BAB IV
IMPLEMENTASI DAN PENGUJIAN

IV.1 Implentasi

Implementasi menjelaskan pengimplementasian dari sistem yang telah dibuat. Implementasi yang dijelaskan berupa implementasi perangkat keras, implementasi perangkat lunak, implementasi basis data dan implementasi proses.

IV.1.1 Implementasi Perangkat Keras

Implementasi perangkat keras menjelaskan perangkat keras yang digunakan untuk mengimplementasikan sistem yang telah dibuat. Berikut ini merupakan perangkat keras yang digunakan dalam implementasi sistem ini:

1. Processor : 2.27 GHz
2. Memory : 2 GB
3. VGA : 512 MB
4. Hardisk : 320 GB
5. Monitor : 14 inch
6. Mouse dan Keyboard

IV.1.2 Implementasi Perangkan Lunak

Implementasi perangkat lunak menjelaskan perangkat lunak yang digunakan dalam implementasi sistem ini. Perangkat lunak yang digunakan adalah sebagai berikut:

1. Sistem Operasi Windows 8.1
2. Microsoft Visual Studio 2010
3. XAMPP MySql

IV.1.3 Implementasi Basis Data

Implementasi basis data menjelaskan implementasi dari basis data yang dibuat pada sistem ini, meliputi implementasi proses pembuatan *database* dan *table*. Berikut merupakan implementasi basis data pada Tabel IV-1:

Tabel IV-1 Implementasi Basis Data

Pembuatana *database*	`CREATE DATABASE 'sistem_pemilihan_lagu';`
Pembuatan *table* bobot akhir	`CREATE TABLE `sistempemilihanlagu`.`bobotakhir` ( `kelas` INT NOT NULL , `W1` DOUBLE NOT NULL , `W2` DOUBLE NOT NULL , `W3` DOUBLE NOT NULL , `W4` DOUBLE NOT NULL , `W5` DOUBLE NOT NULL , `W6` DOUBLE NOT NULL , `rasio_pembelajaran` FLOAT NOT NULL , PRIMARY KEY ( `kelas` ) ) ENGINE = INNODB`
Pembuatan *table* data *training*	`CREATE TABLE `sistempemilihanlagu`.`tabel_data_training` ( `id` INT NOT NULL AUTO_INCREMENT PRIMARY KEY , `F1` FLOAT NOT NULL , `F2` FLOAT NOT NULL , `F3` FLOAT NOT NULL , `F4` FLOAT NOT NULL , `F5` FLOAT NOT NULL , `F6` FLOAT NOT NULL , `kelas` INT NOT NULL ,`

	INDEX (`kelas`)) ENGINE = INNODB; ALTER TABLE `tabel_data_training` ADD FOREIGN KEY (`kelas`) REFERENCES `sistempemilihanlagu`.`bobotakhir` (`kelas`);
Pembuatan *table* Data *Playlist*	CREATE TABLE `sistempemilihanlagu`.`lagu` (`id ` INT NOT NULL AUTO_INCREMENT PRIMARY KEY , `link_lagu` VARCHAR NOT NULL , `kelas` INT NOT NULL , INDEX (`kelas`)) ENGINE = INNODB; ALTER TABLE `playlist` ADD FOREIGN KEY (`kelas`) REFERENCES `sistempemilihanlagu`.`bobotakhir`(`Kelas`) ON DELETE RESTRICT ON UPDATE RESTRICT

IV.1.4 Implementasi Antarmuka

Implementasi antarmuka menjelaskan dan menggambarkan proses dari setiap antarmuka yang ada di dalam sistem ini.

IV.1.4.1 Tampilan Utama

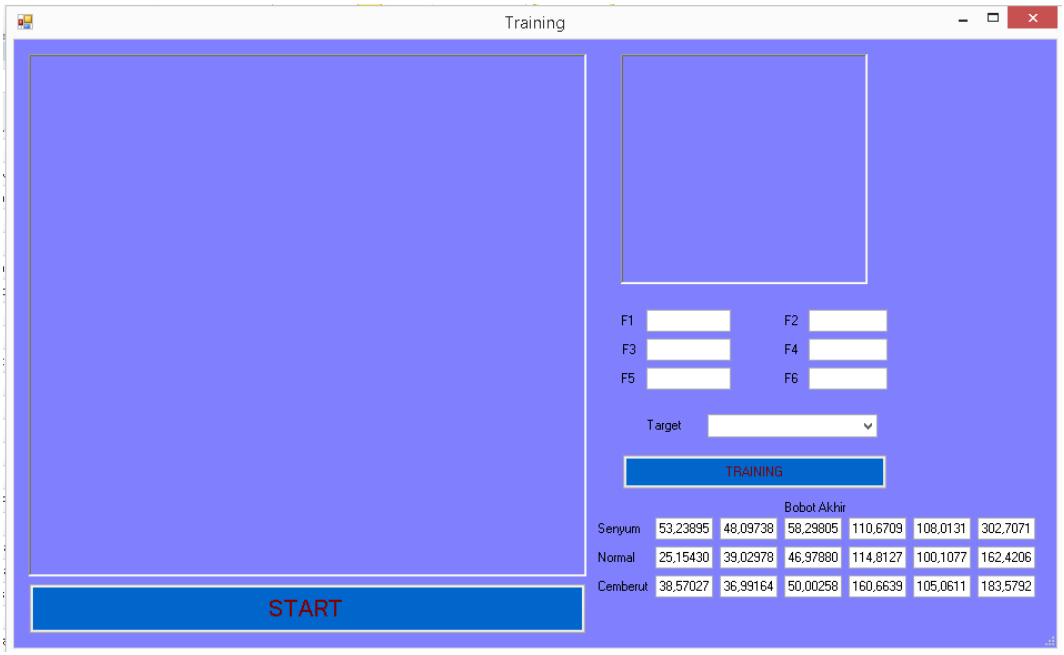
Tampilan utama merupakan tampilan yang pertama kali muncul ketika aplikasi dijalankan. Pada Gambar IV.1 menunjukan gambar dari tampilan utama.



Gambar IV.1 Tampilan Utama

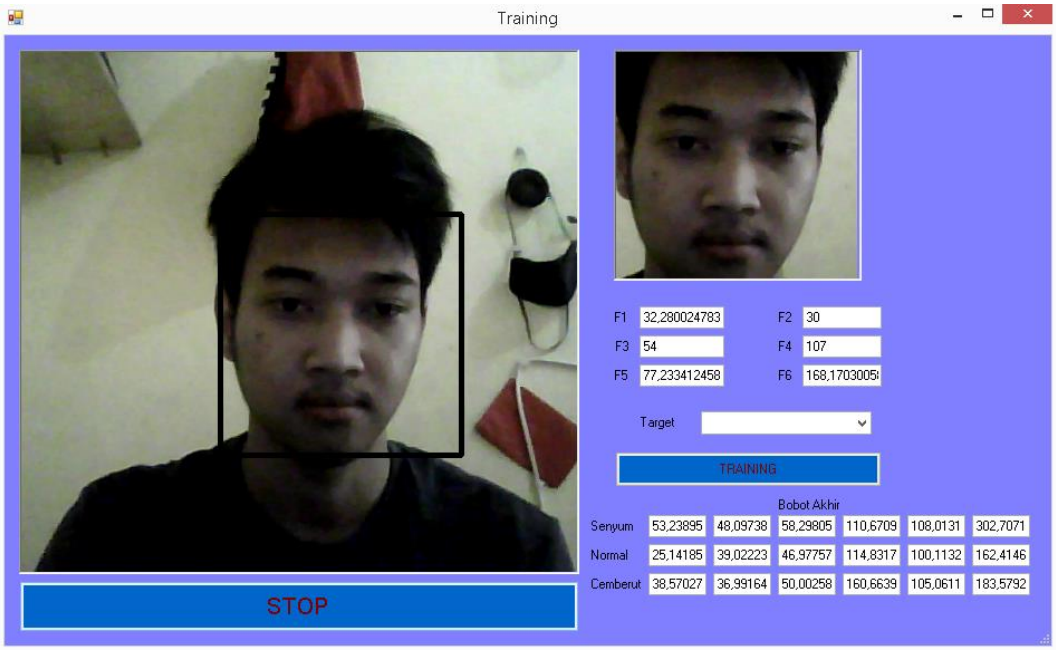
IV.1.4.2 *Training*

Training merupakan proses menambahkan data training dan proses belajar untuk *learning vector quantization.* Tampilan awal *training* digambarkan pada Gambar IV.2.



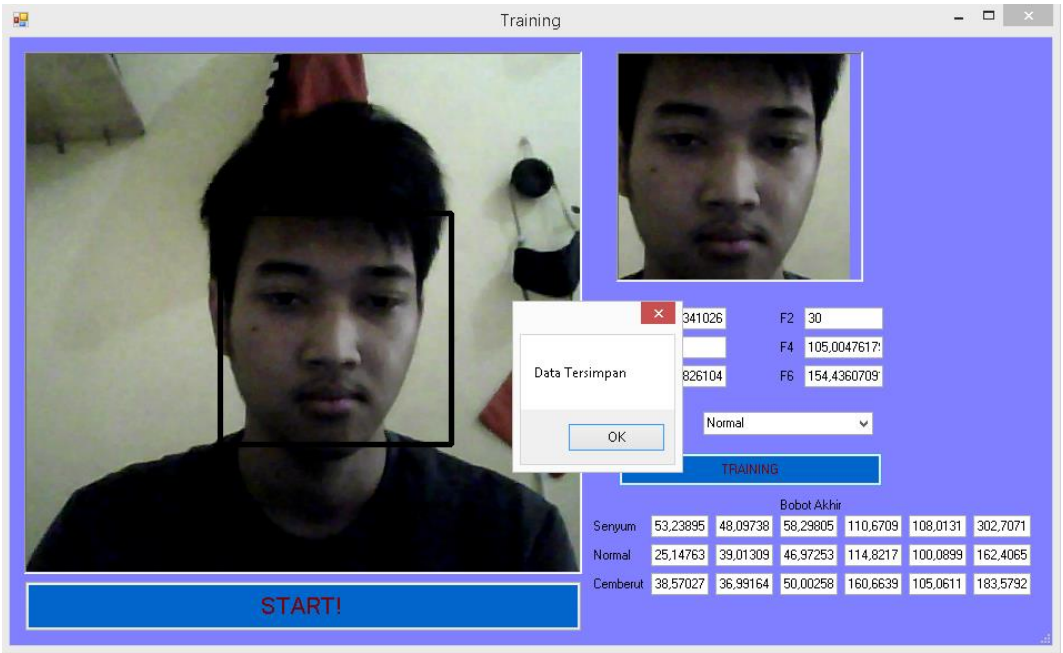
Gambar IV.2 Tampilan Awal *Training*

Untuk proses pengambilan data *training* dapat dilihat pada Gambar IV.3.



Gambar IV.3 Pengambilan Data *Training*

Dan pada Gambar IV.4 menggambarkan tentang proses *training* dan menyimpan data hasil *training* berhasil.



Gambar IV.4 Proses *Training* Berhasil

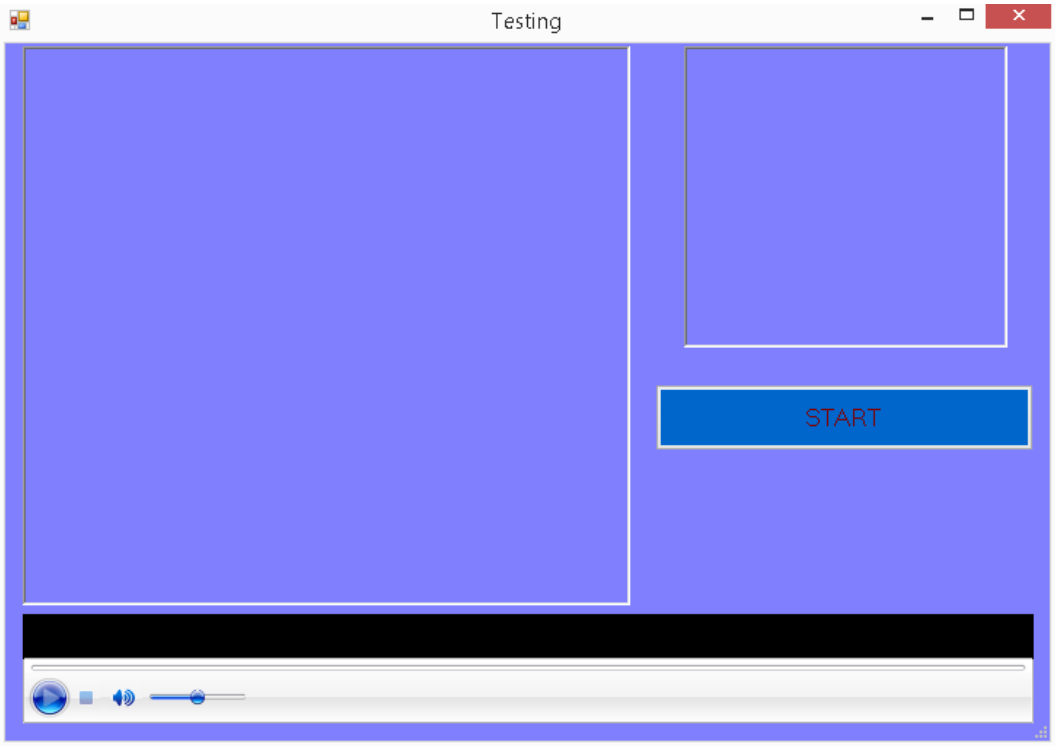
Dan jika target belum dipilih maka akan muncul pesan *error* seperti yang terlihat pada Gambar IV.5.



Gambar IV.5 Target Belum Dipilih

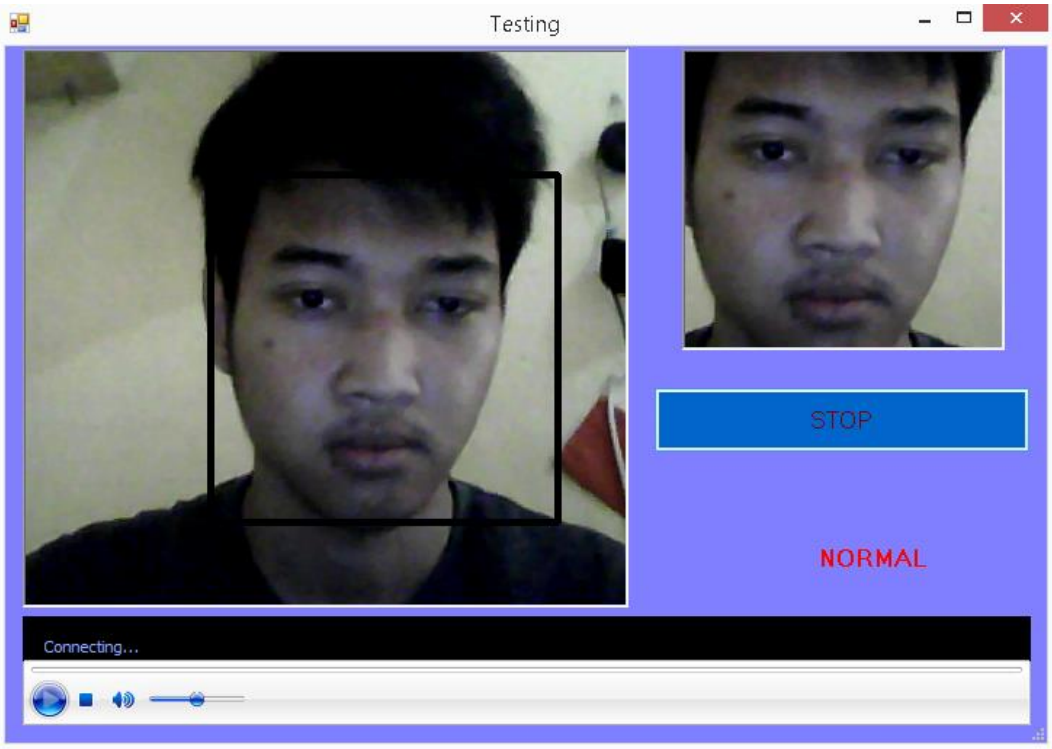
IV.1.4.3 *Testing*

Testing merupakan proses menguji hasil pembelajaran *learning vector quantization* dan menjalankan musik. Untuk gambaran tampilan awal *testing* digambarkan pada Gambar IV.6.



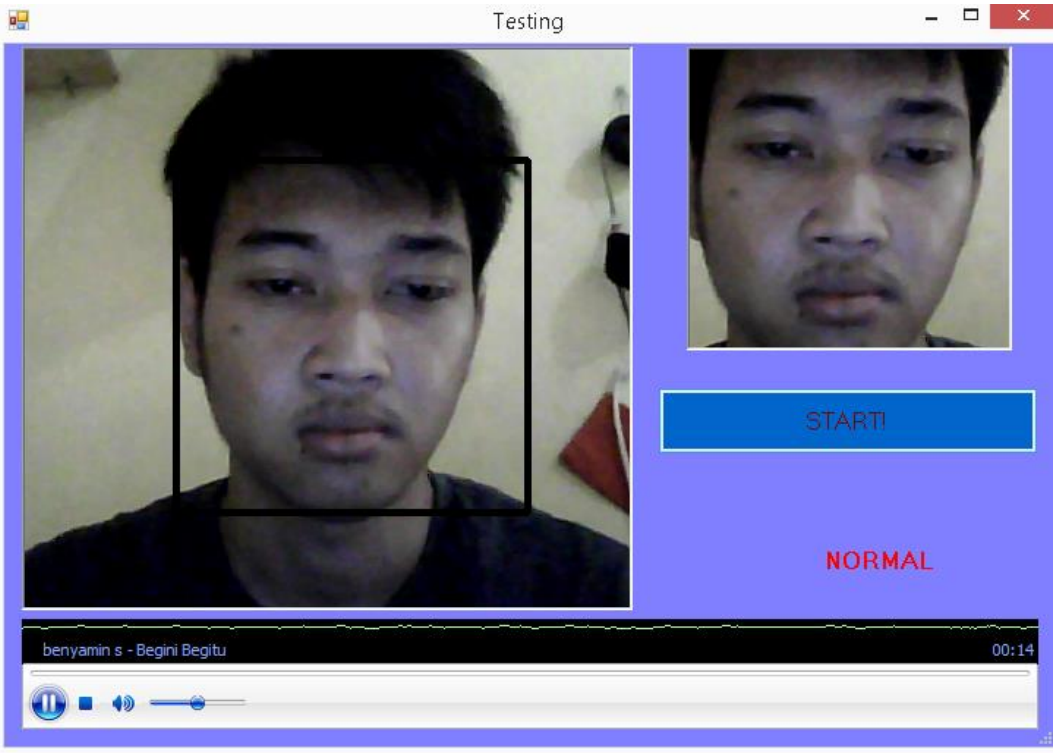
Gambar IV.6 Tampilan Awal Testing

Dan untuk proses pengambilan gambar wajah dapat dilihat pada Gambar IV.7.



Gambar IV.7 Pengambilan Gambar Wajah

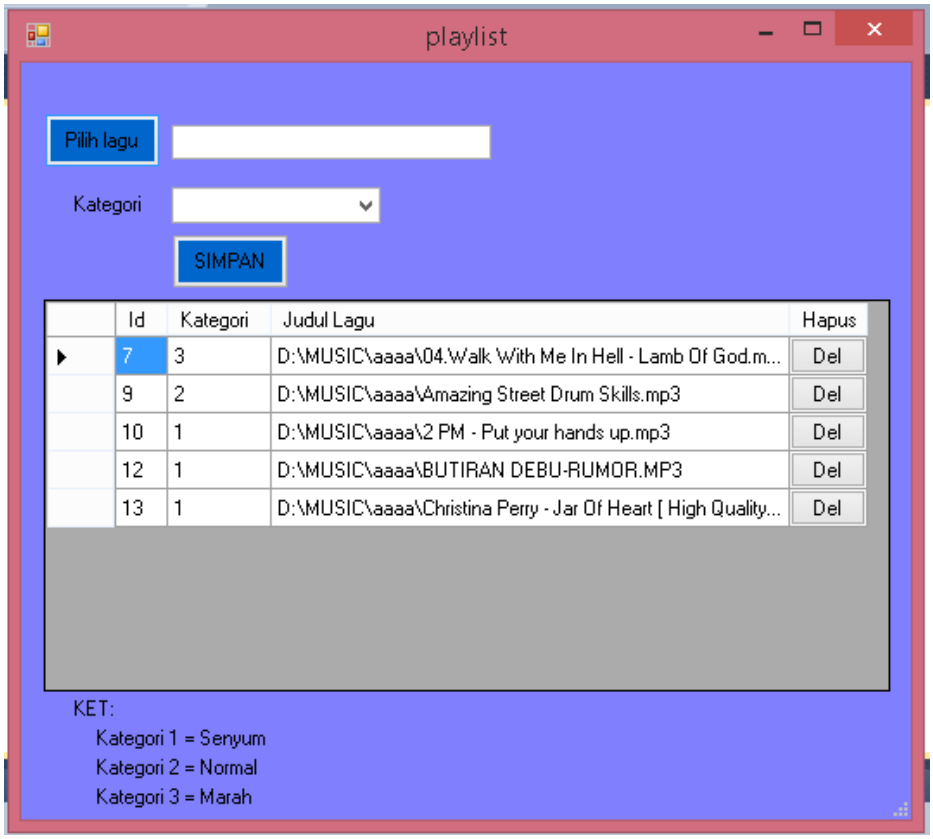
Jika wajah sudah dideteksi dan menekan tombol “STOP” maka proses selanjutnya menjalankan musik sesuai ekspresi pada wajah tersebut, seperti yang terlihat pada Gambar IV.8.



Gambar IV.8 Proses Menjalankan Musik

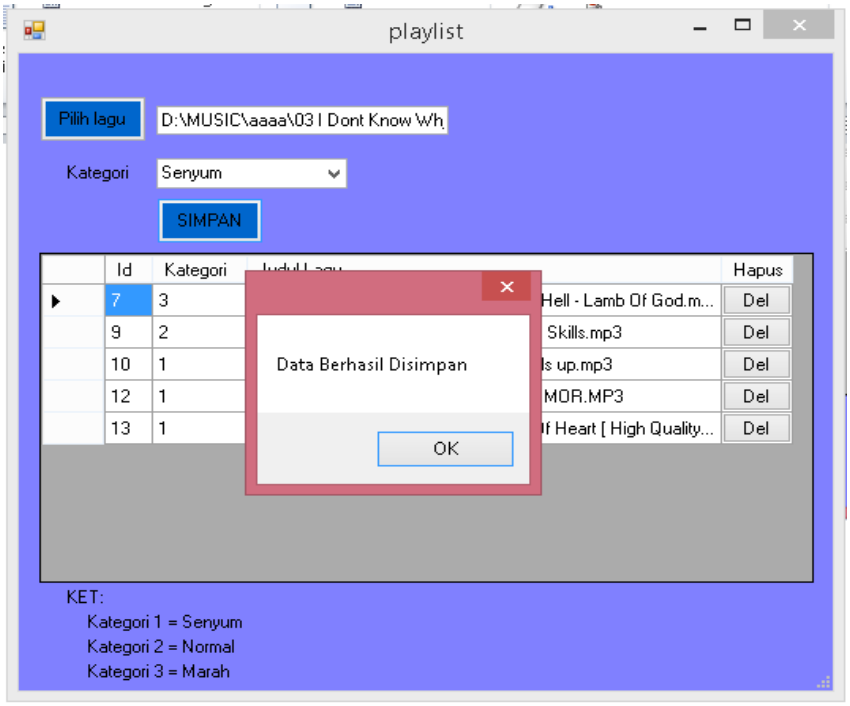
IV.1.4.4 Tambah Lagu

Tambah lagu merupakan proses untuk menambahkan lagu baru atau menghapus lagu yang ada di playlist. Untuk tampilan tamabah lagu dapat dilihat pada Gambar IV.9.



Gambar IV.9 Tampilan Awal Tambah Lagu

Jika lagu dan kategori sudah dipilih selanjutnya tekan tombol simpan untuk menyimpan data lagu baru kedalam playlist. Seperti yang terlihat pada Gambar IV.10.



Gambar IV.10 Tamah Data Lagu Berhasil Disimpan

IV.2 Pengujian

Pengujian ini terdiri dari 2 jenis pengujian, yaitu pengujian *alpha* dan pengujian performansi.

IV.2.1 Pengujian *Alpha*

Pengujian alpha dilakukan pada sisi pengembang yang dianggap pemakai dan merekam semua kesalahan dan masalah pemakaian. Pengujian alpha dilakukan pada sebuah lingkungan yang terkendali.

Pendekatan perancangan kasus uji menggunakan dua metode, yaitu *white box* dan *black box*. *White box* adalah metode perancangan kasus uji yang menggunakan struktur kontrol program. *Black box* adalah metode pengujian yang berfokus pada fungsional dari perangkat lunak.

IV.2.1.1 Rencana Pengujian

Berikut pada Tabel IV-2 adalah rencana pengujian yang akan dijalankan pada aplikasi ini:

Tabel IV-2 Rencana Pengujian

No	Kompunen yang diuji	Macam-macam uji	Jenis uji
1	Algoritma *learning vector quantization*	Jumlah region, *Cyclomatic Complexity* (V(G)), *Independent path*, *graph* matriks, dan *predicate node*	*White box*
2	Training	Proses mendeteksi fitur wajah	*Black box*
		Proses pembelajaran *learning vector quantization*	*Black box*
3	Testing	Proses mendeteksi wajah	*Black box*
		Proses mengenali ekspresi wajah	*Black box*
4	Musik Player	Proses menjalankan musik	*Black box*
5	Tambah Lagu	Menambahkan Data Lagu	*Black box*
		Menghapus Data Lagu	*Black box*

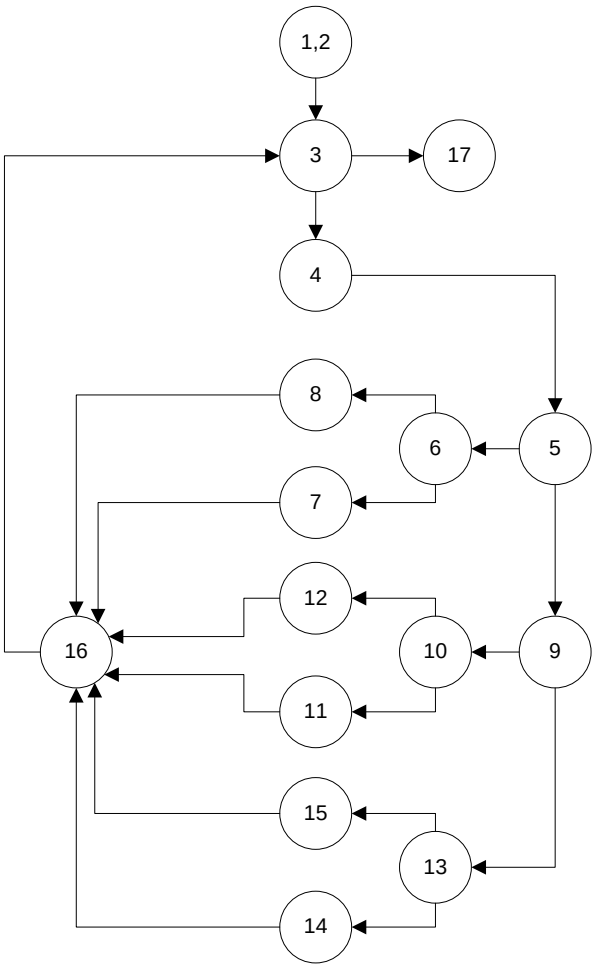
IV.2.1.2 Pengujian *White Box*

Pengujian secara *white box* terhadap algoritma *learning vector quantization* dilakukan untuk meyakinkan semua perintah dan kondisi aplikasi dieksekusi secara minimal. Teknik yang digunakan yaitu *Flow Graph* yang digunakan untuk menggambarkan alur dari algoritma dan *Graph Matriks* yang digunakan untuk merepresentasikan *flow graph* dalam bentuk tabel. Berikut ini *pseudocode* algoritma *learning vector uantization*:

```
1.    int maxPerulangan = 10;
      double errorMin = 0.1;
      int epoch = 0;
      double penguranganRasio = 0.1;
```

```
2.    do{
3.          epoch = epoch + 1;
            double[] hasilAkhir = new double[3];
4.          hitung jarak X terhadap nilai bobot Wj;
5.          if    (hasilAkhir[0]    <=    hasilAkhir[1]    &&
hasilAkhir[0] <= hasilAkhir[2]){
6.            if (kelas == 1) {
7.                    bobotBaru[o]    =    bobotLama[o]    +
(rasioPembelajaran * (data[o] - bobotLama[o]));
               }
                else {
8.                    bobotBaru[o]    =    bobotLama[o]    -
(rasioPembelajaran * (data[o] - bobotLama[o]));
                }
            }
9.         else if (hasilAkhir[1] <= hasilAkhir[2]) {
10.             if (kelas == 2) {
11.                   bobotBaru[o]    =    bobotLama[o]    +
(rasioPembelajaran * (data[o] - bobotLama[o]));
                }
                else {
12.                   bobotBaru[o] = bobotLama[o] -
(rasioPembelajaran * (data[o] - bobotLama[o]));
                }
               }
            else{
13.             if (kelas == 3) {
14.                   bobotBaru[o]    =    bobotLama[o]    +
(rasioPembelajaran * (data[o] - bobotLama[o]));
            }
                  else{
15.                   bobotBaru[o]    =    bobotLama[o]    -
(rasioPembelajaran * (data[o] - bobotLama[o]));            }
          }
16.      rasioPembelajaran     =     rasioPembelajaran     -
(penguranganRasio * rasioPembelajaran);
      }while ((epoch <= maxPerulangan) || (rasioPembelajaran
>= errorMin));
17.   return bobotBaru;
```

Berdasarkan sintaks di atas, dapat dibuat *flow graph* untuk menggambarkan ilustrasi aliran struktur program algoritma *learning vector quantization*. Bagan *Flow graph* tersebut dapat dilihat pada Gambar IV.11 dan tabel keterangan *flow graph* dapat dilihat pada Tabel IV-3.



Gambar IV.11 *Flow Graph*

Tabel IV-3 Keterangan *Flow Graph*

Simbol Flow Graph	Keterangan
○	Node (N)
→	Busur (E)

Jumlah region pada *flow graph* dihasilkan dari jumlah daerah yang berada di dalam node ditambah satu daerah di luar *node*. Rumus penjumlahannya:

Region = daerah_dalam + daerah_luar

= 6 + 1

= 7

Cyclomatic Complexity (V(G)) diperoleh dari perhitungan jumlah busur dikurangi jumlah *node* ditambah konstanta dua. Perhitungannya:

V(G) = E - N + 2

= 21 - 16 + 2

= 7

Independent path ditentukan oleh penelusuran dari node awal sampai node akhir. Berikut ini adalah sejumlah path yang ditemukan dari *flow graph* sebelumnya:

Path 1 = 1-2-3-17

Path 2 = 1-2-3-4-5-6-7-16-3-17

Path 3 = 1-2-3-4-5-6-8-16-3-17

Path 4 = 1-2-3-4-5-9-10-11-16-3-17

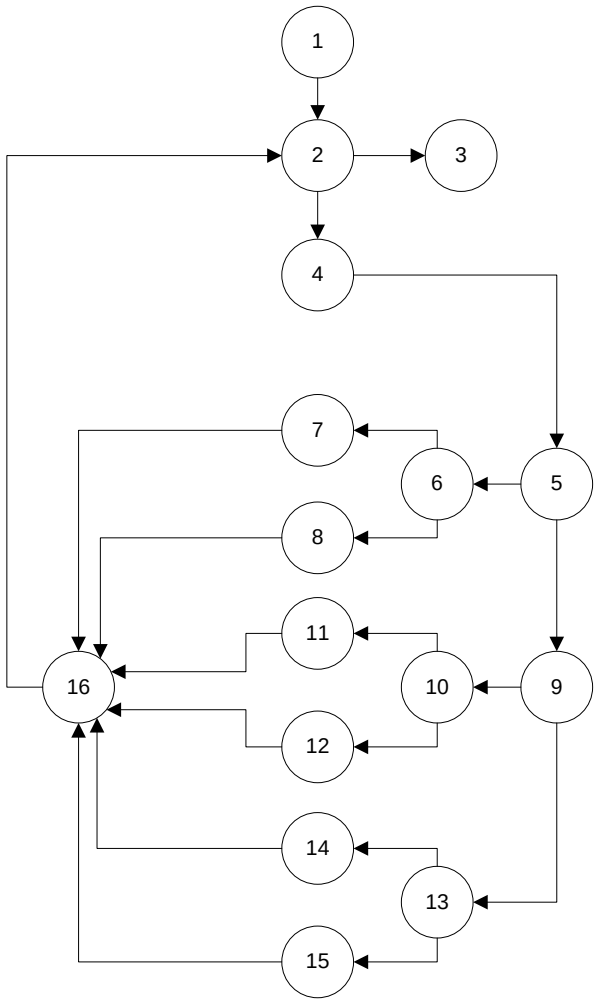
Path 5 = 1-2-3-4-5-9-10-12-16-3-17

Path 6 = 1-2-3-4-5-9-13-14-16-3-17

Path 7 = 1-2-3-4-5-9-13-15-16-3-17

Jadi, pada teknik *flow graph*, jumlah *region*, *Cyclomatic Complexity* dan *Independent Path* yang dihasilkan bernilai sama.

Padak teknik graph matriks, satu *node* pada *flow graph* diwakilkan oleh satu nomor agar memudahkan perhitungan matriks. Gambar IV.12 merupakan *flow graph* hasil pengubahan nomor pada *node*.



Gambar IV.12 *Flow graph* pada *graph matriks*

Tabel IV-4 merupakan matriks yang menyatakan hubungan antara simpul dengan simpul dari busur selanjutnya berdasarkan Gambar IV.12. Kolom Sum-1 dihitung berdasarkan jumlah *node* yang ada dikurang satu. Jika node dari busur selanjutnya tidak ada, maka dinyatakan sebagai elemen kosong dan bernilai nol.

Tabel IV-4 . *Graph* matriks

N	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	Sum-1
1		1															0
2			1	1													1
3																	0
4					1												0
5						1			1								1
6							1	1									1

7															1	0
8															1	0
9										1			1			1
10											1	1				1
11															1	0
12															1	0
13														1	1	1
14															1	0
15															1	0
16		1														0
Jumlah																6

V(G) = jumlah_matriks + 1

= 6 + 1

= 7

Setelah mendapatkan *Cyclomatic Complexity* dari *graph* matriks, perlu dicari nilai *Cyclomatic Complexity* berdasarkan *predicate node* (P), yaitu jumlah simpul yang memiliki lebih dari satu busur. Berdasarkan Gambar IV.12, simpul-simpul yang memiliki jumlah busur lebih dari satu ialah 2, 5, 6, 9, 10, dan 13 sehingga :

Predicate node (P) = 6

Setelah nilai P didapat, untuk mendapatkan nilai *Cyclomatic Complexity* dari *predicate node*, rumus dan perhitungan yang digunakan ialah

V(G) = P + 1

= 6 + 1

= 7

Jadi, nilai hasil dari *graph* matriks dan *predicate node* masing-masing adalah 7.

IV.2.1.3 Pengujian *Black Box*

Pengujian untuk menguji fungsional perangkat lunak yang sudah dibangun dengan menggunakan metode *black box* berdasarkan tabel rencana pengujian yang sudah dibubat.

1. Menggunakan data yang benar

Pada Tabel IV-5 adalah hasil uji dengan menggunakan data uji yang benar.

Tabel IV-5 Kasus dan Hasil Uji Menggunakan Data Yang Benar

No	Kasus/Uji	Skenario Uji	Data Uji	Hasil yang diharapkan	Hasil pengujian
1	Training	Menekan tombol "STAR" dan setelah terdeteksi tekan tombol "STOP"	Gambar wajah secara realtime	Melakukan proses perhitungan nilai fitur wajah dan menghasilkan F1,F2,F3,F4,F5,F6	[√]Berhasil [ ]Tidak berhasil
		Menekan tombol "TRAINING"	Nilai F1,F2,F3, F4,F5,F6, kelas	Melakukan proses pembelajaran untuk *learning vector quantization* , menyimpan hasil pembelajaran ke database dan menampilkan pesan "Data Tersimpan"	[√]Berhasil [ ]Tidak berhasil
2	Testing	Menekan tombol "STAR"	Gambar realtime	Melakukan proses deteksi wajah	[√]Berhasil [ ]Tidak berhasil
		Menekan tombol "STOP"	Citra wajah	Mengenali ekspresi wajah dari si pengguna	[√]Berhasil [ ]Tidak berhasil
3	Musik Player	Ekspresi wajah si pengguna sudah diketahui	Indeks ekspresi	Menjalankan musik sesuai dengan ekspresi wajah dari si pengguna	[√]Berhasil [ ]Tidak berhasil
4	Tambah Data Lagu	Data lagu sudah diinputkan semua	Lagu dan kategori	Data berhasil disimpan	[√]Berhasil [ ]Tidak berhasil
		Menekan Tombol "Del"	Id_lagu	Data berhasil dihapus	[√]Berhasil [ ]Tidak berhasil

2. Menggunakan data yang salah

Pada Tabel IV-6 adalah hasil uji dengan menggunakan data uji yang salah.

Tabel IV-6 Kasus dan Hasil Uji Menggunakan Data Yang Salah

No	Kasus/Uji	Skenario Uji	Data Uji	Hasil yang diharapkan	Hasil pengujian
1	Training	Menekan tombol "STAR" dan setelah terdeteksi tekan tombol "STOP"	Gambar wajah tidak terdeteksi	F1,F2,F3,F4,F5,F6 Tidak akan ditampilkan	[√]Berhasil [ ]Tidak berhasil
		Menekan tombol "TRAINING"	Nilai F1,F2,F3, F4,F5,F6, Nilai kelas belum diisi	Menampilkan form pesan " Inputan belum lengkap"	[√]Berhasil [ ]Tidak berhasil
2	Testing	Menekan tombol "STAR"	Gambar wajah tidak terdeteksi	Tidak akan melakukan proses apapun	[√]Berhasil [ ]Tidak berhasil
		Menekan tombol "STOP"	Citra wajah tidak terdeteksi	Ekspresi wajah tidak akan dikenali	[√]Berhasil [ ]Tidak berhasil
3	Musik Player	Ekspresi wajah si pengguna belum diketahui	Indeks ekspresi belum ada	Musik tidak akan dijalankan	[√]Berhasil [ ]Tidak berhasil
4	Tambah Data Lagu	Data lagu belum lengkap	Lagu	Menampilkan pesan " inputan belum lengkap"	[√]Berhasil [ ]Tidak berhasil
		Menekan Tombol "Del"	Id_lagu tidak dipilih	Data gagal dihapus	[√]Berhasil [ ]Tidak berhasil

IV.2.1.4 Kesimpulan Hasil Pengujian Alpha

Berdasarkan hasil pengujian fungsiaonal dari aplikasi yang dibangun dapat ditarik kesimpulan bahwa aplikasi tidak ada kesalahan proses dan secara fungsional mengeluarkan hasil yang sesuai dengan yang diharapkan.

IV.2.2 Pengujian Performansi

Pengujian performansi dilakukan dengan serangkaian percobaan-percobaan dalam kondisi-kondisi tertentu yang dapat mempengaruhi keefektifan kinerja pendeteksian ekspresi wajah untuk pemilihan lagu.

IV.2.2.1 Proses Pengujian

Proses training algoritma *learning vector quantization* menggunakan 25 orang dengan setiap orang dilakukan pengambilan data sebanyak 3 kali dengan ekspresi wajah yang berbeda, yaitu senyum, normal, dan marah, seperti yang terlihat pada Gambar IV.13. Dari 25 orang tersebut akan diambil sempel sebanyak 10 orang untuk melakukan pengujian dalam proses pendeteksian ekspresi wajah dengan tiga kategori ekspresi, yaitu senyum, normal, dan marah. Pengujian dilakukan dengan tiga tahapan, yaitu pengujian kemiringan wajah, pengujian jarak wajah terhadap kamera, dan pengaruh cahaya terhadap deteksi ekspresi wajah.

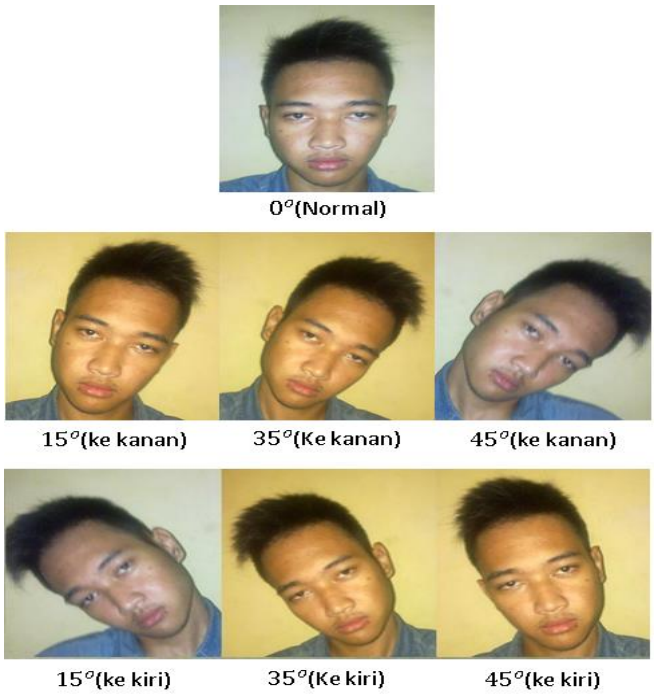


Gambar IV.13 Ekspresi Wajah

IV.2.2.2 Pengujian Kemiringan Wajah

Pengujian ini dilakukan untuk menguji akurasi dari pendeteksian ekspresi wajah berdasarkan sudut kemiringan wajah pengguna. Pengujian dilakukan

terhadap 10 orang dengan beberapa varian sudut kemiringan wajah ke arah kiri atau ke arah kanan, dengan sudut kemiringan 0^o(posisi normal), 15^o, 35^o, dan 45^o. Seperti yang ditunjukan pada Gambar IV.14 menjelaskan tentang varian sudut kemiringan. Dan pengujiannya pada Tabel IV-7.



Gambar IV.14 Sudut Kemiringan Wajah

Tabel IV-7 Pengujian Kemiringan wajah

Percobaan	0^o(Normal)			15^o			35^o			45^o		
	S	**N**	**M**	**S**	**N**	**M**	**S**	**N**	**M**	**S**	**N**	**M**
1	√	–	√	–	–	–	–	–	–	–	–	–
2	√	√	–	–	–	–	–	–	–	–	–	–
3	√	–	√	–	–	–	–	–	–	–	–	–
4	–	√	√	–	–	–	–	–	–	–	–	–
5	√	√	–	–	–	–	–	–	–	–	–	–
6	√	–	√	–	–	–	–	–	–	–	–	–

7	–	√	√	–	–	–	–	–	–	–	–	–
8	√	–	–	–	–	–	–	–	–	–	–	–
9	√	√	–	–	–	–	–	–	–	–	–	–
10	–	√	√	–	–	–	–	–	–	–	–	–
Rata-rata (%)	**63.33%**			**0%**			**0%**			**0%**		

Keterangan:

Tanda " √ " menujukan bahwa deteksi ekspresi wajah akurat

Tanda " – " menunjukan bahwa deteksi ekspresi wajah tidak akurat

S = Senyum

N = Normal

M = Marah

Pada pengujian kali ini dapat dianalisis bahwa sudut kemiringan wajah dari pengguna akan mempengaruhi pendeteksian. Karena jika wajah posisinya miring *library* emgucv tidak dapat mendeteksi wajah sehingga untuk menentukan ekspresi wajah tidak bisa dilakukan oleh karena itu sistem ini hanya akan berjalan ketika posisi wajah pengguna pada posisi tegak lurus. Pada pengujian di atas didapat nilai paling optimal pada posisi wajah adalah yang tegak atau sudut kemiringan 0^o(posisi normal) dengan tingkat akurasi sebesar 63.33%.

IV.2.2.3 Pengujian Jarak Wajah Terhadap *Webcam*

Parameter yang juga sangat berpengaruh pada bagian pendeteksian ekspresi wajah kali ini adalah jarak antara si pengguna dengan kamera yang digunakan. Pada pengujian kali ini dilakukan pada 10 orang dengan posisi tepat pada area wajah yang tegak lurus dan ketinggiannya sejajar dengan *webcam*. Berikut adalah Tabel IV-8 pendeteksian wajah berdasarkan jarak pengguna dengan kamera.

Tabel IV-8 Pengujian Jarak Wajah Terhadap *Webcam*

Percobaan	**Jarak Wajah (Cm)**				
	20	**40**	**60**	**80**	**>80**

	S	N	M	S	N	M	S	N	M	S	N	M	S	N	M
1	√	–	√	√	–	√	–	√	–	–	√	–	–	√	–
2	√	–	–	√	–	√	–	√	–	–	√	–	–	√	–
3	√	–	√	√	–	√	√	√	–	–	√	–	–	√	–
4	√	–	–	–	√	√	–	√	–	–	√	–	–	√	–
5	√	–	–	√	–	–	√	√	–	–	√	–	–	√	–
6	√	–	–	√	√	–	–	√	–	–	√	–	–	√	–
7	√	–	–	√	–	√	–	√	–	–	√	–	–	√	–
8	√	–	–	–	–	√	√	√	–	–	√	–	–	√	–
9	√	–	–	√	√	√	–	–	√	–	√	–	–	√	–
10	√	–	–	√	–	√	–	√	–	–	√	–	–	√	–
Rata-rata (%)	**40%**			**63.33%**			**43.33%**			**30%**			**30%**		

Keterangan:

Tanda " √ " menujukan bahwa deteksi ekspresi wajah akurat

Tanda " − " menunjukan bahwa deteksi ekspresi wajah tidak akurat

S = Senyum

N = Normal

M = Marah

Dari pengujian di atas dapat dianalisa dan disimpulkan bahwa semakin besar jarak wajah dan semakin dekat wajah ke kamera, maka akan berpengaruh pada deteksi ekspresi wajah. Karena pada saat posisi wajah terhadap *webcam* semakin dekat, itu akan membuat gambar yang terlihat hanya sebagian wajah saja dan jika jarak wajah dengan *webcam* semakin jauh, wajah akan terdeteksi namun untuk fitur-fitur wajah akan semakin mengecil itu mengakibatkan fitur-fitur wajah akan susah untuk dideteksi. Jarak wajah dengan kamera yang terbaik dalam pengujian di atas adalah pada jarak wajah dengan kamera 40cm, dengan tingkat akurasi sebesar 63.33%.

IV.2.2.4 Pengaruh Cahaya Terhadap Deteksi Ekspresi Wajah

Pengujian kali ini yaitu tentang pengaruh cahaya terhadap proses deteksi ekspresi wajah. Dalam pengujian pengaruh pencahayaan ini menggunakan jenis lampu neon dan pijar. Pengujian dilakukan terhadap 10 orang dengan kondisi ruangan sebagai berikut, lebar 2m, panjang 5m dan tinggi 2,75m. Dengan jarak wajah terhadap kamera adalah 40 cm. Berikut adalah Tabel IV-9 pengujian pengaruh cahaya terhadap deteksi ekspresi wajah.

Tabel IV-9 Pengaruh Cahaya Terhadap Deteksi Ekspresi Wajah

Percobaan	Jenis Lampu											
	LP 15 W			LP 40 W			LN 5 W			LN 20 W		
	S	N	M	S	N	M	S	N	M	S	N	M
1	√	–	√	√	√	√	√	–	–	√	–	√
2	√	–	√	√	–	√	√	–	√	√	√	–
3	–	–	–	√	√	–	–	–	√	√	–	√
4	√	–	–	√	–	√	–	√	√	√	–	√
5	–	√	√	√	–	√	√	√	√	√	–	√
6	√	–	√	–	–	√	√	–	√	–	√	–
7	–	–	–	√	–	√	√	–	√	√	–	–
8	√	–	√	√	√	–	–	–	–	–	–	√
9	–	–	–	–	–	–	√	–	√	√	–	√
10	√	–	√	√	–	√	√	–	–	√	√	√
Rata-rata (%)	40%			53.33%			53.33%			60%		

Keterangan :

LP = Lampu Pijar

LN = Lampu Neon

W = Watt

S = Senyum

N = Normal

M = Marah

Tanda " √ " menujukan bahwa deteksi ekspresi wajah akurat

Tanda " − " menunjukan bahwa deteksi ekspresi wajah tidak akurat

Dari pengujian di atas, yaitu pengaruh cahaya terhadap deteksi ekspresi wajah, semakin terang tempat untuk mendeteksi wajah maka deteksi akan semakin bagus akurasinya. Dalam proses pengambilan gambar secara *real time* pengaruh cahaya akan sangat berpengaruh karena jika ruangan semakin gelap maka gambar akan semakin banyak warna hitam sehingga deteksi fitur-fiturnya akan susah untuk dideteksi dan jika pencahayaan semakin terang itu akan membuat fitur-fitur dari wajah akan mudah terlihat sehingga pendeteksianya akan lebih baik. Dengan pencahayaan yang paling optimal dalam pengujian di atas adalah pada saat menggunakan lampu neon 20 Watt dengan tingkat akurasi 60%.

IV.2.2.5 Kesimpulan Pengujian Performansi

Dalam pengujian performansi yang dilakukan dengan tiga pengujian yaitu pengujian kemiringan wajah, jarak wajah terhadap kamera, dan pengaruh cahaya dapat ditarik kesimpulan bahwa akurasi dari sistem pendeteksian ekspresi wajah dalam pemilihan lagu adalah 62.22%.

IV.2.3 Data Hasil Pengujian

Data hasil pengujian menjelaskan tentang data inputan yang di dapat berdasarkan ekspresi wajah yang terdeteksi pada proses *testing*. Data hasil pengujian dapat dilihat pada Tabel IV-10.

Tabel IV-10 Data hasil Pengujian

Ekspresi	No	F1	F2	F3	F4	F5	F6	Lagu
Senyum	1	31.02	54	74	106.02	124.02	351.07	Wake Up
	2	51.21	69	77	158.86	147.01	386.75	RAN - Selamat Pagi
	3	35.11	56	76	134.63	129.25	363.57	RAN - Selamat

								Pagi
	4	54.17	76	81	191.21	185.17	313.77	Wake Up
	5	42.58	66	119.04	166.03	273.54	273.54	RAN - Sepeda
Normal	1	30.63	26	39	89	76.06	159.24	Bondan-Kita Selamanya
	2	22.52	28	40	92	82	151.92	Bondan-Tetap Semangat
	3	24.54	26	38	94.02	71.06	155.23	Bondan-Kita Selamanya
	4	22.94	25	37	84.02	81	224.08	RAN - Warna Warni Dunia
	5	27.02	35	47	139.03	101	177.74	Bondan-Kita Selamanya
Marah	1	31.09	59	77	126.14	163	241.81	RAN - Jadi Gila
	2	32.06	57	61	162	154.01	262.52	RAN - Jadi Gila
	3	23.14	57	85	106.23	153.16	237.44	Vierra-Don_tCry
	4	23.71	47	59	141	131.01	218.22	RAN - Jadi Gila
	5	48.71	49	61	175.23	134.13	247.31	Vierra-Don_tCry

Keterangan:

F1 = Jarak alis dengan mata

F2 = Lebar mata

F3 = Lebar mulut

F4 = Jarak kedua alis mata

F5 = Jarak kedua mata

F6 = Jarak alis mata dengan mulut

Berdasarkan Tabel IV-10 dapat dilihat adanya tiga ekspresi berbeda yang diambil nilai fitunya, dan dari setiap ekspresi akan menjalankan lagu sesuai dengan playlist yang sudah tersedia dengan urutan pemutaran lagu secara acak.

IV.2.4 Kesimpulan Pengujian

Dari hasil pengujian *white box* dapat diambil kesimpulan bahwa kompleksitas lagoritma *learning vector quantization* adalah 7. Dan hasil uji *black box* dapat diambil kesimpulan bahwa fungsional dari setiap sistem berjalan dengan baik. Sedangkan dalam pengujian performansi yang dilakukan dengan tiga

pengujian yaitu pengujian kemiringan wajah, jarak wajah terhadap kamera, dan pengaruh cahaya dapat ditarik kesimpulan bahwa akurasi dari sistem pendeteksian ekspresi wajah dalam pemilihan lagu adalah 62.22%.