dari citra uji dapat dilihat pada gambar 3.19, gambar 3.20, dan gambar 3.21.



Gambar 3.19 Histogram Warna R (Citra Latih 3)



Gambar 3.20 Histogram Warna G (Citra Latih 3)



Gambar 3.21 Histogram Warna B (Citra Latih 3)

3.1.2.3 Analisis Metode Jarak Euclidean (Euclidean *Distance*)

Euclidean distance digunakan untuk menghitung jarak antar histogram warna. Dimana dalam tahapan ini menentukan tingkat kemiripan warna dari citra uji dan citra latih. Semakin kecil jaraknya maka citra tersebut semakin mirip dengan citra uji. Metode ini diimplementasikan dengan formula 2.9.

Hasil diperoleh dengan menghitung jarak terdekat, yaitu nilai jarak Euclidean yang paling kecil.

Pada tiap-tiap histogram akan dicari *occurrences* nilai citra uji maupun citra latih kemudian dilakukan pengurangan. *Occurences* adalah jumlah kemunculan nilai pada tiap matrik.

Penerapannya seperti dibawah ini:

Perhitungan jarak dengan formula 2.9 antara citra uji dan citra latih 1 pada channel warna R.

a) Histogram R dari Citra uji memiliki nilai 98, 76, 5, 3, 164, 202, 16, 10, 27, 35, 0, 9, 11, 1, 31, 21, 28, 39, 175, 118, 179, dan 252 dengan *occurrences* 1, 1, 3, 1, 1, 1, 1, 2, 1, 1, 6, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, dan 1.
b) Histogram R dari Citra latih 1 memiliki nilai 98, 54, 0, 1, 99, 191, 50, 29, 10, 4, 17, 97, 3, 2, 9, 90, 42, 22, dan 74 dengan *occurrences* 1, 1, 6, 1, 1, 1, 1, 1, 1, 4, 1, 1, 2, 3, 1, 1, 1, 1, dan 1.

Tabel 3.1 Perhitungan Euclidean Distance Matrik R Citra Uji Dan Citra Latih 1

Citra Uji		**Citra Latih 1**		$(x_i - y_i)^2$
Nilai	***Occurences***	**Nilai**	***Occurences***	
98	1	98	1	0
76	1	76	0	1
5	3	5	0	9
3	1	3	2	1
164	1	164	0	1
202	1	202	0	1

Citra Uji		Citra Latih 1		$(x_i - y_i)^2$
Nilai	***Occurences***	**Nilai**	***Occurences***	
16	1	16	0	1
10	2	10	1	1
27	1	27	0	1
35	1	35	0	1
0	6	0	6	0
9	1	9	1	0
11	1	11	0	1
1	1	1	1	0
31	1	31	0	1
21	1	21	0	1
28	1	28	0	1
39	1	39	0	1
175	1	175	0	1
118	1	118	0	1
179	1	179	0	1
252	1	252	0	1
$\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$				**5.09**

Tabel 3.1 adalah hasil perhitungan jarak antara histogram warna citra uji dan citra latih pada channel warna R, dimana hasil haruslah bernilai *absolute*. dengan Total jarak *Euclidean distance* channel R citra uji dan citra latih 1 adalah 5.09. Selanjutnya adalah melakukan hal yang sama pada channel G dan channel B. Perhitungan jarak dengan formula 2.9 antara citra uji dan citra latih 1 pada channel warna G.

a) Histogram G dari Citra uji memiliki nilai 124, 79, 0, 164, 202, 16, 6, 26, 36, 5, 4, 34, 24, 1, 37, 178, 121, 177, dan 250 dengan *occurrences* 1, 1, 10, 1, 1, 1, 1, 2, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, dan 1.
b) Histogram G dari Citra latih 1 memiliki nilai 125, 62, 0, 97, 194, 60, 30, 11, 8, 41, 99, 5, 7, 1, 4, 3, 2, 91, 40, 31, dan 82 dengan *occurrences* 1, 1, 2, 1, 1, 1, 1, 2, 2, 1, 1, 2, 2, 2, 1, 1, 4, 1, 1, 1, dan 1.

Hasil perhitungannya dapat dilihat pada tabel 3.2.

Tabel 3.2 Perhitungan Euclidean Distance Matrik G Citra Uji Dan Citra Latih 1

Citra Uji		Citra Latih 1		$(x_i - y_i)^2$
Nilai	***Occurences***	**Nilai**	***Occurences***	
124	1	124	0	1
79	1	79	0	1
0	10	0	2	64
164	1	164	0	1
202	1	202	0	1
16	1	16	0	1

Citra Uji		Citra Latih 1		$(x_i - y_i)^2$
Nilai	***Occurences***	**Nilai**	***Occurences***	
6	1	6	0	1
26	2	26	0	4
36	1	36	0	1
5	2	5	2	0
4	1	4	1	0
34	1	34	0	1
24	1	24	0	1
1	1	1	2	1
37	1	37	0	1
178	1	178	0	1
121	1	121	0	1
177	1	177	0	1
250	1	250	0	1
$\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$				**9.11**

Perhitungan jarak dengan formula 2.9 antara citra uji dan citra latih 1 pada channel warna B.

a) Histogram B dari Citra uji memiliki nilai 151, 98, 9, 7, 172, 210, 14, 5, 0, 24, 38, 4, 2, 39, 33, 12, 37, 48, 183, 130, 13, 188, dan 255 dengan *occurrences* 1, 1, 1, 1, 1, 1, 2, 2, 4, 1, 1, 2, 1, 1, 1, 2, 1, 1, 1, 1, 1, 1, dan 1.
b) Histogram B dari Citra latih 1 memiliki nilai 144, 65, 5, 4, 102, 203, 69, 34, 15, 11, 44, 98, 17, 9, 10, 3, 16, 7, 93, 43, 36, dan 84 dengan *occurrences* 1, 1, 3, 4, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 1, 1, 2, 1, 1, 1, dan 1.

Hasil perhitungannya dapat dilihat pada tabel 3.3.

Tabel 3.3 Perhitungan Euclidean Distance Matrik B Citra Uji Dan Citra Latih 1

Citra Uji		Citra Latih 1		$(x_i - y_i)^2$
Nilai	***Occurences***	**Nilai**	***Occurences***	
151	1	151	0	1
98	1	98	1	0
9	1	9	2	1
7	1	7	2	1
172	1	172	0	1
210	1	210	0	1
14	2	14	0	4
5	2	5	3	1
0	4	0	0	16
24	1	24	0	1
38	1	38	0	1
4	2	4	4	4
2	1	2	0	1
39	1	39	0	1
33	1	33	0	1

Citra Uji		Citra Latih 1		$(x_i - y_i)^2$
Nilai	***Occurences***	**Nilai**	***Occurences***	
12	2	12	0	4
37	1	37	0	1
48	1	48	0	1
183	1	183	0	1
130	1	130	0	1
13	1	13	0	1
188	1	188	0	1
255	1	255	0	1
$\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$				**6.78**

Jadi, jarak antar histogram citra uji dan citra latih 1 adalah sebagai berikut : channel R bernilai 5.09, channel G bernilai 9.11 dan channel B bernilai 6.78, maka total jarak *Euclidean Distance* RGB nya adalah 5.09 + 9.11 + 6.78 = 20.98.

Selanjutnya, akan dilakukan langkah-langkah yang sama terhadap citra latih 2. Karena nilai *occurrences* pada citra uji telah didapatkan, maka sekarang akan dilakukan pencarian nilai *occurrences* pada citra latih 2.

Perhitungan jarak dengan formula 2.9 antara histogram citra uji dan citra latih 2 pada channel warna R.

Histogram R dari Citra latih 2 memiliki nilai 15, 9, 60, 241, 249, 255, 8, 37, 245, 252, 5, 153, 247, 250, 77, dan 204 dengan *occurrences* 1, 1, 1, 1, 1, 13, 1, 1, 1, 3, 1, 1, 1, 1, 1, dan 1.

Hasil perhitungannya dapat dilihat pada tabel 3.4.

Tabel 3.4 Perhitungan Euclidean Distance Matrik R Citra Uji Dan Citra Latih 2

Citra Uji		Citra Latih 2		$(x_i - y_i)^2$
Nilai	***Occurences***	**Nilai**	***Occurences***	
98	1	98	0	1
76	1	76	0	1
5	3	5	1	4
3	1	3	0	1
164	1	164	0	1
202	1	202	0	1
16	1	16	0	1
10	2	10	0	4
27	1	27	0	1
35	1	35	0	1
0	6	0	0	36
9	1	9	1	0
11	1	11	0	1
1	1	1	0	1
31	1	31	0	1

Citra Uji		Citra Latih 2		$(x_i - y_i)^2$
Nilai	***Occurences***	**Nilai**	***Occurences***	
21	1	21	0	1
28	1	28	0	1
39	1	39	0	1
175	1	175	0	1
118	1	118	0	1
179	1	179	0	1
252	1	252	3	4
$\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$				**8.06**

Perhitungan jarak dengan formula 2.9 antara histogram citra uji dan citra latih 2 pada channel warna G.

Histogram G dari Citra latih 2 memiliki nilai 80, 73, 120, 255, 251, 252, 74, 65, 253, 254, 71, 181, 117, 250, dan 244 dengan *occurrences* 1, 1, 1, 8, 3, 3, 1, 1, 1, 3, 1, 2, 1, 2, dan 1.

Hasil perhitungannya dapat dilihat pada tabel 3.5.

Tabel 3.5 Perhitungan Euclidean Distance Matrik G Citra Uji Dan Citra Latih 2

Citra Uji		Citra Latih 2		$(x_i - y_i)^2$
Nilai	***Occurences***	**Nilai**	***Occurences***	
124	1	124	0	1
79	1	79	0	1
0	9	0	0	81
164	1	164	0	1
202	1	202	0	1
16	1	16	0	1
6	1	6	0	1
26	2	26	0	4
36	1	36	0	1
5	2	5	0	4
4	1	4	0	1
34	1	34	0	1
24	1	24	0	1
1	1	1	0	1
37	1	37	0	1
178	1	178	0	1
121	1	121	0	1
177	1	177	0	1
250	1	250	2	1
$\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$				**10.24**

Perhitungan jarak dengan formula 2.9 antara histogram citra uji dan citra latih 2 pada channel warna B.

Histogram B dari Citra latih 2 memiliki nilai 180, 171, 216, 255, 186, 115, 183, 231, 253, 178, dan 250 dengan *occurrences* 1, 1, 1, 17, 1, 1, 1, 2, 1, 1, dan 3.

Hasil perhitungannya dapat dilihat pada tabel 3.6.

Tabel 3.6 Perhitungan Euclidean Distance Matrik B Citra Uji Dan Citra Latih 2

Citra Uji		**Citra Latih 2**		$(x_i - y_i)^2$
Nilai	***Occurences***	**Nilai**	***Occurences***	
151	1	151	0	1
98	1	98	0	1
9	1	9	0	1
7	1	7	0	1
172	1	172	0	1
210	1	210	0	1
14	2	14	0	4
5	2	5	0	4
0	4	0	0	16
24	1	24	0	1
38	1	38	0	1
4	2	4	0	4
2	1	2	0	1
39	1	39	0	1
33	1	33	0	1
12	2	12	0	4
37	1	37	0	1
48	1	48	0	1
183	1	183	1	1
130	1	130	0	1
13	1	13	0	1
188	1	188	0	1
255	1	255	17	256
$\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$				**17.46**

Jadi, jarak antar histogram citra uji dan citra latih 2 adalah sebagai berikut : channel R bernilai 8.06, channel G bernilai 10.24 dan channel B bernilai 17.46, maka total jarak *Euclidean Distance* RGB nya adalah 8.06 + 10.24 + 17.46 = 35.76.

Selanjutnya, akan dilakukan langkah-langkah yang sama terhadap citra latih 3. Karena nilai *occurrences* pada citra uji telah didapatkan, maka sekarang akan dilakukan pencarian nilai *occurrences* pada citra latih 3.

Perhitungan jarak dengan formula 2.9 antara histogram citra uji dan citra latih 3 pada channel warna R.

Histogram R dari Citra latih 3 memiliki nilai 19, 185, 244, 255, 25, 14, 81, 252, 16, 21, 13, 113, 251, 18, 24, 72, 224, 8, 4, 10, 13, dan 23 dengan *occurrences* 2, 1, 2, 5, 1, 2, 1, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, dan 1.

Hasil perhitungannya dapat dilihat pada tabel 3.7.

Tabel 3.7 Perhitungan Euclidean Distance Matrik R Citra Uji Dan Citra Latih 3

Citra Uji		Citra Latih 3		$(x_i - y_i)^2$
Nilai	***Occurences***	**Nilai**	***Occurences***	
98	1	98	0	1
76	1	76	0	1
5	3	5	0	9
3	1	3	0	0
164	1	164	0	0
202	1	202	0	0
16	1	16	2	0
10	2	10	1	1
27	1	27	0	1
35	1	35	0	1
0	6	0	0	36
9	1	9	0	1
11	1	11	0	1
1	1	1	0	1
31	1	31	0	1
21	1	21	1	0
28	1	28	0	1
39	1	39	0	1
175	1	175	0	1
118	1	118	0	1
179	1	179	0	1
252	1	252	1	0
$\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$				**7.68**

Perhitungan jarak dengan formula 2.9 antara histogram citra uji dan citra latih 3 pada channel warna G.

Histogram G dari Citra latih 3 memiliki nilai 27, 35, 200, 247, 253, 249, 33, 30, 96, 255, 250, 252, 36, 20, 125, 254, 31, 28, 81, 227, 46, 43, 37, 32, dan 38 dengan *occurrences* 1, 3, 1, 1, 2, 1, 3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, dan 1. Hasil perhitungannya dapat dilihat pada tabel 3.8.

Tabel 3.8 Perhitungan Euclidean Distance Matrik G Citra Uji Dan Citra Latih 3

Citra Uji		Citra Latih 3		$(x_i - y_i)^2$
Nilai	***Occurences***	**Nilai**	***Occurences***	
124	1	124	0	1
79	1	79	0	1
0	9	0	0	81
164	1	164	0	1
202	1	202	0	1
16	1	16	0	1
6	1	6	0	1
26	2	26	0	4
36	1	36	1	0
5	2	5	0	4
4	1	4	0	1

Citra Uji		Citra Latih 3		$(x_i - y_i)^2$
Nilai	***Occurences***	**Nilai**	***Occurences***	
34	1	34	0	1
24	1	24	0	1
1	1	1	0	1
37	1	37	1	0
178	1	178	0	1
121	1	121	0	1
177	1	177	0	1
250	1	250	1	0
$\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$				**10.09**

Perhitungan jarak dengan formula 2.9 antara histogram citra uji dan citra latih 3 pada channel warna B.

Histogram B dari Citra latih 3 memiliki nilai 164, 130, 255, 252, 170, 125, 155, 165, 152, 201, 162, 163, 104, 122, 234, 171, 169, 154, dan 157 dengan *occurrences* 3, 1, 8, 1, 1, 1, 1, 1, 2, 1, 2, 1, 1, 1, 1, 1, 1, 1, dan 1.

Hasil perhitungannya dapat dilihat pada tabel 3.9.

Tabel 3.9 Perhitungan Euclidean Distance Matrik B Citra Uji Dan Citra Latih 3

Citra Uji		Citra Latih 3		$(x_i - y_i)^2$
Nilai	***Occurences***	**Nilai**	***Occurences***	
151	1	151	0	1
98	1	98	0	1
9	1	9	0	1
7	1	7	0	1
172	1	172	0	1
210	1	210	0	1
14	2	14	0	4
5	2	5	0	4
0	4	0	0	16
24	1	24	0	1
38	1	38	0	1
4	2	4	0	4
2	1	2	0	1
39	1	39	0	1
33	1	33	0	1
12	2	12	0	4
37	1	37	0	1
48	1	48	0	1
183	1	183	0	1
130	1	130	1	0
13	1	13	0	1
188	1	188	0	1
255	1	255	8	49
$\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$				**9.84**

Jadi, jarak antar histogram citra uji dan citra latih 3 adalah sebagai berikut : channel R bernilai 7.68, channel G bernilai 10.09 dan channel B bernilai 9.84, maka total jarak *Euclidean Distance* RGB nya adalah 7.68 + 10.09 + 9.84 = 27.61.

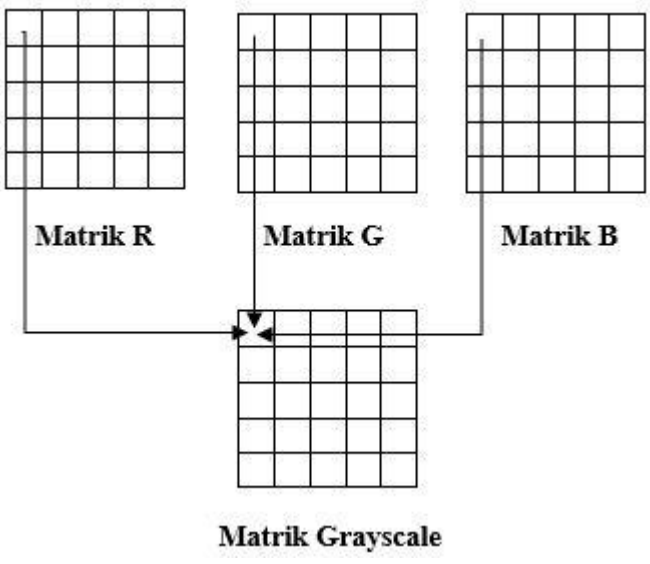
Setelah dilakukan penerapan metode *euclidean distance* pada histogram citra uji dan ketiga citra latih tersebut, didapatkan jarak dari citra uji dengan ketiga citra latih yaitu, dengan citra latih 1 adalah 20.98, dengan citra latih 2 adalah 35.76, dan dengan citra latih 3 adalah 27.61, bisa dilihat bahwa jarak *euclidean* histogram citra uji dan citra latih 1 adalah jarak yang paling dekat, yaitu 20.98.

3.1.2.4 Analisis Metode Pencocokkan Tepi *(Edge Matching)*

Metode *Edge Matching* adalah gabungan antara metode *edge detection* dan *matching process*, dimana sebelum melakukan *matching* (pencocokan) pada suatu gambar, akan dilakukan *edge detection* terlebih dahulu. *Edge Matching* digunakan untuk mencari kecocokan citra dari data uji dan data latih berdasarkan tepi yang telah dideteksi.

3.1.2.5 Analisis Deteksi Tepi (*Edge Detection*)

Pada tahap ini deteksi tepi *(edge detection)* digunakan untuk mencari tepi dari suatu citra yang telah diubah kedalam bentuk citra *grayscale* yang didapatkan dari matrik RGB sebelumnya. Contoh konversi RGB ke *grayscale* dengan citra berukuran 5x5 piksel pada gambar 3.22 :



Gambar 3.22 Matrik R, G, B Ke Matrik Grayscale

Nilai *grayscale* diperoleh dengan menerapkan formula 2.7.

Penerapan metodenya seperti dibawah ini :

Dari formula 2.7 akan diperoleh nilai matrik grayscale dari citra uji dan citra latih pada tabel 3.10, tabel 3.11, tabel 3.12, dan tabel 3.13 :

Tabel 3.10 Matrik Grayscale Citra Uji

119	79	2	1	163	201
16	7	1	1	26	36
0	6	5	6	0	0
33	24	2	1	27	38
176	120	1	1	176	248

Tabel 3.11 Matrik Grayscale Citra Latih 1

119	60	0	0	97	192
58	30	11	7	36	98
7	5	5	6	1	3
3	2	8	9	1	2
90	40	3	2	29	80

Tabel 3.12 Matrik Grayscale Citra Latih 2

73	66	113	250	248	250
67	62	249	252	252	252
64	177	251	252	252	251
112	198	252	250	249	250
234	251	252	250	249	250

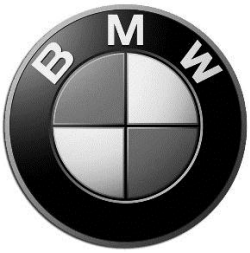
Tabel 3.13 Matrik Grayscale Citra Latih 3

35	38	199	244	251	248
41	33	96	252	249	250
40	42	28	127	249	251
40	39	38	31	81	225
38	46	43	39	36	43

Contoh citra normal (gambar 3.23) yang telah dikonversi kedalam citra *grayscale* (gambar 3.24) :



Gambar 3.23 Citra Normal



Gambar 3.24 Citra Grayscale Dari Gambar 3.4

Setelah diperoleh nilai dari matrik *grayscale*, matrik *grayscale* tersebut kemudian dikonversi kedalam matrik *edge* dengan teknik konvolusi, titik (j,i) dikatakan sebagai tepi (edge) dari suatu citra bila titik tersebut mempunyai perbedaan yang tinggi dengan tetangga. Pada aplikasi ini, proses deteksi tepi menggunakan metode Sobel. Metode ini mengambil prinsip dari fungsi laplacian dan gaussian yang dikenal sebagai fungsi untuk membangkitkan High Pass Filter. Kelebihan dari metode sobel ini adalah kemampuan untuk mengurangi noise sebelum melakukan perhitungan deteksi tepi. Tepi diperoleh dengan mengalikan tiap piksel gambar dengan kernel Sobel vertikal dan horisontal seperti pada gambar 3.25.

Kernel Vertikal =

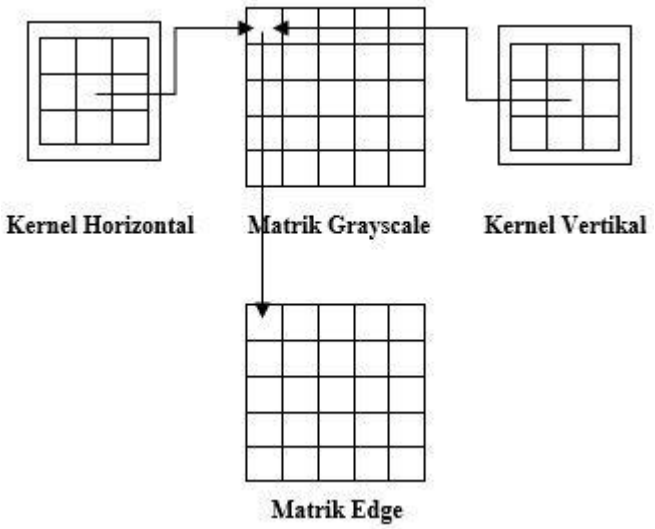
-1	0	1
-2	0	2
-1	0	1

Kernel Horizontal =

1	2	1
0	0	0
-1	-2	-1

Gambar 3.25 Kernel Vertikal Dan Kernel Horizontal Metode Sobel

Contoh penerapan teknik konvolusi pada citra *grayscale* dengan ukuran 5x5 piksel untuk mendapatkan matrik *edge* pada gambar 3.26 :



Gambar 3.26 Konversi Grayscale Kedalam Edge

Dari matrik grayscale citra uji dan citra latih yang telah dicari sebelumnya, akan dilakukan konvolusi terhadap setiap nilainya dengan kernel horizontal dan kernel vertikal, untuk mendapatkan nilai dari setiap bagian matrik, dilakukan perhitungan konvolusi sebagai berikut :

Tabel 3. 14 Matrik Grayscale Citra Uji

119	79	2	1	163	201
16	7	1	1	26	36
0	6	5	6	0	0
33	24	2	1	27	38
176	120	1	1	176	248

Cara mendapatkan nilai konvolusi matrik citra uji pada titik (0,0) dari tabel 3.14 yaitu 119 adalah sebagai berikut :

Horizontal = G1(x,y) = | (0 * 119) + (0 * 79) + (-2 * 16) + (-1 * 7) | = |-39| = 39

Vertikal = G2(x,y) = | ((0 * 119) + (2 * 79) + (0 * 16) + (1 * 7) | = |165| = 165

$G(x,y) = \sqrt{(G1(x,y) * G1(x,y)) + (G2(x,y) * G2(x,y))}$

$= \sqrt{(39 * 39) + (165 * 165)}$

$= 169{,}5 \approx 170$

Hasil konvolusi pada titik (0,0) adalah 170. Setelah itu dilakukan cara yang sama terhadap setiap bagian dari matrik citra uji maupun citra uji, sehingga

didapatkan hasil konvolusi dari citra uji (tabel 3.15), citra latih 1 (tabel 3.16), citra latih 2 (tabel 3.17), dan citra latih 3 (tabel 3.18) sebagai berikut :

Tabel 3. 15 Hasil Konvolusi Dari Citra Uji

170	251	162	348	444	365
326	298	109	255	585	605
67	63	35	40	60	53
497	462	193	273	673	710
279	390	263	376	539	393

Tabel 3.16 Hasil Konvolusi Dari Citra Latih 1

209	313	155	227	507	327
306	305	112	163	528	504
144	123	35	35	182	230
207	170	35	20	143	185
82	170	74	52	150	59

Tabel 3.17 Hasil Konvolusi Dari Citra Latih 2

276	512	985	1041	1008	1064
379	687	748	204	11	1004
654	766	416	4	10	1005
922	581	193	10	9	999
817	780	953	1001	998	1058

Tabel 3.18 Hasil Konvolusi Dari Citra Latih 3

158	433	791	887	1000	1061
147	306	859	708	171	998
156	55	505	924	730	856
166	24	82	479	887	772
177	156	148	183	464	553

Setelah didapatkan hasil konvolusi dari citra uji dan ketiga citra latih tersebut, maka akan dilakukan normalisasi *threshold* untuk menentukan tepian (*edge)*. Nilai normalisasi *threshold* didapatkan dengan menerapkan formula 2.8.

Jika f(*x*,*y*) adalah nilai intensitas *pixel* pada posisi (*x*,*y*) maka *pixel* tersebut diganti putih atau hitam tergantung kondisi berikut :

$f(x,y) = 255$, jika $f(x,y) \geq T$

f(x,y) = 0, jika f(x,y) < T

Diketahui nilai dari hasil konvolusi grayscale citra uji pada tabel 3.19 yaitu :

Tabel 3.19 Hasil Konvolusi Dari Citra Uji

170	251	162	348	444	365
326	298	109	255	585	605
67	63	35	40	60	53
497	462	193	273	673	710
279	390	263	376	539	393

Dari nilai pada tabel 3.19 akan dilakukan normalisasi *threshold* untuk mendapatkan nilai T, dengan menerapkan formula 2.8 seperti dibawah ini :

$$T = \frac{fmaks + fmin}{2} = \frac{710 + 35}{2} = 372{,}5 \approx 373$$

Nilai T dari citra uji adalah 373. Cara ini juga diterapkan pada ketiga citra latih, sehingga didapatkan hasil seperti dibawah ini :

Nilai T dari citra latih 1 adalah 274.

Nilai T dari citra latih 2 adalah 534.

Nilai T dari citra latih 3 adalah 543.

Hasil dari normalisasi *threshold* dari nilai konvolusi citra uji dan ketiga citra latih dapat dilihat pada pada tabel 3.20, tabel 3.21, tabel 3.22, dan tabel 3.23 :

Tabel 3.20 Normalisasi *Threshold* Citra Uji

0	0	0	0	255	0
0	0	0	0	255	255
0	0	0	0	0	0
255	255	0	0	255	255
0	255	0	255	255	255

Tabel 3.21 Normalisasi *Threshold* Citra Latih 1

0	255	0	0	255	255
255	255	0	0	255	255
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0

Tabel 3.22 Normalisasi *Threshold* Citra Latih 2

0	0	255	255	255	255
0	255	255	0	0	255
255	255	0	0	0	255
255	255	0	0	0	255
255	255	255	255	255	255

Tabel 3.23 Normalisasi *Threshold* Citra Latih 3

0	0	255	255	255	255
0	0	255	255	0	255
0	0	0	255	255	255
0	0	0	0	255	255
0	0	0	0	0	255

Setelah dilakukan normalisasi *threshold* terhadap nilai konvolusi pada setiap piksel *grayscale*, citra tersebut diubah kedalam citra biner atau dalam bentuk nilai 0 dan 1, dimana bentuk angka 0 untuk warna hitam dan angka 255 untuk warna putih, sehingga didapatkan hasil sebagai berikut :

1. Bentuk biner citra uji (tabel 3.24).

Tabel 3. 24 Bentuk Biner Citra Uji

0	0	0	0	1	0
0	0	0	0	1	1
0	0	0	0	0	0
1	1	0	0	1	1
0	1	0	1	1	1

2. Bentuk biner citra latih 1 (tabel 3.25).

Tabel 3. 25 Bentuk Biner Citra Latih 1

0	1	0	0	1	1
1	1	0	0	1	1
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0

3. Bentuk biner citra latih 2 (tabel 3.26).

Tabel 3. 26 Bentuk Biner Citra Latih 2

0	0	1	1	1	1
0	1	1	0	0	1
1	1	0	0	0	1
1	1	0	0	0	1
1	1	1	1	1	1

4. Bentuk biner citra latih 3 (tabel 3.27).

Tabel 3. 27 Bentuk Biner Citra Latih 3

0	0	1	1	1	1
0	0	1	1	0	1
0	0	0	1	1	1
0	0	0	0	1	1
0	0	0	0	0	1

Contoh citra grayscale (gambar 3.27) yang telah dikonversi kedalam citra *edge* (gambar 3.28) :



Gambar 3.27 Citra Grayscale



Gambar 3.28 Edge Dari Citra Grayscale Pada Gambar 3.7

3.1.2.6 Analisis Proses Pencocokkan (*Matching Process*)

Matching Process digunakan untuk menghitung jarak antar tepi*(edge)*. Dimana dalam tahapan ini menentukan tingkat kemiripan dari tepi citra uji dan tepi citra latih. Semakin kecil jaraknya maka citra tersebut semakin mirip dengan citra

uji. Pada penelitian ini, proses pencocokkan menggunakan perhitungan *Chamfer*. Perhitungan ini diimplementasikan dengan formula 2.10.

Pengenalan diperoleh dengan menghitung jarak terdekat, yaitu nilai *chamfer* yang paling kecil.

Pada tiap koordinat *edge (i, j)* yang dideteksi pada piksel citra uji maupun citra latih dilakukan pengurangan dan dikuadratkan, setelah itu hasil dari masing-masing koordinat *i* dan *j* akan dilakukan penambahan, kemudian dilakukan penjumlahan dari hasil tiap-tiap koordinat. Perhitungan terpusat pada citra uji dari awal koordinat, jadi jika pada citra latih tidak ditemukan koordinat yang sama maka akan diganti dengan koordinat yang terdekat.

Cara untuk mencocokkan titik *edge* dari citra uji dan citra latih adalah sebagai berikut, diketahui nilai dari citra biner dari citra uji (tabel 3.28) dan citra latih 1 (tabel 3.29) yaitu :

Tabel 3. 28 Citra Biner Dari Citra Uji

0	0	0	0	1	0
0	0	0	0	1	1
0	0	0	0	0	0
1	1	0	0	1	1
0	1	0	1	1	1

Tabel 3. 29 Citra Biner Dari Citra Latih 1

0	1	0	0	1	1
1	1	0	0	1	1
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0

Titik yang akan dicocokkan dari citra uji pada tabel 3.28 adalah titik (0,4), maka titik tersebut akan dibandingkan dengan setiap titik *edge* yang ada pada citra latih 1 pada tabel 3.29.

Tabel 3. 30 Pencarian Titik (0,4) Pada Citra Uji Terhadap Titik Pada Citra Latih 1

Citra Uji		**Citra Latih 1**		$\sqrt{(j_1 - j_2)^2 + (i_1 - i_2)^2}$
I	***j***	***i***	***j***	
0	4	0	1	3
0	4	0	4	0

Citra Uji		Citra Latih 1		$\sqrt{(j_1 - j_2)^2 + (i_1 - i_2)^2}$
I	***j***	***i***	***j***	
0	4	0	5	1
0	4	1	0	4.12
0	4	1	1	3.16
0	4	1	4	1
0	4	1	5	1.41

Penerapannya adalah sebagai berikut:

a) Citra uji memiliki koordinat *edge* (0,4), (1,4), (1,5), (3,0), (3,1), (3,4), (3,5), (4,1), (4,3), (4,4), dan (4,5).
b) Citra latih 1 memiliki koordinat *edge* (0,1), (0,4), (0,5), (1,0), (1,1), (1,4), dan (1,5).
c) Citra latih 2 memiliki koordinat *edge* (0,2), (0,3), (0,4), (0,5), (1,1), (1,2), (1,5), (2,0), (2,1), (2,5), (3,0), (3,1), (3,5), (4,0), (4,1), (4,2), (4,3), (4,4), dan (4,5).
d) Citra latih 3 memiliki koordinat *edge* (0,2), (0,3), (0,4), (0,5), (1,2), (1,3), (1,5), (2,3), (2,4), (2,5), (3,4), (3,5), dan (4,5).

Tabel 3.31 Perhitungan Chamfer Citra Uji Dan Citra Latih 1

Citra Uji		Citra Latih 1		$\sqrt{(j_1 - j_2)^2 + (i_1 - i_2)^2}$
I	***j***	***i***	***j***	
0	4	0	4	0
1	4	1	4	0
1	5	1	5	0
1	5	0	5	1
3	0	1	0	2
3	1	0	1	3
3	1	1	1	2
Total				**8**

Jarak yang dihasilkan dengan perhitungan *chamfer* terhadap *edge* pada citra uji dan citra latih 1 (tabel3.28) adalah 2.45.

Selanjutnya akan dilakukan hal yang sama terhadap *edge* pada citra uji dan citra latih 2.

Tabel 3.32 Perhitungan Chamfer Citra Uji Dan Citra Latih 2

Citra Uji		Citra Latih 2		$\sqrt{(j_1 - j_2)^2 + (i_1 - i_2)^2}$
i	***j***	***i***	***j***	
0	4	0	4	0
1	4	1	2	2

Citra Uji		Citra Latih 2		$\sqrt{(j_1 - j_2)^2 + (i_1 - i_2)^2}$
i	***j***	***i***	***j***	
1	5	1	5	0
3	0	3	0	0
3	1	3	1	0
3	4	2	5	1.41
3	5	3	5	0
4	1	4	1	0
4	3	4	3	0
4	4	4	4	0
4	5	4	5	0
Total				**3.41**

Jarak yang dihasilkan dengan perhitungan *chamfer* terhadap *edge* pada citra uji dan citra latih 2 (tabel 3.29) adalah 4.90.

Selanjutnya akan dilakukan hal yang sama terhadap *edge* pada citra uji dan citra latih 3.

Tabel 3.33 Perhitungan Chamfer Citra Uji Dan Citra Latih 3

Citra Uji		Citra Latih 2		$\sqrt{(j_1 - j_2)^2 + (i_1 - i_2)^2}$
i	***j***	***i***	***j***	
0	4	0	4	1
1	4	1	3	1
1	5	1	5	1
3	0	1	2	2.82
3	1	2	3	2.23
3	4	3	4	0
3	5	3	5	1
4	1	2	4	3.60
4	3	2	5	2.82
4	4	0	5	4.12
4	5	4	5	0
Total				**19.59**

Jarak yang dihasilkan dengan perhitungan *chamfer* terhadap *edge* pada citra uji dan citra latih 3 (tabel 3.30) adalah 3.32.

Setelah dilakukan penerapan metode *edge matching* pada citra uji dan ketiga citra latih tersebut, didapatkan jarak tepi dari citra uji dengan ketiga citra latih yaitu, dengan citra latih 1 adalah 3.41, dengan citra latih 2 adalah 16.12, dan dengan citra latih 3 adalah 8.24,bisa dilihat bahwa jarak citra uji dan citra latih 1 adalah jarak yang paling dekat, yaitu 3.41.

Hasil dari penerapan metode *euclidean distance* dan *edge matching* terhadap citra uji dan ketiga citra latih, ditemukan bahwa tingkat kemiripan citra uji dan citra latih 1 adalah yang paling kecil. Maka bisa disimpulkan bahwa citra latih yang paling mendekati struktur citra uji adalah citra latih 1.

3.1.2.7 Kesimpulan

Dari hasil yang didapatkan melalui perhitungan euclidean *distance* dan edge matching, maka diperoleh data seperti pada tabel 3.34.

Tabel 3. 34 Hasil Penerapan Metode

Jarak Dengan Citra Uji		
	Euclidean *Distance*	***Edge Matching***
Citra Latih 1	20.98	8
Citra Latih 2	35.76	3.41
Citra Latih 3	27.61	19.59

Dari data yang diperoleh dari tabel 3.31, dapat dilihat bahwa citra latih 1 memiliki jarak euclidean *distance* yang paling kecil yaitu, 20.98 dan citra latih 2 memiliki jarak *edge matching* yang paling kecil yaitu, 3.41. Dengan demikian disimpulkan bahwa citra latih 1 adalah citra yang paling menyerupai citra uji menurut warna dan citra latih 2 adalah citra yang paling menyerupai menurut *edge* (tepi). Maka citra latih yang akan dianggap mirip adalah citra latih 1 dan citra latih 2.

3.1.3 Analisis Kebutuhan Sistem

Kebutuhan sistem yang diperlukan dan yang digunakan dalam pembuatan sistem ini terdiri dari beberapa bagian, yaitu: kebutuhan fungsional (tabel 3.31) dan kebutuhan non fungsional (tabel 3.32).

Tabel 3.35 Kebutuhan Fungsional

Nomor	Keterangan
SKPL-F-01	Sistem dapat menganalisa citra yang di-*input*-kan
SKPL-F-02	Sistem dapat memberikan informasi tentang citra yang dianalisis

Tabel 3.36 Kebutuhan Non-Fungsional

Nomor	Keterangan
SKPL-NF-01	Sistem dibangun dalam bentuk aplikasi *desktop.*
SKPL-NF-02	Sistem dapat dijalankan pada perangkat komputer dengan sistem operasi minimal windows 7.

3.1.3.1 Analisis Kebutuhan Non-Fungsional

Analisis kebutuhan nonfungsional adalah sebuah langkah dimana seorang pembangun aplikasi menganalisis sumber daya yang dibutuhkan untuk menggunakan aplikasi yang akan dibangun. Analisis kebutuhan non-fungsional yang dilakukan dibagi dalam dua tahap, yaitu analisis kebutuhan perangkat keras dan analisis kebutuhan perangkat lunak.

3.1.3.1.1 Analisis Kebutuhan Perangkat Keras

Analisis perangkat keras dimaksudkan untuk mengetahui spesifikasi perangkat keras yang digunakan oleh aplikasi ini. Spesifikasi perangkat keras yang digunakan dalam pembangunan aplikasi ini dapat dilihat pada tabel 3.33.

Tabel 3.37 Spesifikasi Perangkat Keras Yang Digunakan Dalam Pembangunan Aplikasi

No	Nama Perangkat	Spesifikasi
1	Prosessor	Intel Core 2 duo 2.00 GHz
2	Monitor	Monitor 14 inch (1366x768)
3	Memori	RAM 3 GB DDR2
4	Harddisk	256 GB SATA 7200rpm

Sedangkan spesifikasi kebutuhan perangkat keras yang dibutuhkan untuk menjalankan aplikasi yang dibangun, yang harus dipenuhi dapat dilihat pada tabel 3.34.

Tabel 3.38 Spesifikasi Kebutuhan Perangkat Keras Yang Dibutuhkan Untuk Menjalankan Aplikasi

No	Nama Perangkat	Spesifikasi
1	Prosessor	Intel Core 2 duo 2.00 GHz
2	Monitor	Monitor 14 inch (1366x768)
3	Memori	RAM 1 GB DDR2
4	Harddisk	80 GB SATA 7200rpm

Spesifikasi perangkat keras pada tabel 3.34 dianjurkan untuk mempersingkat waktu dalam memproses gambar.

3.1.3.1.2 Analisis Kebutuhan Perangkat Lunak

Adapun spesifikasi perangkat lunak yang digunakan dalam pembangunan aplikasi *information retrieval* berbasis citra berdasarkan logo perusahaan dengan menggunakan metode *edge matching* dan *euclidean distance* ini dapat dilihat pada tabel 3.35.

Tabel 3.39 Spesifikasi Perangkat Lunak Yang Digunakan Dalam Pembangunan Aplikasi

No	Nama Perangkat	Spesifikasi
1	Sistem Operasi	Windows 8 Professional
2	JDK *(Java Development Kit)*	JDK 1.7.0
3	NetBeans	NetBeans *IDE* 8.0

Sedangkan spesifikasi kebutuhan perangkat lunak yang dibutuhkan untuk menjalankan aplikasi yang dibangun, yang harus dipenuhi dapat dilihat pada tabel 3.36.

Tabel 3.40 Spesifikasi Kebutuhan Perangkat Lunak Yang Dibutuhkan Untuk Menjalankan Aplikasi

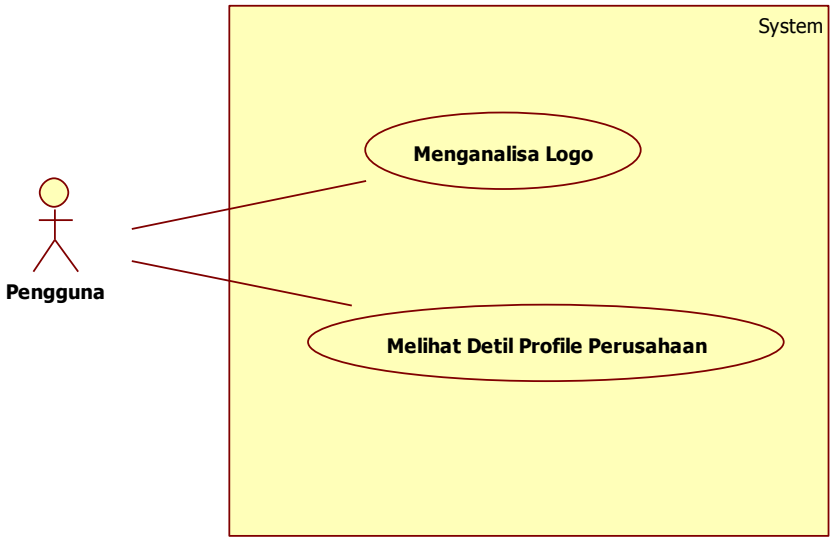
No	Nama Perangkat	Spesifikasi
1	Sistem Operasi	Windows 7
2	JDK *(Java Development Kit)*	JDK 1.7.0
3	NetBeans	NetBeans *IDE* 8.0

3.1.3.2 Analisis Kebutuhan Fungsional

Analisi kebutuhan fungsional adalah segala bentuk data yang dibutuhkan oleh sistem agar sistem dapat berjalan sesuai dengan prosedur yang dibangun. Aplikasi yang dibangun akan dimodelkan menggunakan *Unified Modeling Language* (UML), dan d*tools* yang akan digunakan yaitu *use case* diagram, *activity* diagram, *sequence* diagram, dan *class* diagram.

3.1.3.2.1 Use Case Diagram

Use case diagram digunakan untuk mengetahui apa saja yang dapat dilakukan oleh pengguna/aktor terhadap fungsionalitas yang terdapat pada aplikasi yang dibangun. *Use case* diagram pada aplikasi ini dapat dilihat pada gambar 3.29.



Gambar 3.29 Use Case Diagram

1. *Actor Definition*

Actor Definition berfungsi untuk menjelaskan *Actor* yang terdapat pada *Use case diagram*. *Actor Definition* diterangkan pada Tabel di 3.37.

Tabel 3.41 Actor Definition

No	*Actor*	Deskripsi
1	Pengguna	Orang yang menggunakan Aplikasi

2. *Use Case Definition*

Use Case Definition berfungsi untuk menjelaskan fungsi *Use Case* yang terdapat pada *Use case diagram. Use Case Definition* diterangkan pada tabel di 3.38.

Tabel 3.42 Use Case Definition

No	*Use Case*	Deskripsi
1	Menganalisa Logo	Proses menganalisa logo *input* dan melakukan pencocokan dengan logo yang ada pada *database* dan menampilkan hasil dari pencarian citra berupa gambar yang sama atau menyerupai
2	Melihat Detil Profile Perusahaan	Proses menampilkan detil profile perusahaan dari logo yg ditampilkan.

3. *Use Case* Skenario

Use Case Skenario mendeskripsikan urutan langkah-langkah dalam proses bisnis baik yang dilakukan aktor terhadap sistem maupun yang dilikakukan oleh sistem terhadap aktor. Berdasarkan *use case diagram* pada Gambar diatas maka *use case Skenario* untuk aplikasi yang dibangun akan dijelaskan sebagai berikut.

1. *Use Case* Skenario Menganalisa Logo

Use case skenario dari *use case* menganalisa logo dijelaskan pada tabel 3.39.

Tabel 3.43 Use Case Scenario Menganalisa Logo

Identifikasi	
Nomor	1
Nama	Menganalisa Logo
Tujuan	Proses menganalisa logo *input* dan melakukan pencocokan dengan logo yg ada pada *database* dan menampilkan hasil dari pencarian citra berupa gambar yang sama atau menyerupai
Aktor	Pengguna
Utama	
Kondisi Awal	Pengguna berada di menu utama
Aksi Aktor	**Reaksi Sistem**
1. Pengguna menekan tombol browse untuk mengunggah logo	
	2. Sistem menganalisa histogram warna dari logo input
	3. Sistem menganalisa bentuk dari logo input
	4. Sistem melakukan pencocokkan data dari logo input dan logo yang ada dalam database
Kondisi Akhir	Menampilkan citra yang sama atau menyerupai

2. *Use Case* Skenario Melihat Detil Profile Perusahaan

Use Case Skenario dari *Use Case* melihat detil *profile* perusahaan dijelaskan pada tabel 3.40.

Tabel 3.44 Use Case Skenario Melihat Detil Profile Perusahaan

Identifikasi	
Nomor	2
Nama	Melihat Detil Profile Perusahaan
Identifikasi	
Tujuan	Proses menampilkan detil profile perusahaan dari logo yang ditampilkan.
Aktor	Pengguna
Utama	
Kondisi Awal	Pengguna berada di menu utama yang menampilkan logo yang telah dianalisa terlebih dahulu

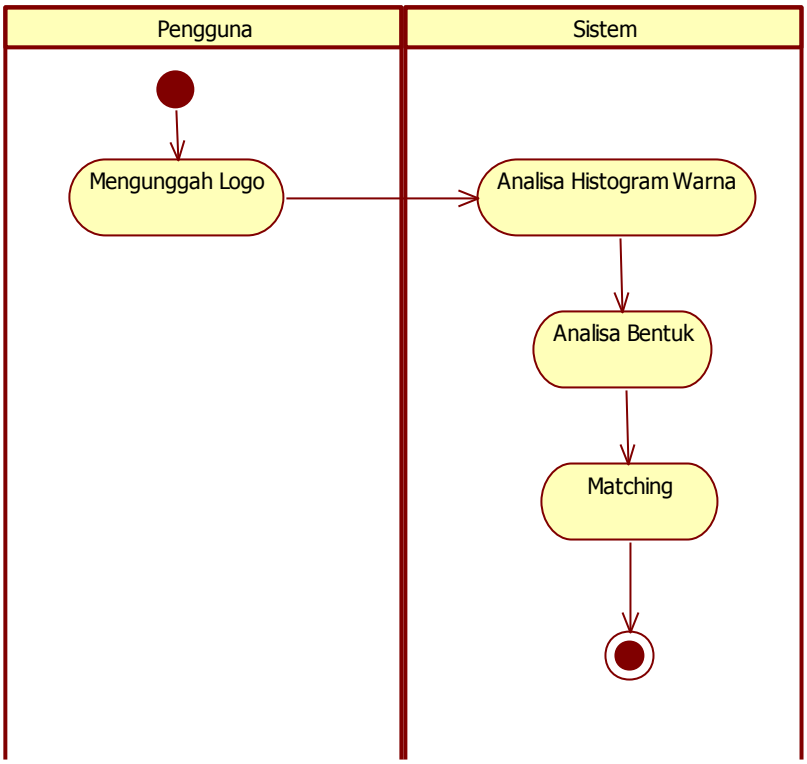
Aksi Aktor	Reaksi Sistem
1. Memilih logo yang ditampilkan	
	2. Sistem menampilkan informasi dari perusahaan.
Kondisi Akhir	Menampilkan informasi perusahaan

3.1.3.2.2 *Activity* Diagram

Activity diagram adalah teknik untuk mendeskripsikan logika procedural, proses bisnis dan aliran kerja dalam banyak kasus. Activity diagram mempunyai peran seperti halnya *flowchart*, akan tetapi perbedannya dengan *flowchart* adalah activity diagram bisa mendukung perilaku parallel sedangkan *flowchart* tidak bisa.

1. *Activity* Diagram Menganalisa Logo

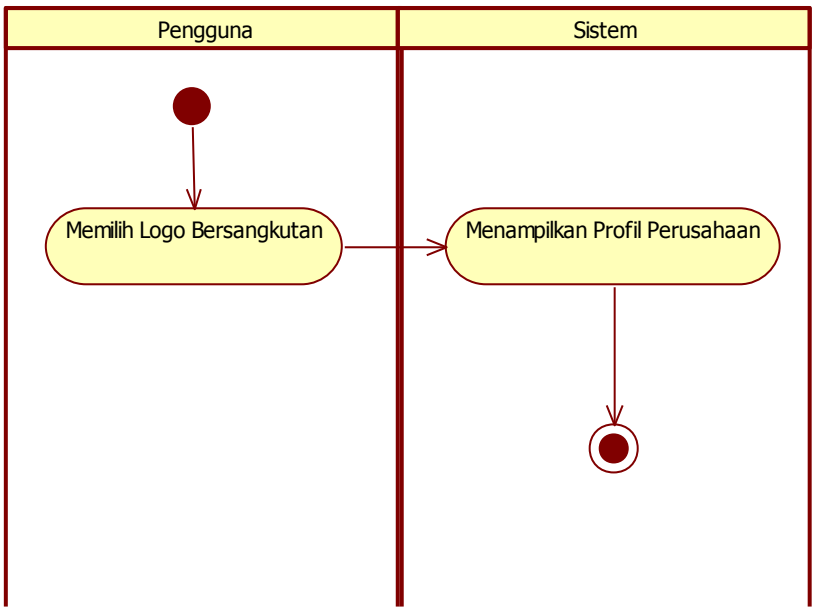
Activity diagram menganalisa logo menggambarkan alur aktivitas pencarian dan pencocokkan logo *input* dengan logo yang ada pada database seperti terlihat pada gambar 3.30.



Gambar 3.30 Activity Diagram Menganalisa Logo

2. *Activity* Diagram Melihat Detil Profile Perusahaan

Activity diagram melihat detil profile perusahaan menggambarkan alur aktivitas pemilihan logo dari citra yang ditampilkan seperti terlihat pada gambar 3.31.



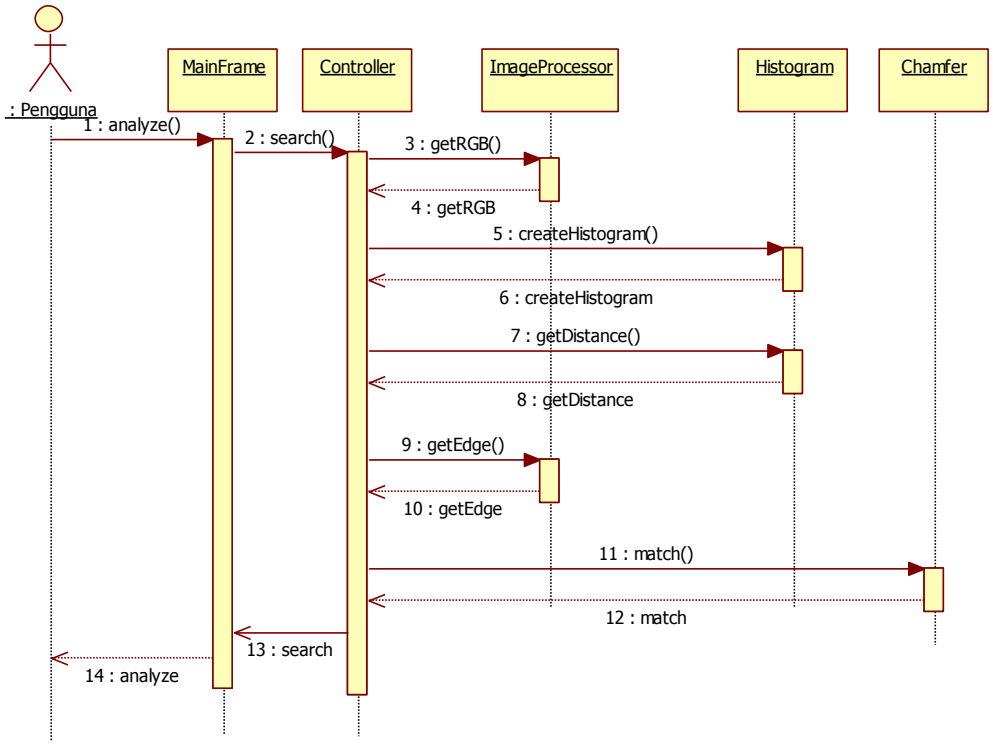
Gambar 3.31 Activity Diagram Melihat Detil Profile Perusahaan

3.1.3.2.3 Sequence Diagram

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) yang digambarkan terhadap waktu. *Sequence* diagram yang terdapat pada aplikasi ini adalah sebagai berikut :

1. *Sequence* Diagram Menganalisa Logo

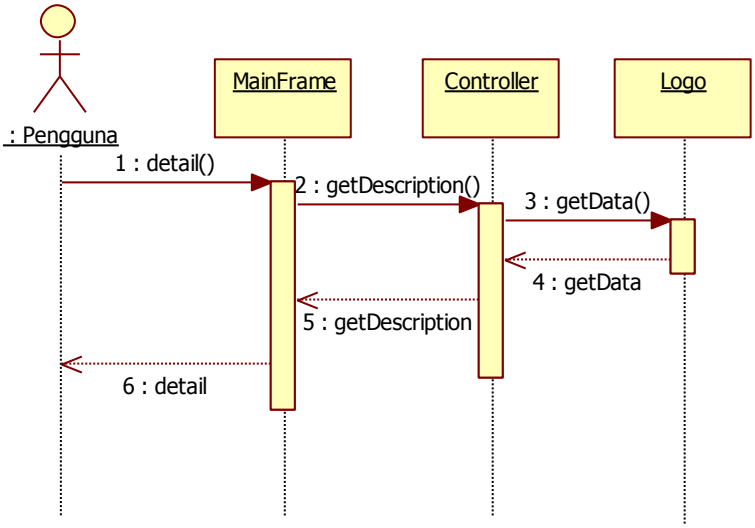
Sequence diagram menganalisa logo merupakan diagram yang mengambarkan interaksi yang terjadi didalam sistem antara pengguna dengan sistem dalam menganalisa logo yang di-*input*-kan oleh pengguna yang kemudian ditampilkan oleh sistem. *Sequence* diagram menganalisa logo aplikasi ini terlihat seperti pada gambar 3.32.



Gambar 3.32 Sequence Diagram Menganalisa Logo

2. *Sequence* Diagram Melihat Detil Profile Perusahaan

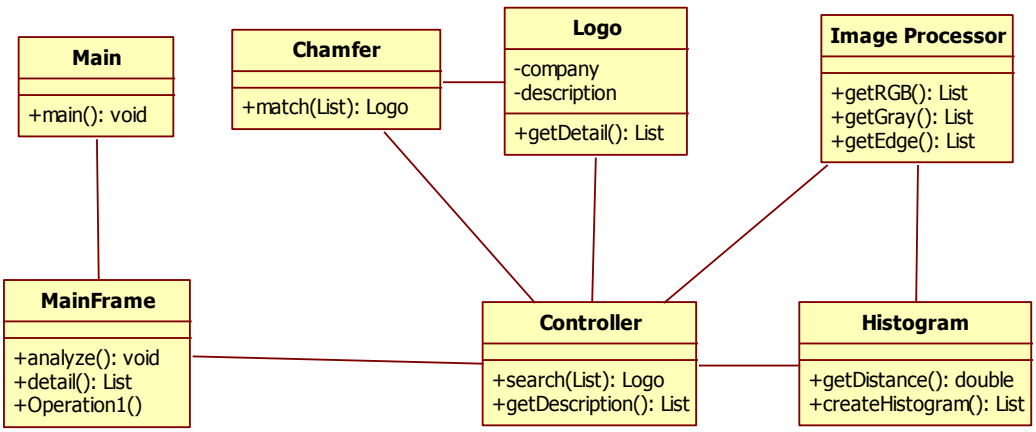
Sequence diagram melihat detil profile perusahaan merupakan diagram yang mengambarkan interaksi yang terjadi didalam sistem antara pengguna dengan sistem dalam menampilkan informasi perusahaan dari logo yang dipilih oleh pengguna yang kemudian ditampilkan oleh sistem. *Sequence* diagram melihat detil profile perusahaan aplikasi ini terlihat seperti pada gambar 3.33.



Gambar 3.33 Sequence Diagram Melihat Detil Profile Perusahaan

3.1.3.2.4 *Class* Diagram

Class diagram digunakan untuk menggambarkan keadan suatu sistem dengan menjelaskan keterhubungan antara suatu *class* dengan *class* yang lain yang terdapat pada sistem. *Class* diagram yang digunakan pada pembangunan aplikasi ini dapat dilihat pada gambar 3.34:



Gambar 3.34 Class Diagram

3.2 Perancangan Sistem

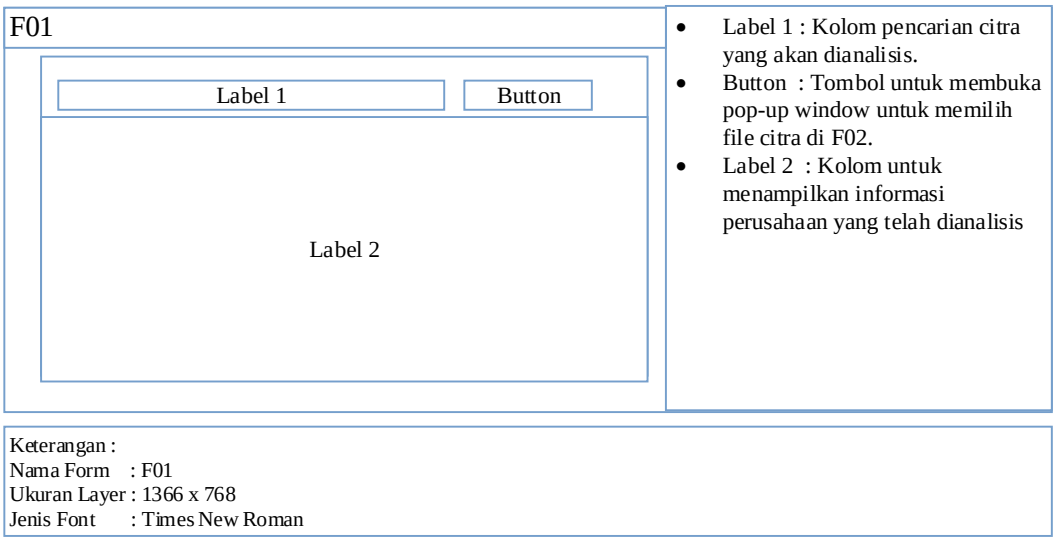
Perancangan sistem merupakan suatu proses yang menggambarkan bagaimana suatu sistem dibangun untuk memenuhi kebutuhan pada fase analisis.

3.2.1 Perancangan Antarmuka

Perancangan antarmuka merupakan sebuah penggambaran, perencanaan, dan pembuatan sketsa atau pengaturan dari beberapa elemen yang terpisah ke dalam satu kesatuan yang utuh dan berfungsi. Adapun perancangan antarmuka aplikasi *information retrieval* berbasis citra berdasarkan logo perusahaan dengan menggunakan metode *edge matching* dan *euclidean distance* adalah sebagai berikut.

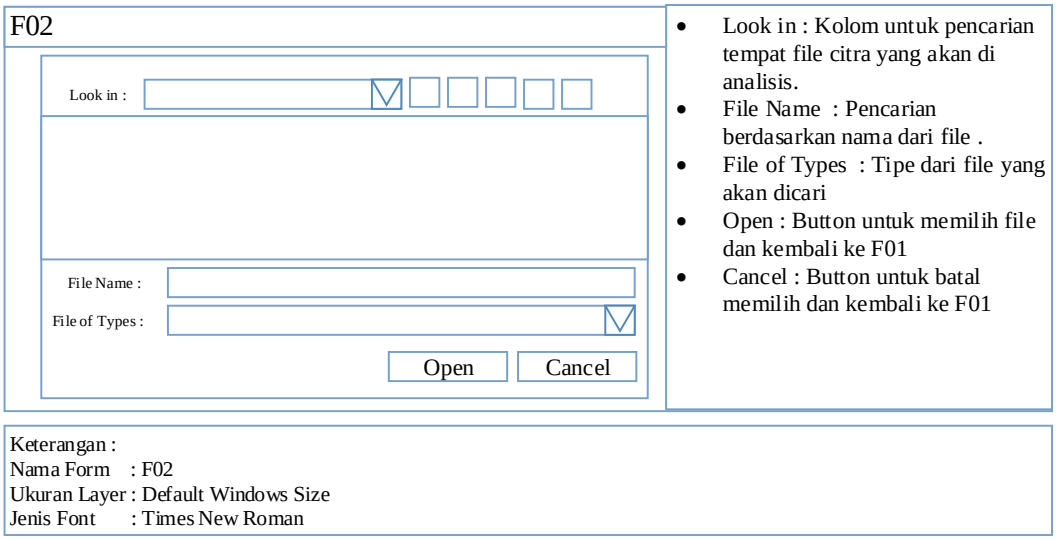
3.2.1.1 Form Perancangan Menu Utama

Form menu utama merupakan *form* yang digunakan sebagai tampilan pada saat pengguna membuka aplikasi. Gambar 3.35 dan gambar 3.36 adalah desain tampilan form aplikasi.



Gambar 3. 35 Form Perancangan Menu Utama

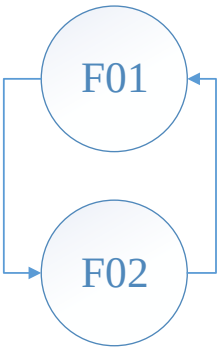
3.2.1.2 Form Perancangan Menu Pop-Up Window



Gambar 3. 36 Form Perancangan Menu Pop-Up Window

3.2.2 Perancangan Jaringan Semantik

Jaringan semantik merupakan alur dari menu-menu yang terdapat didalam sebuah aplikasi yang dibangun. Jaringan semantik yang terdapat didalam aplikasi pengukuran tinggi objek ini dapat dilihat pada gambar 3.37.



Gambar 3.37 Jaringan Semantik Penggun