



Hak Cipta Dilindungi Undang-Undang

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber:
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penyusunan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar Unand.
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin Unand.

**SISTEM PENDETEKSI KANTUK DENGAN DETEKSI SUARA
MENGUAP MENGGUNAKAN METODE MEL FREQUENCY CEPSTRAL
COEFFICIENTS-VECTOR QUANTIZATION (MFCC-VQ) DAN DETEKSI
PERUBAHAN POSISI WAJAH**

TUGAS AKHIR



**TREDO SAPUTRA
06175095**

**JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNIK
UNIVERSITAS ANDALAS
PADANG
2012**

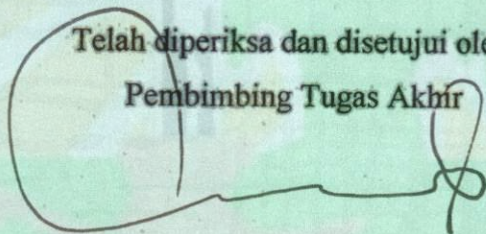
LEMBAR PENGESAHAN

**SISTEM PENDETEKSI KANTUK DENGAN DETEKSI SUARA
MENGUAP MENGGUNAKAN METODA *MEL FREQUENCY CEPSTRAL*
COEFFICIENTS-VECTOR QUANTIZATION (MFCC-VQ) DAN DETEKSI
PERUBAHAN POSISI WAJAH**

LULUS SIDANG TUGAS AKHIR

16 JANUARI 2012

Telah diperiksa dan disetujui oleh,
Pembimbing Tugas Akhir



RAHMADI KURNIA, DR. ENG.
NIP: 19690820 199703 1 002

Mengetahui,

Ketua Jurusan Teknik Elektro



RAHMADI KURNIA, DR. ENG.
NIP: 19690820 199703 1 002

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

Sesungguhnya atas kehendak Allah SWT semua itu dapat terwujud
Tiada kekuatan kecuali dengan pertolongan Allah
(Q.S 27 : 19)

Alhamdulillahirabbil 'alamin
Berkat anugrah dan rahmat-Mu Ya Allah
Dengan Keridhaan-Mu Ya...Rab
Kupersembahkan karya kecilku yang jauh dari kesempurnaan ini
kepada mereka yang kusayangi

Mama dan Papa tercinta, yang telah berjuang untuk
menguliahkanku dengan segenap usaha dan pengorbanannya.
Untuk sementara ini, hanya ini yang mampu aku berikan untuk
membuat mama dan papa bangga, membalas setiap tetes keringat
usaha, setiap tetesan air mata do'a, dan beribu langkah yang
engkau tapaki, yang semata-mata hanya untuk menguliahkanku,
menggapai cita-citaku dan menyongsong masa depanku.

Kemudian Kakakku tersayang, Kak Lung, Kak Ngah dan juga
Adikku tersayang, Riva. Terimakasih atas semua pengorbanan, dan
pengertian kalian untukku, juga untuk bantuan materil dan sprituil
yang telah kalian berikan.

Sungguh semua itulah yang membuatku tegar diatas semua
kondisi dan keadaan yang kita rasakan selama ini, semoga
semua akan berubah menjadi lebih baik....

Semoga rahmat dan karunia Allah SWT
selalu mengiringi perjalanan hidupku
Amin..

Special Thanks to :

My Beloved Family: Mama, Papa, Uni, Isen, Riva, Da Del dan Nayya. Terimakasih banyak atas pengorbanan dan pengertian yang tak ternilai harganya selamanya ini. Do masih ingek payah nyo dulu katiko do nio masuak kuliah ma, pa. Ama manangih katiko do lulus SPMB. Ama jo apa paniang kama mencari pith untuak do kuliah. Maafan do yo ma, pa, lamo na manyusahan ama jo apa. Isen nan manolongan do mambalian formulir samo maantaan do pai mandaftar ka Unand sampai mencari tampek kos bajalan kian kamari di hari paneh garang, makasih yo sen, maaf kalau sampai kini do masih manyusahan isen. Uni nan manelson do katiko pagi ka ujian SPMB, berkat doa uni, lulus do SPMB, maaf yo ni kalau sampai kini do masih manyusahan uni. Riva, rajin-rajin sakolah yo va, mudah-mudahan va bisa lo kuliah di tampek yang va nio. Da Del, maaf yo da, kalau sampai kini do masih manyusahan uda jo isen. Kamanakan Mak Ongah Ainayya, Anak Ayah, capek gadang yo nak, mudah-mudahan bisa menjadi anak yang shaleh dan menjadi kebanggaan bagi ayah dan bunda.

My Sucentong, Nur Hasnah. Ntong akhirnyo abang lulus juo. Makasih yo ntong, ntong yang ndak pernah bosan maagiah abang semangat, walaupun acok bang kecek an ntong nyinya. Namun akhirnyo abang bisa juo meraih gelar sarjana walaupun agak talambek. Jan banyak kandak jo lai ndak ntong..hehehe. Mudah-mudahan apa yang kita cita-citakan dapat tercapai. Amin.

Sahabat terbaikku TIARA ERFIA: Iqbal (Boss, kok jadi wak ka Jakarta bisuak, tolong salamaik an wak ciek yo boss..hehehe), Andry (dima ndan? Apo misi kini tu..?, lah depek lahan untuak bukak taranak ayam tu ndan..?), Rizqan (lah banyak bisnis kini mah ndan..salut wak samo ndan..ajaan wak bisnis ciek yo ndan..hehe), Adhelfi (makasih dokter muda adhel, atas sms motivasinya selama ini, kalau boleh jujur sms yang adhel kirim sesuai dengan suasana hati ayah saat itu..adhel kayaknya berbakat jadi peramal..hehe), Erisa (ibuuu...capek lah jadi dokter,,,baralek abis tu lai,,calon kan lah ado mah...hehe), Rahmi (samo wak wisuda bisuak mi...?hehehe), Fitri (anak ayah yang paling manis, paling banyak makan, tapi badan ndak namuah gapuak..hehehe...capek lah jadi dokter yo nak,,), Irvani (juwik,,lapeh jo stress ayah wik,,makasih atas semangat dan kata-kata klise nya selama ini..hehehe), Adilla (ila, jadi jo ila baralek di umua 23...mudah-mudahan capek ayah depek cucu yo la...hehehe salam samo uda ila). Terimakasih atas persahabatan yang telah kita jalin selama ini. mudah-mudah kita bisa sukses semuanya.

Pak Tam Flower, Ibuk, Ni Gus, Ari dan Gina. Terimakasih do lah buliah tingga tampek apak jo ibuk salamo labiah dari 4 tahun. Sado yang tingga di rumah tu lah do anggap keluarga pak, buk. Maaf kalau salamo do tingga di rumah apak banyak babuek salah, manakak-makak dan lain-lain, baiak singajo ataupun ndak singajo.

Kawan sakos dulu tampek Pak Tam, Adi, Rayes dan Getby. Maafan salah wak yo di, yes, get. Mungkin wak banyak mambuek kawan-kawan tasingguang. Tarimo kasih ateh sadonyo salamo labiah kurang 4 tahun wak tingga basamo. King (semangat yuang, mudah-mudahan capek depek gelar S.H).

Kawan sapamainan di elektro: Duano (Semangat ndan, mudah-mudahan takaja ndan), Eko (makasih lah acok pinjaman FU untuak japuik samba jo pai malala...hehe), Heru (kajaan lah ngguik, Jan acok na bad mood), Rifyal (mas bro, kajaan lah sidang beliau, inget janji wak di audit pas wisuda bisuak..hehe), Gaul (SEMANGKA Gaua!!!), Miko (tunggu wak di basecamp yo sanak), Hary (makasih banyak ri, lah banyak bana manolongan denai salamo ko), Hasbi (makasih banyak bi, lah manolong wak baraja tentang TA ko), Buya (makasih banyak yo buy, lah manolongan wak baraja), Firman (makasih jas nyo pimain, ST juo denai..hehe), Rahmat Mizar (makasih cawek samo dasi nyo maik, makasih lo salamoko lah ditebengan ka kampus...hehehe), Amaik Ichigo (sorry kok acok talabiah na bagarah samo maik, sampai tadando clasmild satangah...hehe), Cander (kajaan lah seminar KP lai mbia, jan main game se karajo..hehe), Renol (semangat nol, mudah-mudahan capek salasai), Aziz (main badminton wak bisuak liak da jis), Pasyak (makasih lah maajaan program yo syak), Yonggi (makasih magnum jo teh gelasnyo puk wah), Hafiz (semangat fiz, ulang liak), Rizky (Gaban, Ada di Database Pak!!!), Andrew (BL wak lai..), Nicko (pelokan kamera ko ciek per).

Rekan-rekan lain yang tak dapat saya sebutkan satu persatu, terimakasih atas persahabatannya selama ini. Hidup ELEKTRO!!!!

Thanks all...

ABSTRAK

Perkembangan teknologi pengolahan suara dan pengolahan citra semakin pesat. Penggabungan kedua teknologi ini memungkinkan manusia untuk membuat suatu sistem baru yang dapat diaplikasikan lebih lanjut. Salah satunya adalah dalam bidang teknologi biometrik. Dalam penelitian ini dirancang suatu sistem teknologi biometrik yaitu sistem pendeteksi kantuk dengan deteksi suara menguap menggunakan Mel-Frequency Cepstral Coefficient-Vektor Quantization (MFCC-VQ) dan deteksi posisi wajah menggunakan metoda Haar Cascade Clasifier. Suara menguap direkam secara online, kemudian di ekstraksi dengan metoda MFCC-VQ. Jarak Euclidean (threshold) digunakan untuk mencocokkan input suara menguap dengan suara menguap dalam filebase. Apabila suara menguap dideteksi oleh sistem maka dilanjutkan dengan pendeteksian wajah. Deteksi wajah yang digunakan adalah deteksi wajah frontal, sehingga posisi wajah miring (dalam posisi tertidur) tidak akan terdeteksi oleh sistem. Apabila sistem tidak dapat mendeteksi wajah selama 2 detik, maka sistem akan mengaktifkan alarm untuk memberikan peringatan. Dari hasil pengujian, tingkat keberhasilan sistem mencapai 83,67% dan tingkat kesalahan sistem adalah 13,33%.

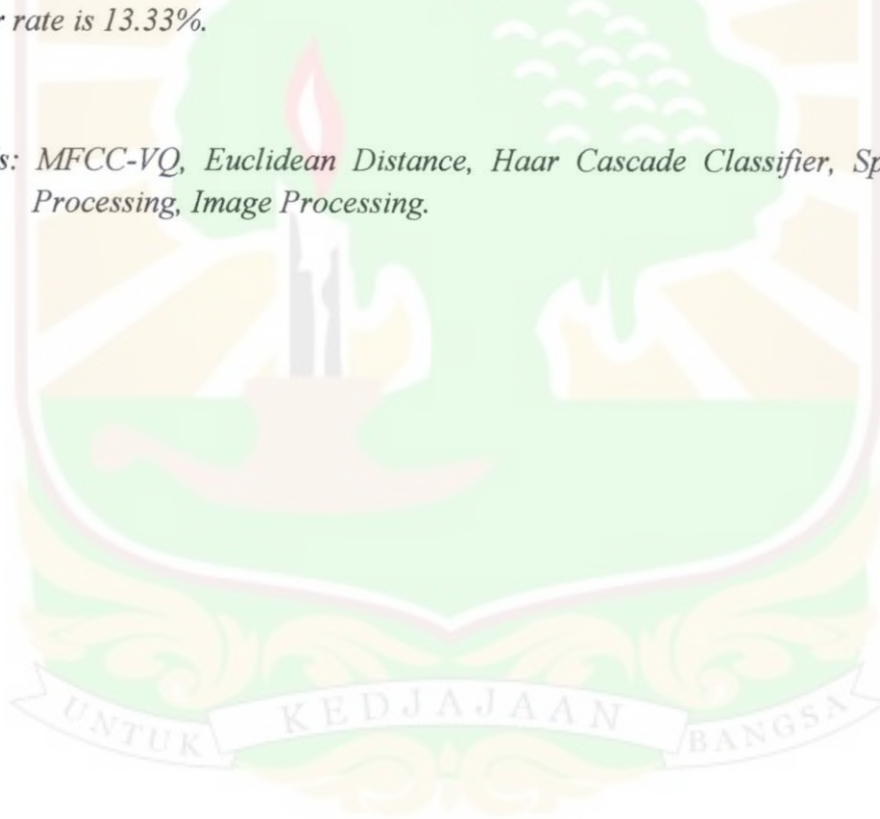
Kata kunci: MFCC-VQ, Jarak Euclidean, Haar Cascade Classifier, Pengolahan Suara, Pengolahan Citra.



ABSTRACT

The development of speech processing technology and image processing are rapidly increasing. The combination of these two technology will enable humans to create a new system that can be applied further. One of them is biometric technology. In this study designed a system of biometric technology that drowsiness detection system with a yawn sound detection using Mel-Frequency Cepstral Coefficient-Vector Quantization (MFCC-VQ) and face detection method using Haar Cascade Clasifier. Yawn sound recorded directly, then extracted by using MFCC-VQ method. Euclidean distance (threshold) is used to match the input with the database. If the system detects a yawning voice then continued with face detection. Face detection is used is the frontal face detection, face tilted position (in sleep position) would not be detected by the system. If the system can't detect the face for 2 seconds, then the system will activate the alarm to give warning. The success rate of drowsiness detection from the test results is 83.67% and error rate is 13.33%.

Keywords: MFCC-VQ, Euclidean Distance, Haar Cascade Classifier, Speech Processing, Image Processing.



KATA PENGANTAR



Alhamdulillah, puji syukur penulis ucapkan kehadiran Allah SWT yang telah memberikan berkah dan rahmat-Nya, sehingga penulis dapat menyelesaikan penelitian dan penulisan Tugas Akhir dengan judul **“Sistem Pendeteksi Kantuk Dengan Deteksi Suara Menguap Menggunakan Metoda *Mel Frequency Cepstral Coefficients-Vector Quantization* (MFCC-VQ) dan Deteksi Perubahan Posisi Wajah”**. Penulisan Tugas Akhir ini merupakan salah satu syarat untuk menyelesaikan pendidikan program strata satu pada Jurusan Teknik Elektro Fakultas Teknik Universitas Andalas Padang.

Dalam penyelesaian penulisan Tugas Akhir ini penulisan mendapat bimbingan, petunjuk dan bantuan dari berbagai pihak. Untuk itu penulis mengucapkan terimakasih kepada:

1. Kedua orang tua (Irwan dan Yetty S.Pd) dan segenap keluarga (Kak Lung, Kak Ngah, Riva, Da Del, dan Nayya) yang telah memberikan kasih sayang, dukungan moril dan materil serta doa yang senantiasa menyertai penulis selama menempuh pendidikan ini. Semoga Allah selalu memberikan kasih sayang yang berlipat ganda kepada mereka.
2. Bapak Rahmadi Kurnia, DR. ENG. selaku Ketua Jurusan Teknik Elektro Fakultas Teknik Universitas Andalas dan sekaligus sebagai pembimbing atas segala bantuan berupa ilmu, bimbingan dan kesabaran yang telah diberikan selama penulis melakukan penelitian.
3. Bapak Mumuh Muharam, MT dan Ibuk Fitrilina, MT yang telah bersedia menjadi penguji dari seminar proposal hingga sidang Tugas Akhir. Terimakasih atas saran dan masukannya yang berharga untuk kesempurnaan Tugas Akhir ini.
4. Bapak dan Ibu Dosen di lingkungan Fakultas Teknik umumnya dan Jurusan Teknik Elektro khususnya atas segala didikan dan ilmu yang telah diberikan.

5. Pegawai Jurusan Da Eka, Da Akbar, Ni Yuli, Ni Neng, Ni Desi dan Ni Reni atas bantuan dan kemudahan dalam segala urusan administrasi.
6. Nur Hasnah. Terimakasih atas doa dan semangatnya selama ini.
7. Dede Pratama. Terimakasih atas bantuan dalam pemahaman program.
8. Hasbi, Buya, Rahmat dan Firman Rekan Tugas Akhir. Terimakasih atas bimbingan dan bantuannya.
9. Duano, Eko, Heru, Rifyal, Hary, Renol, Amaik, Chan, rekan-rekan di gardu dan teman-teman sepermainan yang tak dapat penulis sebutkan satu per satu, terimakasih atas persahabatannya selama ini.
10. Rekan-rekan seperjuangan angkatan 2006, terimakasih atas semuanya.
11. Semua pihak yang telah membantu baik langsung maupun tidak langsung, yang tidak mungkin disebutkan satu persatu.

Penulis mendoakan semoga amal kebajikan mereka diterima disisi Allah SWT dan dibalas dengan pahala yang berlipat ganda. Penulis menyadari bahwa dalam penulisan ini masih banyak terdapat kekurangan, oleh karena itu penulis mengharapkan kritik dan saran yang membangun demi kesempurnaan Tugas Akhir ini. Penulis berharap semoga Tugas Akhir ini dapat bermanfaat bagi semua pembaca.

Padang, 20 Januari 2012

Penulis

DAFTAR ISI

Halaman

HALAMAN PENGESAHAN

HALAMAN PERSEMBAHAN

ABSTRAK..... i

ABSTRACT..... ii

KATA PENGANTAR..... iii

DAFTAR ISI..... v

DAFTAR GAMBAR..... ix

DAFTAR TABEL..... xi

BAB 1 PENDAHULUAN..... 1

1.1. Latar Belakang..... 1

1.2. Perumusan Masalah..... 4

1.3. Tujuan Penelitian..... 4

1.4. Manfaat Penelitian..... 4

1.5. Batasan Masalah..... 4

1.6. Metodologi Penelitian..... 5

1.7. Sistematika Penulisan..... 5

BAB 2 TINJAUAN PUSTAKA..... 7

2.1. Proses Pembentukan Sinyal Suara Pada Manusia..... 7

2.2. Sistem Pengenalan Suara..... 8

2.3. *Mel Frequency Cepstrum Coefficient* (MFCC). 8

2.3.1. Konversi Analog menjadi Digital.....	10
2.3.2. <i>Frame Blocking</i>	10
2.3.3. <i>Windowing</i>	11
2.3.4. <i>Fast Fourier Transform (FFT)</i>	12
2.3.5. <i>Mel-Frequency Wrapping</i>	13
2.3.6. <i>Cepstrum</i>	14
2.4. <i>Vektor Quantization (VQ)</i>	14
2.4.1. Pelatihan Vektor-Vektor	15
2.5. Jarak <i>Euclidean</i>	16
2.6. Pengolahan Citra (<i>Image Processing</i>)	17
2.6.1. Dasar-Dasar Pengolahan Citra Digital	18
2.6.2. Model Warna RGB.....	19
2.6.3. Penskalaan Citra (<i>Scaling</i>)	21
2.6.4. <i>Grayscale</i>	22
2.6.5. <i>Thresholding</i>	22
2.7. Metoda <i>Haar Cascade Classifier</i>	23
2.7.1. <i>Training Data Pada Haar</i>	23
2.7.2. Sistem Kerja Algoritma <i>Haar Cascade Classifier</i>	24
2.7.3. <i>Haar Feature</i>	25
2.7.4. <i>Integral Image</i>	26
2.7.5. <i>Cascade Classifier</i>	27
BAB 3 METODOLOGI PENELITIAN	28
3.1. Jenis Penelitian.....	28
3.2. Sampel Penelitian.....	28

3.3. Desain Penelitian	28
3.4. Teknik Analisa Sistem	29
3.5. Prosedur Penelitian	29
3.5.1. Tinjauan dan Studi Kepustakaan	29
3.5.2. Penyusunan Algoritma Program	29
3.5.3. Diagram Alir Sistem	30
3.5.4. Perancangan dan Pembuatan Program	30
3.5.5. Analisa Keluaran Sistem	30
3.5.6. Penyusunan Laporan Penelitian	30
BAB 4 PERANCANGAN SISTEM	31
4.1. Blok Diagram Sistem	31
4.2. Ekstraksi Parameter Sinyal Suara Menggunakan MFCC	32
4.2.1. <i>Frame Blocking</i>	32
4.2.2. <i>Windowing</i>	32
4.2.3. <i>Fast Fourier Transform (FFT)</i>	32
4.2.4. <i>Mel Frequency Wrapping</i>	33
4.2.5. <i>Discrete Cosine Transform (DCT)</i>	33
4.2.6. <i>Cepstrum</i>	33
4.3. <i>Vector Quantization</i>	33
4.4. Penghitungan Jarak Euclidean	34
4.5. Deteksi Posisi Wajah	36
4.5.1. Pra Proses	36
4.5.2. Input Video	37
4.5.3. Akuisisi Citra	37

4.5.4. Segmentasi	37
4.5.5. Deteksi Wajah Dengan Metoda <i>Haar Cascade Classifier</i>	37
4.5.6. Hitung Waktu.....	38
4.6. Algoritma Program	38
4.6.1. Algoritma MFCC	38
4.6.2. Algoritma <i>Haar Classifier</i>	39
BAB 5 ANALISA HASIL DAN PEMBAHASAN	40
5.1. Hasil Pendeteksian Suara Menguap.....	40
5.1.1. Hasil Ekstraksi Parameter Menggunakan MFCC	40
5.1.2. Hasil <i>Vektor Quantization</i> (VQ)	42
5.1.3. Jarak <i>Euclidean</i>	43
5.1.4. Penentuan Nilai Threshold.....	44
5.2. Hasil Pendeteksian Wajah.....	48
5.3. Hasil Pengujian Sistem	49
5.4. Pembahasan.....	53
BAB 6 KESIMPULAN DAN SARAN	54
6.1. Kesimpulan	54
6.2. Saran	54
DAFTAR PUSTAKA	
LAMPIRAN	

DAFTAR GAMBAR

	Halaman
Gambar 2.1. Organ Penghasil Suara Manusia	7
Gambar 2.2. Blok Diagram Sistem Pengenalan Suara	8
Gambar 2.3. Contoh Sinyal Suara Sepanjang 40 milidetik	9
Gambar 2.4. Blok Diagram MFCC.....	9
Gambar 2.5. Pembagian Sinyal Dalam Bentuk <i>Frame-Frame</i>	11
Gambar 2.6. Perbandingan Sinyal Suara Sebelum dan Sesudah Proses <i>Windowing</i>	12
Gambar 2.7. Hubungan Frekuensi Dengan Skala <i>Mel</i>	13
Gambar 2.8. Contoh <i>Triangular Filter Bank</i> Dengan <i>Mel Scale</i>	14
Gambar 2.9. Formasi <i>VQ Codebook</i> Antara Penutur 1 Dan Penutur 2.....	15
Gambar 2.10. Citra Digital.....	18
Gambar 2.11. Nilai Warna RGB Dalam Heksadesimal.....	19
Gambar 2.12. Model Warna RGB	21
Gambar 2.13. Penskalaan Citra.....	21
Gambar 2.14. Citra <i>Grayscale</i>	22
Gambar 2.15. Citra <i>Thresholding</i>	23
Gambar 2.16. Sistem Kerja Algoritma <i>Haar Cascade Classifier</i>	24
Gambar 2.17. <i>Basic Haar Feature</i>	25
Gambar 2.18. <i>Integral Image</i>	26
Gambar 2.19. Contoh <i>Integral Image</i>	26

Gambar 2.20. <i>Cascade Classifier</i>	27
Gambar 2.21. (a) Ilustrasi <i>Cascade Boosted Classifier</i> (b) Fitur Yang Digunakan Pada Stage Pertama.....	27
Gambar 4.1. Blok Diagram Penelitian	31
Gambar 4.2. Blok Diagram Deteksi Suara Menguap	35
Gambar 4.3. Blok Diagram Deteksi Posisi Wajah	36
Gambar 4.4. Proses <i>Grayscale</i> Pada Citra Wajah	37
Gambar 4.5. Alur Cara Kerja <i>Haar Cascade Classifier</i>	38
Gambar 5.1. Bentuk Sinyal Suara Menguap Filebase Dan Suara Menguap Yang Diujikan.....	40
Gambar 5.2. Spektogram Sinyal Suara Menguap, $M = 100$ Dan $N = 256$	41
Gambar 5.3. Contoh <i>Filterbank</i> Dalam Proses <i>Mel Frequency</i>	41
Gambar 5.4. Plot Vektor Akustik	42
Gambar 5.5. <i>Plot</i> Vektor Akustik Yang Telah Dilatih dan <i>Codebook</i> -nya.....	43
Gambar 5.6. Grafik Pendeteksian Suara	47
Gambar 5.7. Hasil Deteksi Wilayah Wajah Dengan Berbagai Posisi	48
Gambar 5.8. Posisi Wajah Yang Tidak Mampu Dikenali Sistem	49
Gambar 5.9. Tampilan Awal Sistem	50
Gambar 5.10. Tampilan Sistem Ketika Suara Menguap Terdeteksi.....	51
Gambar 5.11. Tampilan Sistem Ketika Sistem Pendeteksi Wajah Aktif.....	51
Gambar 5.12. Tampilan Sistem Ketika Sistem Mendeteksi Objek Tertidur	52

DAFTAR TABEL

Halaman

Tabel 5.1. Penentuan Nilai <i>Threshold</i>	44
Tabel 5.2. Hasil Pengambilan Data Pendeteksian Sinyal Suara Menguap	45
Tabel 5.3. Hasil Pengambilan Data Pendeteksian Sinyal Suara Mirip Menguap ..	46



BAB I

PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi pengolahan citra (*image processing*) dan pengolahan suara (*speech processing*) semakin pesat. Penggabungan kedua teknologi ini menyediakan kemungkinan manusia untuk membuat suatu sistem baru yang dapat digunakan untuk diaplikasikan lebih lanjut. Salah satunya adalah dalam bidang teknologi biometrik. Berbagai studi tentang teknologi biometrik terus dikembangkan. Teknologi biometrik merupakan suatu metode mengenali manusia berdasarkan satu atau lebih bagian tubuh manusia atau kelakuan dari manusia itu sendiri yang memiliki keunikan.

Proses pengenalan wajah yang dilakukan komputer tidak semudah dan secepat dibandingkan dengan proses pengenalan yang dilakukan manusia. Manusia dengan mudah dapat mengenali seseorang sangat cepat tanpa rasanya harus berfikir lama. Manusia juga tidak terpengaruh oleh orientasi wajah orang tersebut, misalnya orang tersebut agak menoleh, menunduk, menengadah, asalkan dalam batas-batas yang masih dapat dilihat. Sedangkan komputer selain lambat dalam pengenalan, juga kesulitan pada orientasi wajah yang berlainan, pencahayaan, latar belakang yang berbeda, potongan rambut, kumis atau jenggot, kacamata atau tidak, laki-laki atau perempuan dan lain sebagainya. Memang otak manusia lebih memiliki keuntungan dalam mengatasi masalah dimana aturan eksplisit tidak mudah diformulasikan, sedangkan komputer memiliki kelebihan dalam bidang matematika dimana aturan-aturan mudah diformulasikan.

Suara dan wajah merupakan dua fitur yang juga sering digunakan dalam berbagai aplikasi biometrik. Suara manusia diolah agar manusia dapat berinteraksi dengan mesin atau komputer. Dengan teknologi pengolahan suara, mesin atau komputer mampu mengenali jenis sebuah suara melalui perbandingan dengan database suara yang telah disimpan.

Sama halnya dengan suara, wajah juga banyak digunakan dalam aplikasi biometrik. Wajah seseorang bisa terdapat dalam foto atau video. Foto atau video merupakan sebuah citra, namun foto adalah citra diam sedangkan video

merupakan citra bergerak. Secara harfiah, citra (*image*) adalah gambar pada bidang dwimatra (dua dimensi). Citra diam adalah citra tunggal yang tidak bergerak, sedangkan citra bergerak adalah rangkaian citra diam yang ditampilkan secara beruntun sehingga memberi kesan pada mata kita sebagai gambar yang bergerak.[12]

Wajah yang dimiliki setiap orang pasti berbeda karena memiliki ciri - ciri tersendiri sehingga bisa digunakan dalam aplikasi seperti keamanan. Wajah bisa digunakan sebagai pengganti *password* atau pin yang rentan terhadap pencurian atau pembobolan.

Dengan teknologi yang canggih, kehidupan manusia menjadi lebih mudah. Berbagai kegiatan sehari - hari dapat diselesaikan lebih cepat dan aman. Fitur biometrik seperti suara dan wajah dapat diaplikasikan dalam kehidupan sehari-hari seperti sistem pendeteksi kantuk.

Kantuk merupakan hal yang wajar terjadi pada setiap orang. Hal ini dapat disebabkan oleh bermacam-macam faktor seperti kelelahan, kurang tidur, makan berlebih dan lainnya. Kantuk biasanya diawali dengan menguap yang merupakan refleksi dari tubuh dimana ia memberikan tanda bahwa jumlah oksigen dalam otak menurun sehingga tidak mampu berkonsentrasi dengan optimal namun orang menguap tidak semuanya mengantuk.

Kantuk memang hal yang lumrah akan tetapi jika terjadi dalam kondisi yang tidak tepat seperti sedang mengendarai kendaraan, belajar, ataupun sedang melakukan pekerjaan yang membutuhkan konsentrasi yang tinggi tentulah tidak bagus. Pada saat yang tidak tepat tersebut kadang kala kita dituntut untuk tetap berkonsentrasi dengan hal yang sedang kita kerjakan namun secara tidak sadar kita mengantuk dan tertidur sehingga hasil kerja kita tidak maksimal.

Untuk itu perlu dibuat suatu sistem yang mampu mendeteksi kantuk yang mana hasil akhir dari sistem ini mampu mengantisipasi seseorang agar tidak tertidur beberapa saat setelah orang tersebut menguap.

Sebelumnya telah ada penelitian tentang pendeteksi kantuk diantaranya:

Neeta Premar [2] yang berjudul *Drowsy Driver Detection System*. Sistem ini menggunakan konsep berbasis *non-intrusive machine vision*. Sistem ini

menggunakan kamera sekuriti yang mengarah langsung ke mata pengemudi dan memonitor matanya untuk mendeteksi kelelahan.

Vural, Esra, and Mujdat Cetin, dkk. [3] yang berjudul "*Automated Drowsiness Detection For Improved Driving Safety*". Sistem ini mengamati perilaku manusia selama mengantuk seperti kedipan mata, gerakan menguap dan gerakan wajah lainnya menggunakan sistem *Facial Action Coding System* (FACS).

Tsuyoshi NAKAGAWA, Taiji KAWACHI, dkk[4] dalam paper yang berjudul "*Drowsiness Detection Using Spectrum Analysis of Eye Movements and Effective Stimuli to Keep Drive Awake*". Sistem ini menilai tingkat mengantuk pengemudi dengan analisis LF/HF untuk gerakan mata.

Penelitian-penelitian tentang deteksi wajah sudah banyak dilakukan, diantaranya:

Setiawan Hadi [7] dalam disertasinya yang berjudul *Pengembangan Metode Pendeteksian Banyak Wajah Pada Citra Digital Kompleks Menggunakan Pendekatan Multiaspek*. Penelitian tersebut membuat suatu sistem yang dapat mendeteksi banyak wajah pada kondisi citra wajah yang kompleks secara waktu nyata (*realtime*). Sistem ini mampu mendeteksi wajah pada citra digital statis, citra video dan citra langsung yang diperoleh melalui *webcam*. Metode yang digunakan adalah pendekatan multiaspek yang melibatkan tiga fase pemrosesan yang disebut kerangka kerja DeWa.

Dewi Agushinta, Adang Suhendra dan Hendra[8] dalam papernya yang berjudul *Ekstraksi Fitur Dan Segmentasi Wajah Sebagai Semantik Pada Sistem Pengenalan Wajah* yang mengembangkan suatu sistem yang memisahkan citra wajah ke dalam komponen-komponen wajah yang kemudian mengekstraksi fitur bagian mata, hidung, mulut dan batas wajah pada citra diam tunggal yang di ambil dari posisi tampak depan. Antara masing-masing fitur di ukur jaraknya yang selanjutnya dikombinasikan dengan fitur lainnya untuk membentuk semantik wajah.

Rainer Lienhart, Alexander Kuranov, dan Vadim Pisarevsky dalam papernya yang berjudul "*Empirical Analysis of Detection Cascades of Boosted Classifiers for Rapid Object Detection*". Sistem yang digunakan mampu

mendeteksi wajah frontal dengan sangat baik, metoda yang digunakan adalah metoda *haar cascade classifier*. Dimana dalam pendeteksian wajah digunakan fitur –fitur haar yang telah dilatih terlebih dahulu. Sistem ini bekerja dengan cara menggabungkan *weak classifier* untuk mendapatkan *strong classifier*.

Berdasarkan penelitian-penelitian tersebut, maka penulis mencoba melakukan penelitian yang akan difokuskan pada deteksi kantuk berdasarkan suara menguap dan perubahan posisi wajah. Sistem ini akan mendeteksi wajah secara realtime setelah terdeteksinya suara menguap dari pengguna kemudian melihat perubahan posisi wajah tersebut. Apabila posisi wajah tersebut dalam posisi tertidur maka sistem akan mengeluarkan suara peringatan berupa *alarm* sehingga orang akan tersadar dari tidurnya.

1.2 Perumusan Masalah

Masalah deteksi kantuk ini dapat dirumuskan sebagai berikut:

1. Sistem akan mendeteksi adanya suara menguap dari pengguna kemudian diberikan masukan citra digital yang diambil secara *realtime*.
2. Sistem akan membandingkan suara menguap dari pengguna dengan suara menguap yang ada pada database.
3. Jika hasil suara cocok maka sistem akan menjalankan program deteksi posisi wajah secara *real time*.
4. Sistem akan mendeteksi wajah, dalam penelitian ini wajah yang dapat dideteksi oleh sistem adalah wajah frontal.
5. Jika sistem tidak mendapatkan wilayah wajah dalam selang waktu tertentu maka sistem akan memberikan suara peringatan.

1.3 Tujuan Penelitian

Penelitian bertujuan untuk membuat suatu sistem pendeteksi kantuk yang mampu mengantisipasi seseorang agar tidak tertidur dalam melakukan aktifitas yang sedang dikerjakannya.

1.4 Manfaat Penelitian

Informasi-informasi yang diperoleh dari sistem pendeteksi kantuk ini diharapkan dapat dimanfaatkan bagi orang memiliki pekerjaan yang membutuhkan kesiagaan dan ketelitian tinggi sehingga mengharuskan mereka untuk tetap fokus, seperti sopir, pengawas pabrik, *security* dan lain sebagainya.

1.5 Batasan Masalah

Untuk mengarahkan pokok permasalahan dalam penelitian ini, penulis mengambil batasan pembahasan sebagai berikut :

1. Video dan suara diperoleh dari rekaman secara *realtime*.
2. Resolusi video adalah 320x240 piksel.
3. Masukan pada sistem berupa rekaman suara dan wajah secara langsung. Rekaman suara dengan format *.wav*, dan wajah dideteksi secara *real time*.
4. Suara menguap yang dikeluarkan adalah suara menguap yang dapat didengar cukup jelas dan dikenali sistem.
5. Wajah yang dapat dideteksi adalah wajah yang menghadap frontal ke kamera.
6. Posisi wajah tertidur tidak frontal ke kamera.
7. Jarak wajah dengan kamera dibatasi antara 40 sampai 50 cm.
8. Kamera yang digunakan adalah webcam jenis VGA.
9. Pemrograman menggunakan software Matlab 7.1 dan *library OpenCV*.

1.6 Metodologi Penelitian

Metodologi penelitian yang digunakan pada tugas akhir ini adalah:

1. Studi literatur sebagai gambaran awal tentang pemahaman konsep dari masalah yang akan dibahas pada tugas akhir ini.
2. Pembuatan program.
3. Simulasi program dan pengujian.
4. Analisis data.
5. Penyusunan laporan akhir.

1.7 Sistematika Penulisan

1. BAB I PENDAHULUAN

Bab ini berisi latar belakang penelitian, tujuan penelitian, manfaat penelitian, batasan masalah, metodologi penelitian dan sistematika penulisan.

2. BAB II TINJAUAN PUSTAKA

Bab ini berisikan teori tentang dasar citra yang mencakup proses pengolahan suara, pembentukan citra, digitalisasi citra dan teori tentang deteksi wajah.

3. BAB III METODOLOGI PENELITIAN

Metodologi penelitian berisikan penjelasan dan uraian mengenai jenis penelitian, sampel penelitian, desain penelitian, teknik analisa sistem dan prosedur penelitian.

4. BAB IV PERANCANGAN SISTEM

Bab ini berisikan tentang komponen sistem dan analisa kerja sistem.

5. BAB V ANALISA HASIL DAN PEMBAHASAN

Berisikan tentang citra hasil pendeteksian suara, hasil pendeteksian wajah dan hasil pengujian sistem.

6. BAB VI PENUTUP

Bab ini berisi kesimpulan dari hasil penelitian dan saran yang disampaikan berdasarkan hasil analisis dan pembahasan dari penelitian ini.

BAB II

TINJAUAN PUSTAKA

2.1. Proses Pembentukan Sinyal Suara Pada Manusia

Suara merupakan modal besar yang dimiliki oleh manusia untuk dapat mengkomunikasikan maksud dan tujuannya. Berbagai hal dapat dilakukan dengan hanya mengeluarkan suara, seperti memberikan perintah kepada manusia lain. Dengan suara, manusia dapat mengerti dengan mudah apa yang diucapkan oleh manusia yang lain.[17]

Suara merupakan sinyal akustik yang dikeluarkan oleh manusia. Suara manusia dihasilkan karena adanya interaksi antara organ-organ pembentuk sinyal. Alat-alat pembentuk sinyal suara pada manusia dapat dilihat pada Gambar 2.1.



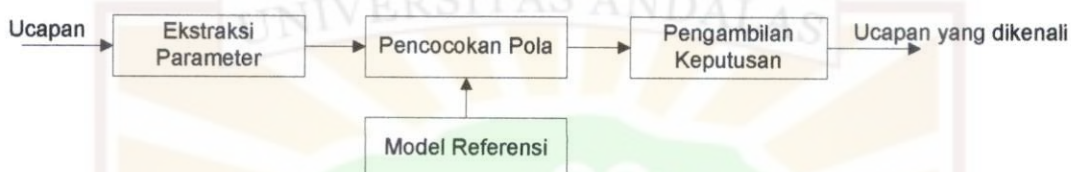
Gambar 2.1. Organ Penghasil Suara Manusia

Suara yang dihasilkan pada organ-organ suara dibangkitkan dengan adanya tekanan udara yang dihasilkan dari paru-paru yang kemudian dialirkan ke batang tenggorok. Udara yang mengalir melalui batang tenggorok akan masuk ke *larynx*, di mana terdapat pita-pita suara, sehingga ketika udara melewatinya ia akan bergetar dan akan menghasilkan pulsa-pulsa udara. Pulsa-pulsa udara ini akan masuk ke *vocal tract*. Pada bagian ini terjadi resonansi pulsa-pulsa udara yang kemudian membentuk pola-pola suara yang kemudian akan menghasilkan sinyal suara. *Vocal tract* pada setiap orang memiliki bentuk dan ukuran yang berbeda satu sama lain sehingga suara yang dihasilkan akan memiliki karakteristik yang berbeda pula pada setiap manusia. Sinyal suara yang dihasilkan dari organ

penghasil suara akan direpresentasikan dalam bentuk gelombang suara. Gelombang suara akan membentuk berbagai jenis bentuk, di mana bentuk inilah yang akan berbeda pada setiap orang [17].

2.2. Sistem Pengenalan Suara

Secara umum, suatu sistem pengenalan suara terdiri atas dua proses utama, yaitu: proses ekstraksi parameter dan pencocokan pola. Proses ini dapat dilihat pada gambar 2.2 berikut ini.



Gambar 2.2. Blok Diagram Sistem Pengenalan Suara

Proses sistem pengenalan suara pada sistem ini dimulai dengan mengambil sinyal suara manusia yang telah direkam dengan sebuah *microphone*. Sinyal ini diinputkan ke komputer melalui *sound card* untuk mengubah sinyal analog menjadi sinyal *digital* agar sistem mudah untuk memproses lebih lanjut. Setelah itu dilakukan ekstraksi parameter yang menghasilkan koefisien parameter dari kata yang diucapkan oleh sumber suara untuk dilatihkan. Proses pelatihan tersebut akan menghasilkan model kata. Model kata ini akan disimpan sebagai model referensi. Model referensi tersebut akan digunakan untuk proses pengujian dalam menentukan keputusan pada pengenalan sumber suara.

2.3. Mel Frequency Cepstrum Coefficients (MFCC)

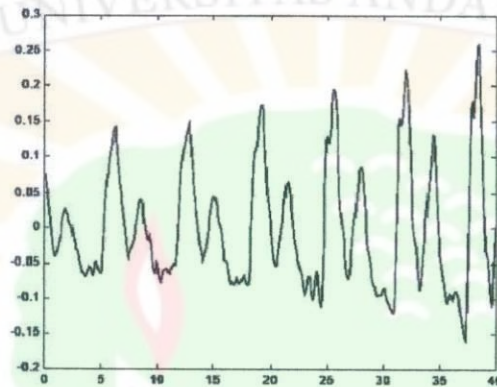
Mel Frequency Cepstrum Coefficients (MFCC) merupakan salah satu metode yang banyak digunakan dalam bidang *speech processing*, baik itu *speech recognition* maupun *speaker recognition*. Metode ini digunakan untuk melakukan *feature extraction*, sebuah proses yang mengonversikan sinyal suara menjadi beberapa parameter.

Keunggulan dari metode ini adalah:

1. Mampu untuk menangkap karakteristik suara yang sangat penting bagi pengenalan suara atau dengan kata lain mampu menangkap informasi-informasi penting yang terkandung dalam sinyal suara.

2. Menghasilkan data seminimal mungkin tanpa menghilangkan informasi-informasi penting yang ada.
3. Mereplikasi organ pendengaran manusia dalam melakukan persepsi terhadap sinyal suara.

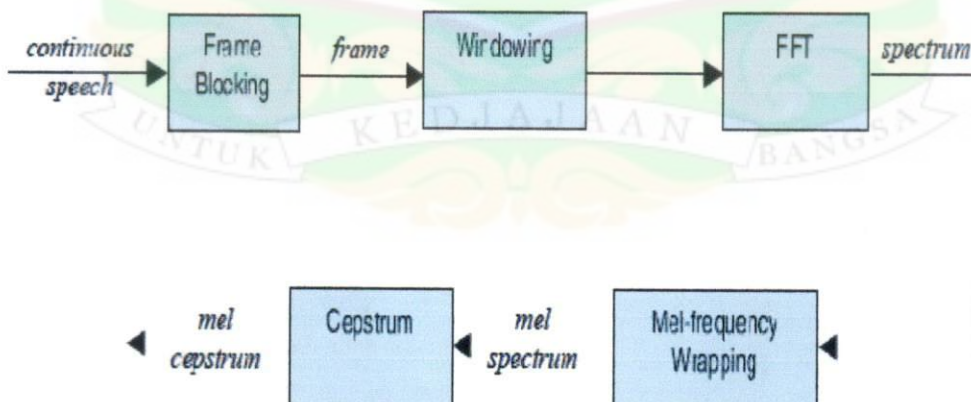
Perhitungan yang dilakukan dalam MFCC menggunakan dasar dari perhitungan *short-term analysis*. Hal ini dilakukan mengingat sinyal suara yang bersifat *quasi stationary* [18]. Contoh dari sinyal suara sepanjang 40 milidetik dapat dilihat pada gambar 2.3.



Gambar 2.3. Contoh Sinyal Suara Sepanjang 40 milidetik

Pengujian yang dilakukan untuk periode waktu yang cukup pendek (sekitar 10 sampai 30 milidetik) akan menunjukkan karakteristik sinyal suara yang *stationary*. Tetapi bila dilakukan dalam periode waktu yang lebih panjang karakteristik sinyal suara akan terus berubah sesuai dengan kata yang diucapkan.

Blok diagram MFCC dapat dilihat pada gambar 2.4 dibawah ini:



Gambar 2.4. Blok Diagram MFCC

2.3.1. Konversi Analog Menjadi Digital

Sinyal-sinyal yang natural pada umumnya, seperti sinyal suara, merupakan sinyal kontinu di mana memiliki nilai yang tidak terbatas. Pada komputer, semua sinyal yang dapat diproses oleh komputer adalah sinyal diskrit atau sering disebut *digital signal*. Agar sinyal natural dapat diproses oleh komputer, maka harus diubah terlebih dahulu dari data sinyal kontinu menjadi diskrit. Hal itu dapat dilakukan melalui 3 proses, diantaranya adalah *sampling* data, kuantisasi, dan pengkodean.

Proses *sampling* adalah suatu proses untuk mengambil data sinyal kontinu untuk setiap periode tertentu. Dengan kata lain, sinyal kontinu dicuplik untuk setiap periode tertentu yang nantinya akan menjadi sinyal diskrit. Dalam melakukan proses *sampling*, berlaku aturan *Nyquist*, yaitu bahwa frekuensi *sampling* (*sampling rate*) minimal harus dua kali lebih tinggi dari frekuensi maksimum sinyal yang akan *disampling*. Jika sinyal *sampling* kurang dari dua kali frekuensi maksimum sinyal yang akan *disampling*, maka akan timbul efek *aliasing*. *Aliasing* adalah suatu efek di mana sinyal yang dihasilkan memiliki frekuensi yang berbeda dengan sinyal aslinya.

Proses kuantisasi adalah proses untuk membulatkan nilai data ke dalam bilangan-bilangan tertentu yang telah ditentukan terlebih dahulu. Semakin banyak level yang dipakai maka semakin akurat pula data sinyal yang disimpan, namun akan menghasilkan ukuran data besar dan proses yang lama. Proses pengkodean adalah proses pemberian kode untuk tiap-tiap data sinyal yang telah terkuantisasi berdasarkan *level* yang ditempati.

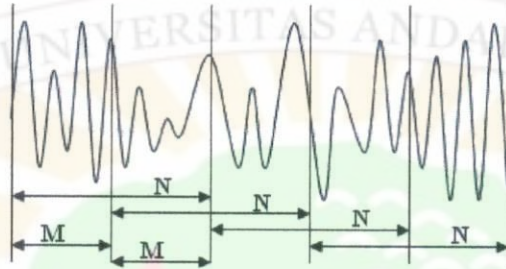
2.3.2. Frame Blocking

Pada tahap ini, sinyal suara diblok menjadi *frame-frame* dengan masing-masing *frame* memuat N cuplikan dan *frame-frame* yang berdekatan dipisahkan sejauh $M < N$. *Frame* pertama terdiri atas N cuplikan pertama. *Frame* kedua dimulai dengan M cuplikan setelah *frame* pertama, dan *overlap* sebesar $N-M$ cuplikan. *Frame* ketiga dimulai dengan $2M$ cuplikan setelah *frame* pertama (atau M cuplikan setelah *Frame* kedua) dan *overlap* sebanyak $N-2M$ cuplikan. Apabila

didefinisikan $X_i(n)$ sebagai *frame* yang ke- i dan ada L *frame* untuk keseluruhan sinyal, maka: $X_i(n) = \check{S}(ML + n)$, di mana $n = 0, 1, \dots, N-1$ dan $L = 0, 1, \dots, L-1$ [10].

Frame pertama $X_0(n)$ meliputi cuplikan sinyal $\check{S}(0), \check{S}(1), \dots, \check{S}(N-1)$. *Frame* kedua $X_1(n)$ meliputi cuplikan sinyal $\check{S}(M), \check{S}(M+1), \dots, \check{S}(M+N-1)$. *Frame* ke- L $X_{L-1}(n)$ meliputi cuplikan sinyal $\check{S}(M(L-1)), \check{S}(M(L-1)+1), \dots, \check{S}(M(L-1)+N-1)$. Nilai tipikal untuk N adalah 256 dan M adalah 100.

Proses ini dapat dilihat pada Gambar 2.5.



Gambar 2.5. Pembagian Sinyal Dalam Bentuk *Frame-Frame*

2.3.3. Windowing

Tahap selanjutnya adalah *me-window* setiap *frame* yang bertujuan untuk meminimalkan ketidakkontinuan sinyal pada permulaan dan akhir dari tiap-tiap *frame*. Prinsipnya adalah untuk meminimasi distorsi dengan menggunakan *window* pada ujung sinyal menjadi nol di awal dan akhir dari tiap-tiap *frame*. Jika didefinisikan *window* sebagai $w(n)$ dengan $0 \leq n \leq N-1$, maka sinyal hasil *windowing* adalah:

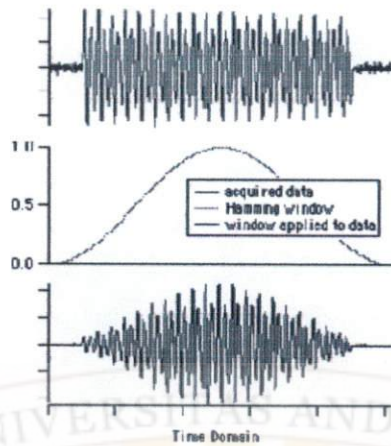
$$Y_1(n) = X(n).w(n), \text{ di mana } 0 \leq n \leq N-1 \quad \dots(2.1)$$

di mana N adalah jumlah cuplikan dalam tiap-tiap *frame* dan X adalah masing-masing *frame*.

Jenis *window* yang sering digunakan adalah *hamming window* yang memiliki bentuk sebagai berikut:

$$w(n) = 0,54 - 0,46 \cos\left(\frac{2\pi n}{N-1}\right) \text{ dengan } 0 \leq n \leq N-1 \quad \dots(2.2)$$

di mana $w(n)$ adalah sinyal *windowing* dan N adalah jumlah cuplikan tiap-tiap *frame*.



Gambar 2.6. Perbandingan Sinyal Suara Sebelum dan Sesudah Proses *Windowing*

2.3.4. *Fast Fourier Transform (FFT)*

Transformasi *fourier* adalah suatu metode yang sangat efisien untuk menyelesaikan transformasi *fourier* diskrit yang banyak dipakai untuk keperluan analisa sinyal seperti pemfilteran, analisa korelasi, dan analisa *spectrum*[18]. Transformasi *fourier* ini dilakukan untuk mentransformaikan sinyal dari *domain* waktu ke *domain* frekuensi. *Output* dari proses ini disebut dengan spektrum atau *periodogram*. FFT adalah DFT (*Discrete Fourier Transfoms*) dengan teknik perhitungan yang cepat dengan memanfaatkan sifat periodikal dari transformasi *fourier*[18]. Perhatikan definisi dari DFT berikut:

$$F(k) = \sum_{n=1}^N f(n) \cdot e^{-j2\pi knT/N} \quad \text{.....(2.3)}$$

atau dapat dituliskan dengan :

$$F(k) = \sum_{n=1}^N f(n) \cos(2\pi knT/N) - j \sum_{n=1}^N f(n) \sin(2\pi knT/N) \quad (2.4)$$

di mana $F(k)$ adalah fungsi dengan *domain* frekuensi, yang merupakan hasil transformasi; dan N menunjukkan jumlah titik.

Proses FFT adalah proses DFT yang dibagi menjadi 2 bagian. Hal ini masih bisa dilanjutkan dengan memperhatikan kembali sifat dari fungsi *sinus* dan *cosinus* pada setengah bagian, seperempat bagian, dan seterusnya sehingga prosesnya menjadi lebih cepat[11]. Untuk n buah data, DFT memerlukan n^2 kali

perhitungan, dan FFT memerlukan $(n/2 + 1) \times n + n/2$ kali perhitungan. Misalkan jumlah data $n=100$, maka dengan menggunakan DFT diperlukan $100 \times 100 = 10.000$ kali perhitungan, sedangkan dengan menggunakan FFT cukup dilakukan $(51 \times 100 + 50) = 5150$ kali perhitungan.

2.3.5. Mel-Frequency Wrapping

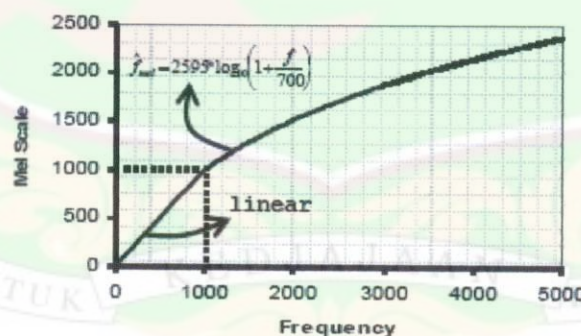
Tahap ini merupakan proses pemfilteran dari spektrum setiap *frame* yang diperoleh dari tahapan sebelumnya, dan menggunakan sejumlah M filter segitiga dengan tinggi satu. Filter ini dibuat dengan mengikuti persepsi telinga manusia dalam menerima suara. Persepsi ini dinyatakan dalam skala “mel” (berasal dari *Melody*) yang mempunyai hubungan tidak linear dengan frekuensi suara [1]. Satu relasi antara frekuensi bunyi (dalam Hz) dengan skala *mel* adalah :

$$mel(f) = 2595 * \log_{10} \left(1 + \frac{f}{700} \right) \quad \dots(2.5)$$

di mana f adalah frekuensi sinyal yang dihitung dalam satuan atau skala Hertz.

Skala frekuensi *mel* merupakan suatu skala yang berjarak secara linier untuk frekuensi di bawah 1000 Hz dan berjarak logaritmis untuk sinyal-sinyal pada frekuensi di atas 1000 Hz.

Bentuk grafik dari skala *mel* tersebut dapat dilihat pada Gambar 2.7.



Gambar 2.7. Hubungan Frekuensi Dengan Skala Mel

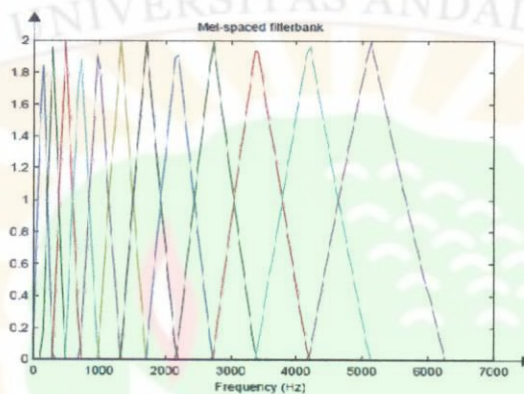
Nilai koefisien spektrum *mel* (*mel spectrum coefficients*) adalah jumlah dari hasil pemfilteran sesuai dengan rumus sebagai berikut:

$$X_i = \log \left(\sum_{f=0}^{N-1} \text{abs}(X(j)) * H_i(f) \right) \quad \dots(2.6)$$

di mana $i = 1, 2, 3, \dots, M$; M adalah banyaknya filter; N adalah banyaknya koefisien FFT; $\text{abs}(X(j))$ adalah magnitude koefisien ke- j dari periodogram yang

dihasilkan oleh transformasi FFT; dan $H_i(f)$ adalah nilai *filter* ke- i pada titik frekuensi f .

Salah satu pendekatan untuk mensimulasikan spektrum adalah dengan menggunakan *filter bank*. *Filter bank* merupakan suatu rangkaian *filter-filter* yang memiliki frekuensi yang berbeda, di mana untuk menghitung MFCC digunakan suatu *filter* untuk tiap-tiap komponen frekuensi *mel* yang diinginkan. *Filter bank* ini digunakan dalam *domain* frekuensi. Gambar 2.8 memperlihatkan bentuk dari *filter bank* MFCC.



Gambar 2.8. Contoh *Triangular Filter Bank* Dengan *Mel Scale*

2.3.6. Cepstrum

Pada tahap ini dilakukan konversi dari koefisien *spektrum mel* kembali ke *domain* waktu dengan menggunakan transformasi kosinus/*Discrete Cosine Transform* (DCT). *Discrete Cosine Transform* (DCT) adalah model transformasi *fourier* yang dikenakan pada fungsi diskrit dengan hanya mengambil bagian cosinus dari eksponensial kompleks[11].

$$C_j = \sum_{i=1}^M X_i * \cos\left(\frac{j*(i-0.5)*\pi}{20}\right) \dots\dots(2.7)$$

dengan $j = 1, 2, 3, \dots, K$; K adalah banyaknya koefisien MFCC yang diinginkan; M adalah banyaknya *filter* segitiga; X_i adalah koefisien *spektrum mel* yang diperoleh dengan persamaan (2.7). Dalam hal ini C_j disebut sebagai koefisien ke j dari *Mel Frequency Cepstrum Coefficients* (MFCC).

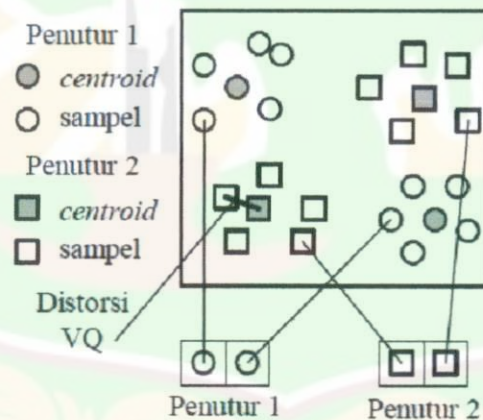
2.4. Vector Quantization (VQ)

Vector Quantization (VQ) adalah proses memetakan vektor-vektor dari ruang vektor besar menjadi jumlah terbatas daerah ruang vektor ^[19]. Masing-

masing daerah disebut kluster dan dapat direpresentasikan oleh pusatnya yang disebut *codeword*. Kumpulan dari semua *codeword-codeword* disebut *codebook*.

Dalam Gambar 2.9 menunjukkan konsep diagram untuk ilustrasi proses pengenalan. Hanya dua penutur dan dua dimensi dari ruang akustik ditunjukkan. Putaran-putaran mengacu pada vektor-vektor akustik dari penutur 1 dengan tanda lingkaran dan penutur 2 dengan tanda kotak. Dalam fasa pelatihan, VQ *codebook* penutur spesifik dibangkitkan untuk masing-masing penutur yang dikenali oleh pengklusteran vektor-vektor akustik pelatihan dari laki-laki atau wanita.

Hasil *codeword-codeword* (*centroid*) ditunjukkan dalam Gambar 2.9 dengan tanda lingkaran hitam dan tanda kotak hitam untuk penutur 1 dan 2. Jarak terdekat antara vektor *codeword* dari *codebook* disebut distorsi VQ. Dalam fasa pengenalan ini, input wicara dari suara tak dikenal adalah “*vector* terkuantisasi” dengan menggunakan masing-masing *codebook* yang dilatih dan jarak total distorsi VQ. Penutur dengan VQ *codebook* dan total distorsinya terkecil akan diidentifikasi.



Gambar 2.9. Formasi VQ Codebook Antara Penutur 1 Dan Penutur 2.

2.4.1. Pelatihan Vektor-Vektor

Selanjutnya vektor-vektor akustik diekstraksi dari input wicara dari seorang penutur sebagai *set* pelatihan vektor-vektor. Sebagaimana penjelasan diatas, langkah penting berikutnya adalah membangun VQ *codebook* dari penutur yang spesifik dengan menggunakan pelatihan vektor-vektor ini. Algoritma ini dikenal sebagai algoritma LBG (Linde, Buzo, dan Gray) [19], untuk kluster *set* L

pelatihan vektor-vektor menjadi *set M codebook* vektor-vektor. Algoritma ini secara formal diimplementasikan dengan prosedur rekursif berikut ini:

1. Desain satu vektor *codebook*, ini adalah *centroid* dari masukan set pelatihan vektor-vektor (karenanya tak diperlukan iterasi disini).
2. Gandakan ukuran *codebook* dengan membagi masing-masing *codebook* sekarang y_n sesuai dengan aturan

$$\begin{aligned} y_n^+ &= y_n(1 + \varepsilon) \\ y_n^- &= y_n(1 - \varepsilon) \end{aligned} \quad \dots(2.8)$$

dimana n berubah dari 1 ke ukuran *codebook* sekarang dan ε adalah parameter pembagi (*splitting*) (misalnya $\varepsilon = 0.01$)

3. Pencarian *neighbor* (tetangga) terdekat: untuk masing-masing pelatihan vektor, tentukan *codebook* dalam *codebook* yang terdekat dan menetapkan vektor-vektor tersebut yang berhubungan dengan sel (berhubungan dengan *codebook* terdekat).
4. *Centroid* terbaru (*update*): mem-perbarui *codebook* dalam masing-masing sel dengan menggunakan *centroid* dari pelatihan vektor-vektor ini yang ditentukan untuk sel tersebut.
5. Iterasi 1: mengulangi langkah 3 dan 4 sampai jarak rata-rata jatuh dibawah nilai ambang.
6. Iterasi 2: mengulangi langkah 2, 3, dan 4 sampai *codebook* dengan ukuran M didesain. Algoritma LBG mendesain M vektor *codebook* dalam langkah ini. Langkah pertama dengan mendesain satu vektor *codebook*, kemudian menggunakan teknik pemecahan (*splitting*) pada *codebook-codebook* untuk menginialisasi pencarian untuk 2 vektor *codebook* dan melanjutkan proses pemecahan sampai M vektor *codebook* yang diinginkan akan dapat ditentukan.

2.5. Jarak Euclidean

Dalam tahap pengenalan penutur, suara penutur yang tak dikenal direperentasikan oleh deretan vektor-vektor ciri $\{x_1, x_2 \dots x_i\}$ dan kemudian ini dibandingkan dengan *codebook* dari database. Untuk mengidentifikasi pembicara

yang tak dikenali, ini dapat dilakukan dengan pengukuran jarak distorsi dari dua kumpulan vektor yang berdasarkan peminiman jarak Euclidean.

Jarak *Euclidean* adalah jarak antar dua titik yang akan diukur dengan suatu aturan, yang dapat dibuktikan oleh aplikasi teorema *Pythagorean* [21]. Persamaan yang digunakan untuk menghitung jarak *Euclidean* dapat didefinisikan dengan jarak *Euclidean* antara dua titik $P = (p_1, p_2 \dots p_n)$ dan $Q = (q_1, q_2 \dots q_n)$.

$$\begin{aligned} & \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \dots + (p_n - q_n)^2} \\ & = \sqrt{\sum_{i=1}^n (p_i - q_i)^2} \end{aligned} \quad \dots(2.9)$$

2.6. Pengolahan Citra (*Image Processing*)

Pengolahan citra (*image processing*) adalah pengolahan suatu citra (gambar) dengan menggunakan komputer secara khusus, untuk menghasilkan suatu citra yang lain.

Sesuai dengan perkembangan komputer itu sendiri, pengolahan citra mempunyai dua tujuan utama, yaitu sebagai berikut:

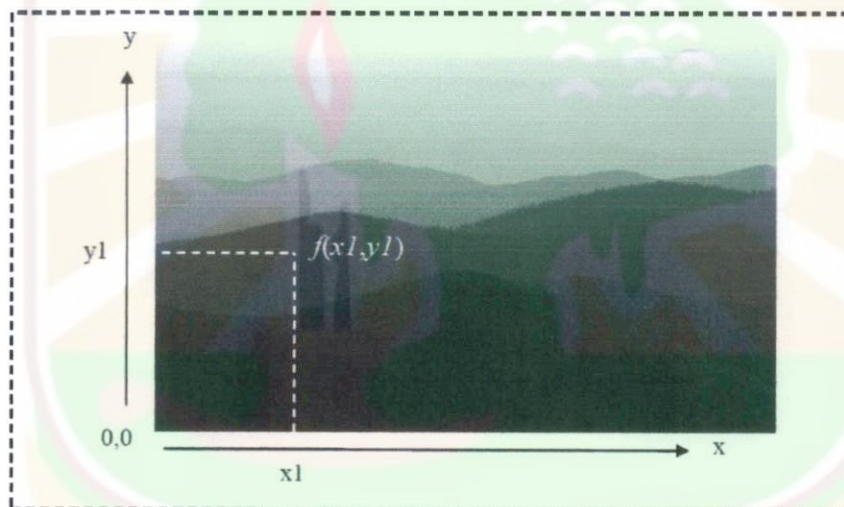
1. Memperbaiki kualitas citra, dimana citra yang dihasilkan dapat menampilkan informasi secara jelas. Hal ini berarti manusia sebagai pengolah informasi (*human perception*).
2. Mengekstraksi informasi ciri yang menonjol pada suatu citra, dimana hasilnya adalah informasi citra dimana manusia mendapatkan informasi ciri dari citra secara numerik atau dengan kata lain *computer* (mesin) melakukan interpretasi terhadap informasi yang ada pada citra melalui besaran-besaran data yang dapat dibedakan secara jelas (berupa besaran numerik).

Dalam perkembangan lebih lanjut, *image processing* dan *computer vision* digunakan sebagai pengganti mata manusia dengan perangkat *input image capture* seperti kamera dan scanner dijadikan sebagai mata dan mesin komputer dijadikan sebagai otak yang mengolah informasi. Sehingga muncul beberapa pecahan bidang yang menjadi penting dalam *computer vision* antara lain: *pattern recognition* (pengenalan pola), *biometric* (pengenalan identifikasi manusia berdasarkan ciri-ciri biologis yang tampak pada badan manusia), *content based*

image and video retrieval (mendapatkan kembali citra atau video dengan informasi tertentu), *video editing* dan lain-lain.

2.6.1. Dasar – Dasar Pengolahan Citra Digital

Citra digital diasumsikan dengan persamaan $f(x,y)$ dimana x menyatakan nomor baris, y menyatakan nomor kolom, dan f menyatakan nilai derajat keabuan dari citra. Sehingga (x,y) adalah posisi dari piksel dan f adalah nilai derajat keabuan pada titik (x,y) . Kecerahan setiap citra disimpan dengan cara pemberian nomor pada setiap piksel. Makin tinggi nomor piksel maka makin gelap (hitam) piksel tersebut. Begitu juga sebaliknya makin rendah nilai piksel tersebut maka makin terang. Sistem yang umum memiliki 256 tingkat kecerahan untuk setiap piksel, yang paling terang adalah 255 dan yang paling gelap adalah 0.



Gambar 2.10. Citra Digital

Citra atau gambar terbagi dalam tiga tipe adalah sebagai berikut:

a. Gambar *Grayscale*

Citra yang terdiri dari satu layer warna dengann derajat keabuan tertentu. Dinyatakan dalam suatu fungsi :

$$f(x,y) \in [0...255] \dots\dots\dots(2.10)$$

b. Gambar Biner

Citra yang hanya memiliki dua nilai yaitu 1 dan 0. Dinyatakan dalam suatu fungsi :

$$f(x,y) \in \{0,1\} \dots\dots\dots(2.11)$$

c. Gambar berwarna

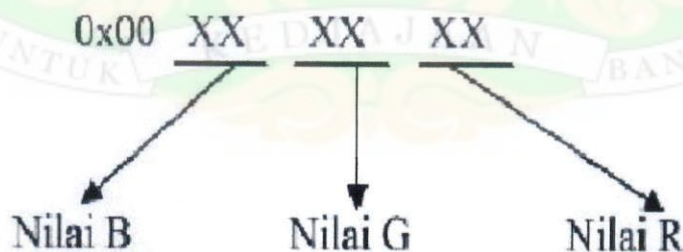
Citra yang terdiri dari tiga layer warna yaitu RGB (*Red-Green-Blue*) dimana *R-layer* adalah matrik yang menyatakan derajat kecerahan untuk warna merah, *G-layer* adalah matrik yang menyatakan derajat kecerahan untuk warna hijau, dan *B-layer* adalah matrik yang menyatakan derajat kecerahan untuk warna biru. Representasi dalam citra digital dinyatakan dalam persamaan :

$$\begin{aligned} fR(x,y) &\Sigma [0...255] \\ fG(x,y) &\Sigma [0...255](2.12) \\ fB(x,y) &\Sigma [0...255] \end{aligned}$$

Proses pengolahan citra digital dengan menggunakan komputer digital adalah terlebih dahulu mentransformasikan citra ke dalam bentuk besaran-besaran diskrit dari nilai tingkat keabuan pada titik-titik elemen citra. Bentuk citra ini disebut citra digital. Elemen-elemen citra digital apabila ditampilkan dalam layar monitor akan menempati sebuah ruang yang disebut dengan piksel (*picture elemen/pixel*).

2.6.2. Model Warna RGB

Dasar dari pengolahan citra adalah pengolahan warna RGB pada posisi tertentu. Dalam pengolahan citra, warna dipresentasikan dengan nilai hexadesimal dari 0x00000000 sampai 0x00ffffff. Warna hitam adalah 0x00000000 dan warna putih adalah 0x00ffffff. Definisi nilai 8 warna di atas seperti Gambar 2.11, variabel 0x00 menyatakan angka dibelakangnya adalah hexadesimal.



Gambar 2.11. Nilai Warna RGB Dalam Heksadesimal

Pengkodean warna RGB dapat ditulis dalam angka hexadesimal (basis 16) untuk masing-masing komponen R, G atau B. Misalkan:

1. Untuk Hitam murni akan ditulis dengan kode #000000 (R=00, G=00, B=00)
2. Untuk Putih sempurna akan ditulis dengan kode #FFFFFF (R=FF, G=FF, B=FF)
3. Untuk Biru murni akan ditulis dengan kode #0000FF (R=00, G=00, B=FF)

Model Warna RGB adalah sebuah model warna *additif* dimana pancaran warna *red* (merah), *green* (hijau), dan *blue* (biru) ditambahkan bersama dengan cara yang bervariasi untuk mereproduksi susunan warna yang lebar. Warna *additif* digunakan untuk *lighting*, video, dan monitor. Monitor sebagai contoh, menciptakan warna dengan memancarkan cahaya melalui merah, hijau dan biru fosfor.

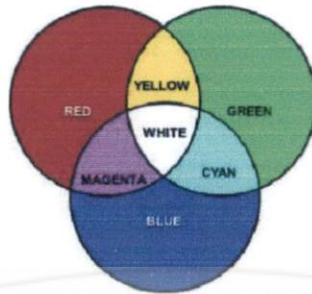
Tujuan utama dari model warna RGB adalah untuk mempresentasikan ulang, dan menampilkan gambar dalam sistem elektronik, misalnya dalam televisi dan komputer. Model warna RGB juga digunakan dalam fotografi konvensional.

RGB adalah sebuah ruang warna yang sifatnya bergantung kepada perangkat. Perangkat yang berbeda akan mendeteksi atau mereproduksi nilai RGB secara berbeda. Untuk membentuk warna dengan RGB, tiga cahaya berwarna balok (satu merah, satu hijau, dan satu biru) harus ditumpangkan (misalnya dengan emisi dari layar hitam, atau dengan refleksi dari layar putih). Masing-masing dari tiga balok disebut sebagai komponen warna, dan masing-masing dapat memiliki intensitas yang berbeda.

RGB sering digunakan dalam perangkat input seperti: TV berwarna dan kamera video, *scanner*, dan kamera digital. Perangkat output seperti: TV dalam berbagai teknologi (CRT, LCD, plasma, dll), komputer dan layar HP, video proyektor, layar LED multiwarna, dan layar lebar seperti JumboTron dan lain-lain.

RGB juga umumnya digunakan dalam pembuatan desain sebuah *website*. Awalnya kedalaman warna yang terbatas pada sebagian besar perangkat keras video memicu keterbatasan palet warna menjadi 216 warna RGB, hal itu ditetapkan oleh *Netscape Color Cube*. Namun, dengan keunggulan menampilkan

24-bit, penggunaan penuh dari 16.7 juta warna dari kode warna RGB HTML tidak lagi menimbulkan masalah bagi sebagian besar pengunjung website.



Gambar 2.12. Model Warna RGB

2.6.3. Penskalaan Citra (*Scaling*)

Penskalaan citra (*scaling*) adalah sebuah operasi geometri yang memberikan efek memperbesar atau memperkecil ukuran citra *input* sesuai dengan variabel penskalaan citranya.

Penskalaan citra (*scaling*) digunakan untuk memperbesar (*zoom-in*) atau memperkecil (*zoom-out*) citra. Rumus yang digunakan untuk penskalaan citra adalah :

$$\begin{aligned} x' &= S_h x \\ y' &= S_v y \end{aligned} \dots\dots\dots(2.13)$$

Keterangan :

S_h = faktor skala horizontal

S_v = faktor skala vertikal

Gambar Asli



320 x 240 pixel

Gambar Hasil *Scaling*



200 x 100 pixel

Gambar 2.13. Penskalaan Citra

2.6.4. Grayscale

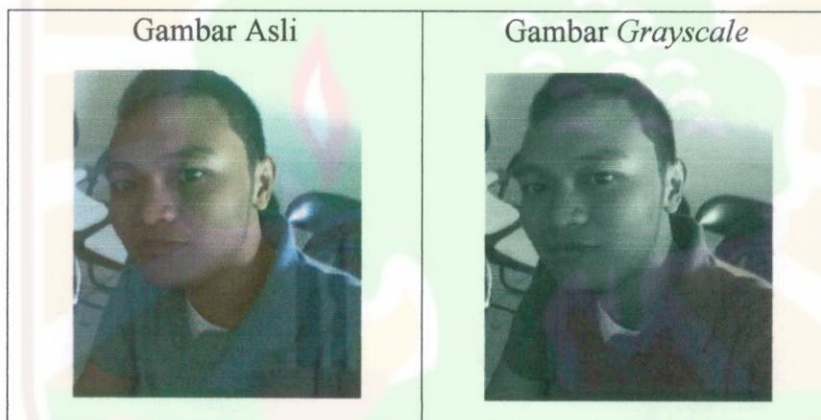
Citra *Grayscale* adalah citra yang hanya memiliki satu nilai kanal pada setiap pixelnya, dengan kata lain nilai bagian *Red* = *Green* = *Blue*. Nilai tersebut digunakan untuk menunjukkan tingkat intensitas. Warna yang dimiliki adalah warna dari hitam, keabuan dan putih. Tingkat keabuan di sini merupakan warna abu dengan berbagai tingkatan dari hitam hingga mendekati putih.

Pengubahan dari citra berwarna ke bentuk *grayscale* mengikuti aturan sebagai berikut :

$$\text{Grayscale} = 0,2989 R + 0,5870 G + 0,1140 B$$

.....2.14

Contoh gambar hasil *grayscale* :



Gambar 2.14. Citra *Grayscale*

2.6.5. Thresholding

Thresholding digunakan untuk mengatur jumlah derajat keabuan yang ada pada citra. Proses *thresholding* ini pada dasarnya adalah proses pengubahan kuantisasi pada citra, sehingga untuk melakukan *thresholding* dengan derajat keabuan dapat digunakan rumus:

$$x = \frac{w}{b} \dots\dots\dots 2.15$$

Keterangan :

w = Nilai derajat keabuan sebelum *thresholding*

b = Jumlah keabuan { b = (256) }

x = Nilai derajat keabuan setelah *thresholding*

Proses *thresholding* akan menghasilkan citra biner, yaitu citra yang memiliki dua nilai tingkat keabuan yaitu hitam dan putih. Secara umum proses *thresholding* citra *grayscale* untuk menghasilkan citra biner adalah sebagai berikut:

$$g(x, y) = \begin{cases} 1 & \text{if } f(x, y) \geq T \\ 0 & \text{if } f(x, y) < T \end{cases} \dots\dots\dots 2.16$$

Berikut ini contoh *thresholding* untuk mengubah citra *grayscale* menjadi 2 warna yaitu hitam dan putih :



Gambar 2.15. Citra *Thresholding*

2.7. Metoda *Haar Cascade Classifier*

Proses deteksi adanya citra wajah dalam sebuah gambar pada *OpenCV*, menggunakan sebuah metode yang dipublikasikan oleh Paul Viola dan Michael Jones tahun 2001^[15]. Umumnya disebut metode *Haar Classifier*. Metode ini merupakan metode yang menggunakan *statistical model (classifier)*. Pendekatan untuk mendeteksi objek dalam gambar menggabungkan empat konsep utama :

1. *Training data*
2. Fitur segi empat sederhana yang disebut fitur *Haar*.
3. *Integral image* untuk pendeteksian fitur secara cepat.
4. Pengklasifikasi bertingkat (*cascade classifier*) untuk menghubungkan banyak fitur secara efisien.

2.7.1. *Training Data Pada Haar*

Metode ini memerlukan 2 tipe gambar objek dalam proses *training* yang dilakukan, yaitu :

1. *Positive Samples*

Berisi gambar obyek yang ingin di deteksi, apabila ingin mendeteksi wajah maka *positive samples* ini berisi gambar wajah, begitu juga obyek lain yang ingin dikenali.

2. *Negative Samples*

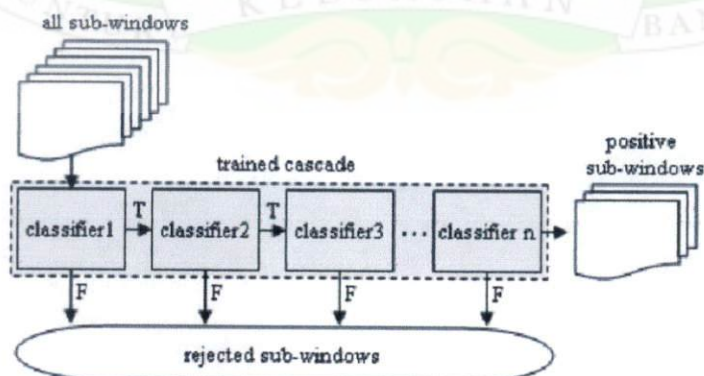
Berisi gambar obyek selain obyek yang ingin dikenali, umumnya berupa gambar *background* (tembok, pemandangan, lantai, dan gambar lainnya). Resolusi untuk sampel negatif disarankan untuk memiliki resolusi yang sama dengan resolusi kamera.

Training dari *Haar* menggunakan dua tipe sampel diatas. Informasi dari hasil *training* ini lalu dikonversi menjadi sebuah parameter model statistik.

2.7.2. Sistem Kerja Algoritma *Haar Cascade Classifier*

Algoritma *Haar* menggunakan metode statistik dalam melakukan pengenalan mata. Metode ini menggunakan *simple haar-like features* dan juga *a cascade of boosted tree classifier*. Classifier ini menggunakan gambar berukuran tetap. Cara kerja dari *haar* dalam mendeteksi wajah adalah menggunakan teknik *sliding window* pada keseluruhan gambar dan mencari apakah terdapat bagian dari gambar yang berbentuk seperti wajah atau tidak. *Haar* juga memiliki kemampuan untuk melakukan *scalling* sehingga dapat mendeteksi adanya wajah yang berukuran lebih besar ataupun lebih kecil dari gambar pada *classifier*.

Tiap *feature* dari *haar-like feature* didefinisikan pada bentuk dari *feature*, diantaranya koordinat dari *feature* dan juga ukuran dari *feature* tersebut. Skema algoritma *Haar Cascade Clasifier* dapat dilihat pada gambar 2.16 dibawah ini:

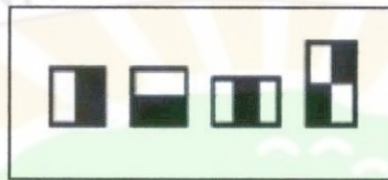


Gambar 2.16. Sistem Kerja Algoritma *Haar Cascade Classifier*

2.7.3. Haar Feature

Haar Feature adalah fitur yang digunakan oleh Viola dan Jones didasarkan pada *Wavelet Haar*. *Wavelet Haar* adalah gelombang tunggal bujur sangkar (satu interval tinggi dan satu interval rendah). Untuk dua dimensi, satu terang dan satu gelap. Selanjutnya kombinasi-kombinasi kotak yang digunakan untuk pendeteksian objek visual yang lebih baik.

Setiap *Haar-like feature* terdiri dari gabungan kotak - kotak hitam dan putih, seperti pada gambar 2.17 dibawah ini:



Gambar 2.17. *Basic Haar Feature*

Adanya fitur *Haar* ditentukan dengan cara mengurangi rata-rata piksel pada daerah gelap dari rata-rata piksel pada daerah terang. Jika nilai perbedaannya itu diatas nilai ambang atau threshold, maka dapat dikatakan bahwa fitur tersebut ada.

Nilai dari *Haar-like feature* adalah perbedaan antara jumlah nilai-nilai piksel *gray level* dalam daerah kotak hitam dan daerah kotak putih:

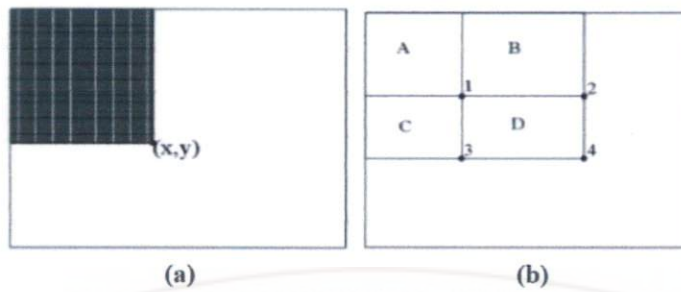
$$f(x) = \text{Sumblack rectangle} - \text{Sumwhite rectangle} \dots\dots\dots(2.17)$$

Kotak *Haar-like feature* dapat dihitung secara cepat menggunakan “*integral image*”.

2.7.4. Integral Image

Integral Image digunakan untuk menentukan ada atau tidaknya dari ratusan fitur *Haar* pada sebuah gambar dan pada skala yang berbeda secara efisien. Pada umumnya, pengintegrasian tersebut berarti menambahkan unit-unit kecil secara bersamaan. Dalam hal ini unit-unit kecil tersebut adalah nilai-nilai piksel. Nilai integral untuk masing-masing piksel adalah jumlah dari semua piksel-piksel dari atas sampai bawah. Dimulai dari kiri atas sampai kanan bawah,

keseluruhan gambar itu dapat dijumlahkan dengan beberapa operasi bilangan bulat per piksel.



Gambar 2.18. *Integral Image*

Seperti yang ditunjukkan oleh gambar 2.18(a) di atas setelah pengintegrasian, nilai pada lokasi piksel (x,y) berisi jumlah dari semua piksel di dalam daerah segiempat dari kiri atas sampai pada lokasi (x,y) atau daerah yang diarsir. Untuk menentukan nilai rata-rata piksel pada area segiempat (daerah yang diarsir) ini dapat dilakukan hanya dengan membagi nilai pada (x,y) oleh area segiempat.

$$P(x, y) = \sum i(x', y') \dots\dots\dots 2.18$$

Untuk mengetahui nilai piksel untuk beberapa segiempat yang lain missal, seperti segiempat D pada gambar 2.18(b) di atas dapat dilakukan dengan cara menggabungkan jumlah piksel pada area segiempat A+B+C+D, dikurangi jumlah dalam segiempat A+B dan A+C, ditambah jumlah piksel di dalam A. Dengan, A+B+C+D adalah nilai dari integral image pada lokasi 4, A+B adalah nilai pada lokasi 2, A+C adalah nilai pada lokasi 3, dan A pada lokasi 1. Sehingga hasil dari D dapat dikomputasikan :

$$D = (A+B+C+D)-(A+B)-(A+C)+A \dots\dots\dots 2.19$$

Contoh integral image:

Image				Integral Image			
0	8	6	1	0	8	14	15
1	5	9	0	1	14	29	30
0	7	5	0	1	21	41	42
2	8	9	2	3	31	60	63

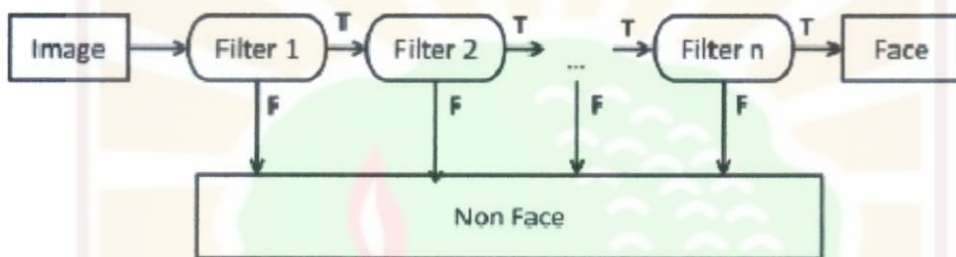


Gambar 2.19. *Contoh Integral Image*

2.7.5. Cascade Classifier

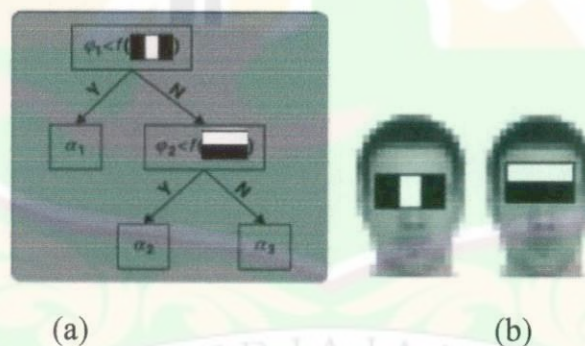
Cascade classifier adalah sebuah rantai *stage classifier*, dimana setiap *stage classifier* digunakan untuk mendeteksi apakah didalam *image sub window* terdapat obyek yang diinginkan (*object of interest*).

Stage classifier dibangun dengan menggunakan algoritma adaptive-boost (*AdaBoost*). Algoritma tersebut mengkombinasikan performance banyak weak classifier untuk menghasilkan *strong classifier*. *Weak classifier* dalam hal ini adalah nilai dari haar-like feature. Jenis *AdaBoost* yang digunakan adalah *Gentle AdaBoost*. Cara kerja dari algoritma ini dapat dilihat pada gambar 2.20 berikut:



Gambar 2.20. *Cascade Classifier*

Untuk lebih jelasnya tentang proses *cascade classifier* dapat dilihat pada gambar 2.21 berikut ini:



Gambar 2.21. (a) Ilustrasi *Cascade Boosted Classifier* (b) Contoh Fitur Yang Digunakan Pada Stage Pertama

Dari gambar dapat kita lihat proses pada stage pertama dari cascade classifier menggunakan fitur haar. Sub image dihitung nilai fitur haar-nya, jika nilainya kurang dari threshold kemudian lanjut dengan fitur kedua, jika masih dibawah threshold maka sub image tersebut akan dibuang atau bukan wajah.

BAB III

METODOLOGI PENELITIAN

3.1 Jenis Penelitian

Berdasarkan tujuan dan manfaat tugas akhir ini, maka penelitian ini termasuk ke dalam jenis penelitian terapan, dimana penelitian ini merupakan pemecahan terhadap suatu masalah untuk tujuan tertentu dan merupakan aplikasi baru dari penelitian yang telah ada dengan berpedoman pada data sekunder (data dari hasil penelitian) yang relevan. Ditinjau dari sifatnya, penelitian ini bersifat eksperimen.

3.2 Sampel Penelitian

Berdasarkan batasan masalah mengenai sistem kerja pendeteksi kantuk yang dijabarkan dalam bab I tugas akhir ini, maka sampel penelitian yang akan digunakan yaitu berupa suara manusia yang direkam menggunakan *microphone* dan posisi wajah manusia yang didapat secara langsung dengan cara deteksi wajah secara *real time* menggunakan *webcam*.

Penelitian ini memakai objek secara langsung, yaitu ketika input suara menguap dikenali sistem, maka sistem akan mengaktifkan kamera *webcam* untuk mendeteksi wajah secara *real time* untuk mengetahui posisinya, jika posisi wajah tidak frontal atau dalam posisi tertidur pada selang waktu tertentu, maka sistem akan mengeluarkan suara peringatan berupa *alarm*.

3.3 Desain Penelitian

Sistem ini dimulai dengan proses input sinyal suara menguap menggunakan *microphone*. Kemudian dilakukan ekstraksi parameter terhadap masukan sinyal suara sehingga didapatkan parameter dari sinyal tersebut. Pada sistem pengenalan suara ini, ekstraksi parameter dilakukan dengan menggunakan metode *Mel Frequency Cepstrum Coefficients* (MFCC), dimana parameter yang ingin didapatkan adalah koefisien *Cepstrum Mel-frequency*. Kemudian dilakukan *vector quantization* untuk mendapatkan jarak *Euclidean*. Selanjutnya apabila parameter masukan sinyal suara menguap tersebut sama atau hampir sama dengan parameter

dari sinyal suara yang sudah di-*training* pada *database* maka akan dilanjutkan dengan proses pengolahan citra.

Pada sistem pengolahan citra, video wajah manusia akan disegmentasi untuk memisahkan wilayah wajah dan bukan wajah secara real time. Apabila dalam selang waktu tertentu tidak ditemukan wilayah wajah, maka diasumsikan bahwa manusia tersebut dalam posisi tertidur dan sistem akan mengeluarkan suara peringatan untuk membangunkan.

3.4 Teknik Analisa Sistem

Kinerja sistem yang akan dieksperimenkan dianalisa dengan menggunakan penilaian objektif. Hasil-hasil yang didapatkan melalui eksperimen (keluaran sistem) dibandingkan dengan teori- teori yang berasal dari literatur yang ada.

3.5 Prosedur Penelitian

Agar penelitian dapat lebih terarah dan efektif, penulis telah menyusun dan akan mengikuti prosedur penelitian sebagai berikut:

3.5.1 Tinjauan dan Studi Kepustakaan

Studi literatur berupa diskusi dengan pembimbing, mengambil bahan dari internet dan buku-buku panduan. Studi literatur juga dilakukan terhadap bahasa pemrograman *MATLAB 7.1* dan *library openCV* yang dalam penelitian ini menjadi sarana aplikatif.

3.5.2 Penyusunan Algoritma Program

Program yang dirancang dalam tugas akhir ini terbagi atas 3 (tiga) algoritma pokok, yaitu:

- a. Algoritma ekstraksi parameter menggunakan metode *Mel Frequency Cepstrum Coefficients* (MFCC). Algoritma ini digunakan untuk memperoleh nilai koefisien *Cepstrum Mel-frequency* dari masukan berupa sinyal suara menguap.
- b. Algoritma deteksi wajah untuk menentukan posisi wajah.
- c. Algoritma suara alarm untuk memberikan peringatan.

3.5.3 Diagram Alir Sistem

Dapat dilihat pada "Gambar 4.1 Blok Diagram Penelitian" bab IV.

3.5.4 Perancangan dan Pembuatan Program

Berdasarkan algoritma yang telah ditentukan di atas, dibuat program pengenalan suara dan pengolahan citra (pendeteksi wajah), dengan menggunakan bahasa pemrograman Matlab.

3.5.5 Analisa Keluaran Sistem

Analisa keluaran sistem mencakup analisa pengenalan suara menguap dengan menggunakan metode *Mel Frequency Cepstrum Coefficients* (MFCC), analisa deteksi posisi wajah.

3.5.6 Penyusunan Laporan Penelitian

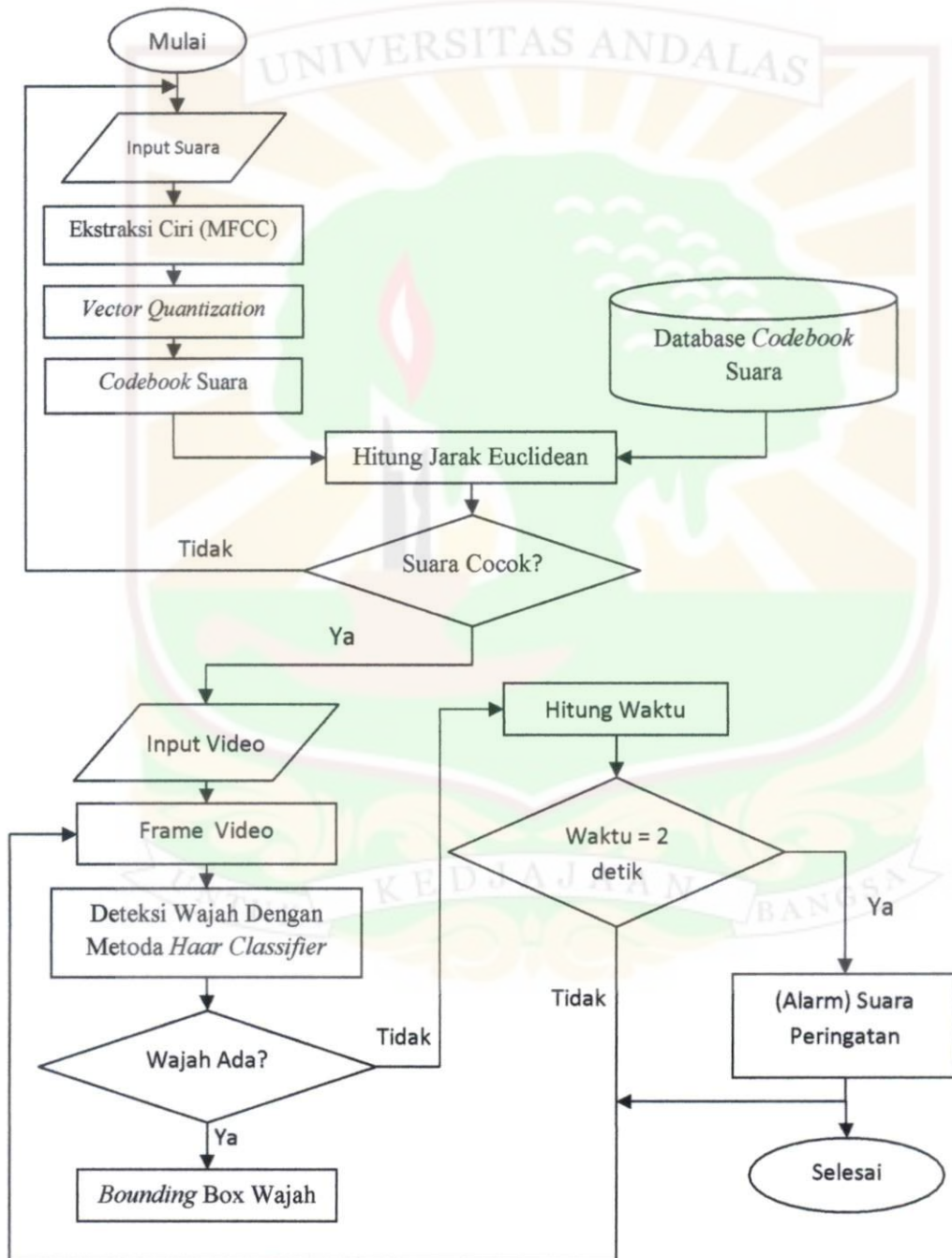
Berisikan penjelasan secara tertulis dan tergambar keseluruhan proses penelitian ini, dengan rincian isi yang telah dijabarkan didalam subbab 1.7 tugas akhir ini mengenai sistematika penulisan.

BAB IV

PERANCANGAN SISTEM

4.1. Blok Diagram Sistem

Penelitian ini dilakukan melalui beberapa tahapan untuk mendapatkan hasil akhir. Rencana umum dari penelitian ini dapat digambarkan seperti blok diagram pada gambar 4.1 berikut.



Gambar 4.1. Blok Diagram Penelitian

4.2. Ekstraksi Parameter Sinyal Suara Menggunakan MFCC

Proses pendeteksian suara menguap dimulai dengan penangkapan sinyal suara menguap secara *online* dengan menggunakan *microphone* yang terkoneksi dengan komputer (laptop). Kemudian secara otomatis sinyal *disampling* dengan frekuensi *sampling* sebesar 12000 Hz dan *bit rate* data 16 *bit*. Sinyal suara menguap ini akan dijadikan sebagai masukan pada sistem dengan format *.wav*. Kemudian dilakukan ekstraksi parameter terhadap masukan sinyal suara ini sehingga didapatkan parameter dari sinyal tersebut. Pada sistem pengenalan suara ini, ekstraksi parameter dilakukan dengan menggunakan metode MFCC, dimana parameter yang ingin didapatkan adalah koefisien *Cepstrum Mel-frequency*.

4.2.1. Frame Blocking

Pada tahap ini, sinyal suara menguap dikelompokkan ke dalam *frame-frame* dengan ukuran masing-masing *frame* (N) sebesar 256 data cuplikan. *Frame* ini berurutan dengan pemisahan antara kedua bingkai (M) sebesar 100 data cuplikan, sehingga *overlapping* sebesar 156 cuplikan.

4.2.2. Windowing

Langkah selanjutnya adalah proses *windowing*, yaitu membuat sinyal menuju nol pada awal dan akhir setiap *frame*. Pada sistem ini, jenis *window* yang dipakai adalah *Hamming window* dengan bentuk :

$$w(n) = 0,54 - 0,46 \cos\left(\frac{2\pi n}{N-1}\right) \text{ dengan } 0 \leq n \leq N-1 \quad \text{.....(3.1)}$$

Bentuk ini akan diterapkan oleh :

$$y[n] = x[n] * w[n] \quad \text{.....(3.2)}$$

4.2.3. Fast Fourier Transform (FFT)

Tahap berikutnya, setelah dilakukan *windowing*, adalah *Fast Fourier Transform* (FFT). Proses FFT ini bertujuan untuk mengonversi masing-masing N sampel sinyal dari domain waktu ke domain frekuensi. Hasil yang diperoleh dari proses ini disebut dengan spektrum.

4.2.4. *Mel Frequency Wrapping*

Spektrum yang dihasilkan dari proses FFT kemudian diproses dengan *Mel Frequency Wrapping*. Tahap ini merupakan proses pemfilteran dari spektrum setiap *frame* dengan menggunakan sejumlah M filter segitiga. Filter ini dibuat dengan mengikuti persepsi telinga manusia dalam menerima suara. Persepsi ini dinyatakan dalam skala '*mel*' yang mempunyai hubungan tidak linear dengan frekuensi suara. Dalam hal ini, skala *mel-frequency* adalah linear untuk frekuensi di bawah 1000 Hz dan logaritmik untuk frekuensi di atas 1000 Hz. Pada sistem ini digunakan pendekatan *filterbank* sebanyak 20 filter. Satu filter digunakan untuk masing-masing komponen *mel-frequency* yang diinginkan. Hasil dari proses ini ialah *mel* spektrum.

4.2.5. *Discrete Cosine Transform (DCT)*

Langkah terakhir dari proses utama ekstraksi parameter MFCC ialah transformasi kosinus diskrit atau *Discrete Cosine Transform (DCT)*. Tujuan dari proses DCT ini adalah untuk mendapatkan nilai *cepstrum*. Nilai *cepstrum* inilah yang merupakan hasil akhir dari ekstraksi parameter MFCC. *Cepstrum* merupakan koefisien parameter MFCC berupa vektor akustik atau vektor ciri. Sistem ini menggunakan orde 20 yang berarti menghasilkan 20 koefisien pertama MFCC untuk tiap satu *frame*.

4.2.6. *Cepstrum*

Untuk mendapatkan nilai *cepstrum* MFCC, maka *mel* frekuensi tersebut harus ditransformasikan kembali menjadi *domain* waktu. Transformasi tersebut dilakukan dengan menggunakan transformasi kosinus diskrit atau *Discrete Cosine Transform (DCT)*. DCT merupakan langkah terakhir dari proses utama MFCC *feature extraction*.

4.3. *Vector Quantization (VQ)*

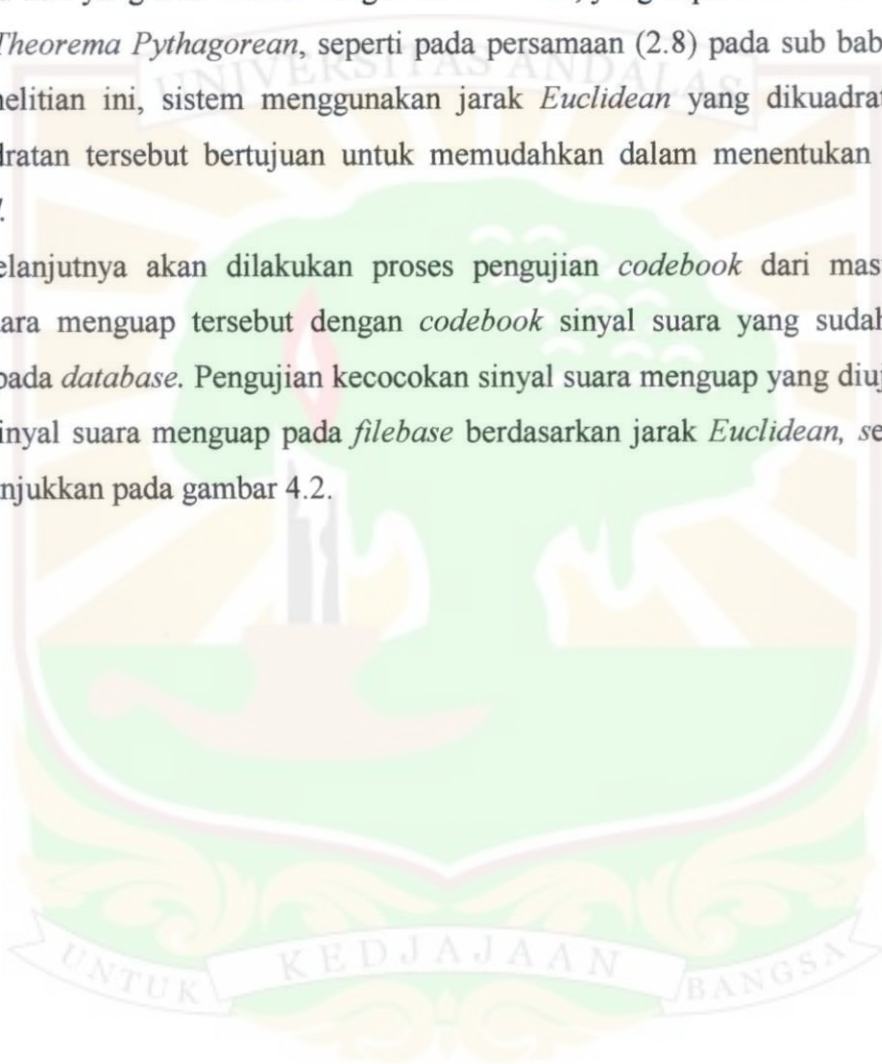
Selanjutnya dilakukan proses kuantisasi vektor (*Vector Quantization*) dari hasil MFCC, dengan tujuan untuk memetakan vektor-vektor dari ruang vektor besar menjadi jumlah terbatas daerah ruang vektor. Masing-masing daerah disebut

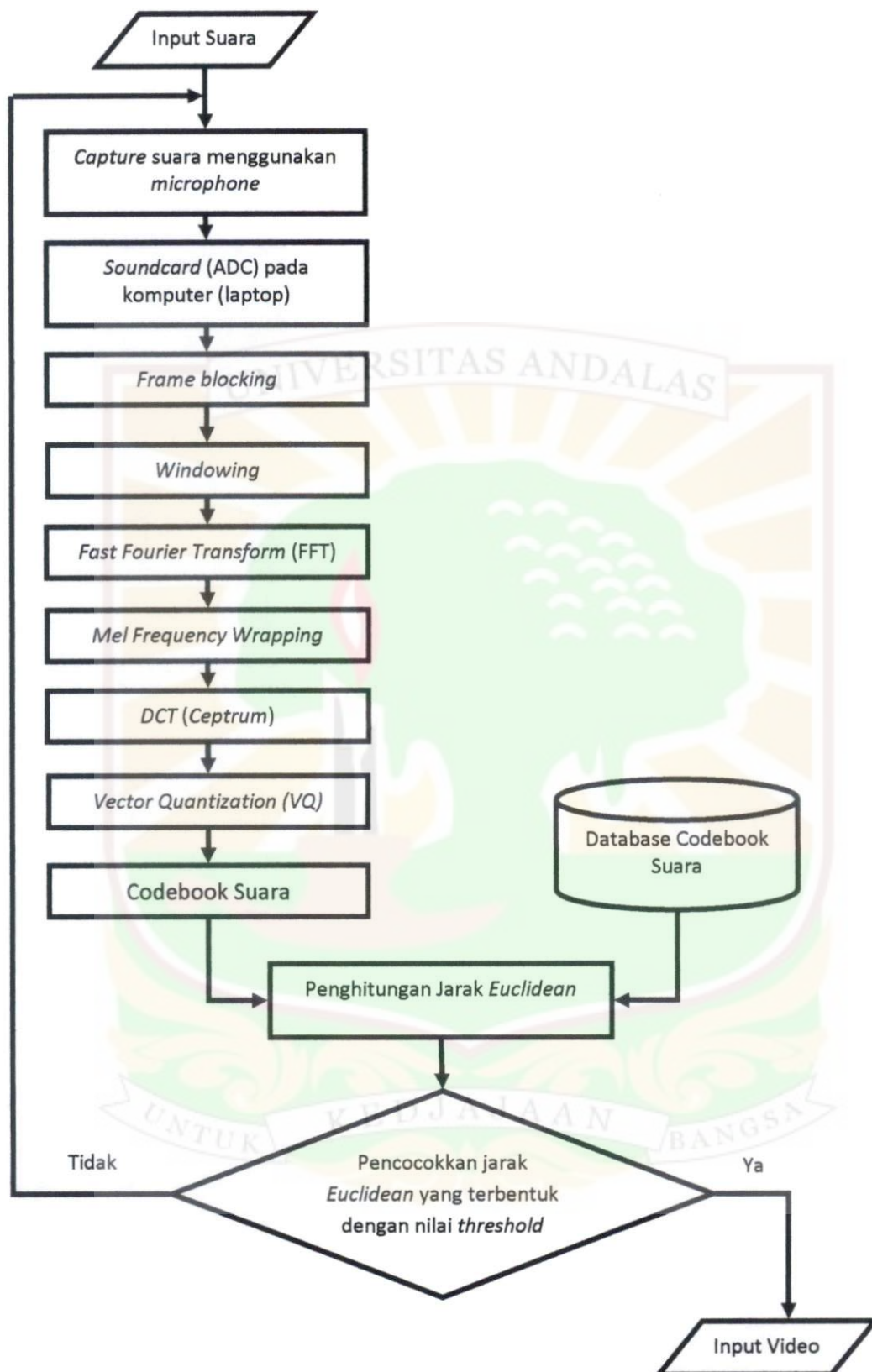
kluster dan dapat direpresentasikan oleh pusatnya yang disebut *codeword*. Kumpulan dari semua *codeword-codeword* disebut *codebook*.

4.4. Penghitungan Jarak *Euclidean*

Tahap berikutnya adalah penghitungan jarak *Euclidean* dari *codebook* antara *input* suara penutur dengan suara *filebase*. Jarak *Euclidean* adalah jarak antara dua titik yang akan diukur dengan suatu aturan, yang dapat dibuktikan oleh aplikasi *Theorema Pythagorean*, seperti pada persamaan (2.8) pada sub bab 2.5. Pada penelitian ini, sistem menggunakan jarak *Euclidean* yang dikuadratkan. Pengkuadratan tersebut bertujuan untuk memudahkan dalam menentukan nilai *threshold*.

Selanjutnya akan dilakukan proses pengujian *codebook* dari masukan sinyal suara menguap tersebut dengan *codebook* sinyal suara yang sudah di-*training* pada *database*. Pengujian kecocokan sinyal suara menguap yang diujikan dengan sinyal suara menguap pada *filebase* berdasarkan jarak *Euclidean*, seperti yang ditunjukkan pada gambar 4.2.





Gambar 4.2. Blok Diagram Deteksi Suara Menguap

Pengujian kecocokan ini berdasarkan nilai *threshold* dari jarak *Euclidean* yang diperoleh. Jika jarak *Euclidean* yang diperoleh berada didalam rentang nilai *threshold*, maka suara masukan dianggap cocok dengan suara pada *database*, dan apabila jarak *euclidean* yang diperoleh berada diluar rentang nilai *threshold* maka suara masukan dianggap tidak cocok dengan suara pada *database*. Jika hasil yang diperoleh adalah kecocokan antara masukan sinyal suara menguap dengan sinyal suara yang sudah di-*training* pada *database*, maka akan dilanjutkan dengan proses pengolahan citra.

4.5. Deteksi Posisi Wajah

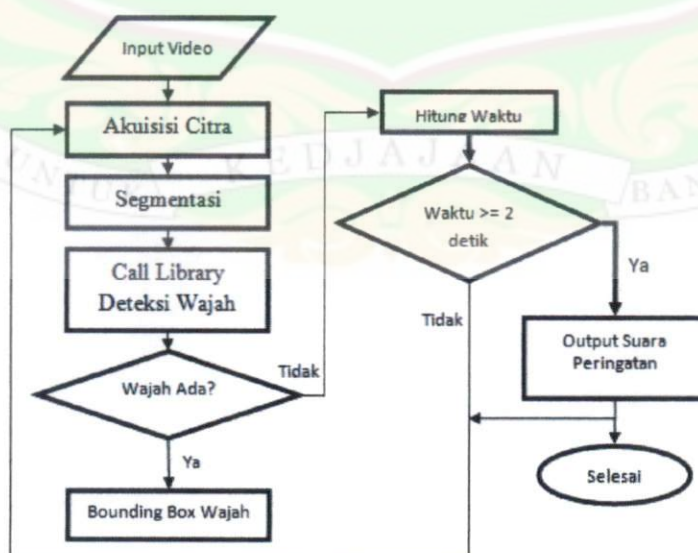
4.5.1. Pra Proses

Langkah pertama yang dilakukan dalam praproses adalah mendeteksi wajah dari sebuah citra image dalam hal ini menggunakan library openCV. Beberapa file yang telah tersedia adalah sebagai berikut:

1. *haarcascade_frontalface_alt.xml*
2. *FaceDetectDll.h*
3. *FaceDetectDll_RC1.dll*

Kemudian dari file-file yang telah tersedia akan digunakan untuk mendeteksi wajah.

Adapun proses deteksi posisi wajah dapat dilihat pada blok diagram pada gambar 4.3 berikut.



Gambar 4.3. Blok Diagram Deteksi Posisi Wajah

4.5.2. Input Video

Video digital diambil dengan menggunakan alat penangkap video (video imager) yaitu kamera *webcam*.

4.5.3. Akuisisi Citra

Akuisisi citra merupakan proses pengambilan beberapa *frame* (citra digital) secara otomatis (*online*) dari video wajah yang sedang diamati dengan menggunakan *webcam* atau dikenal dengan istilah *video capture*.

4.5.4. Segmentasi

Proses segmentasi diawali dari konversi citra dari ruang warna RGB ke *grayscale*.



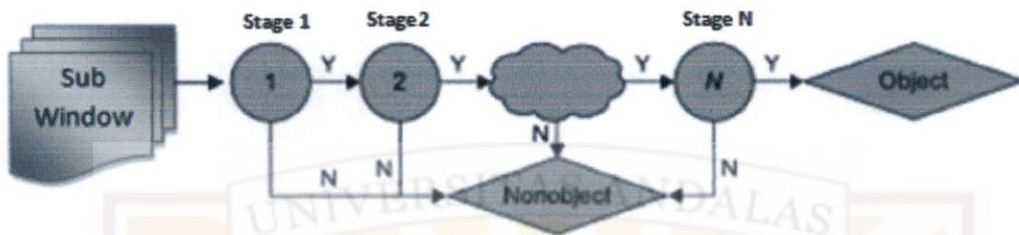
Gambar 4.4. Proses *Grayscale* Pada Citra Wajah

4.5.5. Deteksi Wajah Dengan Metoda *Haar Cascade Classifier*

Untuk mendeteksi wajah menggunakan metoda *haar cascade classifier*. Algoritma Haar menggunakan metode statistik dalam melakukan pendeteksian wajah. Metode ini menggunakan *simple haar-like features* dan juga *a cascade of boosted tree classifier*. *Classifier* ini menggunakan gambar berukuran tetap. Cara kerja dari *haar* dalam mendeteksi wajah adalah menggunakan teknik *sliding window* dengan ukuran 20x20 pixel pada keseluruhan gambar dan mencari apakah terdapat bagian dari gambar yang berbentuk seperti wajah atau tidak. Haar juga memiliki kemampuan untuk melakukan *scalling* sehingga dapat mendeteksi adanya wajah yang berukuran lebih besar ataupun lebih kecil dari gambar pada *classifier*.

Masing-masing stage memiliki fitur *haar* untuk mendeteksi bagian wajah, apabila suatu sub image tidak mampu melewati salah satu stage maka sub image

tersebut langsung dikategorikan sebagai bukan wajah. Namun jika image mampu melewati hingga stage terakhir maka digolongkan ke dalam wajah. Untuk lebih jelasnya dapat dilihat dari alur kerja algoritma *haar* seperti pada gambar 4.5 dibawah ini.



Gambar 4.5. Alur Kerja *Haar Cascade Clasifier*

Setiap stage memiliki nilai threshold dan beberapa fitur (lampiran C4). Apabila hasil penjumlahan nilai haar fitur lebih kecil dari stage threshold maka image langsung ditolak atau digolongkan sebagai bukan wajah. Sebaliknya jika nilai haar fitur lebih besar dari stage threshold maka akan dilanjutkan klasifikasi dengan stage berikutnya hingga stage terakhir.

4.5.6. Hitung Waktu

Dari hasil pendeteksian wajah yang dilakukan apabila wajah tidak terdeteksi maka timer akan aktif. Fungsi dari pewaktu ini adalah untuk menghitung lamanya wajah tidak terdeteksi. Jika wajah tidak terdeteksi selama lebih dari 2 detik maka sistem akan mengeluarkan bunyi alarm untuk memberikan peringatan agar tersadar dari tidurnya.

4.6. Algoritma Program

Dalam tugas akhir ini terdapat 2 algoritma utama. Masing-masing algoritma dapat dijelaskan sebagai berikut:

4.6.1. Algoritma MFCC

Algoritma pendeteksian suara menguap dalam tugas akhir ini melalui beberapa tahap. Masing-masing tahap tersebut adalah sebagai berikut:

- a. Pengambilan suara menguap secara *online* dengan menggunakan *microphone* yang terkoneksi dengan komputer (laptop). Kemudian

secara otomatis sinyal *disampling* dengan frekuensi *sampling* sebesar 12000 Hz dan bit *rate* data 16 bit.

- b. Lakukan proses ekstraksi parameter menggunakan metode MFCC.
- c. Lakukan proses kuantisasi vektor (*Vector Quantization*) dari hasil MFCC, dengan tujuan untuk memetakan vektor-vektor dari ruang vektor besar menjadi jumlah terbatas daerah ruang vektor. Masing-masing daerah disebut kluster dan dapat direpresentasikan oleh pusatnya yang disebut *codeword*. Kumpulan dari semua *codeword-codeword* disebut *codebook*.
- d. Pemberian nilai *threshold* yang diperoleh berdasarkan data pengujian sampel suara menguap dengan *data base*. Pemberian nilai *threshold* sebagai acuan dalam proses pengujian *codebook* dari masukan sinyal suara menguap tersebut dengan *codebook* sinyal suara yang sudah di-*training* pada *database*. *Database* yang digunakan dalam penelitian ini berupa *file notepad* yang berisikan *codebook* suara menguap penulis.
- e. Pengujian masukkan suara menguap dengan suara menguap pada *database* berdasarkan jarak *Euclidean*.

4.6.2. Algoritma Haar Classifier

Algoritma *Haar* pada penelitian ini terdiri dari beberapa tahap, yaitu:

- a. Deteksi wajah dengan ukuran *sliding window* yaitu 20x20 pixel.
- b. Jika tidak ditemukan wilayah wajah hingga akhir *window*, kemudian perbesar ukuran skala menjadi 1,2 atau 24x24 pixel.
- c. Hitung nilai *haar* pada setiap gambar yang dilewati dengan metoda *cascade classifier*.
- d. Ulangi langkah (b) dengan skala lebih besar hingga ditemukan wilayah wajah.
- e. Apabila sub image mampu melewati semua stage, lakukan *rectangular* untuk memberi kotak pada area wajah.

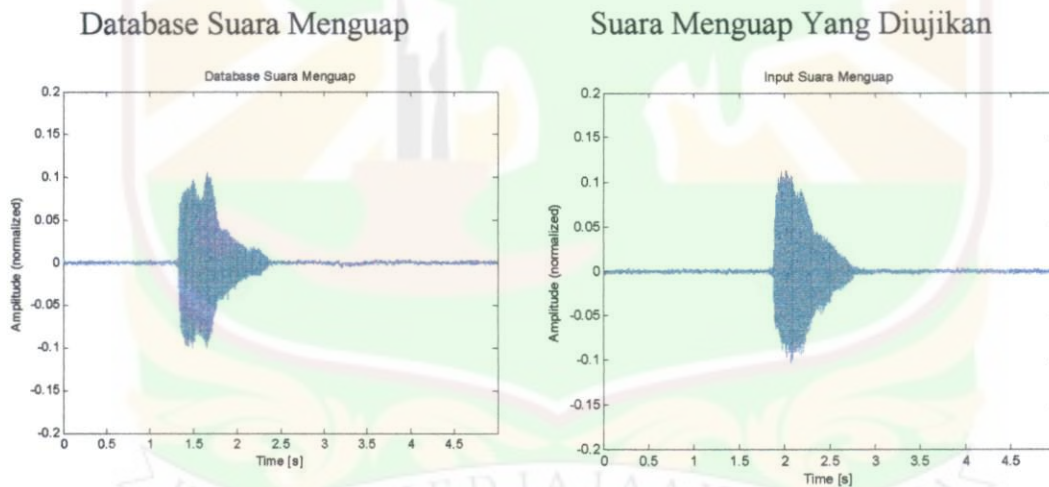
BAB V

ANALISA HASIL DAN PEMBAHASAN

5.1 Hasil Pendeteksian Suara Menguap

Pada tugas akhir ini, sistem dirancang mampu membedakan antara suara menguap asli dengan suara menguap yang tidak asli. Sistem pendeteksian suara menguap ini bersifat verifikasi, yang artinya sistem hanya mendeteksi suara menguap yang sesuai dengan suara *filebase* (.wav). Ekstraksi parameter sinyal suara yang digunakan dalam penelitian ini adalah *Mel Frequency Cepstral Coeffisients* (MFCC).

Suara menguap merupakan suara yang sangat unik dimana terdapat kemiripan karakteristik antara suara menguap yang satu dengan suara menguap yang lainnya dalam satu pembicara. Kemiripan bentuk sinyal suara menguap pada *filebase* dengan sinyal suara menguap yang diujikan, dapat dilihat pada gambar 5.1.

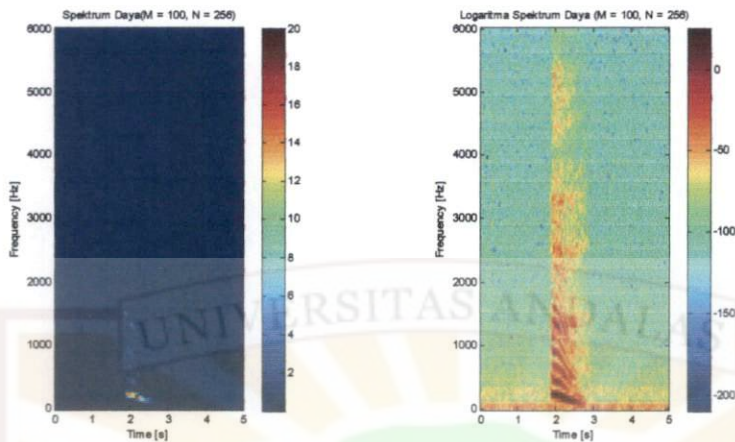


Gambar 5.1. Bentuk Sinyal Suara Menguap Filebase Dan Suara Menguap Yang Diujikan

5.1.1. Hasil Ekstraksi Parameter Menggunakan MFCC

Untuk mendapatkan karakteristik sinyal yang spesifik diperlukan ekstraksi parameter sehingga mudah untuk dianalisa karakteristiknya. Tahapan proses MFCC dapat dilihat pada subbab 4.2. Setelah pemrosesan dengan *windowing*, sinyal akan dianalisis dalam *domain* frekuensi dengan cara mengubah *domain*

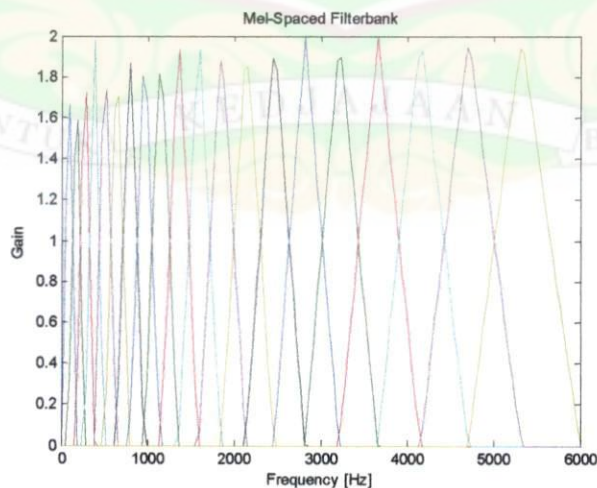
waktu ke *domain* frekuensi menggunakan transformasi *fourier*. Hasilnya dapat dilihat pada gambar 5.2 spektrogram sinyal berikut :



Gambar 5.2. Spektrogram Sinyal Suara Menguap, $M = 100$ Dan $N = 256$

Gambar 5.2 menunjukkan bahwa pada spektrogram sinyal suara menguap, daerah yang mengandung tingkat energi tertinggi ditampilkan dalam warna merah, daerah merah terletak antara 1,9 dan 3 detik, sebagian besar energi terkonsentrasi pada frekuensi rendah (antara 0 Hz dan 2 kHz).

Pada proses *mel-frequency wrapping* digunakan *filter bank*. *Filter bank* berfungsi untuk mengubah spektrum sinyal ke sebuah representasi yang lebih dekat/mencerminkan perilaku telinga manusia. Untuk meniru telinga manusia, *filter* didistribusikan linear untuk frekuensi rendah (dibawah 1 kHz), sedangkan untuk frekuensi yang lebih tinggi (diatas 1 kHz) distribusi dari *filter* adalah logaritmik. Contoh *filter bank* dapat dilihat pada gambar 5.3.

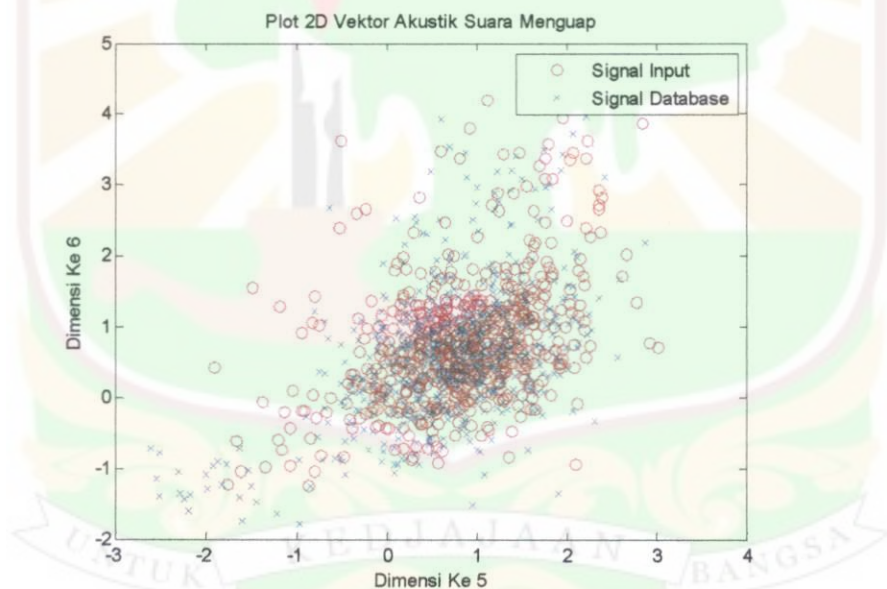


Gambar 5.3. Contoh *Filterbank* Dalam Proses *Mel Frequency*.

Gambar 5.3 menunjukkan bahwa *filter bank* ini berperilaku seperti suksesi dari histogram pada spektrum. *Filter bank* memodifikasi spektrum frekuensi ke dalam 20 buah filter. Setiap *filter-filter bank* memiliki respon frekuensi segitiga. *Filter* segitiga ini merupakan penjangka zona spektrum frekuensi. Proses selanjutnya, hasil *filter bank* ini diubah kembali ke dalam *domain* waktu menggunakan *Discrete Cosine Transform* (DCT).

5.1.2. Hasil *Vector Quantization* (VQ)

Pada sistem ini proses VQ dengan menggunakan algoritma LBG (*Linde, Buzo, and Grey*) akan menghasilkan sebuah *codebook* dengan 16 *codeword*. Untuk memeriksa ruang akustik (MFCC vektor), dari 20 dimensi vektor akustik, dipilih dua dimensi (contoh dimensi lima dan enam) yang akan mewakili masing-masing suara menguap. Berikut ini gambar titik data pada bidang 2D. *Plot* Vektor Akustik pada bidang 2 dimensi ditunjukkan pada gambar 5.4.

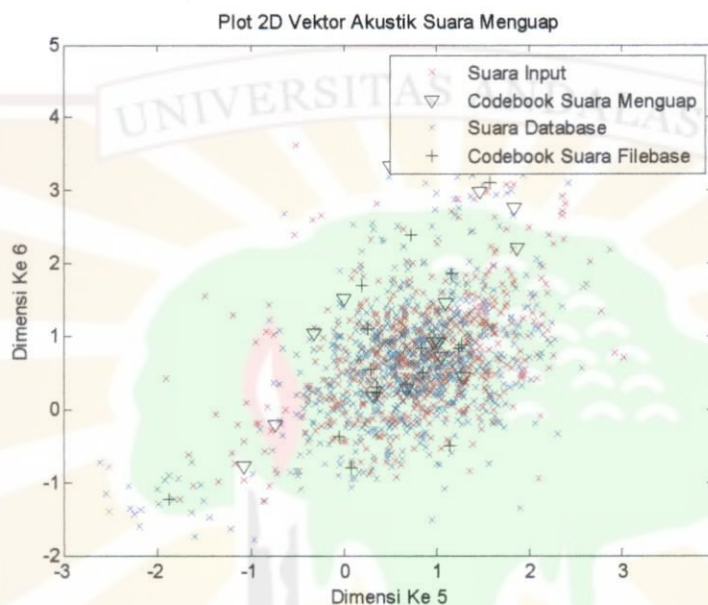


Gambar 5.4. Plot Vektor Akustik

Plot vektor akustik dimensi lima dan enam terbentuk dari dua sinyal, yaitu sinyal suara menguap pada *filebase* (*signal 1*) ditandai dengan lingkaran warna merah dan sinyal suara menguap yang diujikan (*signal 2*) ditandai dengan tanda silang berwarna biru. Antara titik-titik vektor akustik kedua sinyal ada yang saling tumpang tindih dan ada yang saling berjauhan atau berada pada daerah-daerah tertentu untuk digunakan secara eksklusif oleh salah satu sinyal. Hal inilah yang

menjadi dasar untuk mendeteksi suara menguap yang satu dengan suara menguap yang lainnya.

Proses selanjutnya adalah pelatihan vektor akustik dengan menggunakan algoritma *Linde, Buzo, and Grey* (LBG), sehingga didapatkan vektor akustik yang spesifik untuk tiap sinyal suara. *Plot* vektor akustik yang telah dilatih (*codebook*) ditunjukkan pada gambar 5.5.



Gambar 5.5. Plot Vektor Akustik Yang Telah Dilatih dan *Codebook*-nya

Pada gambar 5.5 dapat dilihat bahwa *codebook* dari sinyal suara menguap pada *filebase* ditunjukkan oleh tanda (+) dan *codebook* sinyal suara menguap yang diujikan ditunjukkan oleh tanda (v). Dan dapat diamati bahwa terdapat 16 *centroid* tiap *codebook* yang spesifik pada tiap sinyal suara menguap.

5.1.3. Jarak *Euclidean*

Jarak *euclidean* dihitung dari vektor akustik spesifik (*codebook*) dari tiap suara. Jarak ini dihitung dengan perkalian vektor antara *codebook* spesifik tersebut. Jarak *euclidean* yang diperoleh dari data yang diujikan akan dibandingkan dengan nilai *threshold*. Jika dari hasil pengujian diperoleh jarak *euclidean* sinyal suara yang diujikan lebih kecil dari nilai *threshold* maka sinyal suara tersebut dianggap sama (hampir sama) dengan suara pada *filebase* dan sistem akan mendeteksi suara bahwa suara yang diujikan tersebut adalah suara menguap. Namun jika dari hasil pengujian diperoleh jarak *euclidean* sinyal suara

yang diujikan lebih besar dari nilai *threshold* maka sinyal suara tersebut dianggap tidak sama dengan suara pada *filebase*.

5.1.4. Penentuan Nilai *Threshold*

Penentuan nilai *threshold* sistem diambil dengan menganalisis jarak *euclidean* atau nilai yang terbentuk dari pengujian sistem. Berdasarkan hasil pengujian yang dilakukan sebanyak 10 kali berturut-turut, dengan frekuensi sampling 12 kHz, ternyata suatu sinyal suara menguap memiliki suatu ciri yang istimewa yaitu kemiripan antara sinyal suara yang satu dengan sinyal suara menguap yang lainnya dalam satu pembicara. Kemiripan sinyal suara menguap ini ditandai dengan jarak *Euclidean* yang tidak terlalu jauh berbeda antara satu dengan yang lain. Data 10 kali pengujian sinyal suara menguap pada *filebase* dengan sinyal suara menguap yang diujikan dapat dilihat pada Tabel 5.1.

Tabel 5.1. Penentuan Nilai *Threshold*

No.	Sinyal Suara Menguap	Jarak <i>Euclidean</i>
1.	Suara1.wav	4.6219
2.	Suara2.wav	4.4116
3.	Suara3.wav	5.0421
4.	Suara4.wav	4.9893
5.	Suara5.wav	5.3879
6.	Suara6.wav	5.9915
7.	Suara7.wav	4.5390
8.	Suara8.wav	3.7773
9.	Suara9.wav	5.8398
10.	Suara10.wav	5.5737
Nilai <i>Threshold</i>		5.9915

Tabel 5.1. menunjukkan bahwa nilai *threshold* yang diperoleh dari 10 kali pengujian adalah sebesar 5,9915. Nilai *threshold* ini akan dijadikan acuan dalam pengambilan data pendeteksian sinyal suara menguap. Dari 10 kali pengujian yang dilakukan berturut-turut, diperoleh jarak *euclidean* yang hampir sama dengan suara menguap pada *filebase*, yaitu berkisar antara 3,7773-5,9915. Hal ini

menandakan bahwa terdapat kemiripan yang signifikan antara suara menguap yang satu dengan suara menguap yang lainnya dalam satu pembicara.

Hasil pengambilan data pendeteksian sinyal suara menguap yang dilakukan sebanyak 15 kali secara *online* dengan menggunakan *microphone* yang terkoneksi dengan perangkat lunak Matlab 7.01, pengujian data dilakukan berdasarkan nilai *threshold* dapat dilihat pada tabel 5.2 berikut ini.

Tabel 5.2. Hasil Pengambilan Data Pendeteksian Sinyal Suara Menguap

No.	Sinyal Suara Menguap	Jarak <i>Euclidean</i>	Hasil
1.	Data1	5.1040	Benar
2.	Data2	3.5053	Benar
3.	Data3	5.1174	Benar
4.	Data4	4.7613	Benar
5.	Data5	4.3249	Benar
6.	Data6	4.9953	Benar
7.	Data7	6.5285	Salah
8.	Data8	4.9531	Benar
9.	Data9	5.2005	Benar
10.	Data10	7.8388	Salah
11.	Data11	5.8221	Benar
12.	Data12	4.4521	Benar
13.	Data13	3.2485	Benar
14.	Data14	5.6922	Benar
15.	Data15	4.1321	Benar

Tabel 5.2 menunjukkan bahwa sistem dapat mendeteksi sinyal suara menguap dengan benar sebanyak 13 kali dan 2 kali kesalahan. Dengan demikian, persentase keberhasilan sistem dalam mendeteksi sinyal suara menguap adalah sebesar 86,67 % dan kesalahan sistem dalam mendeteksi sinyal suara menguap adalah sebesar 13,33 %.

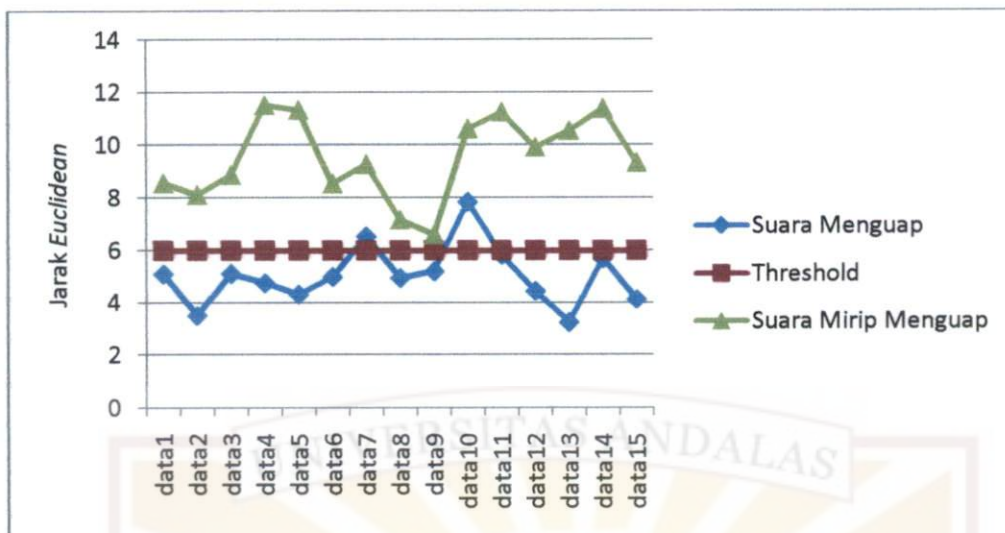
Hasil pengujian antara sinyal suara menguap pada *filebase* dengan sinyal suara mirip suara menguap dapat dilihat pada tabel 5.3 berikut ini:

Tabel 5.3. Hasil Pengambilan Data Pendeteksian Sinyal Suara Mirip Menguap

No.	Selain Suara Menguap	Jarak <i>Euclidean</i>	Hasil
1.	TesSatu.wav	8.5518	Benar
2.	TesDiam2.wav	8.1078	Benar
3.	TesWooh.wav	8.8606	Benar
4.	TesMancis.wav	11.5157	Benar
5.	TesLima.wav	11.3243	Benar
6.	TesKuap3.wav	8.5325	Benar
7.	TesBatuk.wav	9.2597	Benar
8.	TesDiam.wav	7.1630	Benar
9.	TesKotak.wav	6.5957	Benar
10.	TesKaleng.wav	10.6129	Benar
11.	TesHoo.wav	11.2473	Benar
12.	TesSsshhh.wav	9.9204	Benar
13.	TesHoam.wav	10.5365	Benar
14.	TesHembus.wav	11.3808	Benar
15.	TesWah.wav	9.3317	Benar

Pada Tabel 5.3 terlihat bahwa dari 15 kali data pengujian, semua sinyal suara mirip menguap (tidak asli) dapat ditolak oleh sistem. Dapat disimpulkan keberhasilan sistem dalam mendeteksi suara selain suara menguap asli adalah 100%.

Hasil pendeteksian sinyal suara menguap asli dan sinyal suara mirip menguap ditunjukkan pada Gambar 5.6.



Gambar 5.6. Grafik Pendeteksian Suara

Gambar 5.6 menunjukkan bahwa dari 15 kali data pengujian dengan menggunakan ekstraksi parameter MFCC, sistem dapat mendeteksi sinyal suara menguap dengan tepat, dengan persentase keberhasilan sistem mencapai 86,67 %. Keberhasilan sistem mendeteksi sinyal suara menguap disebabkan oleh beberapa faktor diantaranya:

- Sinyal suara menguap memiliki karakteristik yang hampir sama dengan sinyal suara menguap lainnya dalam satu pembicara.
- Sinyal suara menguap memiliki karakteristik yang jauh berbeda dengan sinyal suara selain suara menguap (ucapan berupa kata, angka, siul, dan lain-lain) meskipun dalam satu pembicara.
- Kemiripan kharakteristik sinyal suara menguap disebabkan karena proses terbentuknya sinyal suara menguap hampir sama dalam setiap kondisi.

Persentase kesalahan sistem dalam mendeteksi sinyal suara menguap mencapai 13,33 % . Hal ini disebabkan oleh beberapa faktor diantaranya:

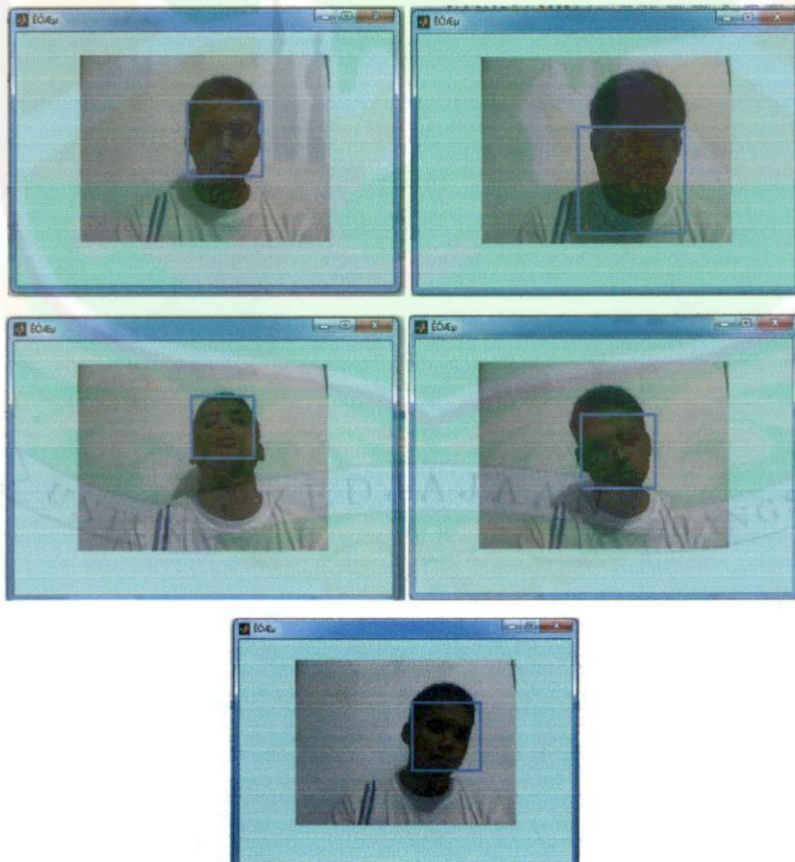
- Kondisi ruangan pada saat proses perekaman suara sangat rentan terhadap *noise* (kebisingan).
- Kualitas *microphone* yang digunakan dalam pendeteksian sinyal suara menguap sangat sensitive terhadap *noise*, sehingga tidak semua sinyal suara menguap terdeteksi sempurna.
- Kualitas hardware dan software yang digunakan.

5.2 Hasil Pendeteksian Wajah

Untuk dapat mendeteksi wajah digunakan metoda *haar cascade classifier* dan *library openCV*. Pada penelitian ini program deteksi wajah menggunakan modul "*FaceDetectDll_RC1.dll*". Didalamnya sudah tersedia database wajah dan metoda-metoda pendeteksian wajah secara real time. Kemudian dari file-file yang telah tersedia akan digunakan untuk mendeteksi wajah.

Berdasarkan hasil penelitian yang dilakukan oleh "*Rainer Lienhart, Alexander Kuranov, and Vadim Pisarevsky.2002*"^[20] dengan menggunakan 5000 data positif wajah frontal and 3000 data sampel negatif selain wajah. Dengan memanfaatkan database yang telah digunakan oleh penelitian diatas, penulis melakukan pengujian deteksi wajah dengan data uji 30 wajah berbeda dengan posisi wajah frontal (lampiran C5), sistem mampu mendeteksi 100% wajah frontal.

Contoh hasil pendeteksian wajah dapat dilihat pada gambar 5.7 dibawah ini.



Gambar 5.7. Hasil Deteksi Wilayah Wajah Dengan Berbagai Posisi

Dari gambar 5.7 diatas dapat dilihat hasil wajah terdeteksi dalam berbagai posisi. Hasil yang ditampilkan tersebut merupakan hasil deteksi wilayah wajah dari berbagai posisi selama bagian wajah lengkap dan dalam jangkauan kamera maka sistem mampu mendeteksi wajah.

Pada penelitian ini, diambil empat macam posisi tidur diantaranya yaitu posisi wajah miring ke kanan, posisi wajah miring ke kiri, posisi wajah menunduk dan posisi wajah mendongak. Hasil dari keempat posisi tersebut dapat kita lihat pada gambar 5.8 hasil deteksi posisi wajah dibawah ini.



Gambar 5.8. Posisi Wajah Yang Tidak Mampu Dikenali Sistem

Dari hasil yang terlihat pada gambar 5.8 diatas, dapat diketahui bahwa sistem tidak mampu mendeteksi wajah dalam posisi tertidur. Hal ini dikarenakan sistem memiliki keterbatasan dalam mendeteksi wajah dimana wajah yang terlalu miring, terlalu menunduk dan terlalu menengadah tidak akan dikenali sistem. Sistem hanya mampu mendeteksi wajah yang bagiannya lengkap dan frontal ke kamera. Dengan memanfaatkan kelemahan sistem maka dapat dijadikan acuan untuk mendeteksi posisi wajah seseorang dalam keadaan tertidur.

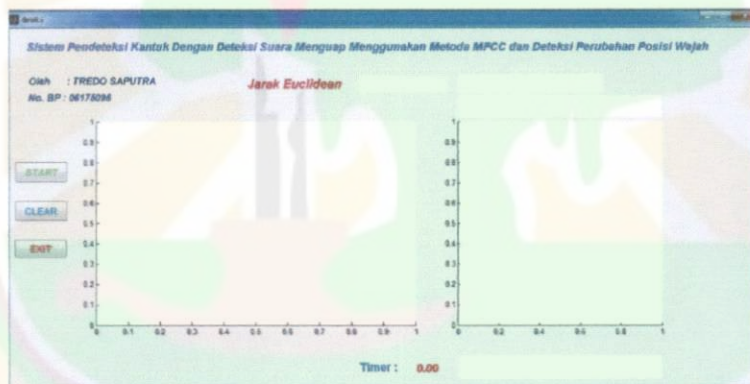
5.3. Hasil Pengujian Sistem

Sistem pendeteksi kantuk merupakan sistem yang menggabungkan proses pendeteksian suara menguap dengan MFCC-VQ dengan proses deteksi wajah

secara real time. Sistem diawali dengan proses mendeteksi suara menguap yang diinputkan melalui *microphone* kemudian suara tersebut diproses dengan metoda MFCC-VQ, apabila suara menguap yang diinputkan sesuai dengan suara database atau berada dibawah rentang nilai *threshold* maka sistem akan dilanjutkan dengan proses deteksi wajah secara real time.

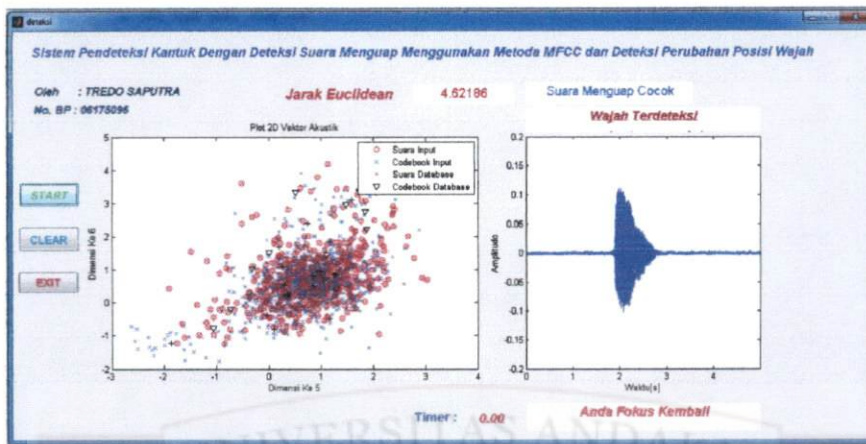
Dalam proses deteksi wajah, sistem akan mendeteksi wajah tiap frame video, apabila wajah terdeteksi maka sistem akan menandai wilayah tersebut dengan kotak (*bounding box*). Jika sistem tidak menemukan wilayah wajah, maka sistem akan mengaktifkan timer untuk menghitung waktu lamanya wajah tidak terdeteksi, apabila wajah tidak terdeteksi selama 2 detik, maka sistem akan membunyikan alarm untuk memberikan peringatan segera agar seseorang tersadar dari rasa ingin tidurnya.

Untuk proses lebih jelasnya, dapat dilihat pada penjelasan gambar 5.9 berikut:



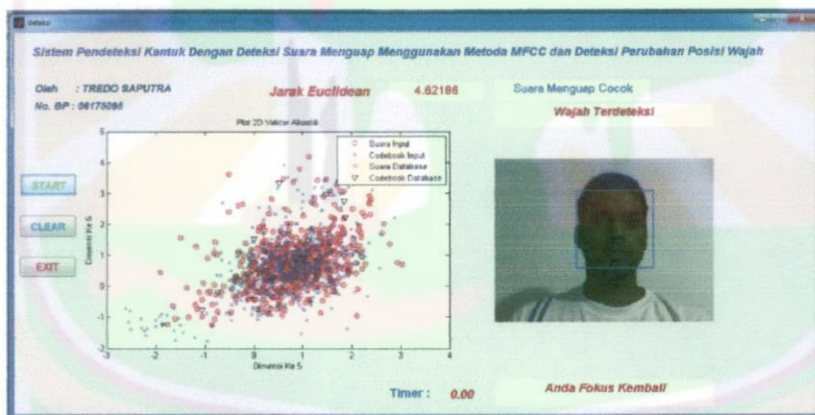
Gambar 5.9. Tampilan Awal Sistem

Gambar 5.9 diatas adalah tampilan awal dari sistem pendeteksi kantuk, saat tombol "start" ditekan, maka sistem akan merekam suara dan selanjutnya sistem akan mengolah suara yang diterima tersebut. Kemudian sistem akan menampilkan plot sinyal dari suara yang masuk tersebut dengan plot sinyal dari suara database. Hasil dari plot sinyal suara yang diterima dan plot sinyal suara menguap yang ada pada database, dapat dilihat pada gambar 5.10 berikut ini:



Gambar 5.10. Tampilan Sistem Ketika Suara Menguap Terdeteksi

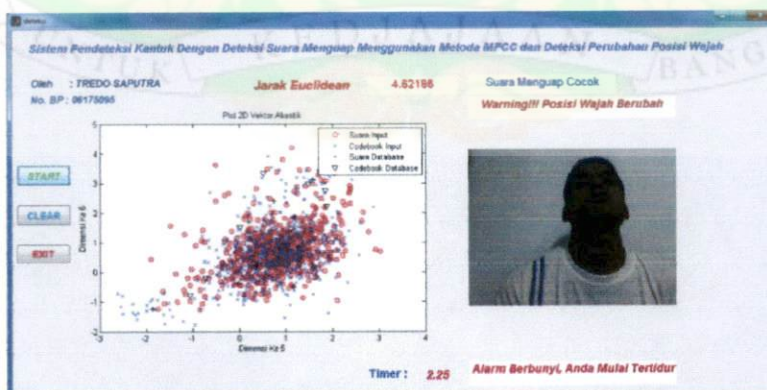
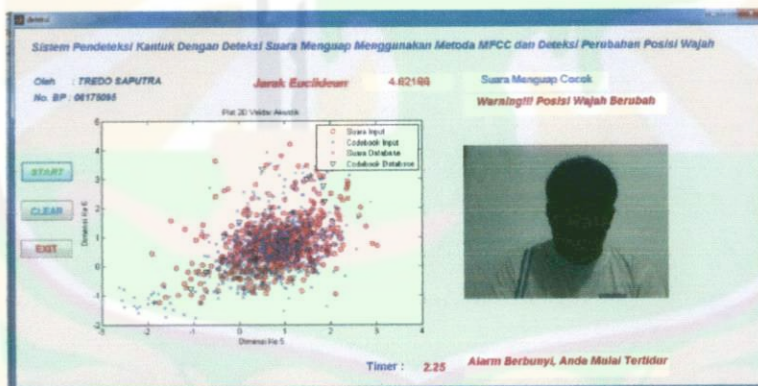
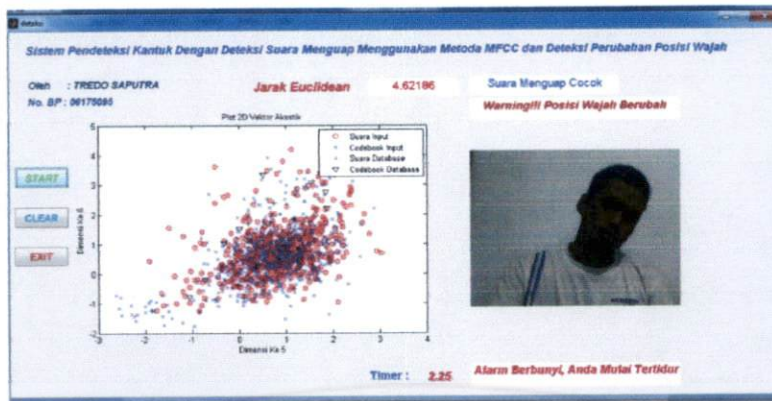
Dari gambar 5.10 terlihat bahwa jarak *euclidean* suara yang didapatkan dari input suara menguap adalah 4.62186 (lebih kecil dari nilai *threshold*) maka sistem akan menyimpulkan bahwa suara yang diujikan adalah suara menguap. Setelah suara menguap terdeteksi maka akan dilanjutkan dengan proses deteksi wajah secara real time. Seperti terlihat pada gambar 5.11.



Gambar 5.11. Tampilan Sistem Ketika Sistem Pendeteksi Wajah Aktif

Dari gambar 5.11 terlihat bahwa posisi wajah masih terdeteksi oleh sistem, karena wajah masih menghadap frontal ke kamera. Sistem akan terus mendeteksi wajah selama posisi wajah masih bisa dikenali sistem. Ketika posisi wajah berubah atau posisi wajah terlalu miring, maka sistem tidak dapat mendeteksi adanya wajah dan timer akan aktif untuk menghitung lamanya wajah tidak terdeteksi.

Pada saat wajah tidak terdeteksi, timer akan mulai menghitung waktu. Apabila wajah tidak terdeteksi maka sistem akan membunyikan alarm. Seperti terlihat pada gambar 5.12 berikut ini:



Gambar 5.12. Tampilan Sistem Ketika Sistem Mendeteksi Objek Tertidur

Dari gambar 5.12 terlihat posisi wajah seseorang dalam keadaan tertidur. Posisi tersebut sudah melewati batas waktu yaitu 2 detik. Dengan demikian sistem akan mengaktifkan alarm untuk memberikan peringatan agar seseorang tersebut tersadar dari tidurnya sehingga bisa fokus lagi.

5.4. Pembahasan

Dari pengujian sistem yang telah dilakukan. Sistem sudah bisa berjalan dengan baik walaupun dalam pendeteksian suara menguap masih terdapat beberapa suara menguap yang tidak dikenali. Dimana dari 15 suara menguap yang diujikan didapat 13 suara menguap terdeteksi dan 2 suara menguap tidak terdeteksi, sehingga dengan demikian tingkat akurasi sistem dalam mendeteksi suara menguap adalah 86,67 % dan tingkat kesalahan 13,33 %. Sedangkan dalam proses pendeteksian wajah secara *real time*, sistem mampu mendeteksi wajah frontal dengan baik. Dari 30 kali pengujian sistem mampu mendeteksi 100% wajah frontal. Dengan demikian secara keseluruhan keberhasilan sistem pendeteksi kantuk dengan deteksi suara menguap dan deteksi posisi wajah adalah 86,67% dan tingkat kesalahan 13,33%.

Secara keseluruhan, apa yang menjadi tujuan penulis pada sub bab 1 sudah tercapai. Jadi, sistem yang dibuat pada penelitian ini telah bekerja sebagaimana yang diharapkan.

BAB VI

PENUTUP

6.1 Kesimpulan

Berdasarkan analisa terhadap hasil yang didapat, maka dapat dirangkum beberapa simpulan sebagai berikut:

1. Suara menguap sangat menentukan keberhasilan dari sistem, apabila suara menguap yang didapat tidak dikenali maka sistem tidak akan berjalan dengan baik.
2. Secara keseluruhan tingkat akurasi sistem pendeteksi kantuk yang telah diujikan adalah 86,67 % dengan tingkat kesalahan 13,33 %.
3. Sistem ini telah berhasil mendeteksi kantuk berdasarkan deteksi suara menguap dan deteksi posisi wajah dengan baik walaupun hasil yang didapat belum sempurna.

6.2 Saran

Setelah menganalisis cara kerja sistem dan hasil keluaran sistem (hasil pengolahan), untuk penelitian dan pengembangan sistem selanjutnya, penulis menyarankan beberapa hal berikut:

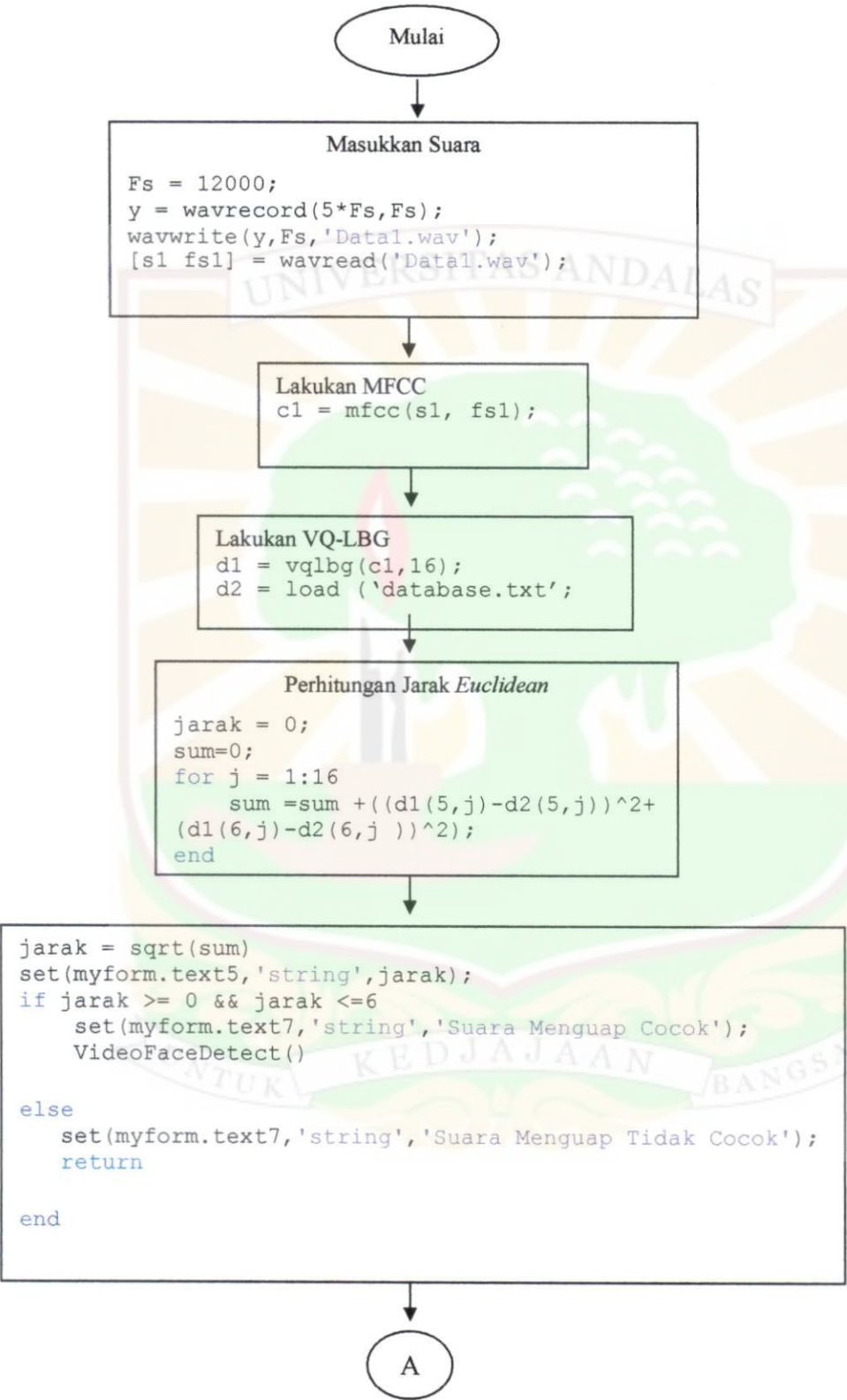
1. Suara menguap yang dikenali sistem masih terbatas yaitu hanya satu database, diharapkan pada penelitian selanjutnya sistem mampu mengenali suara menguap dengan banyak database.
2. Sistem hanya mendeteksi wajah, untuk penelitian selanjutnya agar bisa ditambahkan detektor mata dan detektor mulut menguap sehingga hasil yang didapat bisa lebih akurat lagi.

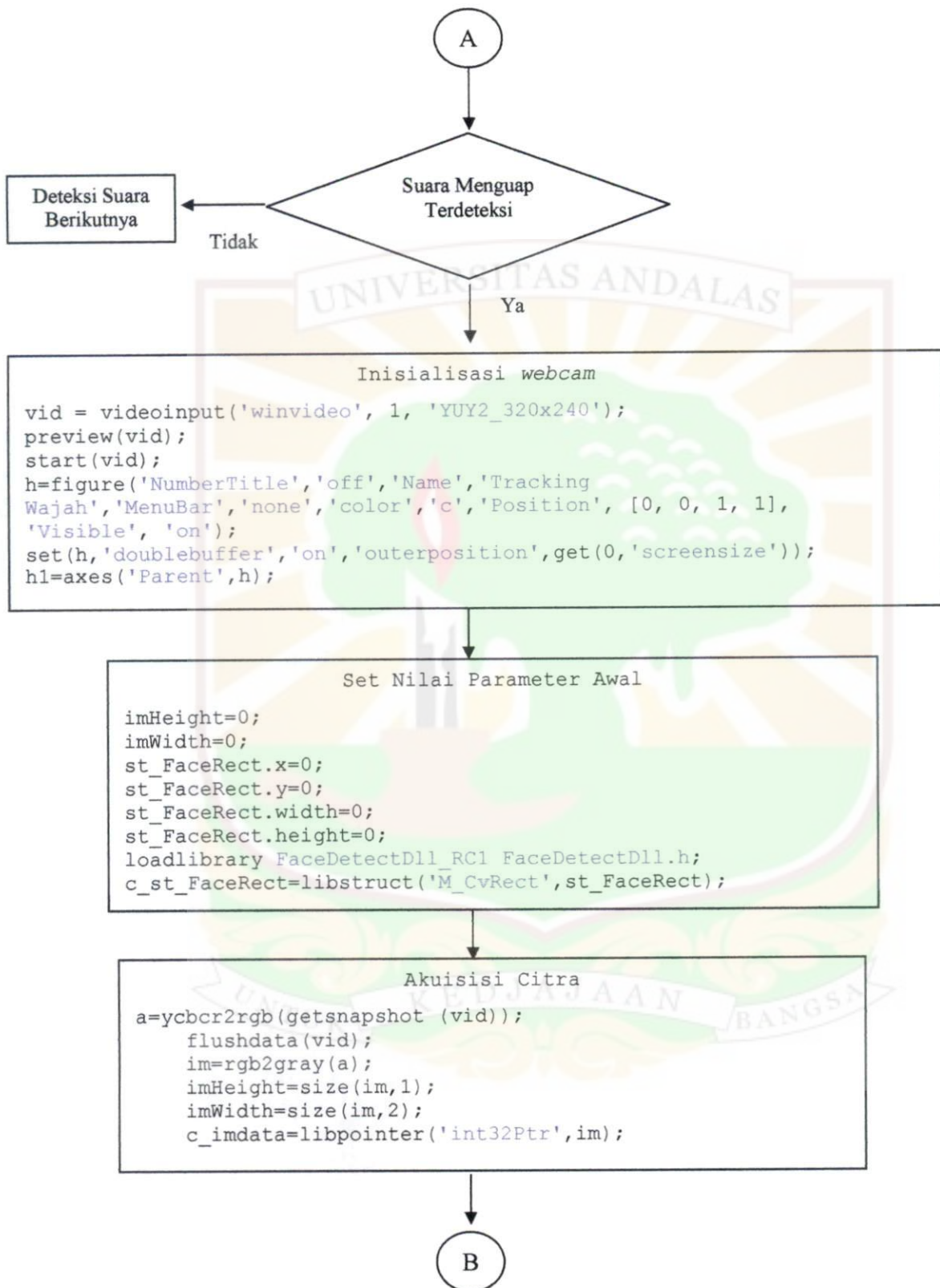
DAFTAR PUSTAKA

1. Tri Putriyati Permata. 2009. "*Segmentasi Iris Mata Menggunakan Metode Deteksi Tepi dan Operasi Morfologi*". Program Studi Sistem Informasi, Universitas Gunadarma, Depok.
2. Premar, Neeta. 2002. "*Drowsy Driver Detection System*". Department of Electrical and Computer Engineering, Ryerson University.
3. Vural, Esra, and Mujdat Cetin, etc. "*Automated Drowsiness Detection For Improved Driving Safety*". Sabanci University Faculty of Engineering and Natural Sciences Orhanli, Istanbul, University of California San Diego Institute of Neural Computation La Jolla, San Diego.
4. KANAGAWA, Tsuyoshi, and Taiji KAWACHI, dkk. 2006. "*Drowsiness Detection Using Spectrum Analysis of Eye Movements and Effective Stimuli to Keep Drive Awake*". Proceedings of Smart Systems 2006 and ICMA 2006.
5. Yusron Rijal, Riza Dhian Ariefianto. 2008. "*Deteksi Wajah Berbasis Segmentasi Model Warna Menggunakan Template Matching Pada Objek Bergerak*". Program Studi Sistem Komputer, Sekolah Tinggi Manajemen Informatika & Teknik Komputer, Surabaya.
6. Achmad Hidayatno, R. Rizal Isnanto, dan Dhody Kurniawan. 2006. "*Penentuan Wilayah Wajah Manusia Pada Citra Berwarna Berdasarkan Warna Kulit Dengan Metode Template Matching*". Teknik Elektro Fakultas Teknik Universitas Diponegoro, Semarang.
7. Hadi, S. 2008. "*Pengembangan Model Generatif Pengenalan Wajah pada Latar Belakang, Pose dan Iluminasi yang Bervariasi*". Disertasi, Institut Teknologi Bandung.
8. Agushinta, Dewi, Adang Suhendra dan Hendra. "*Ekstraksi Fitur dan Segmentasi Wajah Sebagai Semantik Pada Sistem Pengenalan Wajah*". National Conference on Computer Science & Information Technology VII, Universitas Gunadarma.
9. Munir, R. 2004 "*Pengolahan Citra Digital dengan Pendekatan Algoritmik*". Informatika Bandung.
10. Razaki, R. 2008. "*Tracking Objek Bergerak Dalam Arah Linear Berdasarkan Seleksi Bentuk*". Tugas Akhir, Teknik Elektro Fakultas Teknik Universitas Andalas, Padang.

11. Irfan Syafur Bachtiar. 2007. *Aplikasi Pengenalan Wicara HMM untuk Kendali Robot PDA*. Teknologi Telekomunikasi, Politeknik Elektronika Negeri Surabaya, Surabaya
12. Hendrawan. 2003. "Pendahuluan: Handout Mata Kuliah Pemrosesan Citra Dan Video". Bandung: Teknik Elektro ITB
13. Yu-Jin, Zhang. 2006. *Advance in Image and Video Segmentation*. Harshey: IRM Press
14. Gora, Winastwan. 2006. "BAB I. Dasar Digital Video". Belajarsendiri.com. Diakses tanggal 28 Juni 2007
15. P. Viola and M. Jones. 2001 "Robust Real-time Object Detection", Second International Workshop on Statistical and Computational Theories of Vision - Modeling, Learning, Computing, and Sampling.
16. Yuda, Aksara Cipta. 2008. *Object Tracking Pada Gerakan Non-Linier Berdasarkan Informasi Warna*. Tugas Akhir: Teknik Elektro Fakultas Teknik Universitas Andalas, Padang.
17. Silvana, Meza. 2006. *Optimalisasi Bobot Jaringan Syaraf Tiruan Menggunakan Algoritma Genetik dalam Identifikasi Suara*. Tugas Akhir. Padang: Teknik Elektro Universitas Andalas.
18. Rahmat, Abmierdal. 2010. *Perancangan Sistem Verifikasi Penutur Menggunakan Metoda Mel Frequency Cepstral Coefficients-Vector Quantisation (MFCC-VQ) dan Pengenalan Kata Menggunakan Metoda Logika Fuzzy Berdasarkan Kata Warna "Merah", "Hijau", dan "Biru"*. Tugas Akhir. Padang: Teknik Elektro Universitas Andalas.
19. Mustofa, Ali. 2007. *Sistem Pengenalan Penutur dengan Metode Mel-frequency Wrapping*. Journal Teknik Elektro. Malang: Universitas Brawijaya.
20. Lienhart, Rainer, Alexander Kuranov, Vadim Pisarevsky. 2002. *Empirical Analysis of Detection Cascades of Boosted Classifiers for Rapid Object Detection*. Microprocessor Research Lab, Intel Labs Intel Corporation, Santa Clara, CA 95052, USA

Lampiran A-1. Diagram Alir Proses Pendeteksian Sinyal Suara Menguap





B

Segmentasi Citra

Proses Segementasi Dengan Program Yang Terkompilasi Dalam FaceDetectDll_RC1.dll dan FaceDetect.h

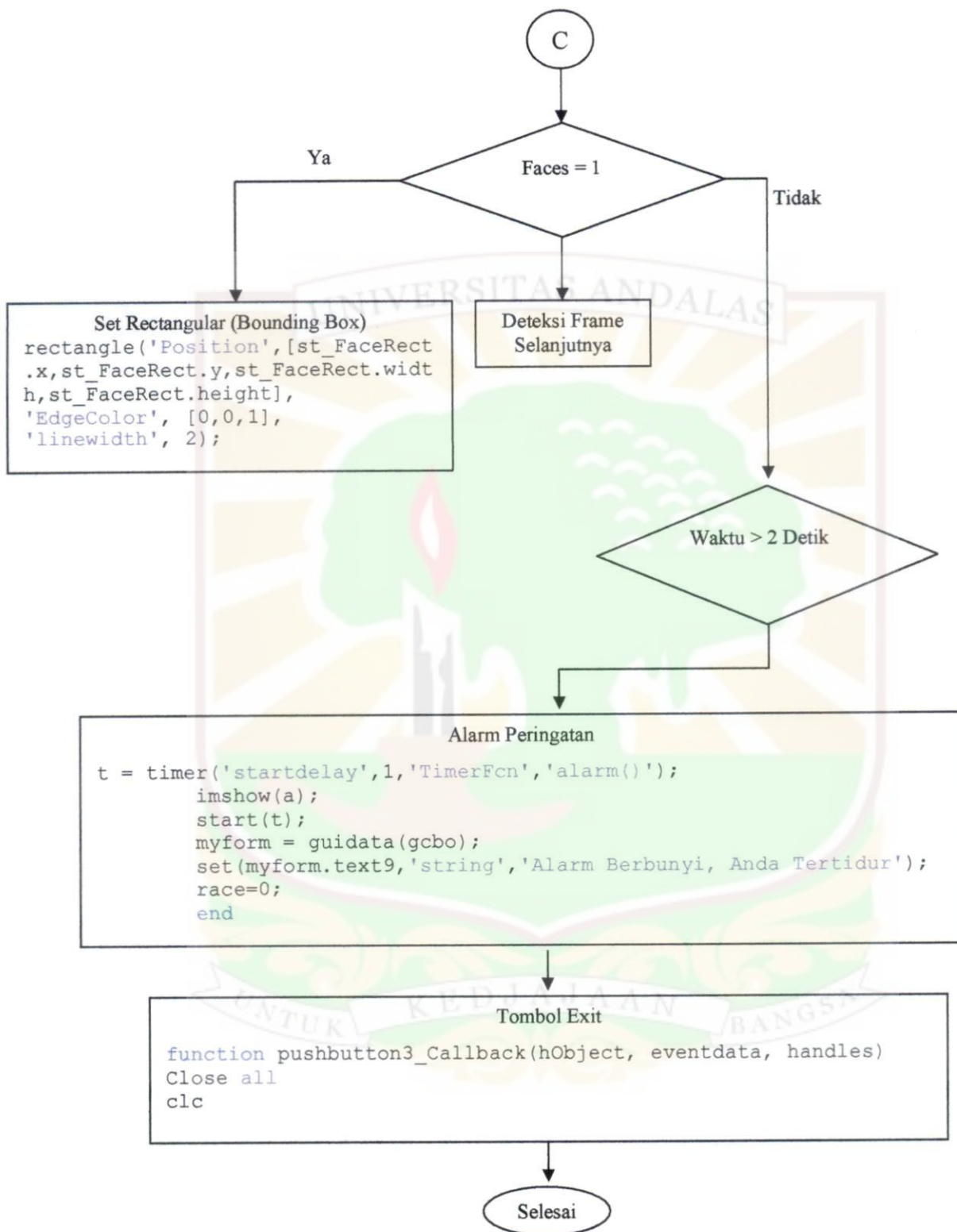
Deteksi Wajah

```
faces=calllib('FaceDetectDll_RC1','FaceDetect',imHeight,imWidth,c_imdat
a,c_st_FaceRect);

if faces==1
    st_FaceRect=get(c_st_FaceRect);
    imshow(a);

rectangle('Position',[st_FaceRect.x,st_FaceRect.y,st_FaceRect.width,st_
FaceRect.height], 'EdgeColor', [0,0,1], 'linewidth', 2);
    myform = guidata(gcbo);
    set(myform.text9,'string','');
    myform = guidata(gcbo);
    set(myform.text10,'string','0.00');
    race=0;
    detik=0;
else
    race=race+0.25;
    if (race>2)
        t = timer('startdelay',1,'TimerFcn','alarm()');
        imshow(a);
        start(t);
        myform = guidata(gcbo);
        set(myform.text9,'string','Alarm Berbunyi, Anda Tertidur');
        race=0;
    end
    detik=detik+0.25
    myform = guidata(gcbo);
    set(myform.text10,'string',detik);
end
drawnow;
```

C



Lampiran A-2. Program GUI

```
function varargout = deteksi(varargin)
% DETEKSI M-file for deteksi.fig
%   DETEKSI, by itself, creates a new DETEKSI or raises the
existing
%   singleton*.
%
%   H = DETEKSI returns the handle to a new DETEKSI or the
handle to
%   the existing singleton*.
%
%   DETEKSI('CALLBACK',hObject,eventData,handles,...) calls the
local
%   function named CALLBACK in DETEKSI.M with the given input
arguments.
%
%   DETEKSI('Property','Value',...) creates a new DETEKSI or
raises the
%   existing singleton*. Starting from the left, property
value pairs are
%   applied to the GUI before deteksi_OpeningFcn gets called.
An
%   unrecognized property name or invalid value makes property
application
%   stop. All inputs are passed to deteksi_OpeningFcn via
varargin.
%
%   *See GUI Options on GUIDE's Tools menu. Choose "GUI allows
only one
%   instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help deteksi

% Last Modified by GUIDE v2.5 26-Aug-2011 01:19:09

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                  'gui_Singleton',   gui_Singleton, ...
                  'gui_OpeningFcn', @deteksi_OpeningFcn, ...
                  'gui_OutputFcn',  @deteksi_OutputFcn, ...
                  'gui_LayoutFcn',   [] , ...
                  'gui_Callback',    []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
```

```

% --- Executes just before deteksi is made visible.
function deteksi_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to deteksi (see VARARGIN)

% Choose default command line output for deteksi
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes deteksi wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = deteksi_OutputFcn(hObject, eventdata,
handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;
axes(handles.axes1);

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
clear all
clc
myform = guidata(gcbo);

set(myform.figure1, 'CurrentAxes', myform.axes3);
Fs = 12000;
rekam= wavrecord(5*Fs,Fs,'int16');
wavwrite(rekam,Fs,'Data1.wav');

[s1 fs1] = wavread('SUARA1.wav');
[s2 fs2] = wavread('DATABASESUARA.wav');

%Signal input s1
disp('> Sinyal input s1');
t = 0:1/fs1:(length(s1) - 1)/fs1;
plot(t, s1);
axis([0, (length(s1) - 1)/fs1 -0.2 0.2]);
title('Plot Sinyal Input Suara Menguap');

```



```

xlabel('Waktu[s]');
ylabel('Amplitudo')

%MFCC
c1 = mfcc(s1, fs1);
c2 = mfcc(s2, fs2);

%*****
%Pengenalan Suara Menguap (Verifikasi)
%*****
% %Vektor akustik
d1 = vqldg(c1,16);
d2 = vqldg(c2,16);

set(myform.figure1,'CurrentAxes',myform.axes1);

disp('> Question 7: 2D plot of accoustic vectors');
c1 = mfcc(s1, fs1);
c2 = mfcc(s2, fs2);
plot(c1(5, :), c1(6, :), 'or');
hold on;
plot(c2(5, :), c2(6, :), 'xb');
xlabel('Dimensi Ke 5');
ylabel('Dimensi Ke 6');
legend('Sinyal Input', 'Sinyal Database');
title('Plot 2D Vektor Akustik');

plot(c1(5, :), c1(6, :), 'xr')
hold on
plot(d1(5, :), d1(6, :), 'vk')
plot(c2(5, :), c2(6, :), 'xb')
plot(d2(5, :), d2(6, :), '+k')
xlabel('Dimensi Ke 5');
ylabel('Dimensi Ke 6');
legend('Suara Input', 'Codebook Input', 'Suara Database',
'Codebook Database');
title('Plot 2D Vektor Akustik');

%Perhitungan Jarak Euclidean
jarak = 0;
sum=0;

for j = 1:16
    sum =sum + ((d1(5,j)-d2(5,j ))^2+(d1(6,j)-d2(6,j ))^2);
end

%Pencocokan Jarak Euclidean
jarak = sqrt(sum)
set(myform.text5,'string',jarak);
%%Menentukan batasan jarak Euclidian%%
if jarak >= 0 && jarak <=6
    set(myform.text7,'string','Suara Menguap Cocok');
    set(myform.text12,'string','Kamera Akan Diaktifkan');
set(myform.figure1,'CurrentAxes',myform.axes3);
%%%%%%%%% menjalankan program pengambilan citra
%%%%%%%%%

```

```

    VideoFaceDetect()
else
    set(myform.text7,'string','Suara Menguap Tidak Cocok');
    return
end

% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
close all;

% --- Executes on button press in pushbutton3.
function pushbutton3_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton3 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
myform = guidata(gcbo);
set(myform.text9,'string','');
set(myform.text7,'string','');
set(myform.text12,'string','');
set(myform.text10,'string','');
set(myform.text9,'string','');
set(myform.text5,'string','');
clear all;
clc;

function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
%        str2double(get(hObject,'String')) returns contents of
edit1 as a double

% --- Executes during object creation, after setting all
properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    empty - handles not created until after all
CreateFcns called

% Hint: edit controls usually have a white background on Windows.
%        See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```


Lampiran A-3. Program FaceDetectDll.h

```
#ifdef __cplusplus
```

```
extern "C" {
```

```
#endif
```

```
typedef struct M_CvRect
```

```
{
```

```
    int x;
```

```
    int y;
```

```
    int width;
```

```
    int height;
```

```
}M_CvRect;
```

```
__declspec(dllexport) int sub(int iLeft,int iRight);
```

```
__declspec(dllexport) int FaceDetect(int height,int width,int *imdata,M_CvRect  
*faceRect );
```

```
#ifdef __cplusplus
```

```
}
```

```
#endif
```



Lampiran B-1. Program Deteksi Suara Menguap Dengan Metoda MFCC

```
%Persiapan Matlab
function main
clear all
clc
Fs = 12000;
rekam= wavrecord(4*Fs,Fs,'int16');
wavwrite(rekam,Fs,'Data1.wav');
[s1 fs1] = wavread('suara1.wav');
[s2 fs2] = wavread('DATABASESUARA.wav');

%Sinyal input s1
disp('> Sinyal input s1');
t = 0:1/fs1:(length(s1) - 1)/fs1;
plot(t, s1);
axis([0, (length(s1) - 1)/fs1 -0.2 0.2]);
title('Database Suara Menguap');
xlabel('Time [s]');
ylabel('Amplitude (normalized)');
%pause
%close all

%Pre-emphasis

%MFCC
c1 = mfcc(s1, fs1);
c2 = mfcc(s2, fs2);

%*****
%Pengenalan Suara Menguap (Verifikasi)
%*****
% %Vektor akustik
d1 = vq1bg(c1,16);
d2 = load('database.txt');

%Perhitungan Jarak Euclidean
jarak = 0;
sum=0;

for j = 1:16
    sum =sum +((d1(5,j)-d2(5,j))^2+(d1(6,j)-d2(6,j))^2);
end

%Pencocokan Jarak Euclidean
jarak = sqrt(sum)
%%Menentukan batasan jarak Euclidian%%
if jarak >= 0 && jarak <= 6
%menjalankan program pengambilan citra
    disp ('Suara Menguap Cocok')
    VideoFaceDetect()
else
    disp ('Suara Menguap Tidak Cocok')

end
```


Lampiran B-2. Program Deteksi Wajah Secara Real Time

```
%=====
%                               PROGRAM DETEKSI WAJAH                               %
%                               OLEH :                               %
%                               TREDO SAPUTRA                               %
%                               06175095                               %
%                               PEMBIMBING : RAHMADI KURNIA, DR.ENG.       %
%=====

clear all;

imamem(3000000);

vid = videoinput('winvideo',1,'YUY2_320x240');

preview(vid);

start(vid);

h=figure('NumberTitle','off','Name','Tracking
Wajah','MenuBar','none','color','c','Position',[0,0,1,1],
'Visible','on');

set(h,'doublebuffer','on','outerposition',get(0,'screensize'));

h1=axes('Parent',h);

hold on;

axis off;

imHeight=0;
imWidth=0;

st_FaceRect.x=0;
st_FaceRect.y=0;
st_FaceRect.width=0;
st_FaceRect.height=0;

loadlibrary FaceDetectDll_RC1 FaceDetectDll.h;
c_st_FaceRect=libstruct('M_CvRect',st_FaceRect);

race=0;
detik=0;

while ishandle(h)

    a=ycbcr2rgb(getsnapshot (vid));

    flushdata(vid);

    im=rgb2gray(a);
```

```

imHeight=size(im,1);
imWidth=size(im,2);
c_imdata=libpointer('int32Ptr',im);

faces=calllib('FaceDetectDll_RC1','FaceDetect',imHeight,imWidth,c_
imdata,c_st_FaceRect);

if faces==1

    st_FaceRect=get(c_st_FaceRect);

    imshow(a);
rectangle('Position',[st_FaceRect.x,st_FaceRect.y,st_FaceRect.widt
h,st_FaceRect.height], 'EdgeColor', [0,0,1], 'linewidth', 2);
    myform = guidata(gcbo);
    set(myform.text9,'string','Anda Fokus Kembali');
    set(myform.text12,'string','Wajah Terdeteksi');
    myform = guidata(gcbo);
    set(myform.text10,'string','0.00');
    race=0;
    detik=0;

else
    myform = guidata(gcbo);
    set(myform.text12,'string','Warning!!! Posisi Wajah
Berubah');
    race=race+0.25;
    if (race>2)
        t = timer('startdelay',1,'TimerFcn','alarm()');
        imshow(a);
        start(t);
        myform = guidata(gcbo);
        %set(myform.text12,'string','Wake Up, Fokus');
        set(myform.text9,'string','Alarm Berbunyi, Anda Mulai
Tertidur');
        race=0;
        end
        detik=detik+0.25
        myform = guidata(gcbo);
        set(myform.text10,'string',detik);
    end
    drawnow;

end;

clear c_st_FaceRect;
unloadlibrary('FaceDetectDll_RC1');
delete(vid);
disp('Program Deteksi Wajah Dimatikan,Refresh Memory');

```


Lampiran B-3. Listing Program Penentuan Nilai *Threshold*

```
clear all
clc

[s1 fs1] = wavread('suara1.wav');
[s2 fs2] = wavread('suaraDatabase.wav');

%Sinyal input s1
disp('> Sinyal input s1');
t = 0:1/fs1:(length(s1) - 1)/fs1;
plot(t, s1)
axis([0, (length(s1) - 1)/fs1 -0.4 0.5]);
title('Plot of signal s1.wav');
xlabel('Time [s]');
ylabel('Amplitude (normalized)')
%pause
close all

%Pre-emphasis

%MFCC
c1 = mfcc(s1, fs1);
c2 = mfcc(s2, fs2);

d1 = vq1bg(c1,16);
d2 = vq1bg(c2,16);
%Perhitungan Jarak Euclidean
jarak = 0;
sum=0;
for j = 1:16
    sum =sum + ((d1(5,j)-d2(5,j ))^2+(d1(6,j)-d2(6,j ))^2);
end
%Pencocokan Jarak Euclidean
jarak = sqrt(sum)
```

Lampiran B-4. Listing Program *Mel Frequency Cepstral Coefficient* (MFCC)

```
%*****
*
% M-File : mfcc
%*****
*
function r = mfcc(s, fs)
% MFCC
%
% Inputs: s  contains the signal to analyze
%         fs is the sampling rate of the signal
%
% Output: r contains the transformed signal

m = 100;
n = 256;
l = length(s);

%Frame blocking
nbFrame = floor((l - n) / m) + 1;

for i = 1:n
    for j = 1:nbFrame
        M(i, j) = s((j - 1) * m) + i);
    end
end

%Windowing (Hamming Window)
h = hamming(n);
M2 = diag(h) * M;

%Fast Fourier Transform (FFT)
for i = 1:nbFrame
    frame(:,i) = fft(M2(:, i));
end

t = n / 2;
tmax = 1 / fs;

%Mel Frequency Wrapping
m = mel_fb(20, n, fs);
n2 = 1 + floor(n / 2);
z = m * abs(frame(1:n2, :)).^2;

%Discrete Cosine Transform (DCT)
r = dct(log(z));
```


Lampiran B-5. Listing Program *Vector Quantization*

```
%*****
%
% M-File : vqlbg
%*****
%
function r = vqlbg(d,k)
% VQLBG Vector quantization using the Linde-Buzo-Gray algorithm
%
% Inputs: d contains training data vectors (one per column)
%         k is number of centroids required
%
% Output: r contains the result VQ codebook (k columns, one for
each centroids)

e = .01;
r = mean(d, 2);
dpr = 10000;

for i = 1:log2(k)
    r = [r*(1+e), r*(1-e)];

    while (1 == 1)
        z = disteu(d, r);
        [m,ind] = min(z, [], 2);
        t = 0;
        for j = 1:2^i
            r(:, j) = mean(d(:, find(ind == j)), 2);
            x = disteu(d(:, find(ind == j)), r(:, j));
            for q = 1:length(x)
                t = t + x(q);
            end
        end
        if ((dpr - t)/t) < e
            break;
        else
            dpr = t;
        end
    end
end
end
```

Lampiran B-6. Listing Program Filterbank

```
function m = melfb(p, n, fs)
% MELFB          Determine matrix for a mel-spaced filterbank
%
% Inputs:        p   number of filters in filterbank
%               n   length of fft
%               fs   sample rate in Hz
%
% Outputs:       x   a (sparse) matrix containing the filterbank
%               amplitudes
%               size(x) = [p, 1+floor(n/2)]
%
% Usage:         For example, to compute the mel-scale spectrum of
%               a column-vector signal s, with length n and sample
%               rate fs:
%
%               f = fft(s);
%               m = melfb(p, n, fs);
%               n2 = 1 + floor(n/2);
%               z = m * abs(f(1:n2)).^2;
%
%               z would contain p samples of the desired mel-scale
%               spectrum
%
%               To plot filterbanks e.g.:
%
%               plot(linspace(0, (12500/2), 129), melfb(20, 256,
%               12500)'),
%               title('Mel-spaced filterbank'), xlabel('Frequency
%               (Hz)');
%
%               f0 = 700 / fs;
%               fn2 = floor(n/2);
%
%               lr = log(1 + 0.5/f0) / (p+1);
%
%               % convert to fft bin numbers with 0 for DC term
%               b1 = n * (f0 * (exp([0 1 p p+1] * lr) - 1));
%
%               b1 = floor(b1(1)) + 1;
%               b2 = ceil(b1(2));
%               b3 = floor(b1(3));
%               b4 = min(fn2, ceil(b1(4))) - 1;
%
%               pf = log(1 + (b1:b4)/n/f0) / lr;
%               fp = floor(pf);
%               pm = pf - fp;
%
%               r = [fp(b2:b4) 1+fp(1:b3)];
%               c = [b2:b4 1:b3] + 1;
%               v = 2 * [1-pm(b2:b4) pm(1:b3)];
%
%               m = sparse(r, c, v, p, 1+fn2);
```


Lampiran B-7. Listing Program *Block Frames*

```
%*****
%
% M-File : blockFrames
%*****
%
function M3 = blockFrames(s, fs, m, n)
% blockFrames: Puts the signal into frames
%
% Inputs: s contains the signal to analyze
% fs is the sampling rate of the signal
% m is the distance between the beginnings of two frames
% n is the number of samples per frame
%
% Output: M3 is a matrix containing all the frames
%
%
%*****

l = length(s);
nbFrame = floor((l - n) / m) + 1;
for i = 1:nbFrame
    for j = 1:n
        M(i, j) = s((j - 1) * m + i);
    end
end
h = hamming(n);
M2 = diag(h) * M;
for i = 1:nbFrame
    M3(:, i) = fft(M2(:, i));
end
```

Lampiran B-8. Listing Program *Euclidean distances*

```
*****
*
% M-File : disteu
%*****
*
function d = disteu(x, y)
% DISTEU Pairwise Euclidean distances between columns of two
matrices
%
% Input:
%   x, y:   Two matrices whose each column is an a vector
data.
%
% Output:
%   d:      Element d(i,j) will be the Euclidean distance
between two
%           column vectors X(:,i) and Y(:,j)
%
% Note:
%   The Euclidean distance D between two vectors X and Y is:
%    $D = \sum((x-y).^2).^0.5$ 

[M, N] = size(x);
[M2, P] = size(y);

if (M ~= M2)
    error('Matrix dimensions do not match.')
end

d = zeros(N, P);

if (N < P)
    copies = zeros(1,P);
    for n = 1:N
        d(n,:) = sum((x(:, n+copies) - y) .^2, 1);
    end
else
    copies = zeros(1,N);
    for p = 1:P
        d(:,p) = sum((x - y(:, p+copies)) .^2, 1)';
    end
end

d = d.^0.5;
```


Lampiran C.1. Tabel Pengujian Suara Menguap Untuk Mencari Nilai Threshold

No	Suara Database	Jarak Euclidean 20 Dimensi										
		1 dan 2	3 dan 4	5 dan 6	7 dan 8	9 dan 10	11 dan 12	13 dan 14	15 dan 16	17 dan 18	19 dan 20	rata-rata
1	Suara1	15.6486	6.3387	4.6219	2.9118	3.8660	3.0706	2.2603	1.8314	1.9306	1.3172	4.3797
2	Suara2	14.4525	5.3621	4.4117	2.6343	2.8792	2.3897	1.8555	1.7664	2.1039	1.4309	3.9286
3	Suara3	24.1670	6.0689	5.0421	2.8084	5.3625	3.1729	2.9577	2.4438	1.9587	1.3879	5.5370
4	Suara4	14.4952	6.5758	4.9894	2.6201	4.3574	2.5581	2.7272	2.1970	1.6401	1.7955	4.3956
5	Suara5	17.8305	7.2518	5.3880	4.7411	3.8761	1.8769	2.0988	1.7346	1.5081	1.1963	4.7502
6	Suara6	8.5095	7.2929	5.9916	4.0148	3.4422	3.5057	1.8468	2.5224	1.5060	1.5909	4.0223
7	Suara7	17.5447	5.8088	4.5391	3.0570	2.7542	2.3636	2.0842	1.8333	1.5895	1.1130	4.2687
8	Suara8	12.9925	7.0232	3.7773	3.3258	3.9061	3.6259	2.9445	2.1949	1.5029	1.5022	4.2795
9	Suara9	14.7662	5.1679	5.8398	2.8443	3.0699	2.3693	1.2589	1.4223	1.6613	1.6675	4.0067
10	Suara10	7.5205	6.3216	5.5737	3.3173	3.4204	3.3109	2.5796	2.5784	1.4094	1.5674	3.7599

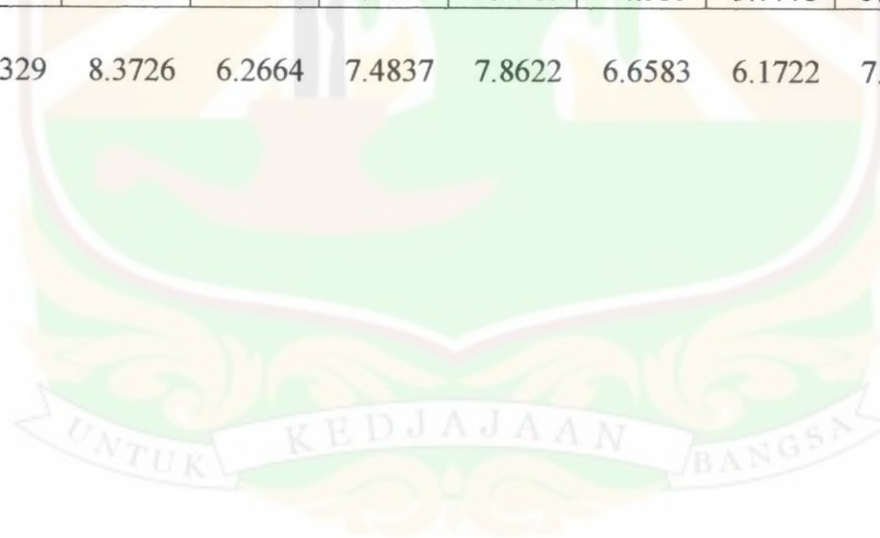
Lampiran C.2 Tabel Pengujian Suara Menguap Database dengan Suara Menguap

No	Suara Menguap yang Diuji	Dimensi										rata-rata
		1 dan 2	3 dan 4	5 dan 6	7 dan 8	9 dan 10	11 dan 12	13 dan 14	15 dan 16	17 dan 18	19 dan 20	
1	Data1	23.5818	6.1858	3.8366	4.9763	3.6685	3.1553	2.9394	3.0351	1.5101	1.6160	5.4505
2	Data2	22.7060	6.3456	5.0181	5.3244	3.7827	2.8755	3.0409	3.1389	1.8460	1.2217	5.5300
3	Data3	18.5678	6.9521	5.6070	4.9838	4.8281	4.0383	2.8040	2.2272	1.6590	1.3885	5.3056
4	Data4	20.8781	5.1149	5.1874	5.7209	5.2926	2.7252	2.9706	3.4424	2.0329	1.3170	5.4682
5	Data5	21.4989	7.5156	5.5653	4.6834	3.5691	3.4829	2.7319	2.5014	1.9743	1.7334	5.5256
6	Data6	8.6955	6.6022	5.0477	4.2015	2.3329	2.0420	2.5841	2.6677	2.5673	1.6979	3.8439
7	Data7	16.5825	6.8009	6.9486	6.6619	2.8995	2.3429	2.7514	2.6911	1.7145	1.5944	5.0988
8	Data8	20.9192	7.0048	5.4790	6.7425	3.6484	2.9554	3.6242	3.4859	1.6417	1.6021	5.7103
9	Data9	16.2195	7.1122	5.6016	4.8724	4.1246	3.0264	2.0714	2.5900	1.9606	1.3432	4.8922
10	Data10	12.0485	6.7670	5.0894	4.3052	3.7798	3.0011	2.3468	2.5269	1.3431	1.4575	4.2665
11	Data11	23.2357	7.7499	4.4827	5.8721	3.5755	3.2753	3.0836	2.7861	2.1898	1.5498	5.7801
12	Data12	15.4124	9.3137	6.2076	5.9174	3.4960	2.6292	2.6859	2.1435	1.4350	1.5147	5.0755
13	Data13	29.7748	8.0391	5.9893	6.1663	5.9591	3.9505	4.6276	4.8284	1.8597	1.9483	7.3143
14	Data14	19.4872	8.9503	5.6522	5.1851	4.0804	3.3186	3.2068	2.5341	1.9168	2.1410	5.6473
15	Data15	17.1180	7.1203	5.6354	4.9450	3.4346	2.6720	3.1671	2.6252	2.0521	1.6670	5.0437

Lampiran C3. Tabel Pencarian Nilai Threshold

	suara 1	suara 2	suara 3	suara 4	suara 5	suara 6	suara 7	suara 8	suara 9	suara 10	Database
suara1	0	4.9391	6.2929	3.8135	5.8320	4.3866	4.9299	4.4485	5.4211	4.3610	4.6219
suara2	4.9391	0	6.7329	4.3652	4.9250	5.5050	4.0382	2.7823	5.6219	4.5056	4.4116
suara3	6.2929	6.7329	0	6.2664	5.9435	7.8622	5.7775	6.1722	6.3631	8.3726	5.0421
suara4	3.8135	4.3652	6.2664	0	5.5495	4.4587	3.8669	3.7314	5.7528	4.9066	4.9893
suara5	5.8320	4.9250	5.9435	5.5495	0	6.1559	4.2592	4.5185	7.4837	6.2602	5.3879
suara6	4.3866	5.5050	7.8622	4.4587	6.1559	0	5.6885	4.8876	7.5525	4.1951	5.9915
suara7	4.9299	4.0382	5.7775	3.8669	4.2592	5.6885	0	3.6516	6.6583	5.2025	4.5390
suara8	4.4485	2.7823	6.1722	3.7314	4.5185	4.8876	3.6516	0	5.2632	4.3332	3.7773
suara9	5.4211	5.6219	6.3631	5.7528	7.4837	7.5525	6.6583	5.2632	0	6.4456	5.8398
suara10	4.3610	4.5056	8.3726	4.9066	6.2602	4.1951	5.2025	4.3332	6.4456	0	5.5737
Database	4.6219	4.4116	5.0421	4.9893	5.3879	5.9915	4.539	3.7773	5.8398	5.5737	0

Max 6.2929 6.7329 8.3726 6.2664 7.4837 7.8622 6.6583 6.1722 7.5525 8.3726 5.9915



Lampiran C.4. Jumlah Stage, Banyak Fitur dan Stage Threshold Database Wajah

No	Banyak Fitur	Stage Threshold
Stage 0	3	0.82
Stage 1	16	6.96
Stage 2	21	9.50
Stage 3	39	18.41
Stage 4	33	15.32
Stage 5	44	21.01
Stage 6	50	23.92
Stage 7	51	24.53
Stage 8	56	27.15
Stage 9	71	34.55
Stage 10	80	39.11
Stage 11	103	50.61
Stage 12	111	54.62
Stage 13	102	50.17
Stage 14	135	66.67
Stage 15	137	67.70
Stage 16	140	69.23
Stage 17	160	79.25
Stage 18	177	87.70
Stage 19	182	90.25
Stage 20	211	104.75
Stage 21	213	105.76

Lampiran C.5. Tabel Pengujian Deteksi Wajah Frontal

No	Wajah	Posisi Wajah	Keterangan
1	Hasbi	Frontal	Terdeteksi
2	Nurul	Frontal	Terdeteksi
3	Eko	Frontal	Terdeteksi
4	Rifyal	Frontal	Terdeteksi
5	Tredo	Frontal	Terdeteksi
6	Fadli	Frontal	Terdeteksi
7	Duano	Frontal	Terdeteksi
8	Hary	Frontal	Terdeteksi
9	Rahmat	Frontal	Terdeteksi
10	Renol	Frontal	Terdeteksi
11	Sitti	Frontal	Terdeteksi
12	Lora	Frontal	Terdeteksi
13	Siska	Frontal	Terdeteksi
14	Erit	Frontal	Terdeteksi
15	Mey	Frontal	Terdeteksi
16	Winda	Frontal	Terdeteksi
17	Yonggi	Frontal	Terdeteksi
18	Friski	Frontal	Terdeteksi
19	Maulidson	Frontal	Terdeteksi
20	Risky	Frontal	Terdeteksi
21	Zul	Frontal	Terdeteksi
22	Anton	Frontal	Terdeteksi
23	Arya	Frontal	Terdeteksi
24	Bentar	Frontal	Terdeteksi
25	Hafiz	Frontal	Terdeteksi
26	Andre	Frontal	Terdeteksi
27	Hamda	Frontal	Terdeteksi
28	Ega	Frontal	Terdeteksi
29	Adi	Frontal	Terdeteksi
30	Afri	Frontal	Terdeteksi