BAB III
ANALISIS DAN PERANCANGAN SISTEM

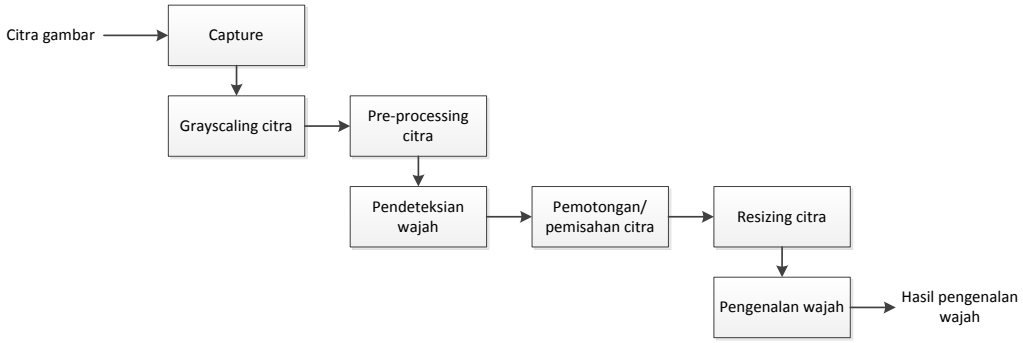
3.1 Analisis Masalah

Analisis masalah merupakan langkah awal dari suatu analisis sistem. Langkah ini diperlukan untuk mengetahui permasalahan apa saja yang terjadi di dalam sistem. Setelah melakukan penelitian maka terdapat beberapa permasalahan yang timbul dalam sistem pengenalan wajah berbasis *principal component analysis* (PCA), yakni masih terdapat kesalahan dalam mengenali citra wajah 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 wajah 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 *photometric normalization* akan dicoba untuk diterapkan pada sistem. Algoritma ini memiliki dua metode khusus yakni *isotrophic smoothing* untuk membantu dalam penghalusan citra gambar dan metode *multiscale retinex* yang mampu untuk membantu dalam mengolah citra gambar yang pencahayaannya kurang. Sehingga akan didapatkan hasil berupa seberapa besarkah dampak dari penerapan algoritma ini dalam menangani kesulitan yang terjadi pada sistem.

3.2 Analisis Sistem

Pada bagian ini menjelaskan tentang metode yang akan diterapkan pada sistem. Terutama penjelasan untuk bagian-bagian pada algoritma pengolahan citra yang akan diterapkan pada sistem. Dalam sistem pengenalan wajah terdapat beberapa tahapan utama. Tahapan proses utama sistem tersebut dapat dilihat pada gambar 3.1.



Gambar 3. 1 Proses utama sistem

Berikut penjelasan dari tiap tahapannya adalah sebagai berikut :

1. Penangkapan/*capture* pada citra masukan

   Merupakan langkah pengambilan data citra pada media masukan kedalam sistem. Citra masukan dapat berupa file gambar atau foto yang didalamnya mengandung objek wajah atau juga dapat melalui pengambilan gambar melalui *webcam*
2. Grayscaling citra

   Merupakan proses perubahan warna pada citra kedalam derajat keabuan
3. Pre-processing citra

   Merupakan proses pengolahan citra. Pada tahap ini citra gambar akan diolah menggunakan metode pada algoritma *photometric normalization*.
4. Pendeteksian wajah

   Merupakan proses pendeteksian objek wajah pada citra gambar masukan
5. Pemotongan citra

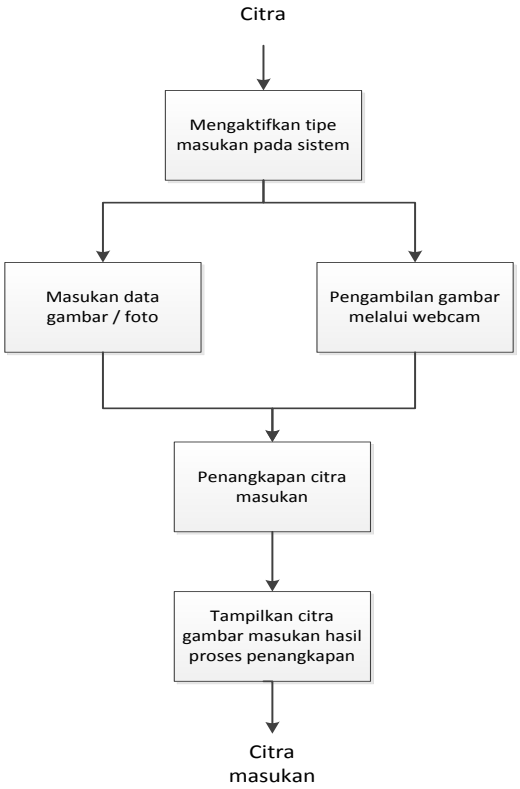
   Merupakan proses pemotongan/pemisahan objek pada citra
6. Resizing citra

   Merupakan proses perubahan ukuran citra kedalam ukuran yang telah ditentukan
7. Pengenalan wajah

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

3.2.1 Penangkapan/capture Pada Citra Masukan

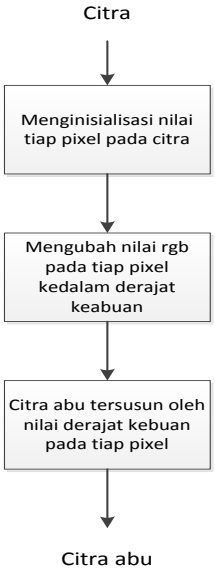
Penangkapan citra masukan yang dilakukan oleh sistem terhadap tipe citra yang dimasukan yaitu dapat berupa file gambar dengan menggunakan menu *input* atau juga dapat diambil secara langsung melalui kamera *webcam*. Untuk lebih jelasnya dapat dilihat pada gambar 3.2 berikut.



Gambar 3. 2 Alur proses penangkapan citra masukan

3.2.2 Grayscaling Citra

Proses pengubahan nilai warna RGB pada setiap *pixel* citra gambar kedalam derajat keabuan. Untuk lebih jelasnya dapat dilihat pada alur diagram pada gambar 3.3 berikut.



Gambar 3. 3 Alur proses *grayscaling* citra

3.2.3 Pre-processing Citra

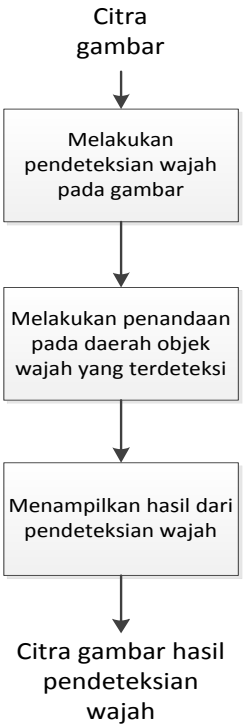
Proses pengolahan pada citra dengan metode yang telah ditentukan sebelum dilakukan proses selanjutnya terhadap citra. Untuk lebih lengkapnya dapat dilihat pada gambar 3.4.



Gambar 3. 4 Alur proses *pre-processing* citra

3.2.4 Pendeteksian Wajah

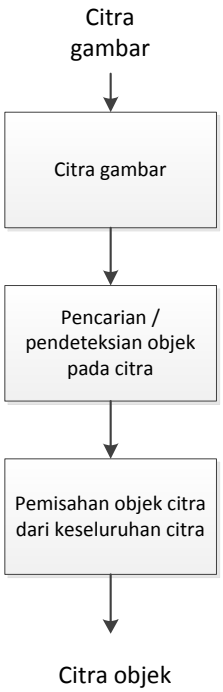
Proses untuk melakukan pencarian objek wajah pada citra gambar kemudian objek wajah pada gambar akan diberi tanda. Untuk alurnya dapat dilihat pada gambar 3.5.



Gambar 3. 5 Alur proses pendeteksian wajah

3.2.5 Pemotongan Citra

Proses pemisahan citra yang mengandung informasi mengenai suatu objek dari keseluruhan citra. Untuk alurnya dapat dilihat pada gambar 3.6.



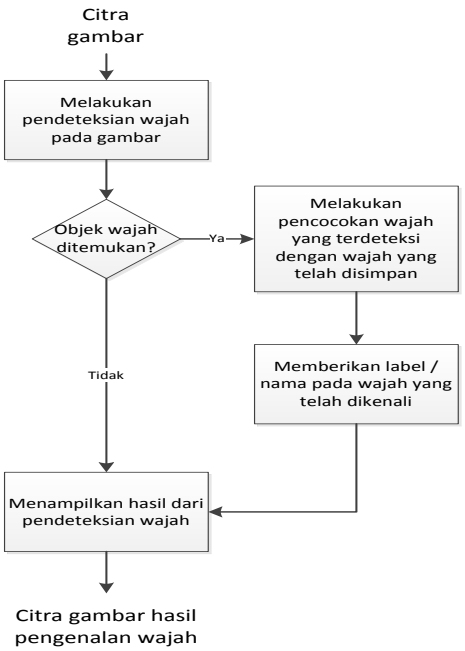
Gambar 3. 6 Alur proses pemotongan citra

3.2.6 Resizing Citra

Proses untuk membentuk citra gambar kedalam suatu ukuran yang telah ditentukan. Untuk ukuran citra latih dengan citra uji maka kedua citra tersebut akan ditentukan dan disamakan ukurannya, dengan ukuran yang ditentukan adalah 100 x 100 *pixel*.

3.2.7 Pengenalan Wajah

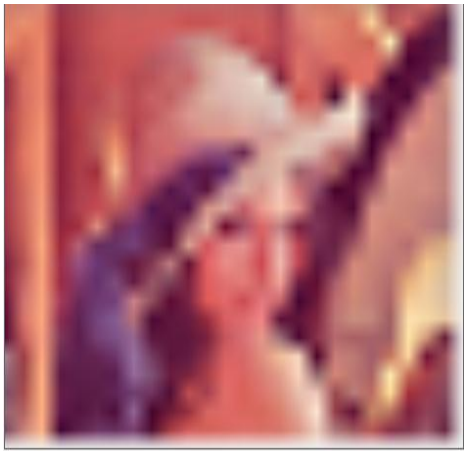
Proses untuk mengenali wajah dari citra gambar masukan dengan wajah yang telah disimpan didalam sistem sehingga wajah yang diujikan akan dikenali sebagai wajah dari seseorang. Untuk alurnya dapat dilihat pada gambar 3.7.



Gambar 3. 7 Alur proses pengenalan wajah

3.3 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 3.8 berikut.



Gambar 3. 8 Contoh citra gambar RGB

Gambar 3.8 memiliki ukuran 25x24 *pixel* dengan model warna RGB. Maka untuk nilai *pixel*nya dapat dilihat pada gambar 3.9.

206 126 109	207 128 111	218 136 114	146 56 80	163 72 82	179 88 91	191 97 95	191 97 97	193 101 97	190 100 100	189 98 100	191 100 98	191 99 97	187 95 94	187 96 98	161 74 86	210 132 116	209 126 107	205 122 102	208 140 113	206 166 134	164 73 84	177 91 93	199 105 100	139 76 90
223 129 108	232 145 111	230 126 103	158 48 67	174 68 74	193 82 83	206 93 91	204 94 92	208 94 92	211 92 89	205 81 81	201 82 82	206 93 93	206 91 88	206 93 91	177 69 83	217 118 114	226 138 113	225 129 105	214 111 98	254 211 152	179 88 85	211 104 101	136 45 72	84 27 64
232 142 113	222 117 90	233 132 106	152 43 66	175 66 76	198 82 85	202 91 89	209 88 92	206 93 96	201 105 114	199 136 135	202 154 144	202 126 121	193 75 85	206 88 89	180 72 84	208 104 102	219 125 113	222 123 102	215 112 97	206 102 85	255 245 181	110 18 55	80 15 60	93 33 64
202 98 89	210 106 90	238 135 105	141 34 61	168 64 78	188 77 82	201 84 85	200 86 85	182 87 97	188 101 109	196 122 124	213 171 152	223 190 173	215 186 172	174 75 85	170 57 77	221 113 105	208 103 100	221 114 102	219 110 96	227 114 101	184 109 103	72 2 50	90 17 58	105 45 75
147 44 66	216 114 95	236 132 106	143 41 66	172 65 79	187 79 81	201 85 88	182 74 83	184 84 90	200 107 107	198 103 104	211 161 151	224 190 182	227 199 189	235 222 207	129 33 58	224 116 104	201 96 97	148 48 71	228 110 97	222 111 102	67 0 46	94 18 59	97 31 77	204 139 129
144 47 67	214 109 98	234 134 110	137 35 62	172 64 79	181 69 79	196 99 98	186 74 83	200 99 102	192 96 97	205 135 134	220 178 161	220 182 173	222 193 183	223 201 187	215 200 193	202 81 76	212 100 97	65 6 45	255 214 190	104 36 71	97 21 62	76 7 53	205 125 128	200 141 124
161 56 71	215 109 94	240 142 111	141 37 61	178 63 73	187 85 87	196 97 91	186 71 81	192 87 95	206 119 115	218 152 146	214 161 149	214 163 154	222 180 169	214 179 170	222 200 194	216 148 137	206 134 123	230 210 196	247 233 212	117 64 94	75 0 46	163 84 104	222 143 128	193 130 123
156 52 70	218 113 94	242 147 115	144 38 64	172 58 71	189 96 92	225 124 98	178 63 79	204 107 108	205 112 113	212 131 128	220 148 138	221 149 136	223 161 151	210 157 154	206 153 145	219 185 170	224 200 186	229 197 186	236 186 170	63 0 39	105 27 71	227 146 137	212 137 123	195 132 123
160 55 72	227 119 97	244 149 114	140 34 60	175 60 72	181 88 88	243 159 112	185 77 89	199 100 103	208 107 110	187 98 122	150 80 124	144 80 119	107 38 79	216 156 150	224 187 168	223 196 181	217 179 169	186 106 106	154 93 105	64 0 44	191 108 117	214 142 129	205 130 124	191 130 121
166 61 74	220 114 94	243 151 116	139 33 60	175 62 74	161 48 61	250 189 144	184 78 89	202 98 113	166 75 103	98 32 87	93 22 73	90 35 100	208 154 148	211 172 157	222 183 166	158 98 108	188 107 121	172 103 100	76 0 47	96 22 62	225 152 137	211 143 127	205 136 129	189 129 124
164 60 76	209 106 93	246 155 121	139 33 61	173 64 78	185 67 77	214 137 113	194 80 94	148 59 90	87 20 71	108 43 98	89 18 60	192 135 150	218 175 160	216 150 141	243 196 183	203 115 115	149 81 111	71 0 41	89 16 56	155 83 99	209 144 129	207 138 126	209 139 128	193 129 123
164 58 73	207 105 96	247 157 122	139 37 66	178 66 81	197 75 80	189 91 88	164 72 103	80 22 76	98 28 75	82 17 70	214 146 136	217 166 155	194 92 103	236 156 145	247 198 176	226 146 141	118 47 79	100 26 64	79 8 51	213 141 131	207 141 123	199 135 122	193 126 123	180 121 122
159 54 72	203 100 91	248 161 122	146 37 61	162 46 62	225 143 127	122 46 81	113 54 112	96 34 83	77 10 57	166 97 109	226 174 157	109 6 41	160 81 104	222 101 102	193 128 131	119 30 62	127 53 87	105 33 67	116 43 76	219 144 127	205 129 117	205 132 119	188 120 122	170 118 120
146 47 68	193 92 87	252 170 128	159 39 56	185 98 118	104 45 93	102 50 104	118 66 123	102 37 76	89 18 54	219 164 148	210 84 94	227 134 135	230 146 147	211 87 96	233 178 170	190 68 85	113 43 78	104 34 68	163 88 99	210 137 123	198 128 121	183 109 112	212 154 127	216 173 132
117 32 63	184 83 84	250 162 123	158 48 75	152 81 128	107 56 117	104 45 92	100 23 58	131 56 82	192 129 129	99 16 50	230 97 99	233 142 135	224 127 121	207 71 80	239 187 175	193 69 83	96 24 60	119 48 79	213 135 123	200 132 118	189 119 117	197 130 115	244 197 135	219 187 153
126 30 58	198 100 99	251 163 123	158 43 67	132 58 108	101 49 106	128 77 125	101 28 66	152 85 101	126 67 90	124 32 63	226 92 93	225 114 114	226 125 122	206 94 96	244 164 151	110 29 66	97 14 49	171 95 108	216 140 120	195 126 119	180 108 110	234 176 124	240 203 155	217 190 161
104 17 51	208 115 113	250 165 127	168 60 88	107 41 89	110 57 113	115 63 120	203 130 139	147 82 94	91 11 51	122 37 67	226 91 89	225 105 106	205 76 83	227 120 125	204 104 111	68 6 52	112 25 56	190 109 111	205 137 117	193 126 120	175 100 102	247 203 150	238 205 161	220 192 163
87 11 51	216 115 105	252 168 128	141 43 79	95 34 81	117 69 126	135 92 147	182 112 120	77 10 52	96 23 61	104 27 63	151 43 60	217 95 98	221 108 111	242 133 133	92 16 54	96 21 58	130 38 66	189 117 116	202 142 125	195 136 126	160 93 99	255 225 173	246 216 168	233 198 156
85 12 53	197 95 92	249 162 129	128 56 89	93 27 74	120 66 116	125 82 136	149 103 145	94 23 61	93 24 62	95 24 60	121 33 62	198 86 92	216 112 116	229 151 137	156 113 122	76 8 52	138 50 74	181 105 109	172 95 100	167 98 104	159 95 102	255 229 164	210 151 122	99 27 57
169 129 141	158 61 71	255 167 126	108 42 80	96 33 80	101 37 82	108 54 107	152 112 160	117 60 95	93 22 58	98 28 68	145 51 73	218 98 100	217 103 104	224 143 130	243 190 152	248 215 192	78 0 38	202 126 120	189 123 117	162 93 106	199 115 94	255 215 159	115 17 52	157 77 87
144 106 126	155 67 71	255 176 131	92 25 67	93 31 75	96 32 75	91 28 80	149 107 157	127 82 116	107 28 64	122 36 62	169 62 73	228 103 99	217 101 98	219 126 114	225 159 141	251 213 189	147 104 109	211 134 121	186 118 112	169 98 101	255 237 174	194 107 87	172 64 76	147 79 94
123 82 107	168 77 80	255 175 130	101 28 71	89 28 70	91 28 68	97 39 87	111 62 112	114 53 94	80 14 54	194 79 84	166 64 78	226 110 105	219 95 90	219 116 109	221 144 126	234 190 173	240 201 187	156 73 77	184 124 120	190 107 96	255 224 153	165 47 59	153 58 76	136 72 93
151 103 118	178 98 96	255 176 129	97 25 73	83 20 64	96 34 75	78 14 57	148 102 150	71 4 44	113 29 58	152 48 61	215 102 98	223 105 100	219 102 93	216 103 93	220 126 112	224 170 153	255 231 211	149 62 73	97 16 53	194 136 104	189 118 93	127 34 67	147 65 86	120 58 82
123 73 97	200 131 120	255 179 137	66 2 50	98 33 77	111 53 101	104 49 95	121 64 112	81 14 48	171 60 67	215 92 90	214 97 86	219 103 95	222 112 104	219 110 100	216 117 105	222 150 133	238 204 188	175 118 120	159 86 93	187 86 84	138 45 71	128 54 74	156 87 107	103 38 65

Gambar 3. 9 Nilai *pixel* pada citra gambar RGB

Gambar 3.9 memiliki ukuran 25x24 *pixel* dengan model warna RGB. Selanjutnya tentunya gambar tersebut akan melalui proses *grayscaling* dimana warna pada citra tersebut akan diubah kedalam derajat keabuan

menggunakan persamaan (2.2). Berikut gambar 3.10 hasil dari *grayscaling* terhadap citra gambar sebelumnya.



Gambar 3. 10 Citra gambar abu

Gambar 3.10 memiliki ukuran 25x24 *pixel* dengan model warna abu. Untuk nilai *pixel*nya dapat dilihat pada gambar 3.11.

130	131	138	73	85	99	107	107	110	110	108	109	108	105	106	87	136	129	125	139	160	86	101	114	87
133	144	130	66	81	94	105	105	106	104	95	95	105	103	105	85	128	140	132	118	198	97	115	63	44
143	120	135	62	81	96	103	102	106	118	142	156	133	91	101	87	115	132	128	119	108	227	39	35	49
107	113	137	54	80	91	97	98	100	113	131	170	188	185	89	75	122	114	122	118	122	115	24	37	60
62	119	136	60	81	91	99	89	97	117	114	163	191	199	218	51	124	108	66	119	120	21	38	52	143
64	117	138	54	80	84	109	89	111	107	142	177	183	193	199	199	93	111	24	211	54	41	28	135	142
72	116	143	56	79	97	106	87	101	127	157	163	166	181	180	200	152	138	208	228	79	22	98	147	135
69	119	148	57	74	105	127	80	118	122	139	153	153	165	162	156	184	198	197	186	18	49	152	141	136
72	124	149	53	76	98	154	92	112	119	115	101	98	58	161	185	194	180	115	103	20	120	146	136	134
76	120	151	53	78	64	182	93	114	93	55	45	60	158	171	182	107	120	109	22	42	155	145	141	134
76	113	155	53	80	83	138	97	78	42	66	38	145	175	154	197	125	97	20	36	95	146	142	143	134
74	113	156	57	83	90	101	91	44	50	40	150	168	106	161	197	153	64	45	28	146	143	138	132	128
71	109	159	56	64	147	65	78	55	31	108	174	28	96	115	136	49	71	51	61	147	134	136	128	124
64	102	166	57	113	66	72	88	56	36	165	101	144	155	103	181	86	61	52	99	141	133	118	152	165
50	94	160	68	103	80	65	42	72	136	35	112	150	136	89	189	87	42	65	140	135	126	133	183	180
49	110	161	63	81	71	97	47	97	80	51	107	126	135	107	169	49	33	107	142	131	116	167	192	184
37	125	163	80	62	79	85	141	92	32	55	105	118	92	133	117	26	44	118	138	131	109	192	195	186
31	123	165	64	55	91	113	122	30	42	46	60	109	121	145	36	40	56	124	143	139	102	212	205	189
32	105	162	74	48	87	103	120	42	43	42	51	100	125	155	120	28	67	114	105	107	104	212	149	44
137	75	164	60	54	57	76	130	76	40	47	68	112	116	148	184	211	20	132	128	104	118	202	38	89
116	78	171	45	51	52	50	126	97	47	53	77	116	113	133	161	210	110	138	123	106	220	110	79	91
94	88	170	49	47	47	59	82	72	33	93	79	121	107	125	147	189	201	83	129	113	206	64	74	85
112	106	170	47	40	53	34	121	23	47	63	113	116	112	112	132	171	227	75	36	133	118	54	80	72
85	135	175	23	53	73	69	84	31	74	105	107	113	122	119	124	153	203	125	96	96	63	68	100	53

Gambar 3. 11 Nilai *pixel* pada citra gambar abu

Setelah citra keabuan didapatkan maka citra akan dinormalisasikan menggunakan metode yang dipilih dari metode terdapat pada algoritma *photometric normalization*. Untuk citra sebelumnya akan dilakukan proses *isotrophic smoothing* pada gambar. Perhitungan dilakukan pada tiap *pixel* kecuali *pixel* samping yang berfungsi sebagai *border*, dengan menggunakan persamaan (2.20) atau dengan rumus berikut :

Nilai *pixel* = *pixel* + C * (Pn + Ps + Pw + Pe). Dimana Pd adalah letak dimensi sekitar *pixel* dan diisi dengan nilai *pixel* – Pd.

Kemudian nilai dari tiap *pixel* baru yang didapat akan disusun menjadi gambar yang baru dengan rumus yang didapat dari persamaan (2.21).

k = 255 / (max – min)

Nilai pixel baru = k * (nilai *pixel* – min)

Dimana max adalah nilai *pixel* baru yang terbesar dan min adalah nilai *pixel* baru yang terkecil pada hasil perhitungan sebelumnya. Berikut gambar 3.12 hasil dari perhitungan metode *isotrophic smoothing*.



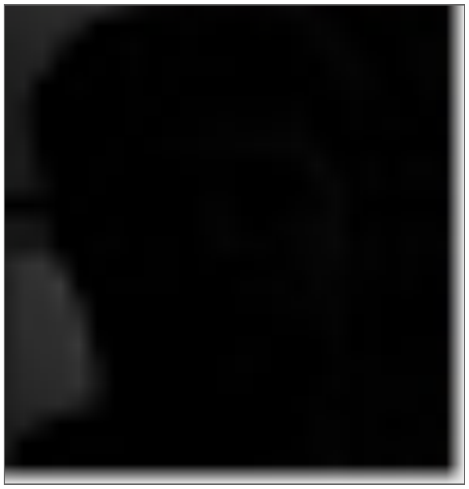
Gambar 3. 12 Citra gambar hasil pengolahan metode *isotrophic smoothing*

Gambar 3.12 memiliki ukuran 25x24 *pixel* dengan tampilan efek dari hasil *smoothing*. Untuk nilai *pixel*nya dapat dilihat pada gambar 3.13.

123	124	125	112	114	117	119	119	119	119	119	119	119	118	119	115	125	123	122	125	130	114	118	120	115
124	139	131	94	113	115	121	118	118	115	102	99	114	119	123	101	130	133	127	97	200	68	153	98	106
126	110	143	92	114	120	120	117	118	122	140	146	122	83	124	109	119	130	124	122	71	255	55	94	107
119	114	147	83	116	115	116	118	114	117	121	140	150	170	70	102	134	114	134	119	127	134	75	100	109
110	131	143	92	116	117	119	110	111	128	100	130	142	139	209	23	147	122	88	113	155	54	110	78	126
110	127	146	84	118	106	131	107	130	105	129	145	135	138	139	189	72	136	0	225	60	113	62	159	126
112	123	149	86	115	120	119	109	110	127	141	130	128	138	129	161	126	101	189	190	112	65	126	143	124
111	125	152	88	108	123	136	92	133	119	127	140	134	152	127	107	142	156	154	175	47	83	163	124	125
112	130	152	83	113	115	152	101	121	128	130	116	121	37	144	144	161	153	94	132	60	144	133	121	124
112	125	154	83	121	72	191	93	136	124	88	91	67	163	138	148	79	119	152	60	80	163	126	127	124
112	118	158	81	118	106	143	114	116	81	129	53	155	160	110	164	111	129	59	100	118	137	125	131	124
112	120	157	84	122	104	122	124	86	116	62	164	181	83	144	166	164	88	110	67	162	128	125	122	123
111	117	160	85	81	189	82	115	110	78	123	195	7	112	113	125	56	123	102	93	148	120	130	117	122
110	113	169	73	153	80	112	136	103	55	208	79	155	157	92	177	108	108	94	118	136	127	104	131	131
107	107	160	88	132	116	104	84	103	190	38	131	144	128	79	182	111	90	94	150	125	120	111	149	134
107	122	160	82	119	101	141	68	136	114	92	126	121	141	99	175	83	81	133	136	124	107	138	144	134
104	136	155	107	100	114	96	170	130	75	105	135	129	87	138	142	73	94	133	129	125	98	153	138	135
103	135	161	86	99	122	130	139	67	109	103	95	127	124	154	48	108	98	135	140	140	86	168	154	135
103	124	158	104	91	125	121	138	85	107	102	94	118	124	145	137	36	121	126	104	114	93	177	147	106
125	75	169	89	108	100	105	148	111	94	103	108	130	114	128	154	222	11	150	133	109	89	205	30	115
121	85	178	75	109	108	84	152	130	99	97	111	129	116	121	124	174	93	146	120	91	211	94	114	115
116	95	173	84	107	102	114	103	122	78	145	95	135	110	123	123	141	178	71	156	95	202	72	110	114
120	102	169	87	97	111	70	175	65	106	88	138	122	118	112	119	128	196	77	45	158	124	85	117	112
114	124	133	102	108	112	111	114	103	112	118	119	120	122	121	122	128	138	122	117	117	110	111	117	108

Gambar 3. 13 Nilai *pixel* pada citra hasil *smoothing*

Selanjutnya untuk metode *multiscale retinex*, metode ini ditujukan untuk mengolah gambar yang pencahayaanya kurang terang. Contoh citra gambar dengan pencahayaan kurang terang dan telah mengalami proses *grayscaling* yang sama seperti sebelumnya menjadi citra masukan pada sistem. Berikut contoh gambar 3.14 citra masukan.



Gambar 3. 14 Contoh citra gambar dengan pencahayaan kurang terang

Gambar 3.14 memiliki ukuran 25x24 *pixel* dengan derajat keabuan dan tampilan kurang bercahaya. Untuk nilai *pixel*nya dapat dilihat pada gambar 3.15.

37	40	43	44	45	49	33	16	4	3	1	3	3	2	2	6	8	1	0	0	2	1	2	1	2
36	38	40	41	29	0	0	0	0	0	1	0	0	0	1	0	1	5	3	0	0	0	1	0	2
31	35	38	21	0	0	0	3	0	0	0	0	0	0	0	0	0	3	5	1	0	0	1	3	2
31	33	31	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	3	5	1	0	1	3	2
28	32	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	3	3	0	0	3	1
30	20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0
29	21	0	0	0	0	0	0	0	0	0	1	3	1	1	3	3	0	0	1	0	1	0	0	1
24	25	0	0	0	0	0	0	0	1	1	3	3	3	5	5	6	3	0	0	0	0	0	0	0
25	26	5	0	0	0	0	0	0	0	0	0	0	0	0	0	1	9	1	0	3	0	3	3	0
24	24	12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	7	0	3	0	1	1	2
0	0	1	0	0	0	0	0	0	0	0	3	3	0	0	0	1	1	7	0	0	0	0	0	0
23	23	18	0	0	0	0	0	0	0	0	3	3	0	0	3	3	6	7	0	1	0	1	1	0
21	20	21	0	0	1	1	0	0	0	0	1	5	3	3	0	3	5	7	1	3	0	3	1	3
35	48	51	25	0	0	0	0	0	0	0	0	0	0	0	0	3	5	5	1	3	0	1	0	1
30	47	49	58	1	0	0	0	0	0	0	0	0	1	1	3	0	5	3	1	0	0	1	0	1
31	46	46	53	31	0	0	0	0	0	0	0	0	0	0	1	3	3	3	0	0	0	0	0	0
31	46	45	47	47	0	0	0	0	0	0	0	0	0	0	0	0	1	3	0	0	0	0	0	0
31	43	45	46	56	3	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	1
31	41	43	43	50	19	0	0	0	0	0	0	0	1	0	0	0	0	1	1	1	0	0	0	1
31	40	41	43	50	23	0	0	0	0	0	0	0	0	0	0	0	0	3	0	1	1	1	1	1
31	37	38	41	30	10	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	1	0	1
25	31	14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	3	0	1	0	0	1	0
29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	3	0	1	1	1	1	2
18	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	5	0	1	0	3	0	2	

Gambar 3. 15 Nilai *pixel* pada citra masukan

Selanjutnya tentunya gambar tersebut lebih diperjernih dengan menggunakan fungsi *gaussian bluring*. Berikut hasilnya terhadap citra gambar 3.16.



Gambar 3. 16 Contoh citra gambar hasil *gaussian bluring*

Gambar 3.16 memiliki ukuran 25x24 *pixel*. Untuk nilai *pixel*nya dapat dilihat pada gambar 3.17.

37	39	41	40	34	26	16	8	3	1	1	1	1	1	1	3	3	3	1	0	0	1	0	1	1
36	37	37	33	25	17	11	6	2	1	0	0	0	0	1	2	2	2	2	1	0	0	1	1	1
34	34	30	22	10	3	0	0	0	0	0	0	0	0	0	0	1	2	2	2	0	0	1	1	2
31	28	21	10	2	0	0	0	0	0	0	0	0	0	0	0	0	1	2	2	1	0	1	1	2
29	22	12	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	1	1
26	17	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0
24	16	7	0	0	0	0	0	0	0	0	1	1	1	2	2	2	1	0	0	0	0	0	0	0
25	17	8	0	0	0	0	0	0	0	0	1	1	1	2	2	3	2	1	0	0	0	0	0	0
24	18	10	1	0	0	0	0	0	0	0	0	1	1	1	1	3	3	2	1	0	1	0	1	1
16	13	7	2	0	0	0	0	0	0	0	0	0	0	0	0	1	3	3	2	0	1	0	1	1
15	13	8	3	0	0	0	0	0	0	0	1	1	0	0	0	1	3	3	2	0	0	0	0	0
14	14	9	4	0	0	0	0	0	0	0	2	2	1	1	1	2	4	3	2	0	0	0	1	0
28	28	22	12	2	0	0	0	0	0	0	1	1	1	1	1	3	4	4	3	1	1	0	1	1
33	35	35	22	9	0	0	0	0	0	0	0	1	1	1	1	2	4	3	2	1	1	0	1	1
39	42	47	34	18	3	0	0	0	0	0	0	0	0	0	1	2	3	2	1	0	0	0	0	0
38	41	48	41	26	8	0	0	0	0	0	0	0	0	0	0	1	2	2	1	0	0	0	0	0
38	40	46	46	31	15	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0
37	39	44	46	34	19	2	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0
36	38	42	46	37	22	5	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
35	37	40	42	34	20	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
32	32	31	28	21	12	3	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0
25	22	17	13	9	4	1	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0
17	13	5	1	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	1	0	0	0	1	1
11	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	1	0	1	1	1	1

Gambar 3. 17 Nilai *pixel* pada citra hasil gaussian bluring

Setelah citra sebelumnya didapatkan maka citra akan dinormalisasikan menggunakan metode pada algoritma *photometric normalization*. Untuk citra sebelumnya akan dilakukan proses pencerahan pada gambar menggunakan algoritma pada *multiscale retinex*. Dimana proses perhitungannya akan dihitung menggunakan besaran faktor yang memiliki nilai sebesar 1 dengan rumus sebagai berikut.

Nilai *pixel* ditambahkan dengan nilai faktor * nilai *pixel* tersebut

Kemudian nilai yang didapatkan sebelumnya akan dihitung kembali pada setiap nilai *pixel* gambar hasil *graysacing* yang sebelumnya dan disimpan kedalam sebuah *array* dua dimensi dengan menggunakan rumus berikut.

Nilai pada array ditambahkan dengan nilai *pixel* gambar abu / (nilai sebelumnya + 1)

Selanjutnya setelah nilainya didapat maka akan dilakukan perhitungan untuk menormalisasikan citra gambar menggunakan perhitungan pada persamaan (2.21).

k = 255 / (max – min)

Nilai *pixel* baru = k * (nilai pada *array* – min)

Dimana max adalah nilai *pixel* baru yang terbesar dan min adalah nilai *pixel* baru yang terkecil pada hasil perhitungan sebelumnya.

Berikut gambar 3.18 hasil dari perhitungan metode *multiscale retinex.*



Gambar 3. 18 Contoh citra gambar hasil pengolahan *multiscale retinex*

Gambar 3.18 memiliki ukuran 25x24 *pixel* dengan tampilan efek dari hasil pencerahan. Untuk nilai *pixel*nya dapat dilihat pada gambar 3.19.

83	85	87	91	109	154	165	151	85	128	42	128	128	85	85	128	170	21	0	0	170	42	170	42	85
83	85	89	102	95	0	0	0	0	0	85	0	0	0	42	0	28	142	85	0	0	0	42	0	85
75	85	104	78	0	0	0	255	0	0	0	0	0	0	0	0	0	85	142	28	0	0	42	128	57
82	97	120	0	0	0	0	85	0	0	0	0	0	0	0	0	0	0	85	142	42	0	42	128	57
79	118	0	0	85	0	0	0	0	0	0	0	0	0	0	0	0	0	42	128	128	0	0	128	42
94	94	0	0	0	0	0	0	0	0	0	0	0	0	0	0	85	0	0	42	0	0	0	0	0
99	105	0	0	0	0	0	0	0	0	0	42	128	42	28	85	85	0	0	85	0	85	0	0	85
78	118	0	0	0	0	0	0	0	85	85	128	128	128	142	142	128	85	0	0	0	0	0	0	0
85	116	39	0	0	0	0	0	0	0	0	0	0	0	0	0	21	191	28	0	255	0	255	128	0
120	146	128	0	0	0	0	0	0	0	0	0	0	0	0	0	0	64	149	0	255	0	85	42	85
0	0	9	0	0	0	0	0	0	0	0	128	128	0	0	0	42	21	149	0	0	0	0	0	0
130	130	153	0	0	0	0	0	0	0	0	85	85	0	0	128	85	102	149	0	85	0	85	42	0
62	59	78	0	0	85	85	0	0	0	0	42	212	128	128	0	64	85	119	21	128	0	255	42	128
87	113	120	92	0	0	0	0	0	0	0	0	0	0	0	0	85	85	106	28	128	0	85	0	42
64	93	87	141	4	0	0	0	0	0	0	0	0	85	85	128	0	106	85	42	0	0	85	0	85
68	93	80	107	98	0	0	0	0	0	0	0	0	0	0	85	128	85	85	0	0	0	0	0	0
68	95	81	85	125	0	0	0	0	0	0	0	0	0	0	0	0	42	128	0	0	0	0	0	0
69	91	85	83	136	13	0	0	0	0	0	0	0	0	0	0	0	0	128	0	0	0	0	0	85
71	89	85	78	112	70	0	0	0	0	0	0	0	85	0	0	0	0	85	42	85	0	0	0	85
73	89	85	85	121	93	0	0	0	0	0	0	0	0	0	0	0	0	255	0	85	85	85	85	85
80	95	101	120	116	65	0	0	0	0	0	0	0	0	0	0	0	42	0	0	85	85	85	0	85
82	115	66	0	0	0	0	0	0	0	0	0	0	0	0	0	0	128	128	0	85	0	0	85	0
137	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	85	85	0	85	85	85	42	85
128	0	0	0	0	0	0	0	0	0	0	0	0	0	0	85	0	28	142	0	85	0	128	0	85

Gambar 3. 19 Nilai *pixel* pada citra hasil pencerahan citra

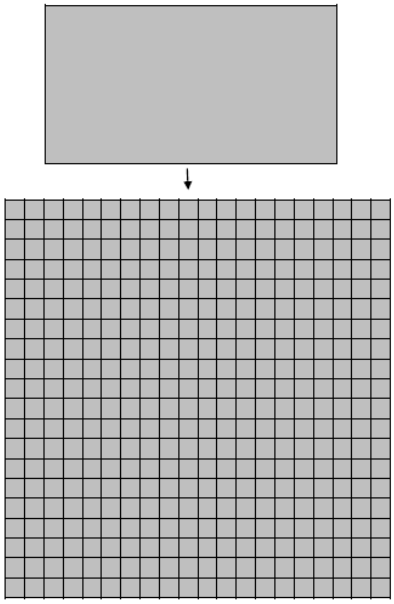
Kemudian setelah proses penormalisasian gambar selesai dilakukan maka tahapan selanjutnya adalah proses pendeteksian objek wajah yang terkandung pada citra gambar masukan dengan menggunakan library *Haarcascade Frontal Face*, yaitu suatu *library* berupa *file* templet berekstensi xml. *Library* ini mengadopsi metode Viola-Jones yang memiliki fungsi untuk mendeteksi objek berupa wajah 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 haarcascade_frontalface_alt_tree.xml :

```
   -->
- <opencv_storage>
  - <haarcascade_frontalface_tree_alt type_id="opencv-haar-classifier">
      <size>20 20</size>
    - <stages>
      - <_>
          <!-- stage 0 -->
        - <trees>
          - <_>
              <!-- tree 0 -->
            - <_>
                <!-- root node -->
              - <feature>
                - <rects>
                    <_>2 7 14 4 -1.</_>
                    <_>2 9 14 2 2.</_>
                  </rects>
                  <tilted>0</tilted>
                </feature>
                <threshold>3.7895569112151861e-003</threshold>
                <left_val>-0.9294580221176148</left_val>
                <right_val>0.6411985158920288</right_val>
              </_>
            </_>
          - <_>
              <!-- tree 1 -->
            - <_>
                <!-- root node -->
```

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 wajah. Gambar 3.20 akan menjelaskan pembentukan *sub-window* 20x20.



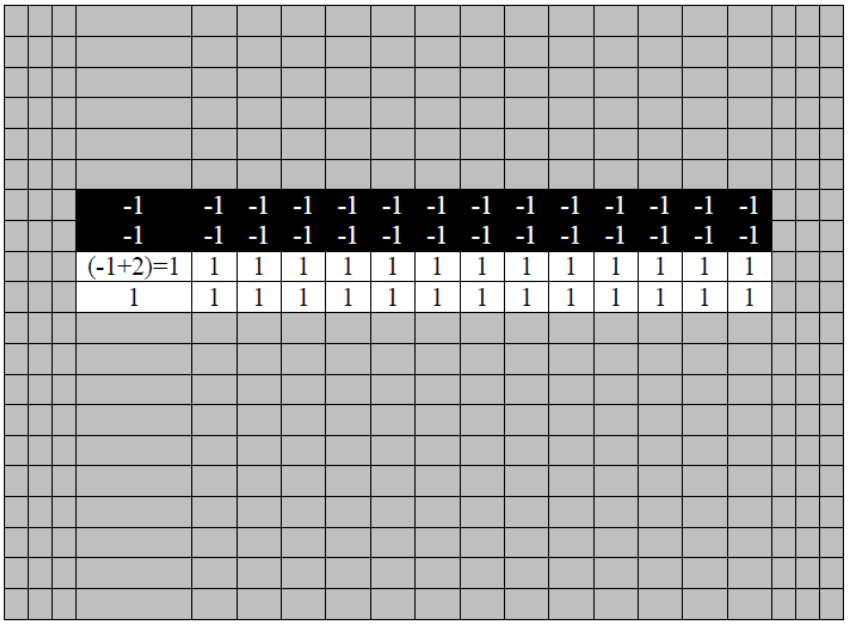
Gambar 3. 20 Pembentukan *sub-window*

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 wajah, 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>
<_>3 7 14 4 -1.</_>
<_>3 9 14 2 2.</_>
</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. 21 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 wajah maka fitur dianggap tidak ada, jika diatas *treshold* pendeteksian wajah 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,

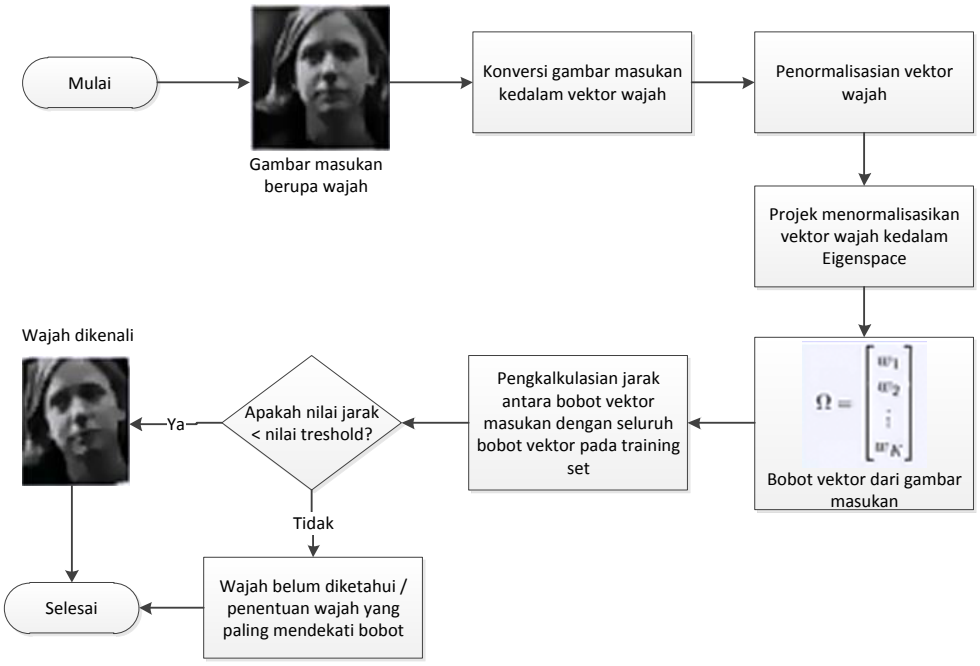
berhasil melalui keseluruhan tingkat *classifier* maka *sub-window* dianggap sebagai wajah.

Setelah wajah terdeteksi, maka citra wajah tersebut akan dipisahkan dari keselruhan citra gambar. Setelah itu citra wajah akan diubah ukurannya sesuai dengan ukuran citra wajah yang tersimpan pada sistem kemudian kedua citra wajah tersebut akan dicocokan menggunakan metode PCA.

3.4 Analisis Metode PCA Untuk Pengenalan Wajah

Dalam sistem pengenalan wajah, *Principal Component Analysis* (PCA) merupakan salah satu komponen yang penting. PCA sering digunakan sebagai alat untuk menganalisis data dan membuat model prediksi seperti pengenalan wajah. PCA juga dapat menunjukan struktur didalam data dan sangat baik dalam menjelaskan variasi fitur didalam data tersebut.

Analisis PCA ini merupakan analisis yang mendeskripsikan tahapan proses PCA dalam mengenali sebuah wajah. Alur proses dari PCA dalam mengenali sebuah wajah dapat dilihat pada gambar berikut 3.22.



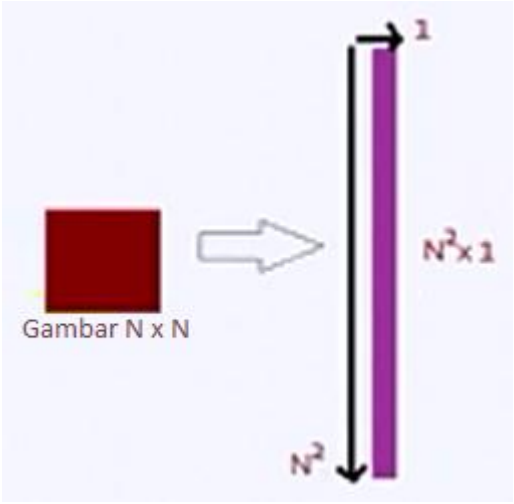
Gambar 3. 22 Alur proses PCA

Proses pengenalan wajah dengan PCA mempunyai beberapa tahapan. Secara garis besarnya ada beberapa bagian utama yaitu :

1. Konversi gambar masukan kedalam vektor
2. Penormalisasian vektor wajah
3. Menormalisasikan vektor wajah kedalam Eigenspace
4. Pencarian bobot dari vektor wajah masukan
5. Pengkalkulasian jarak bobot vektor

3.4.1 Konversi Gambar Masukan Kedalam Vektor

Pada tahap ini, gambar yang dimaksud akan diubah kedalam sebuah vektor dikarenakan metode pada PCA tidak bekerja secara langsung terhadap gambar, maka untuk mengubah gambar kedalam sebuah vektor akan dilakukan langkah sebagai berikut, misalkan terdapat gambar yang berukuran N x N maka gambar tersebut akan diubah menjadi bentuk vektor dengan ukuran N^2 x 1 seperti yang terlihat pada gambar berikut 3.23.



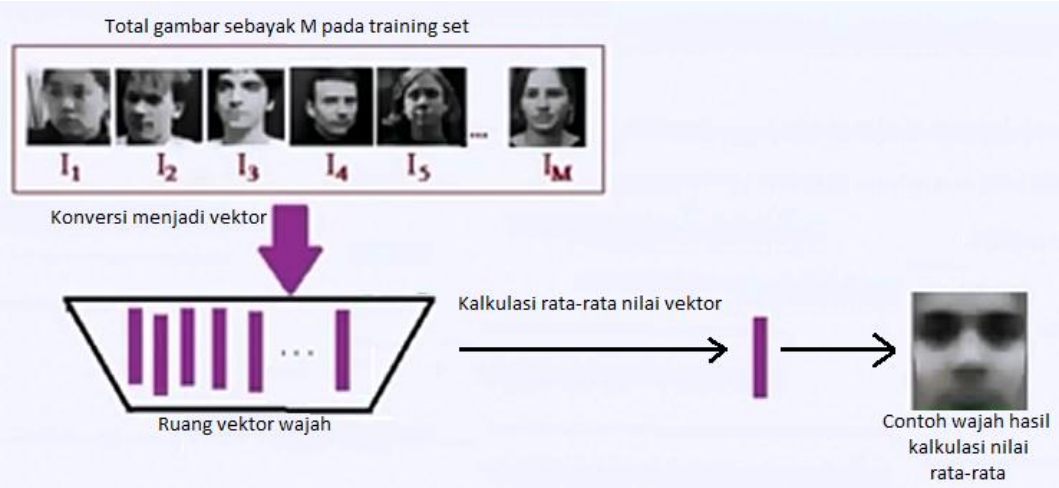
Gambar 3. 23 Proses konversi gambar kedalam vektor

Atau juga dapat dicontohkan dengan persamaan berikut.

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1N} \\ a_{21} & a_{22} & \dots & a_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N1} & a_{N2} & \dots & a_{NN} \end{bmatrix}_{N\times N} \longrightarrow \begin{bmatrix} a_{11} \\ \vdots \\ a_{1N} \\ \vdots \\ a_{2N} \\ \vdots \\ a_{NN} \end{bmatrix}_{N^2\times 1} \quad (3.1)$$

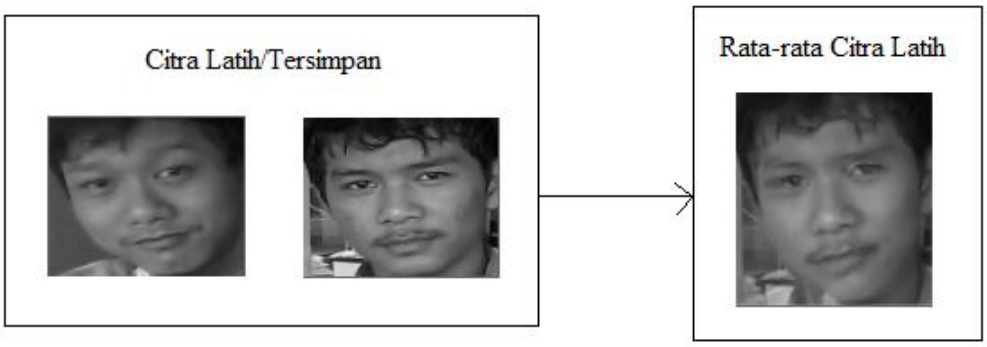
3.4.2 Penormalisasian Vektor Wajah

Pada tahap ini, penormalisasian vektor wajah akan dilakukan dengan cara mencari nilai rata-rata pada seluruh vektor wajah yang ada pada *training set* maka akan ditemukan nilai rata-rata dari gambar berupa fitur umum yang terdapat pada wajah seperti mata, hidung, dan mulut. Kemudian setelah mendapatkan nilai rata-rata tersebut, maka nilai vektor wajah yang dimaksud akan dikurangi dengan nilai rata-rata dengan tujuan untuk mendapatkan fitur khusus pada wajah. Berikut contoh gambar 3.24 untuk proses pencarian nilai rata-rata pada *training set*.



Gambar 3. 24 Proses pencarian nilai rata-rata pada training set

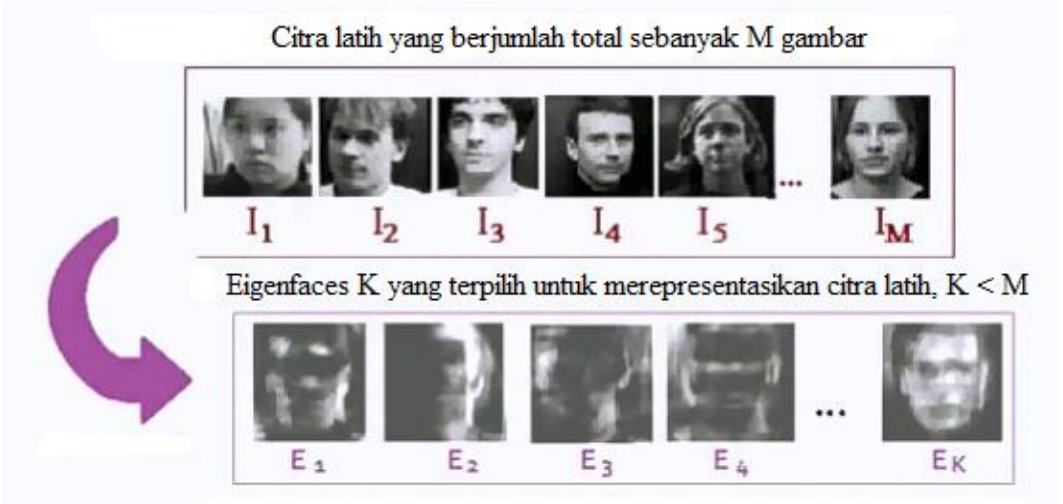
Maka untuk lebih jelasnya, berikut akan dicontohkan beberapa gambar wajah dari citra latih/tersimpan dan gambar wajah dari hasil pencarian rata-ratanya, berikut pada gambar 3.25.



Gambar 3. 25 Gambar hasil rata-rata citra latih

3.4.3 Menormalisasikan Vektor Wajah Kedalam Eigenspace

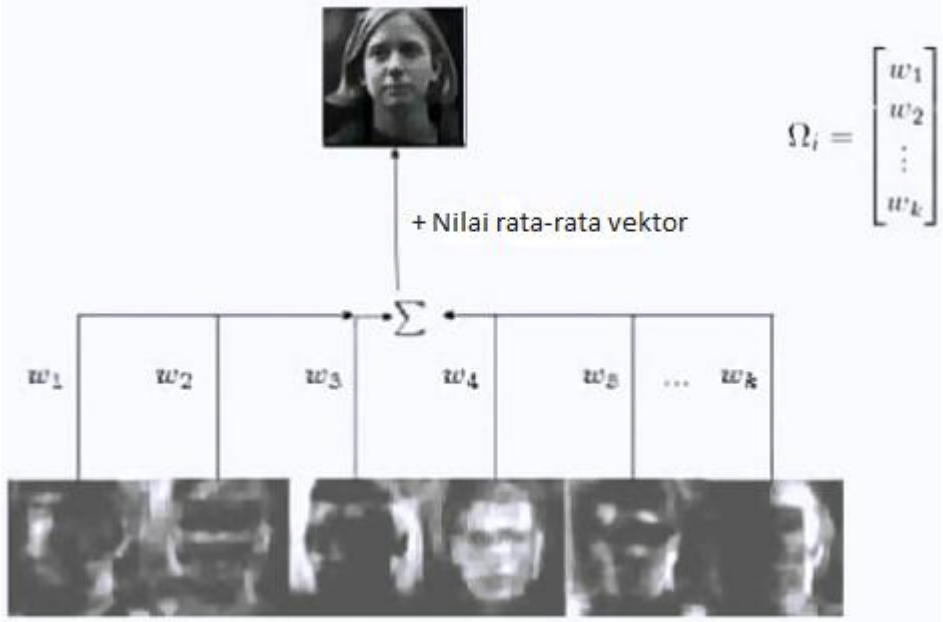
Pada tahap ini tiap vektor akan dinormalisasikan menjadi sebuah dimensi yang lebih kecil berupa *eigenface* dengan tujuan untuk merepresentasikan wajah masukan sebagai kombinasi *eigenface*. Untuk proses ini, gambar dari *eigenface* dapat dilihat pada gambar 3.26 berikut.



Gambar 3. 26 *Eigenface* dari citra latih

3.4.4 Pencarian Bobot Dari Vektor Wajah Masukan

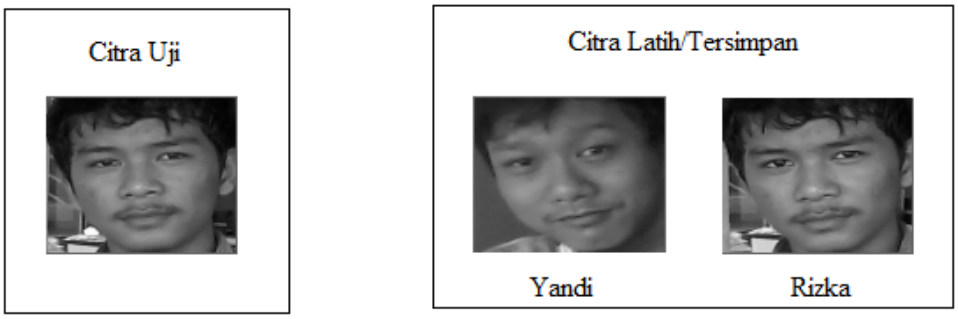
Pada tahap ini akan dilakukan penghitungan bobot nilai dari wajah masukan yaitu dari jumlah bobot dari setiap nilai yang terdapat pada kombinasi *eigenface* yang didapat ditambah dengan bobot nilai eigenface rata-rata vektor. Berikut contoh gambar 3.27 untuk tahap ini.



Gambar 3. 27 Proses perhitungan bobot dari vektor gambar masukan

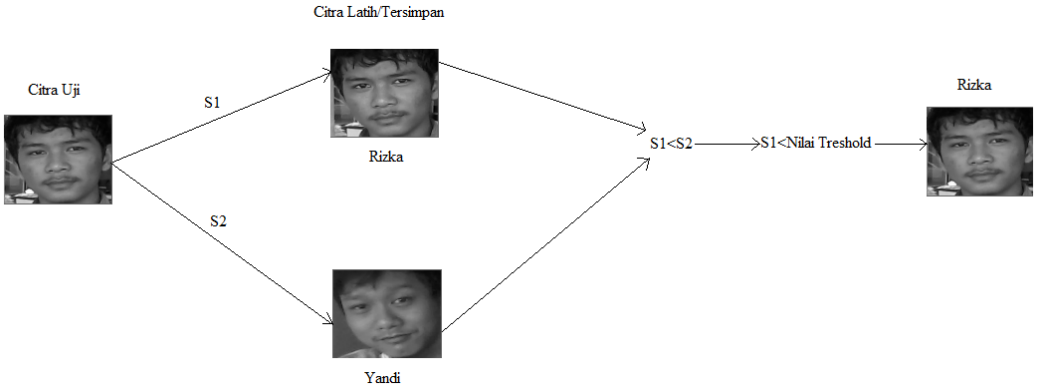
3.4.5 Pengkalkulasian Jarak Bobot Vektor

Pada tahap ini akan dilakukan perhitungan jarak antara bobot nilai pada vektor gambar masukan dengan bobot nilai tiap vektor gambar wajah yang terdapat pada *training set*. Setelah itu jika nilai jarak tersebut termasuk dalam nilai *treshold* yang ditentukan maka wajah akan dikenali sebagai wajah yang terdapat pada training set dengan bobot nilai yang paling mendekati dan apabila bobot terdekat memiliki nilai yang melebihi batas *treshold*, maka citra uji tidak dikenali. Berikut gambar 3.28 untuk contoh citra yang akan diujikan pada citra yang telah tersimpan didalam sistem.



Gambar 3. 28 Contoh citra uji dan citra latih

Maka untuk pencarian nilai kemiripan dari nilai jarak terdekat antara citra uji dengan citra tersimpan berdasarkan contoh data citra yang terdapat pada gambar 3.28 akan digambarkan menggunakan tahapan dengan alur pada gambar 3.29 berikut.



Gambar 3. 29 Alur pencarian nilai kemiripan citra uji

Dari gambar 3.29 maka dapat diketahui bahwa citra uji telah dikenali sebagai sampel nomor 1 pada data citra latih dikarenakan nilai jarak dari sampel nomor 1 adalah nilai yang paling kecil atau paling mendekati dan nilai tersebut berada didalam kriteria *treshold*. Nilai yang terdekat kemiripannya pada sistem ialah sebesar 0 dan dapat dikatakan nilai yang paling mirip, ini menunjukan bahwa gambar dari cira latih dan citra uji adalah sama. Sedangkan untuk batas nilai *treshold* adalah sebesar 3000, maka apabila nilai kemiripan yang terdekat masih berada pada rentang antara 0 dan 3000, citra uji tersebut dikenali sebagai seseorang yang terdapat pada citra latih. Sedangkan untuk nilai kemiripan diatas

3000 maka citra uji dikenali kurang mirip dengan seseorang yang terdapat pada citra latih, misalkan termasuk pada *unknown face*.

3.5 Penerapan Algoritma Photometric Normalization

Algoritma photometric normalization didefinisikan sebagai salah satu algoritma yang didalamnya terdapat metode-metode untuk pengolahan citra pada gambar yang mampu mendukung proses *pre-processing* citra gambar pada sistem pengenalan wajah sehingga mampu meningkatkan performa dari ferivikasi pengenalan wajah.

3.5.1 Penerapan Algoritma Metode Isotrophic Smooting

Metode penghalusan gambar dengan menggunakan metode *isotrophic smoothing* bekerja dengan cara mengolah pencahayaan yang dapat diperkirakan sebagai versi samar-samar (*blur*) dari gambar aslinya. Metode ini menggunakan rumus persamaan (2.20).

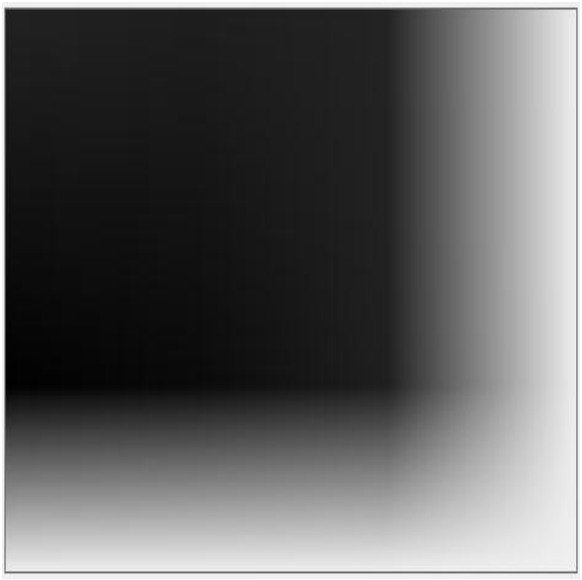
$$I(x,y) = L(x,y) + c[\nabla L_N(x,y) + \nabla L_S(x,y) + \nabla L_E(x,y) + \nabla L_W(x,y)]$$

Dimana ∇L_D mengacu pada turunan untuk tiap masing-masing keempat pengaturan arah sekitar *pixel* (*neighbouring pixel*) dan besaran dari nilai *smoothing* ditentukan oleh parameter *c*.

Maka dalam kasus ini, langah-langkah yang dilakukan untuk menerapkan persamaan (2.20) berikut kedalam sistem adalah dengan melakukan penghitungan dengan menggnakan metode isotrophic smoothing. Penghitungan ini bertujuan untuk merubah nilai dari tiap *pixel* pada gambar dengan menggunakan rumus yang telah ditentukan, dimana nilai *pixel* yang baru akan menjadi penyusun citra gambar yang baru.

3.5.2 Penghitungan Isotrophic Smoothing Pada Citra Gambar

Penghitungan ini pertama-tama dilakukan pada citra gambar abu yang didapat dari hasil *grayscaling*. Berikut contoh citra gambar 3.30 hasil dari proses *grayscaling*.



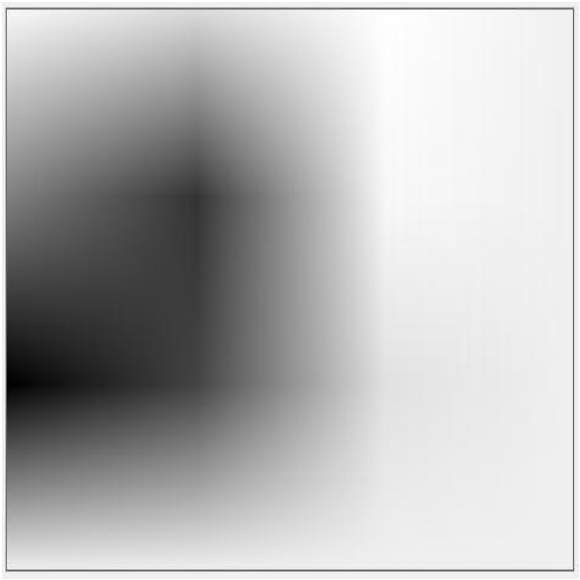
Gambar 3. 30 Contoh citra gambar abu

Gambar 3.30 memiliki ukuran 3x3 *pixel* dengan model warna abu. Untuk nilai *pixel*nya dapat dlihat pada gambar 3.31.

34	30	35
18	20	34
1	10	31

Gambar 3. 31 Nilai *pixel* pada citra gambar abu

Setelah citra keabuan didapatkan maka citra akan dinormalisasikan menggunakan metode *isotrophic smoothing*. Berikut gambar 3.32 hasil dari metode *isotrophic smoothing*.



Gambar 3. 32 Contoh citra gambar setelah proses *smoothing*

Gambar 3.32 memiliki ukuran 3x3 *pixel* dengan tampilan efek dari hasil *smoothing*. Untuk nilai *pixel*nya dapat dlihat pada gambar 3.33.

248	218	255
128	52	248
0	68	225

Gambar 3. 33 Nilai *pixel* pada citra gambar hasil *smoothing*

Sesuai dengan *code* sebelumnya perhitungan pada *pixel*, dilakukan pada *pixel* gambar abu. Perhitungan dilakukan pada tiap *pixel* kecuali *pixel* samping yang berfungsi sebagai *border*. Pada contoh diketahui *pixel* dalam yang bukan *border* ialah *pixel* pada koordinat (2, 2) maka nilai *pixel* dimasukan kedalam rumus persamaan pada *code* dengan nilai *constraint* sebesar 1.

Berikut pencarian nilai perbedaan pada setiap nilai *pixel* sekitar *pixel* yang dimaksud

Pn = 20 – 30

Ps = 20 – 10

Pw = 20 – 18

Pe = 20 – 34
Maka didapatkan hasil
Pn = -10
Ps = 10
Pw = 2
Pe = -14

Kemudian nilai yang didapat dimasukan kedalam perhitungan berikut
Nilai *pixel* = *pixel* + C * (Pn + Ps + Pw + Pe)
Sehingga
Nilai *pixel* = 20 + 1 * (-10 + 10 + 2 + (-14))
Nilai *pixel* = 20 + 1 * -12
Nilai *pixel* = 20 – 12
Sehingga nilai *pixel* baru yang didapat adalah = 8

Setelah perhitungan nilai *pixel* baru maka dapat dilihat bahwa nilai minimal *pixel* adalah 1 dan nilai maksimal *pixel* adalah 35. Kemudian akan dilakukan perhitungan untuk menormalisasikan nilai dari tiap *pixel* menggunakan nilai yang didapat. Berikut perhitungan untuk normalisasi nilai *pixel* gambar baru
k = 255 / (max – min)
Nilai pixel baru = k * (nilai pixel – min)

Contoh untuk nilai *pixel* baru pada koordinat (1, 1) dengan nilai *pixel* sebesar 1 maka
k = 255 / (35 – 1)
k = 255 / 34
k = 7,5
Nilai pixel baru = 7,5 * (1 – 1)
Nilai pixel baru = 7,5 * 0
Maka nilai *pixel* koordinat (1, 1) menjadi 0

Contoh untuk nilai *pixel* baru pada koordinat (2, 2) dengan nilai *pixel* sebesar 8 maka

Nilai pixel baru = 7,5 * (8 – 1)

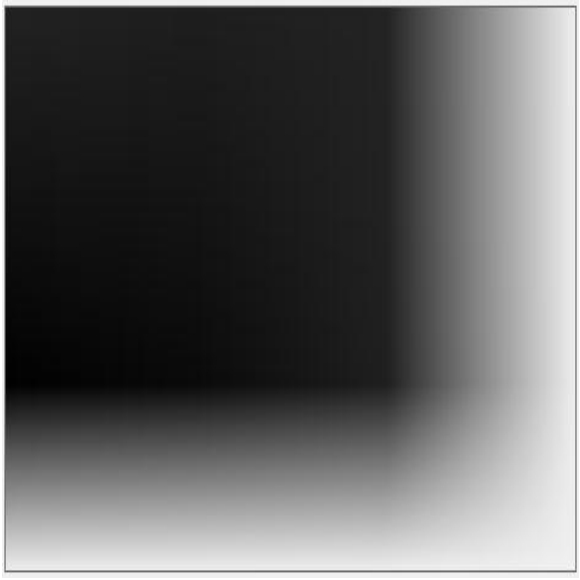
Nilai pixel baru = 7,5 * 7

Maka nilai *pixel* koordinat (2, 2) menjadi 52,5 atau sama dengan 52

Berikut adalah contoh perhitungan pada beberapa koordinat *pixel*, setelah perhitungan selesai pada seluruh *pixel* maka nilai-nilai yang didapat adalah nilai *pixel* yang terdapat pada nilai *pixel* dari hasil *smothing* yang tetera pada gambar nilai *pixel* sebelumnya.

3.5.3 Penerapan Algoritma Metode Multiscale Retinex

Metode ini bertujuan untuk mengolah citra gambar terhadap citra gambar yang memiliki kekurangan pencahayaan. Gambar citra yang terlihat gelap akan dibuat menjadi tampak lebih bercahaya. Perhitunganya pertama-tama dilakukan pada citra gambar yang sudah melalui proses grayscaling. Berikut contoh citra gambarnya.



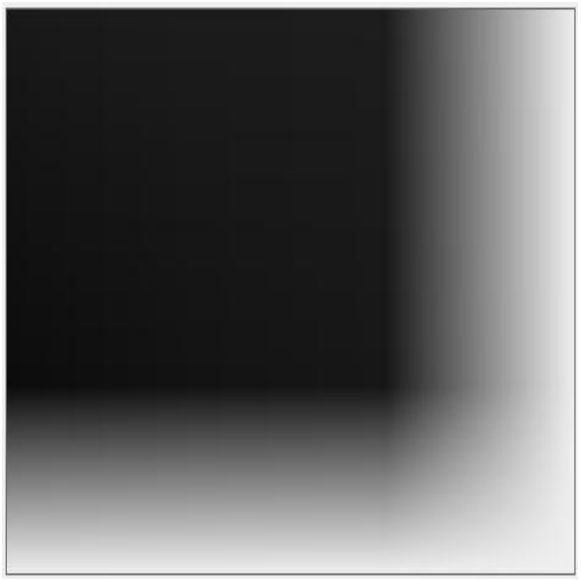
Gambar 3. 34 Contoh citra gambar abu

Gambar 3.34 memiliki ukuran 3x3 *pixel* dengan model warna abu. Untuk nilai *pixel*nya dapat dlihat pada gambar 3.35.

34	30	35
18	20	34
1	10	31

Gambar 3. 35 Nilai *pixel* pada citra gambar abu

Setelah citra keabuan didapatkan maka citra akan diperjernih menggunakan *gaussian blur*. Berikut gambar 3.36 hasil dari proses penjernihan.



Gambar 3. 36 Contoh citra gambar setelah proses penjernihan

Gambar 3.36 memiliki ukuran 3x3 *pixel* dengan nilai *pixel* yang dapat dlihat pada gambar 3.37.

25	28	29
18	23	26
12	19	23

Gambar 3. 37 Nilai *pixel* pada citra gambar hasil penjernihan

Kemudian tiap nilai *pixel* yang didapat akan dihitung menggunakan besaran faktor yang memiliki nilai sebesar 1 dengan rumus sebagai berikut.

Nilai *pixel* dijumlahkan dengan faktor * nilai *pixel* tersebut

Maka penghitungan mulai dilakukan pada nilai pixel yang terdapat pada koordinat awal yaitu koordinat (1, 1) hingga koordinat akhir yaitu (3, 3). Berikut perhitungan pada tiap *pixel*.

Koordinat (1, 1)

12 dijumlahkan dengan 1 * 12

Maka hasilnya sebesar 24

Koordinat (1, 2)

19 dijumlahkan dengan 1 * 19

Maka hasilnya sebesar 38

Koordinat (1, 3)

23 dijumlahkan dengan 1 * 23

Maka hasilnya sebesar 46

Koordinat (2, 1)

18 dijumlahkan dengan 1 * 18

Maka hasilnya sebesar 36

Koordinat (2, 2)

23 dijumlahkan dengan 1 * 23

Maka hasilnya sebesar 46

Koordinat (2, 3)

26 dijumlahkan dengan 1 * 26

Maka hasilnya sebesar 52

Koordinat (3, 1)

25 dijumlahkan dengan 1 * 25

Maka hasilnya sebesar 50

Koordinat (3, 2)

28 dijumlahkan dengan 1 * 28

Maka hasilnya sebesar 56

Koordinat (3, 3)

29 dijumlahkan dengan 1 * 29

Maka hasilnya sebesar 58

Kemudian nilai yang didapatkan sebelumnya akan dihitung kembali pada setiap nilai *pixel* gambar hasil graysacing yang sebelumnya dan disimpan kedalam sebuah *array* dua dimensi dengan ukuran [2, 2] atau sama dengan 3x3 menggunakan rumus berikut

Nilai pada array dijumlahkan dengan nilai *pixel* gambar abu / (nilai sebelumnya + 1)

Berikut perhitungan untuk tiap *array* dengan nilai *default array* yang masih kosong

0 dijumlahkan dengan 1 / (24 +1)

0 dijumlahkan dengan 1 / 25

0 dijumlahkan dengan 0,040

= 0,040

0 dijumlahkan dengan 10 / (38 +1)

0 dijumlahkan dengan 10 / 39

0 dijumlahkan dengan 0,256

= 0,256

0 dijumlahkan dengan 31 / (46 +1)

0 dijumlahkan dengan 31 / 47

0 dijumlahkan dengan 0,659

= 0,659

0 dijumlahkan dengan 18 / (36 +1)

0 dijumlahkan dengan 18 / 37

0 dijumlahkan dengan 0,486

= 0,486

0 dijumlahkan dengan 20 / (46 +1)

0 dijumlahkan dengan 20 / 47

0 dijumlahkan dengan 0,425

= 0,425

0 dijumlahkan dengan 34 / (52 +1)

0 dijumlahkan dengan 34 / 53

0 dijumlahkan dengan 0,641

= 0,641
0 dijumlahkan dengan 34 / (50 +1)
0 dijumlahkan dengan 34 / 51
0 dijumlahkan dengan 0,667
= 0,667
0 dijumlahkan dengan 30 / (56 +1)
0 dijumlahkan dengan 30 / 57
0 dijumlahkan dengan 0,526
= 0,526
0 dijumlahkan dengan 35 / (58 +1)
0 dijumlahkan dengan 35 / 59
0 dijumlahkan dengan 0,593
= 0,593

Sehingga didapatkan nilai minimal *pixel* sebesar 0,040 nilai maksimal *pixel* sebesar 0,667. Kemudian setelah nilai sebelumnya didapat maka akan dilakukan perhitungan untuk menormalisasikan citra gambar menggunakan rumus berikut.
k = 255 / (max – min)
k = 255 / (0,667 – 0,040)
k = 255 / 0,627
maka nilai k adalah sebesar 406,698
setelah nilai k berhasil didapat maka untuk nilai pada tiap *pixel* akan diisi menggunakan rumus berikut.
Nilai *pixel* baru = k * (nilai pada *array* – min)
Sehingga untuk perhitungan berikutnya adalah sebagai berikut.
Koordinat (1, 1) = 406,698 * (0,040 -0,040)
= 406, 698 * 0
= 0
Koordinat (1, 2) = 406,698 * (0,256 -0,040)
= 406, 698 * 0.210
= 85.40658

Koordinat (1, 3) = 406,698 * (0,659 -0,040)

= 406, 698 * 0.619

= 251,746

Koordinat (2, 1) = 406,698 * (0,486 -0,040)

= 406, 698 * 0,448

= 182,200

Koordinat (2, 2) = 406,698 * (0,425 -0,040)

= 406, 698 * 0,385

= 156,578

Koordinat (2, 3) = 406,698 * (0,641 -0,040)

= 406, 698 * 0,601

= 244.425

Koordinat (3, 1) = 406,698 * (0,667 -0,040)

= 406, 698 * 0,627

= 254,999

Koordinat (3, 2) = 406,698 * (0,526 -0,040)

= 406, 698 * 0,486

= 197,655

Koordinat (3, 3) = 406,698 * (0,593 -0,040)

= 406, 698 * 0,553

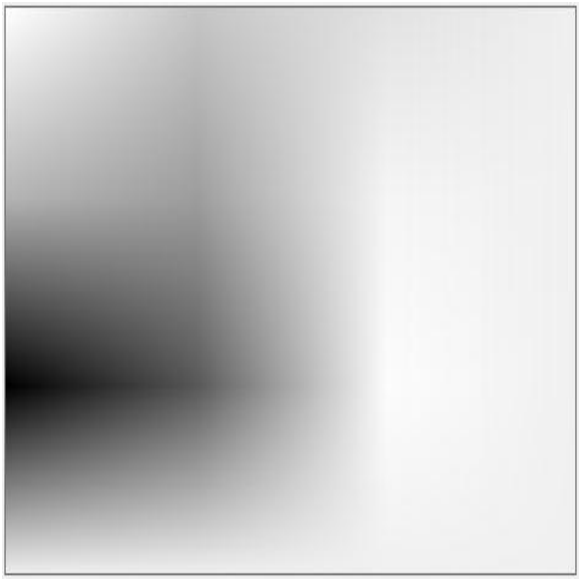
= 224.903

Sehingga nilai *pixel* untuk citra gambar baru akan menjadi seperti pada gambar 3.38 berikut.

255	198	225
182	157	245
0	86	252

Gambar 3. 38 *Pixel* pada citra gambar hasil perhitungan *multiscale retinex*

Dengan demikian gambar hasil akhir untuk metode *multiscale retinex* adalah seperti berikut.



Gambar 3. 39 Citra gambar hasil proses *multiscale retinex*

3.6 Analisis Kebutuhan Perangkat Lunak

Pada tahap ini, menjelaskan analisis spesifikasi dari kebutuhan perangkat lunak yang akan dibangun yaitu meliputi spesifikasi kebutuhan perangkat lunak fungsional (SKPL-F) dan sepesifikasi kebutuhan perangkat lunak non-fungsional (SKPL-NF)

1. Spesifikasi Kebutuhan Fungsional

   Spesifikasi kebutuhan fungsional berisi mengenai fungsi atau layanan apa yang harus disediakan atau bias yang dilakukan oleh sistem, bagaimana sistem berperilaku terhadap suatu kondisi tertentu, dan apa yang harus dilakukan oleh sistem jika terdapat suatu masukan (*input*) tertentu. Detail dari spesifikasi kebutuhan fungsional dapat dlihat pada tabel 3.1.

Tabel 3. 1 Spesifikasi Kebutuhan Fungsional

Nomor	Spesifikasi Kebutuhan Fungsional
SKPL-F-001	Sistem harus dapat menangkap citra gambar
SKPL-F-002	Sistem harus dapat mendeteksi kamera
SKPL-F-003	Sistem harus dapat menampilkan citra gambar uji
SKPL-F-004	Sistem harus dapat menormalisasikan citra

SKPL-F-005	Sistem harus dapat mendeteksi wajah
SKPL-F-006	Sistem harus dapat memisahkan objek wajah pada citra
SKPL-F-007	Sistem harus dapat menampilkan wajah yang terdeteksi
SKPL-F-008	Sistem harus dapat menyimpan wajah yang terdeteksi
SKPL-F-009	Sistem harus dapat mengenali wajah yang terdeteksi

2. Spesifikasi Kebutuhan Non-Fungsional

   Spesifikasi kebutuhan non-fungsional merupakan penjelasan mengenai batasan-batasan fungsi dari sistem yang akan dibangun. Detail mengenai spesifikasi kebutuhan non-fungsional dapat dilihat pada tabel 3.2.

Tabel 3. 2 Spesifikasi Kebutuhan Non-Fungsional

Nomor	Spesifikasi Kebutuhan Non-Fungsional
SKPL-NF-001	Sistem memerlukan masukan berupa data gambar/foto atau dari kamera webcam dengan resolusi yang cukup baik
SKPL-NF-002	Sistem memerlukan sudut gambar wajah yang cukup lurus kedepan untuk pendeteksian wajah

3.7 Analisis Kebutuhan Non-Fungsional

Analisis kebutuhan non-fungsional dilakukan untuk mengetahui spesifikasi kebtuhan untuk aplikasi. Spesifikasi kebutuhan melibatkan analisis pengguna, perangkat keras, dan analisis perangkat lunak. Adapun penjelasan analisis kebutuhan non-fungsional adalah sebagai berikut :

1. Analisis Pengguna

   Aplikasi ini dapat digunakan dengan adanya interaksi antara pengguna dengan antarmuka (*interface*) pada sistem. Sehingga karakteristik pengguna untuk menggunakan aplikasi ini adalah sebagai berikut :

   a. Pengguna harus memiliki kemampuan mengoperasikan perangkat komputer.
   b. Pengguna tidak memiliki kelainan pada mata.

2. Analisis Perangkat Keras

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

Tabel 3. 3 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 GB

3. Analisis Perangkat Lunak

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

   Tabel 3. 4 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

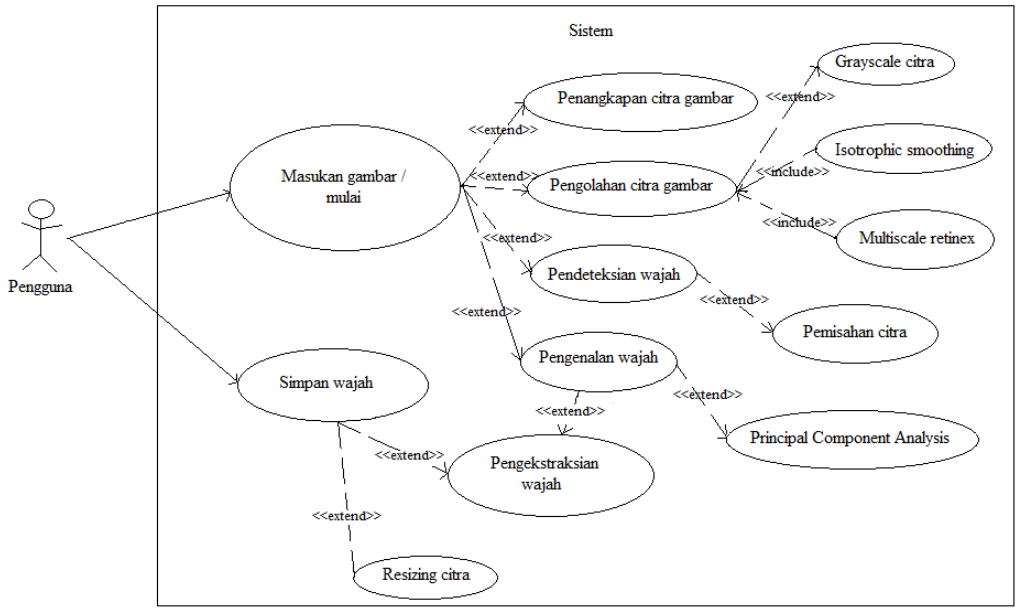
3.8 Analisis Kebutuhan Fungsional

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.

3.8.1 Use Case Diagram

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 wajah yang digambarkan dengan *use case diagram* :



Gambar 3. 40 *Use Case diagram*

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

1. Definisi Actor

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

2. Definisi Use Case

Use case berfungsi untuk mewakili apa yang sistem bisa lakukan. Definisi *use case* bisa dilihat pada tabel 3.5.

Tabel 3. 5 Definisi *use case*

No	*Use Case*	Deskripsi
1	Masukan gambar / mulai	Proses untuk memulai pengenalan objek wajah pada citra gambar masukan
2	Simpan wajah	Proses untuk menyimpan objek wajah yang terdeteksi dari citra gambar masukan menjadi citra latih

3	Penangkapan citra gambar	Proses untuk menangkap citra gambar masukan pada sistem
4	Pengolahan citra gambar	Proses untuk mengolah citra gambar menjadi citra gambar yang baru
5	Pendeteksian wajah	Proses untuk mendeteksi objek wajah pada citra gambar masukan
6	Pengenalan wajah	Proses untuk mengenali objek wajah yang terdeteksi dengan menggunakan metode pengenalan yang telah ditentukan
7	Pengekstraksian wajah	Proses pengekstraksian objek wajah pada citra uji
8	Resizing citra	Proses untuk merubah ukuran citra gambar kedalam kuran yang telah ditentukan
9	Grayscale citra	Proses untuk merubah citra warna RGB kedalam derajat keabuan
10	Isotrophic smoothing	Metode untuk proses smoothing terhadap citra gambar
11	Multiscale retinex	Metode untuk memproses citra gambar agar pencahayaannya lebih terang
12	Pemisahan citra	Proses untuk memisahkan objek wajah yang terkandung pada citra gambar
13	Principal Component Analysis	Metode untuk membandingkan wajah dari citra uji dengan wajah dari citra latih

3. Skenario Use Case

Skenario *use case* merupakan hasil instansiasi dari setiap *use case*. Berikut adalah skenario yang telah didefinisikan sebelumnya :

Nama *use case* : Masukan gambar / mulai

Skenario :

Tabel 3. 6 Skenario *use case* masukan gambar / mulai

Identifikasi		
No	1	
Nama	Masukan gambar / mulai	
Tujuan	Memulai pengenalan wajah	
Deskripsi	Proses untuk memulai proses pendeteksian dan pengenalan wajah	
Aktor	Pengguna	
Skenario Utama		
Kondisi Awal	Pengguna berada pada tampilan utama sistem	
Aksi Aktor		Reaksi Sistem
1. Pengguna memilih jenis masukan citra untuk memulai pengenalan wajah		
		2. Menampilkan citra masukan pada *image box*
		3. Menandai wajah yang tedeteksi
		4. Mengenal wajah yang terdeteksi
		5. Tampilkan nama pada wajah yang telah dikenali
Kondisi akhir		Objek wajah telah dikenali sebagai wajah dari seseorang

Nama *use case* : Simpan wajah

Skenario :

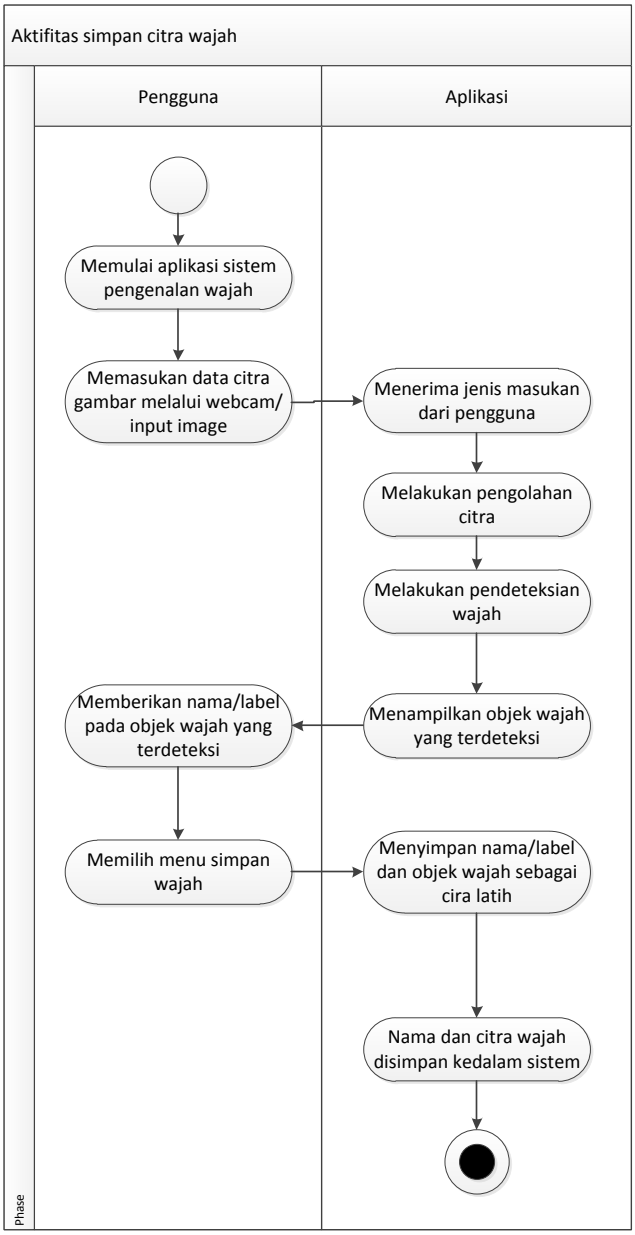
Tabel 3. 7 Skenario *use case* simpan wajah

Identifikasi	
No	2
Nama	Simpan wajah
Tujuan	Menyimpan objek wajah sebagai citra latih
Deskripsi	Proses untuk memberi nama terhadap objek wajah

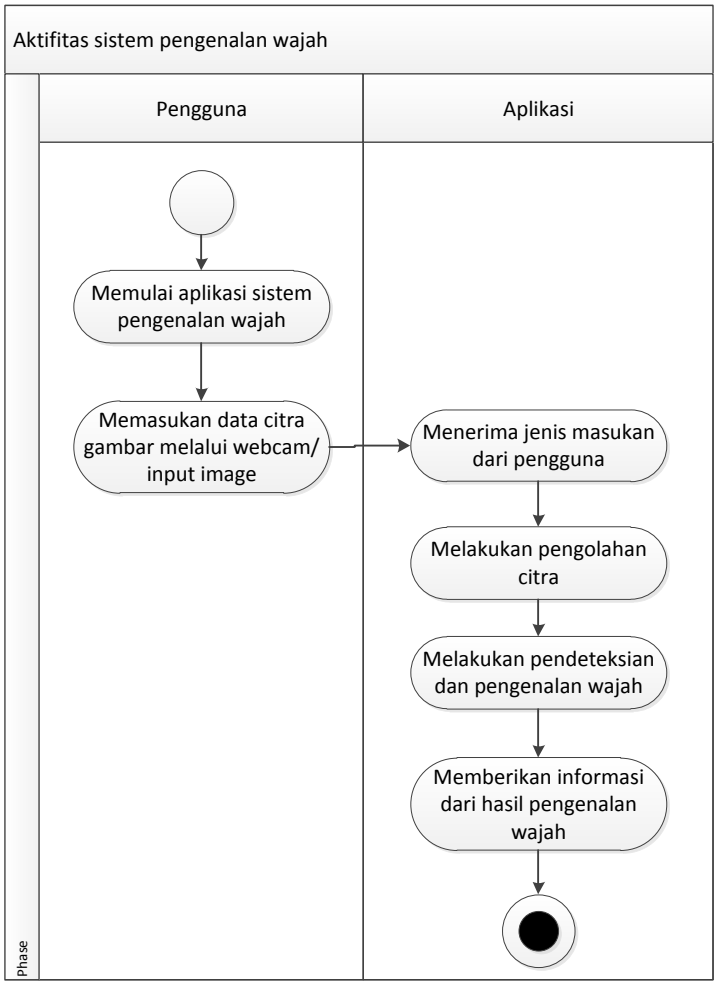
	kemudian menyimpannya
Aktor	Pengguna
Skenario Utama	
Kondisi Awal	Player berada di tampilan utama sistem
Aksi Aktor	Reaksi Sistem
1. Pengguna memilih jenis masukan citra untuk memulai pengenalan wajah	
	2. Menampilkan citra masukan pada *image box*
	3. Menandai wajah yang tedeteksi
	4. Menampilkan wajah yang terdeteksi pada *image box extracted face*
5. Pengguna memberi nama pada wajah lalu menekan tombol *add face*	
	6. Menyimpan nama dan objek wajah
Kondisi akhir	Wajah dan nama berhasil disimpan

3.8.2 Activity Diagram

Activity diagram menggambarkan berbagai alur aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alur berawal dan bagaimana mereka berakhir. *Acitivity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. Berikut gambar *activity diagram* untuk aktivitas simpan wajah dan pengenalan wajah pada sistem :



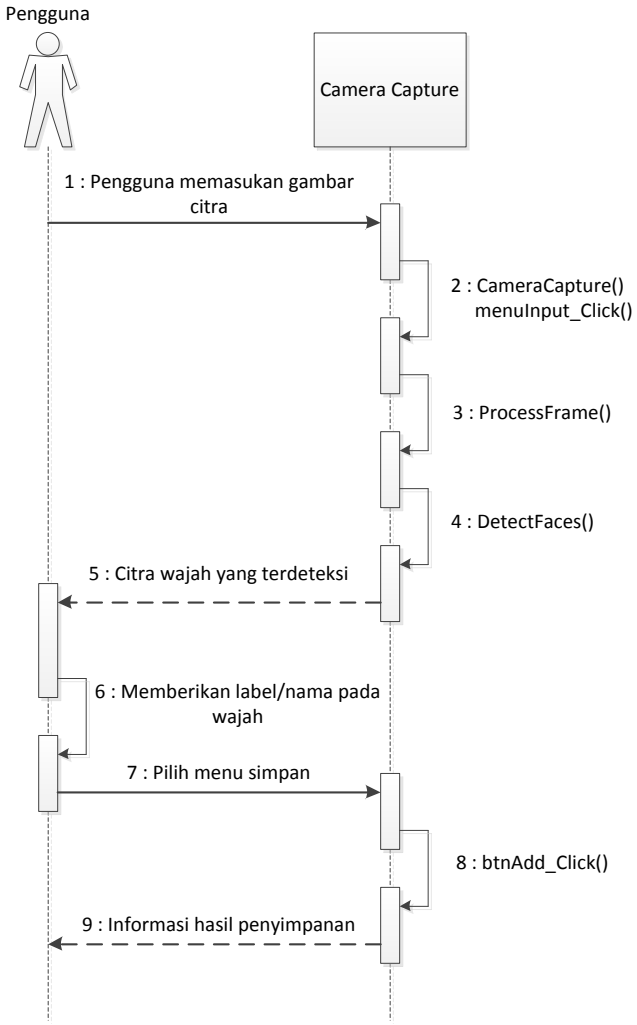
Gambar 3. 41 *Activity diagram* simpan citra wajah



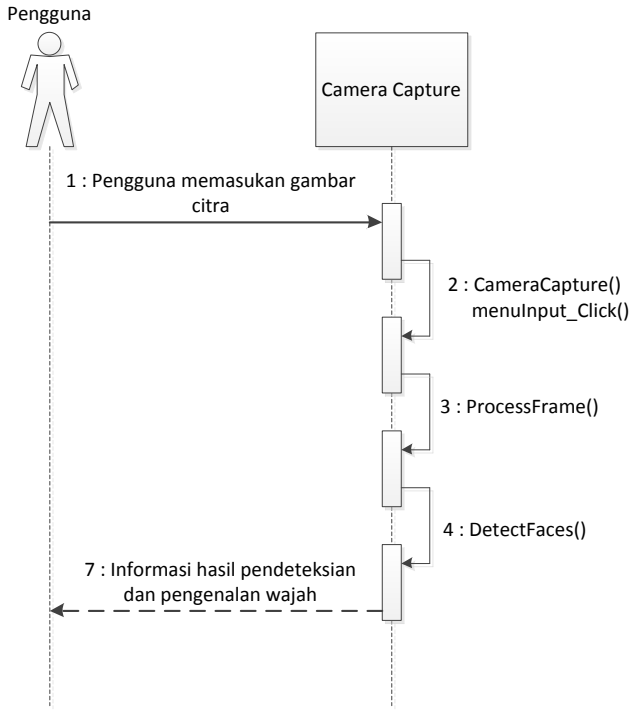
Gambar 3. 42 *Activity diagram* pengenalan wajah

3.8.3 Sequence Diagram

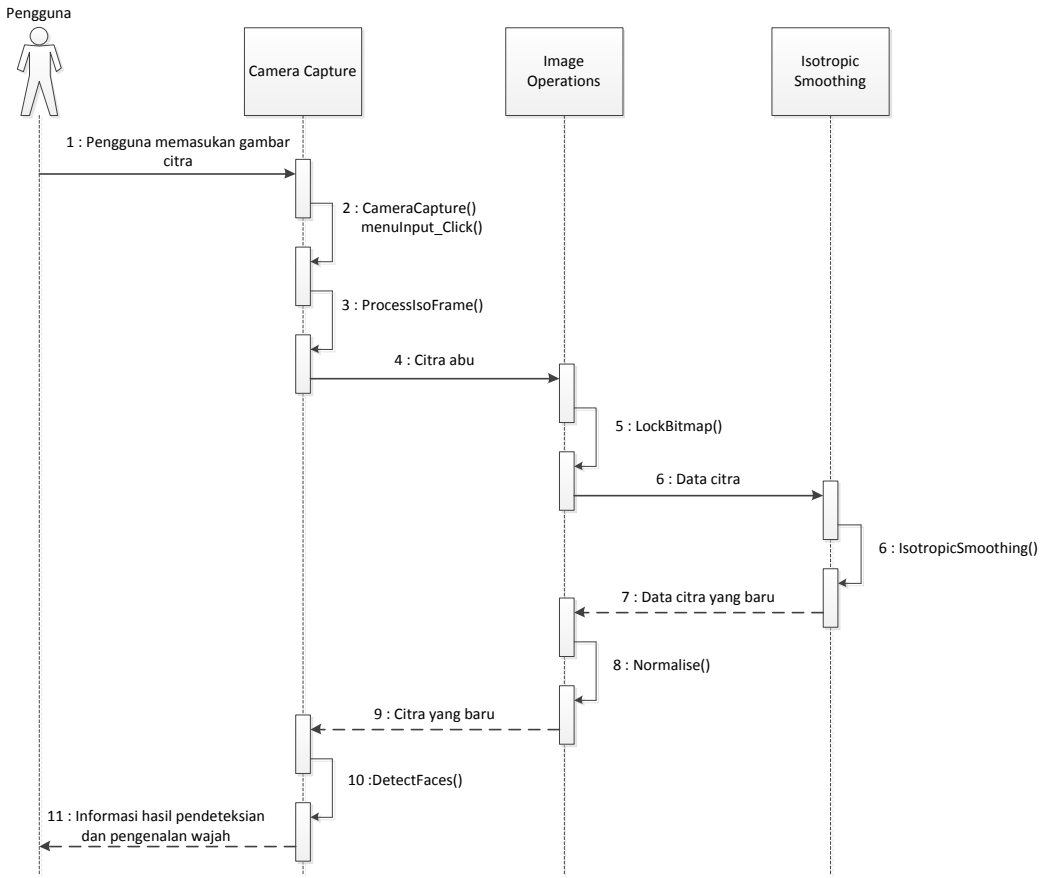
Sequence diagram menggambarkan interaksi antar objek di dalam dan disekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa informasi yang digambarkan terhadap waktu. *Sequence diagram* terdiri atas dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait). Berikut *sequence diagram* dari proses simpan data wajah, pengenalan wajah, *isotrpic smoothing* dan *multiscale retinex* :



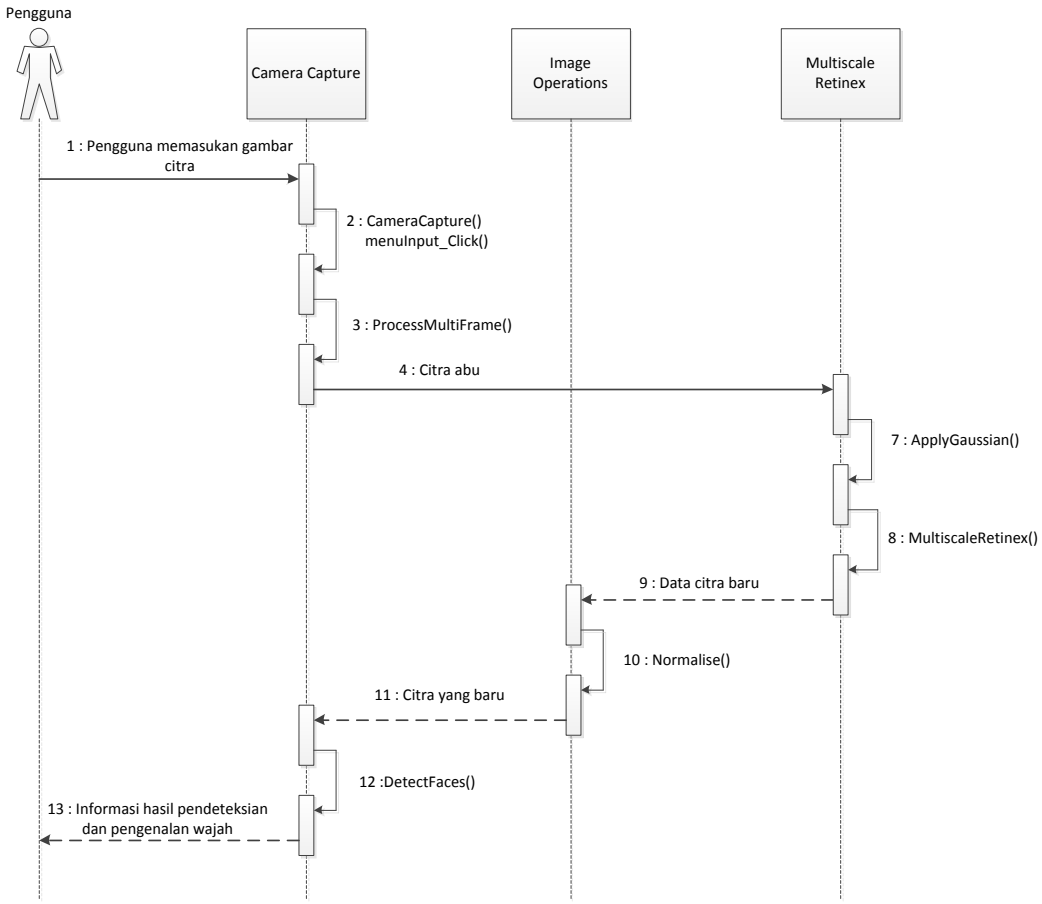
Gambar 3. 43 *Sequence diagram* proses simpan data wajah



Gambar 3. 44 *Sequence diagram* proses pengenalan wajah



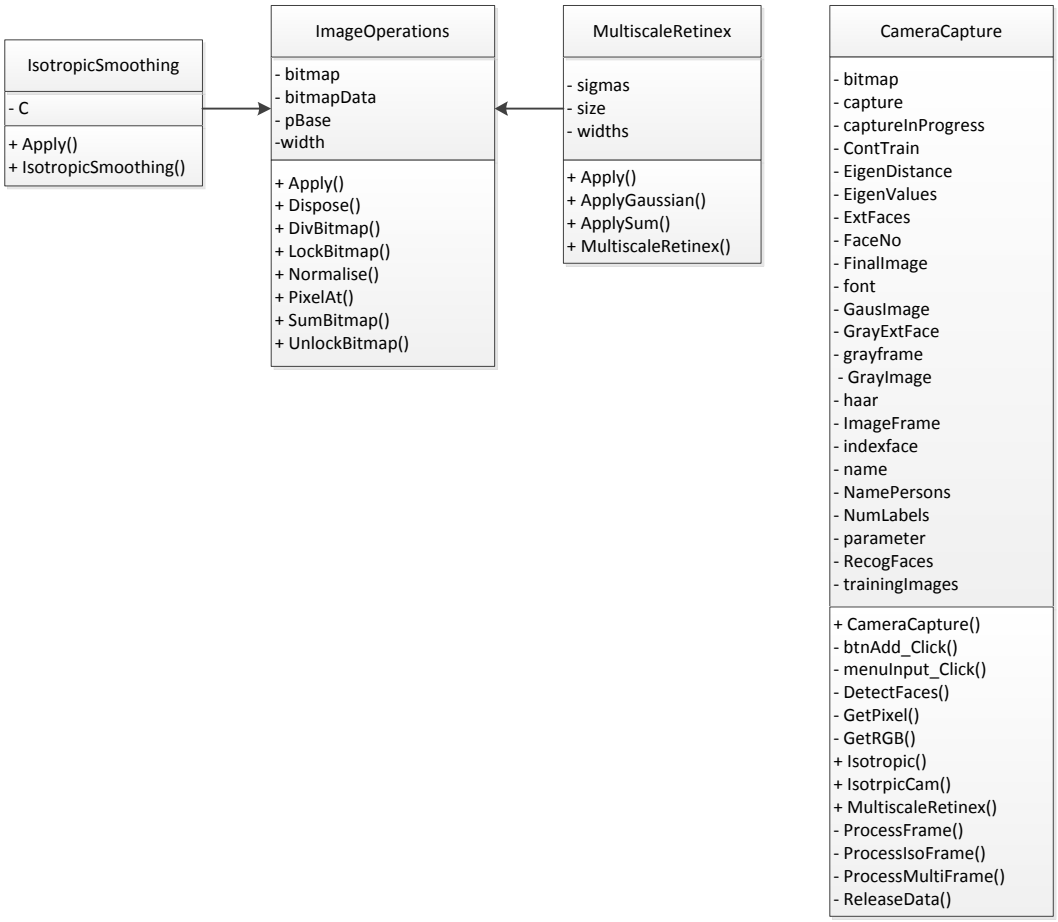
Gambar 3. 45 *Sequence diagram* proses *isotropic smoothing*



Gambar 3. 46 *Sequence diagram* proses *multiscale retinex*

3.8.4 Class Diagram

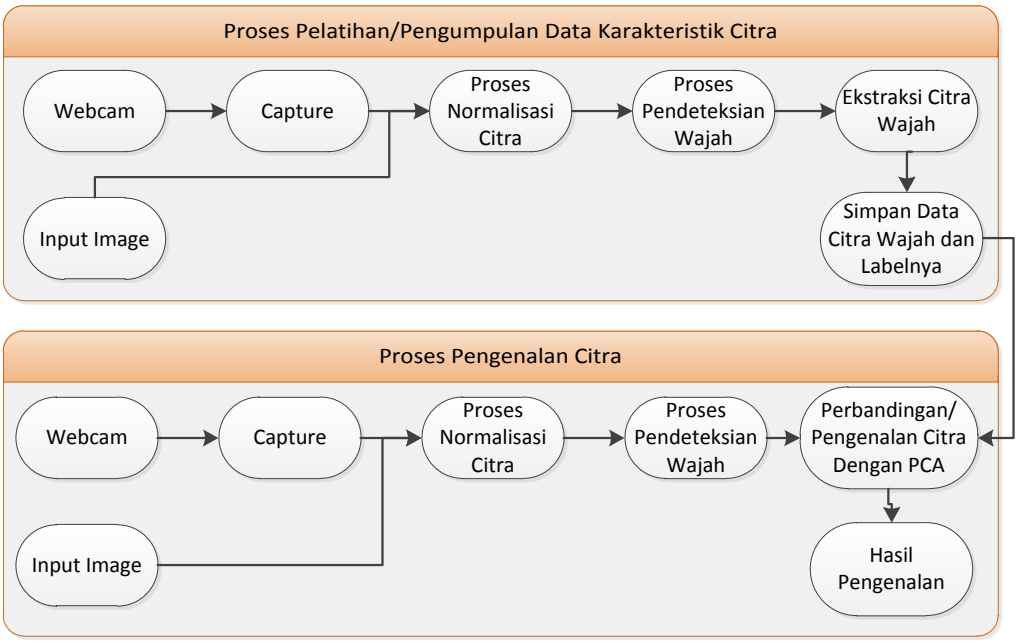
Class diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi, berikut *class diagram* sistem :



Gambar 3. 47 *Class diagram*

3.8.5 Diagram Blok

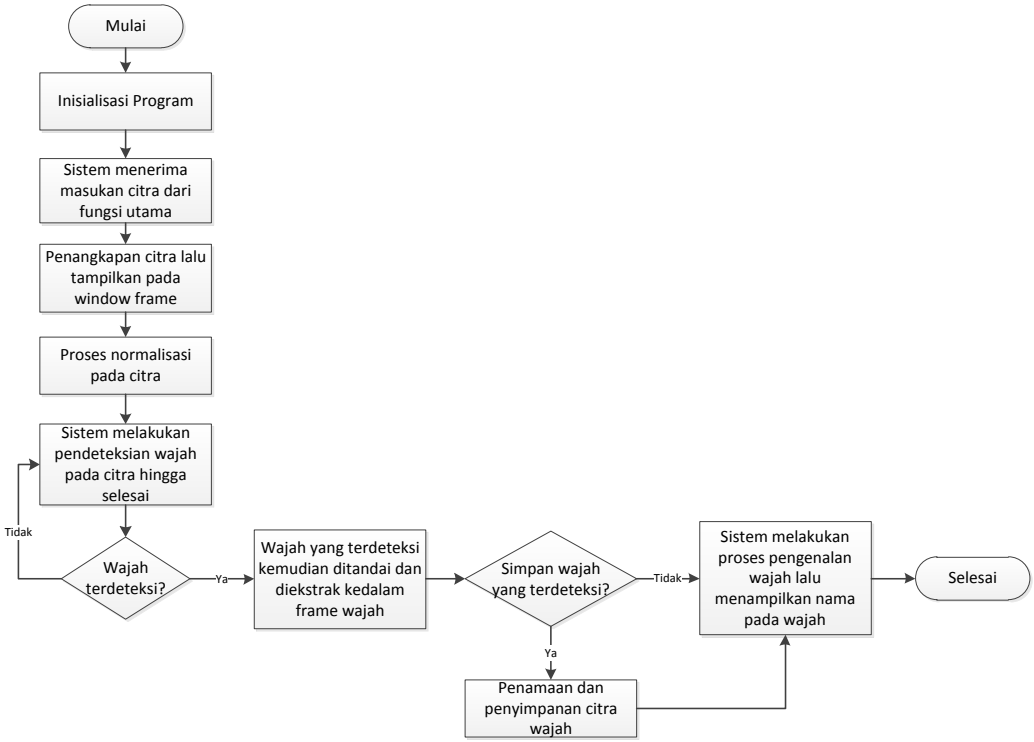
Sistem ini terdiri dari dua blok utama yaitu blok proses pelatihan dan proses pengenalan. Untuk blok proses pelatihan terdiri dari *webcam*, *input image* dan *software* yang akan memproses citra masukan kedalam suatu ciri pada citra yang memiliki karakteristik tertentu dan kemudian hasil ciri tersebut akan disimpan. Sedangkan untuk blok proses pengenalan terdiri dari *webcam*, *input image* dan *software* yang akan memproses citra masukan kedalam suatu ciri berkarakteristik citra lalu akan dibandingkan dengan karakteristik citra yang telah disimpan sebelumnya sehingga akan memberikan keluaran berupa hasil dari pengenalan citra. Berikut adalah diagram blok secara keseluruhan :



Gambar 3. 48 Diagram blok keseluruhan sistem

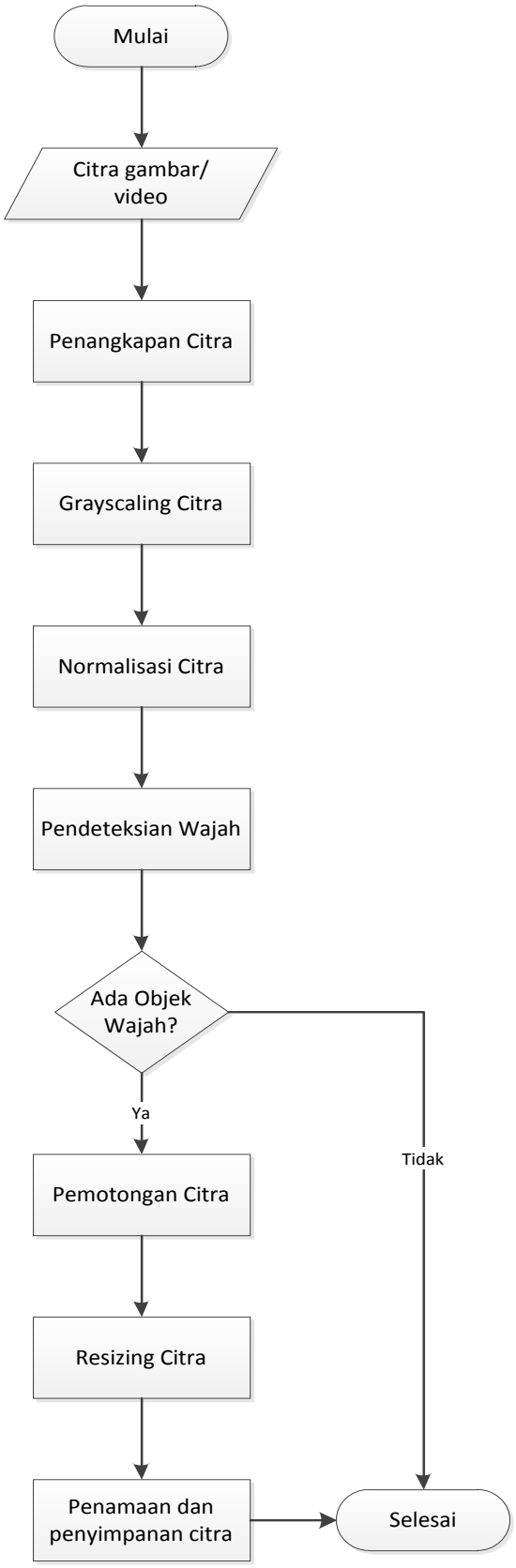
3.8.6 Diagram Alir

Berikut ini adalah diagram alir yang akan digunakan dalam membangun sistem serta penerapan algoritma program untuk mendukung sistem pengenalan wajah :



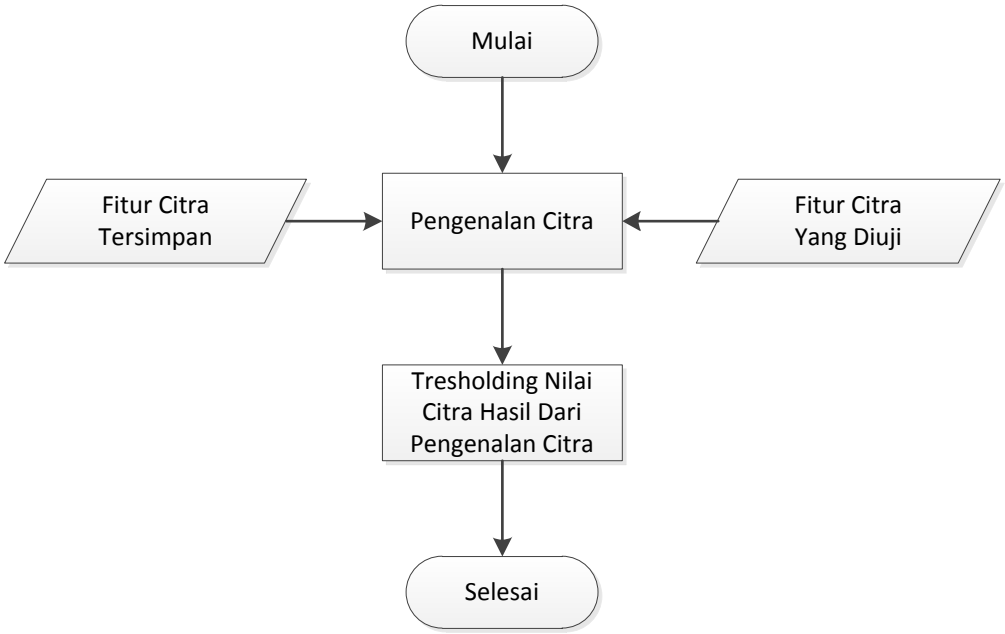
Gambar 3. 49 Diagram alir keseluruhan sistem

Gambar 3.49 menunjukan diagram alir dari keseluruhan sistem yang dirancang. Dimana pada sistem pengenalan wajah ini diawali dengan inisialisasi yang dibutuhkan oleh program dan diakhiri dengan hasil dari pengenalan wajah. Program akan menerima citra gambar/*video* yang selanjutnya citra tersebut akan dinormalisasikan terlebih dahulu dengan metode yang telah ditentukan sebelum dilakukannya proses pendeteksian wajah. Setelah proses pendeteksian wajah maka program akan menampilkan wajah-wajah yang telah terdeteksi kedalam suatu *frame* khusus, dimana wajah yang terdeteksi tersebut akan dilakukan proses pengenalan atau juga wajah tersebut dapat diberi nama kemudian disimpan. Program ini akan selesai apabila tombol untuk menutup program ditekan.


Mulai
Citra gambar/ video
Penangkapan Citra
Grayscaling Citra
Normalisasi Citra
Pendeteksian Wajah
Ada Objek Wajah?
Ya
Tidak
Pemotongan Citra
Resizing Citra
Penamaan dan penyimpanan citra
Selesai

Gambar 3. 50 Diagram alir proses simpan objek citra wajah

Gambar 3.50 menggambarkan proses penyimpanan citra wajah, dimana aliran ini menunjukan bagaimana masukan citra ditangkap, kemudian diselaraskan dan dideteksi objek wajah yang tekandung didalamnya, sampai citra wajah disimpan kedalam sebuah file.



Gambar 3. 51 Diagram alir proses pengenalan citra

Gambar 3.51 menjelaskan dimana proses pengenalan terjadi. Proses ini akan membandingkan kemiripan dari dua citra, yaitu citra yang telah disimpan di *database* atau *file external* dengan citra yang diujikan kedalam program. Proses ini nantinya akan menghasilkan suatu nilai yaitu nilai kemiripan. Nilai kemiripan ini nantinya akan melalui proses *tresholding*, jika nilai kemiripan berada didalam nilai *treshold*, maka citra yang sedang diujikan tersebut akan dikenali sebagai suatu citra yang memiliki nilai didalam *treshold* tersebut.

3.9 Perancangan Sistem

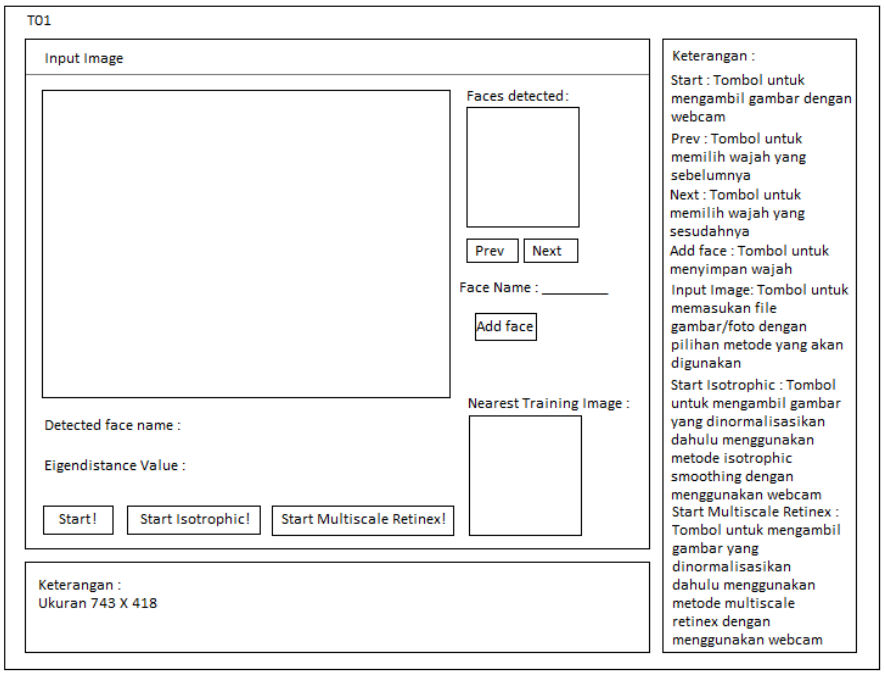
Perancangan sistem bertujuan untuk memahami rancangan dari sistem yang akan dibangun misalnya seperti rancangan pada antarmuka sistem.

3.9.1 Antarmuka Sistem

Perancangan antarmuka bertujuan untuk memberikan gambaran tentang aplikasi yang akan dibangun, sehingga akan mempermudah dalam pembuatan aplikasi. Antarmuka sistem yang dirancang menggunakan antarmuka yang berbasis *windows form application* yang terdiri dari :

1. Tombol *input Image* untuk memasukan sebuah *file* gambar/foto
2. Beberapa tombol *start* untuk memulai proses pengambilan gambar melalui *webcam*
3. Tombol *pause/stop* untuk menghentikan proses pengambilan gambar melalui *webcam* yang terdapat pada tombol *start* setelah proses pengambilan gambar dimulai
4. Tiga buah *image box*, dimana yang satunya untuk menampilkan citra yang telah dimasukan dan diproses dengan ukuran 470 x 299 *pixels*. Sedangkan dua lagi berguna untuk menangkap citra wajah yang terdeteksi dan menampilkan citra latih termirip dengan ukuran 110 x 90 *pixels*
5. Beberapa tombol navigasi yang berfungsi untuk menseleksi citra wajah dari beberapa citra wajah yang terdeteksi oleh sistem
6. Serta tombol add face untuk menyimpan gambar wajah

Berikut contoh gambar dari antarmuka sistem :

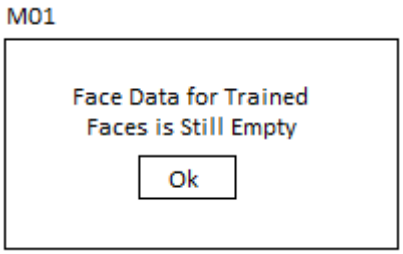


Gambar 3. 52 Antarmuka sistem

3.9.2 Perancangan Pesan

Pada perancangan pesan ini menjelaskan mengenai informasi yang akan diberiktahukan kepada pengguna melalui sebuah pesan yang akan ditampilkan oleh sistem. Berikut adalah pesan yang terdapat pada sistem :

1. Pesan pemberitahuan bahwa citra latih masih belum ada pada sistem



Gambar 3. 53 Pesan keterangan citra latih

Gambar 3.53 menjelaskan bahwa citra latih pada sistem masih belum ada atau kosong.

2. Pesan pemberitahuan bahwa citra wajah tersebut adalah citra wajah paling awal



Gambar 3. 54 Pesan keterangan posisi objek wajah pertama

Gambar 3.54 menjelaskan bahwa objek citra wajah yang ditampilkan adalah wajah pertama yang terkandung pada citra masukan.

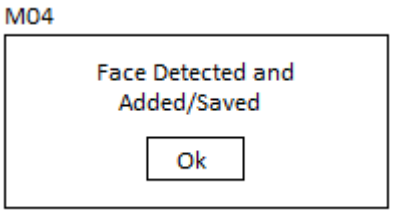
3. Pesan pemberitahuan bahwa citra wajah tersebut adalah cira wajah terakhir



Gambar 3. 55 Pesan keterangan posisi objek wajah terakhir

Gambar 3.55 menjelaskan bahwa objek citra wajah yang ditampilkan adalah wajah paling akhir yang terkandung pada citra masukan.

4. Pesan pemberitahuan bahwa citra wajah berhasil tersimpan pada sistem



Gambar 3. 56 Pesan keterangan simpan citra wajah

Gambar 3.56 menjelaskan bahwa objek citra wajah yang ditampilkan telah berhasil tersimpan menjadi citra laih.