UNIVERSITAS INDONESIA

IDENTIFIKASI IRIS MATA DENGAN MENGGUNAKAN METODE *HIDDEN MARKOV MODEL*

SKRIPSI

BAMBANG SETIAWAN
06 06 04 2355

FAKULTAS TEKNIK
PROGRAM STUDI TEKNIK ELEKTRO
DEPOK
JULI 2009

UNIVERSITAS INDONESIA

IDENTIFIKASI IRIS MATA DENGAN MENGGUNAKAN METODE *HIDDEN MARKOV MODEL*

SKRIPSI

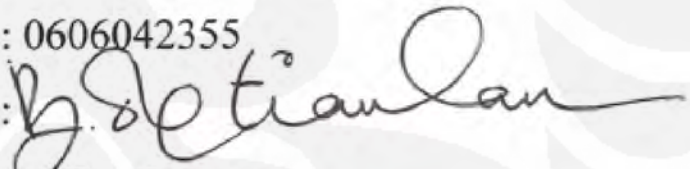
Diajukan sebagai salah satu syarat untuk memperoleh gelar sarjana

BAMBANG SETIAWAN
06 06 04 2355

FAKULTAS TEKNIK
PROGRAM STUDI TEKNIK ELEKTRO
DEPOK
JULI 2009

HALAMAN PERNYATAAN ORISINALITAS

Skripsi ini adalah hasil karya sendiri,
dan semua sumber baik yang dikutip maupun dirujuk
telah saya nyatakan dengan benar

Nama	: Bambang Setiawan
NPM	: 0606042355
Tanda Tangan	: 
Tanggal	: 7 Juli 2009

LEMBAR PENGESAHAN

Skripsi ini diajukan oleh	:
Nama	: Bambang Setiawan
NPM	: 0606042355
Program Studi	: Teknik Elektro
Judul Skripsi	: Identifikasi Iris Mata dengan Menggunakan Metode *Hidden Markov Model*

Telah berhasil dipertahankan di hadapan Dewan Penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Sarjana Teknik ada Program Studi Teknik Elektro, Fakultas Teknik, Universitas Indonesia

DEWAN PENGUJI

Pembimbing : Dr. Ir. Arman Djohan M.Eng ( )

Penguji : Ir. Rochmah N Sukardi Ny MSc ( )

Penguji : Prima Dewi Purnamasari ST., MT., MSc ( )

Ditetapkan di : Ruang Multimedia B LT.2 DTE Depok

Hari / Tanggal : Selasa, 7 Juli 2009

KATA PENGANTAR/UCAPAN TERIMA KASIH

Puji dan syukur saya panjatkan kehadirat Allah SWT, dengan hidayah serta inayah-Nya alhamdulillah saya diberi kekuatan untuk dapat menyelesaikan skripsi ini dengan baik. Penulisan Skripsi ini dilakukan dalam rangka memenuhi salah satu syarat untuk mencapai gelar Sarjana Teknik Jurusan Teknik Elektro pada Fakultas Teknik Universitas Indonesia. Saya menyadari bahwa, tanpa bantuan dan bimbingan dari berbagai pihak, dari masa perkuliahan sampai pada penyusunan skripsi ini, sangatlah sulit bagi saya untuk menyelesaikan skripsi ini. Oleh karena itu, saya mengucapkan terima kasih kepada:

(1) Dr. Ir. Arman Djohan M.Eng, selaku dosen pembimbing yang telah menyediakan waktu, tenaga, dan pikiran untuk mengarahkan saya dalam penyusunan skripsi ini;

(2) Dr. Ir. Muhammad Asvial M.Eng, selaku ketua Departemen Teknik Elektro dan Dr. Ir. Dodi Sudiana M.Eng, selaku sekretaris Departemen Teknik Elektro yang telah banyak membantu saya, sehingga dapat terlaksananya sidang skripsi;

(3) Untuk Ayahanda tercinta yang wafat tanggal 11 Juli 2009. Setelah mendengar Ananda Lulus beliau tersenyum lebar, syukur alhamdulillah, kemudian berkata " Bapak sekarang tidak punya beban", kemudian dengan tenang dan damai menghembuskan nafas terakhirnya. Skripsi ini ku persembahkan untuk Mu Ayah;

(4) Ibunda dan keluarga saya yang telah memberikan bantuan dukungan material dan maoral;

(5) Nur Edi Prasetyo Affan dan teman-teman lainnya yang tidak bisa disebutkan satu persatu yang telah memberi dukungan dan masukan dalam menyelesaikan skripsi ini; dan

(6) Rani Tri Anggraini yang selalu setia mendukung dan memberi semangat.

Akhir kata, Semoga Allah SWT berkenan membalas segala kebaikan semua pihak yang telah banyak membantu. Mudah-mudahan skripsi ini bisa membawa manfaat bagi pengembangan ilmu.

Depok, 7 Juli 2009

Penulis

PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademik Universitas Indonesia, saya yang bertanda tangan di bawah ini:

Nama : Bambang Setiawan
NPM : 0606042355
Program Studi : Teknik Elektro
Departemen : Teknik Elektro
Fakultas : Teknik
Jenis karya : Skripsi

demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Indonesia **Hak Bebas Royalti Noneksklusif (*Non-exclusive Royalty-Free Right*)** atas karya ilmiah saya yang berjudul :

Identifikasi Iris Mata dengan Menggunakan Metode *Hidden Markov Model*

beserta perangkat yang ada. Dengan Hak Bebas Royalti Noneksklusif ini Universitas Indonesia berhak menyimpan, mengalihmedia/formatkan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan mempublikasikan tugas akhir saya tanpa meminta izin dari saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Depok
Pada tanggal : 7 Juli 2009
Yang menyatakan

(Bambang Setiawan)

ABSTRAK

Nama : Bambang Setiawan
Program Studi : S1 Ekstensi Teknik Elektro
Judul : Identifikasi Iris Mata dengan Metode *Hidden Markov Model*

Skripsi ini bertujuan untuk membuat identifikasi iris mata menggunakan *Hidden Markov Model* dengan proses ekstraksi fitur berupa segmentasi, *edge detection* dan polarisasi.

Proses yang pertama adalah pengambilan citra mata. Kemudian dilakukan proses segmentasi terhadap gambar mata yang didapat tersebut untuk memisahkan bagian iris dan pupilnya. Selanjutnya dilakukan proses polarisasi untuk memisahkan bagian iris dengan pupil kedalam bentuk polar. Hasil polarisasi inilah yang akan dikenali oleh sistem pengenal. Proses pengenalan iris mata dalam skripsi ini menggunakan *Hidden Markov Model* yang dilakukan melalui dua tahapan yaitu proses pelatihan data *(training)* yang dilakukan untuk melatih sistem pengenal yang bekerja, agar dapat mengetahui setiap garis-garis pada iris matanya, serta proses pengenalan iris mata itu sendiri *(recognition)* yang digunakan untuk mengenali iris mata yang ingin diuji. Seluruh proses yang dilakukan dibuat menggunakan sebuah perangkat lunak. Dari hasil uji coba yang diperoleh, sistem ini dapat mengenali iris mata yang diuji dengan tingkat akurasi mencapai 100%.

Kata Kunci :

Identifikasi Iris Mata, Proses Segmentasi, Deteksi Tepi, Polarisasi**,** *Hidden Markov Model*

ABSTRACT

Name : Bambang Setiawan
Study Program: S1 Extension Electrical Engineering
Title : Iris Identification Using Hidden Markov Model Method

A Software of iris identification using hidden markov model is developed. The input image is extracted by using segmentation and polaritation process.

The first process is taking of human eye image. Then do the process of segmentation of the image that is to separate iris and pupil from the eyes. Then do the process of polarization to separate iris with the pupil into the polar form. Results of this polarization will be recognized by the user's system. The process of introduction of human iris in this script use the Hidden Markov Model which is done through two stages of the process of *training* is to train a system that works, so that each can know the lines on the eye iris, and the introduction of the iris itself (recognition) that is used to identify iris that you want to test. The whole process is created using a software. From the results of the trials obtained, this system can recognize the iris eyes tested with a level of accuracy reached 100%.

Key words:

Identification Iris, Segmentation Process, Edge Detection , Polaritation**,** Hidden Markov Model

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR SINGKATAN

ROI	*Region Of Interest*
RGB	R*ed Green Blue*
DFT	*Discrete Fourier Transform*
FFT	*Fast Fourier Transform*
VQ	*Vector Quantization*
HMM	*Hidden Markov Model*
PIN	*Personal Identity Number*
LoP	*Log of Probability*

BAB I

PENDAHULUAN

1.1 Latar Belakang

Sistem biometrik saat ini telah mencapai perkembangan yang luar biasa dalam menggantikan sistem verifikasi konvensional. Pemanfaatan anggota tubuh secara unik untuk membedakan antara satu orang dengan orang lain, telah banyak dibuktikan dan memberikan hasil yang lebih akurat dalam pengidentifikasian. Penggunaan iris mata, telapak tangan, sidik jari, bentuk wajah sampai kepada suara telah dikembangkan untuk keperluan tersebut.

Salah satu pemanfaatan organ tubuh untuk diidentifikasi adalah dengan memanfaatkan iris mata. Iris diketahui memiliki tingkat pembeda yang cukup baik untuk dapat mengklasifikasikan tiap individu.

Skripsi ini dimaksudkan untuk bisa menjadi dasar dalam teknik pengidentifikasian, yang nantinya bisa dikembangkan lagi menjadi lebih kompleks seperti dalam bidang kedokteran atau untuk penyelesaian kasus kriminal.

Macam-macam teknik *Artificial Intelligent* (AI) yaitu *Hidden Markov Model (HMM) , Fuzzy Logic* atau *Neural Network.* Teknik *Fuzzy Logic* adalah yang paling sederhana, hanya saja variasi kondisi pada tiap citranya sangat terbatas. Sedangkan teknik *Neural Network* memerlukan proses pembelajaran dan iterasi yang sangat banyak dan panjang. Dengan teknik *Hidden Markov Model*, waktu proses akan jauh lebih cepat dari *Neural Network* dan diharapkan hasil yang didapat akan lebih akurat [4]. Dalam skripsi ini teknik *Artificial Intelligent* yang akan digunakan yaitu teknik *Hidden Markov Model.*

1.2 Rumusan Masalah

Dalam skripsi ini permasalahan yang akan dibahas adalah bagaimana cara mendeteksi citra iris mata yang baik dan akurat dengan mengunakan metode *Hidden Markov Model,* serta menerapkan prinsip-prinsip pengolahan citra pada proses pra-pengolahan dan ekstraksi fitur citra.

1.3 Batasan Masalah

Sesuai dengan rumusan yang telah dipaparkan, maka batasan yang diberlakukan dalam skripsi ini adalah bagaimana merancang algoritma identifikasi citra iris mata dengan menganalisa ciri / kemiripan menggunakan metode *Hidden Markov Model.*

1.4 Tujuan Penulisan

Tujuan dari penulisan skripsi ini adalah untuk merancang dan menguji algoritma pengolahan citra identifikasi iris mata dengan menggunakan metode *Hidden Markov Model.*

1.5 Metodologi Penulisan

Dalam penulisan skripsi ini, metode yang dilakukan meliputi tahap – tahap sebagai berikut:

1. Studi literatur mengenai mata dan iris dengan melakukan pengumpulan data, pencarian informasi melalui buku-buku dan internet.
2. Studi literatur mengenai pemrosesan citra digital, terutama yang berhubungan dengan pemrosesan citra iris mata yang dibutuhkan dalam merancang algoritma.
3. Merancang algoritma untuk beberapa proses yang dibutuhkan, baik dengan menciptakan algoritma baru ataupun memodifikasi algoritma yang telah ada sebelumnya.
4. Membuat sistem simulasi dan pengujian menggunakan perangkat lunak komputasi numerik.

5. Menganalisis dan menyimpulkan hasil pengujian yang dilakukan.
6. Dokumentasi dan laporan.

1.6 Sistematika Penulisan

Sistematika penulisan pada skripsi ini adalah sebagai berikut :

BAB I PENDAHULUAN

Menerangkan latar belakang, tujuan penulisan, batasan masalah dan sistematika penulisan.

BAB II DASAR TEORI

Menerangkan prinsip-prinsip dasar pengolahan citra dan prinsip-prinsip dasar HMM secara umum yang diaplikasikan dalam suatu Sistem Identifikasi Citra Biometrik.

BAB III PERANCANGAN SISTEM

Menerangkan teknik pengolahan citra yang digunakan pada proses pra-pengolahan, ekstraksi fitur citra serta pembentukan, pelatihan dan pengujian *Hidden Markov Model* pada Sistem Identifikasi Citra Iris.

BAB IV UJI COBA DAN ANALISA SISTEM

Menerangkan penggunaan dan pengujian Sistem Identifikasi Citra Iris yang telah dibuat bersama dengan hasil uji coba yang telah dilakukan dan analisanya.

BAB V KESIMPULAN

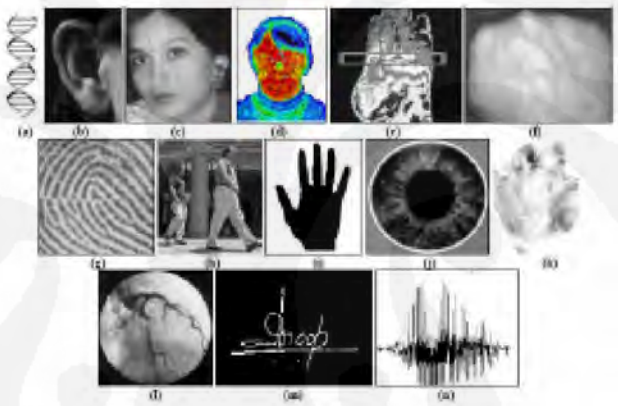
Berisi kesimpulan berdasarkan hasil uji coba Sistem Identifikasi Iris.

BAB II

DASAR TEORI

2.1 Sistem Biometrik

Biometrik berasal dari bahasa Yunani yaitu *bios* = hidup dan *metron* = ukuran, suatu ukuran pengenalan makhluk hidup yang berbasis pada tubuhnya yang unik. Dalam Teknologi Informasi, biometrik lebih sering dipakai sebagai alat otentikasi dengan cara menganalisa karakteristik tubuh manusia yang digunakan, misalnya sidik jari, retina mata, bentuk wajah, cetakan tangan, suara, iris mata dan lain-lain seperti terlihat pada gambar 2.1 di bawah ini.



Gambar 2.1 Karakteristik biometrik [5]

Untuk penggunaan sebagai otentikasi, biometrik harus terlebih dahulu dimasukkan ke dalam *data base* sebuah sistem. Sidik jari biometrik seseorang hanya akan berfungsi bila sidik jari orang tersebut telah terlebih dahulu dimasukkan ke dalam *data base* sistem, sehingga sistem dapat mengenalinya.

Biometrik adalah identitas manusia yang unik, tetapi biometrik bukanlah sesuatu yang dapat dengan mudah dirahasiakan. Kita sulit menyembunyikan biometrik yang kita punyai. Biometrik juga tidak dapat memperbaiki kesalahan yang telah terjadi, sekali biometrik kita dicuri, tidak ada cara untuk mengamankannya kembali. Tidak mungkin manusia mengubah sidik jarinya,

karena sidik jarinya itu telah dicuri orang lain dan digunakan untuk melakukan kriminalitas, misalnya.

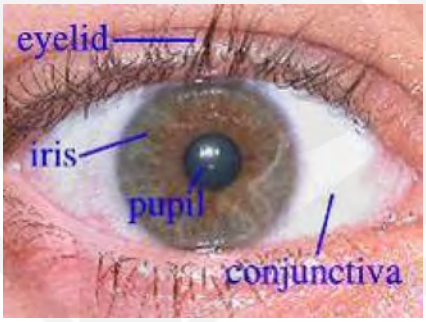
Biometrik sangat sulit dipalsukan, membutuhkan keahlian khusus dan biaya yang tidak sedikit untuk memalsukan biometrik seseorang. Sulit dan mahal sekali untuk memalsukan retina mata, sidik jari, iris mata atau bagian tubuh lainnya. Tetapi beberapa biometrik dapat dengan mudah dicuri, tindakan ini jauh lebih murah dan mudah daripada memalsukannya. Misalkan seorang pencuri telah mengambil gambar sidik jari seseorang untuk mengelabui sistem, sehingga pada saat gambar sidik jari tersebut discan oleh *fingerprint reader* sistem mengira itu adalah sidik jari yang benar. Namun pada kenyataannya untuk mengelabui sidik jari pun tidaklah mudah.

Kita dapat menggunakan biometrik untuk berbagai keperluan yang biasa, misalnya untuk membuka pintu, sebagai alat absensi, atau untuk menghidupkan mesin. Tetapi biometrik tidak dapat digunakan untuk hal-hal yang bersifat rahasia. Walaupun biometrik sangat bagus dan berguna tetapi bukanlah sebuah kunci, karena tidak dapat disembunyikan, tidak dapat dilakukan pengacakan dan tidak dapat ditingkatkan atau dihancurkan. Seperti halnya *password*, kita sebaiknya tidak menggunakan satu password untuk mengunci dua hal yang berbeda, juga sebaiknya tidak menyandi dengan kunci yang sama terhadap dua aplikasi yang berbeda. Dapat dengan mudah dibayangkan betapa tidak amannya penggunaan biometrik untuk hal-hal seperti itu.

Dapat disimpulkan bahwa biometrik akan berfungsi baik hanya bila sistem dapat memeriksa dua hal yaitu pertama, bahwa biometrik itu datang dari orang yang tepat, dan kedua, bahwa biometrik itu klop dengan data base biometrik yang terdapat dalam sistem. Biometrik sangat bagus sebagai pengganti PIN (*personal identity number*) atau pengganti tanda tangan. Tetapi perlu tetap diingat bahwa biometrik tidak dapat dirahasiakan [5].

2.2 Anatomi Iris Mata Manusia

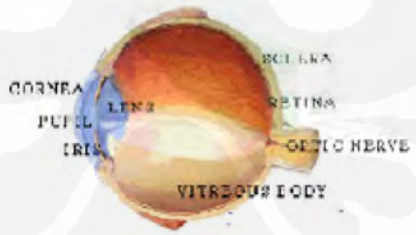
Iris merupakan bagian yang berwarna yang tampak pada bola mata. Bagian iris terlihat seperti pada gambar 2.2 sebagai lingkaran mata yang melingkupi bagian hitam pupil dengan warna-warna tertentu.



Gambar 2.2 Penampakan Iris pada Bagian Mata [8]

2.2.1 Karakteristik Iris

Secara anatomi iris merupakan sebuah organ internal yang dilindungi, terletak di belakang *kornea* dan *aqueous humour*, serta berada di depan lensa mata. Iris merupakan satu-satunya organ internal tubuh yang dapat terlihat dari luar. Iris dapat terlihat cukup jelas pada jarak 1 meter. Gambar 2.3 di bawah ini merupakan struktur anatomi mata.



Gambar 2.3 Struktur Anatomi Mata [2]

Bagian depan dari iris berbentuk tidak teratur, cenderung kasar serta memiliki alur yang tidak rata. Bagian ini dibentuk oleh lapisan yang terdiri dari sel pigmen dan *fibroblast*. Bagian bawah dari lapisan ini adalah jaringan ikat yang berkadar darah rendah (*poorly vascularized*) dengan beberapa serat, *fibroblast* dan *melanocyte*. Bagian selanjutnya merupakan bagian yang kaya akan *supply* darah tertutup oleh jaringan ikat yang longgar [6].

Tekstur dari iris bersifat stokastik. Hal ini disebabkan karena morfologi iris sangat bergantung pada kondisi awal pada fasa *mesoderm embrionik*, saat dimana iris mulai berkembang. Bentuk *fenotif* dari dua iris yang bahkan mempunyai *genotif* yang sama (misalnya pada dua orang yang kembar atau iris kiri dan kanan) akan tidak berkolerasi satu sama lainnya.

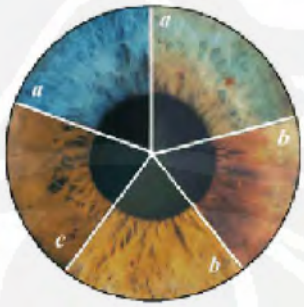
Adapun karakteristik iris adalah :

1. Mempunyai bentuk geometri polar, merupakan sistem koordinat yang alami.
2. Mempunyai tingkat ketidakteraturan yang tinggi, dan mempunyai *entropi* 3,2 bit per milimeter persegi jaringan iris.

2.2.2 Warna Dasar Iris

Warna iris manusia sangat beraneka ragam, tergantung dari ras dan etniknya. Beberapa ahli iridiologi mengklarifikasikan warna iris yang menjadi hitam, cokelat, emas, biru tua, hijau, biru muda, dan abu-abu. Kesemuanya itu dapat dibagi menjadi tiga bagian utama, yaitu : biru, cokelat, dan campuran.

Warna biru dan cokelat merupakan warna iris murni. Warna campuran berasal dari gabungan genetik yang berbeda sebagai akibat perkawinan antar ras, dan merupakan anomali genetik dari warna iris cokelat dan biru. Gambar 2.4 di bawah ini merupakan skema warna iris mata.



Gambar 2.4 Skema Warna Iris Mata [2]

Iris Biru

Warna biru dari iris merupakan refleksi cahaya dari jaringan *ephitel posterior* yang terlihat melalui stroma yang tak berpigmen (lapisan otot pada iris). Iris berwarna dasar biru/abu-abu ditemukan di Nordic, Eropa, dan ras Anglo-Saxon.

Iris Cokelat

Pada iris yang berwarna cokelat, sel pigmen dari stroma memberikan warna pada mata. Warna cokelat ditentukan oleh konsentrasi sel pigmen dalam mata. Warna dasar iris cokelat ditemukan di Asia, Afrika, Indian, dan Ras Semit.

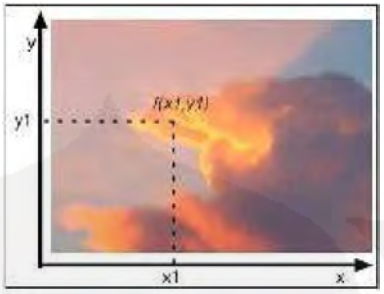
Iris Campuran

Iris campuran merupakan hasil pencampuran antara warna iris biru dan cokelat. Terdapat berbagai macam variasi antara biru dan cokelat. Warna iris campuran mempunyai dasar genetik biru.

Selain warna dasar iris, seringkali kita menemukan ada warna lain didalam iris. Hal ini menunjukkan adanya disfungsi dari organ di dalam tubuh yang letaknya ditunjukkan oleh letak warna lain tersebut di dalam iris. Warna lain ini antara lain putih, kuning, *orange*, cokelat, merah, dan hitam [2].

2.3 Pengolahan Citra Digital

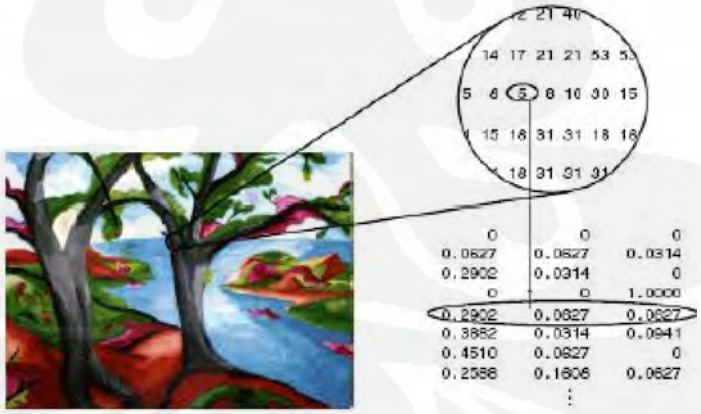
Citra adalah gambar dua dimensi yang dihasilkan dari gambar analog dua dimensi yang kontinu menjadi gambar diskrit melalui proses *sampling*. Citra digital dapat didefinisikan sebagai fungsi dua variabel, *f(x,y)*, dimana *x* dan *y* adalah koordinat spasial sedangkan nilai *f(x,y)* adalah intensitas citra pada koordinat tersebut, hal tersebut diilustrasikan pada gambar 2.5 di bawah ini. Teknologi dasar untuk menciptakan dan menampilkan warna pada citra digital berdasarkan pada penelitian bahwa sebuah warna merupakan kombinasi dari tiga warna dasar, yaitu merah, hijau, dan biru (*Red, Green, Blue* - RGB).



Gambar 2.5 Citra Digital [7]

Citra Berindeks (*Indexed Images*)

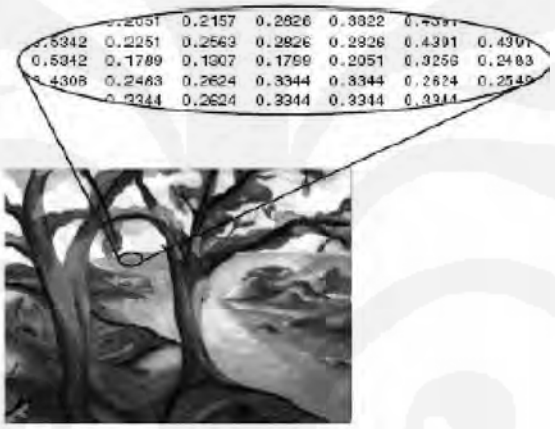
Citra berindeks merupakan citra digital yang mengandung sebuah matriks data x dan matriks peta warna (*colormap*) yang berukuran m x 3 yang mengandung *floating point* dengan nilai antara 0 sampai dengan 1. Setiap baris pada *colormap* adalah data komponen untuk warna merah, hijau dan biru. Citra berindeks menggunakan *direct mapping* untuk menandakan nilai *pixel* pada *colormap*. Warna dari tiap *pixel* ditentukan dengan nilai x yang merupakan indeks pada *colormap*. Sebagai contoh nilai 1 menunjukkan baris pertama pada *colormap*, nilai 2 menunjukkan baris kedua pada *colormap* dan seterusnya seperti terlihat pada gambar 2.6 di bawah ini.



Gambar 2.6 Citra Berindeks [7]

Citra Intensitas (*Intensity Images*)

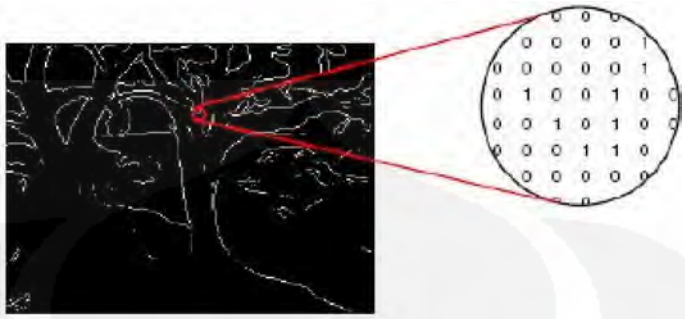
Citra Intensitas disebut juga citra *grayscale*. Citra *grayscale* merupakan citra digital yang mengandung matriks data I yang merepresentasikan nilai dalam suatu *range*. Elemen-elemen dalam matriks intensitas merepresentasikan berbagai nilai intensitas atau derajat keabuan, dimana nilai 0 merepresentaikan warna hitam dan 1, 255 atau 65535 merepresentasikan intensitas penuh atau warna putih. Hal tersebut diilustrasikan pada gambar 2.7 di bawah ini



Gambar 2.7 Citra Intensitas [7]

Citra Biner (*Binary Images*)

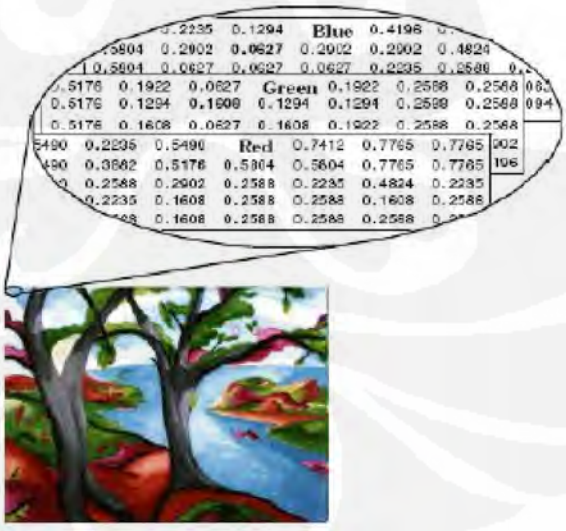
Citra biner merupakan citra yang telah melalui proses pemisahan *pixel* – *pixel* berdasarkan derajat keabuan yang dimiliki. Pembentukan citra biner memerlukan nilai batas keabuan yang akan digunakan sebagai nilai patokan. *Pixel* dengan derajat keabuan lebih besar dari nilai batas akan diberi nilai 1 dan sebaliknya *pixel* dengan derajat keabuan lebih kecil dari nilai batas akan diberi nilai 0 seperti ditunjukkan pada gambar 2.8 dibawah ini.



Gambar 2.8 Citra Biner [7]

Citra RGB (RGB *Image*s)

Citra RGB disebut juga citra *truecolor*. Citra RGB merupakan citra digital yang mengandung matriks data berukuran m x n x 3 yang merepresentasikan warna merah, hijau, dan biru untuk setiap *pixel*nya. Setiap warna dasar diberi rentang nilai. Untuk monitor komputer, nilai rentang paling kecil 0 dan paling besar 255. Pemilihan skala 256 ini didasarkan pada cara mengungkap 8 digit bilangan biner yang digunakan oleh komputer. Sehingga total warna yang diperoleh adalah lebih dari 16 juta warna, sperti terlihat pada gambar 2.9 di bawah ini. Warna dari tiap *pixel* ditentukan oleh kombinasi dari intensitas merah, hijau, dan biru.



Gambar 2.9 Citra RGB [7]

Alasan mengapa dilakukan pengolahan citra digital :

1. Untuk mendapatkan citra asli dari suatu citra yang sudah buruk karena pengaruh derau.
2. Untuk memperoleh citra dengan karakteristik tertentu dan cocok secara visual yang dibutuhkan untuk tahap lebih lanjut dalam pemrosesan analisis citra.

Dalam proses akuisisi, citra yang diolah ditransformasikan dalam suatu representasi numerik . Pada proses selanjutnya representasi numerik tersebutlah yang akan diolah secara digital oleh komputer.

Operasi Pengolahan Citra

Operasi-operasi yang dilakukan dalam pengolahan citra banyak ragamnya, namun secara umum operasi pengolahan citra dapat diklasifikasikan dalam beberapa jenis sebagai berikut:

1. Perbaikan Kualitas Citra (*image enhancement*)

Jenis operasi ini bertujuan untuk memperbaiki citra dengan cara memanipulasi parameter-parameter citra. Dengan operasi ini, ciri-ciri khusus yang khusus yang terdapat didalam citra lebih ditonjolkan.

Contoh-contoh operasi perbaikan citra:

a. Perbaikkan kontras gelap/terang

b. Perbaikkan tepian objek (*edge enhancement*)

c. Penajaman (*sharpening*)

d. Pemberian warna semu (*pseudocoloring*)

e. Penapisan derau (*noise filtering*)

2. Pemugaran Citra (*image restoration*)

Operasi ini bertujuan menghilangkan cacat pada citra. Tujuan pemugaran citra hampir sama dengan operasi perbaikkan citra. Bedanya, pada pemugaran citra penyebab degradasi gambar diketahui.

Contoh-contoh operasi pemugaran citra :

a. Penghilangan kesamaran (*deblurring*)

b. Penghilangan derau (*noise*)

3. Pemampatan Citra (*image compression*)

Jenis operasi ini dilakukan agar citra dapat direpresentasikan dalam bentuk yang lebih kompak sehingga memerlukan memori yang lebih sedikit. Hal penting yang harus diperhatikan dalam pemampatan citra adalah citra yang telah dimampatkan harus tetap mempunyai kualitas gambar yang bagus.

4. Segmentasi Citra (*image segmentation*)

Jenis operasi ini bertujuan untuk memecah suatu citra kedalam beberapa segmen dengan suatu kriteria tertentu. Jenis operasi ini berkaitan erat dengan pengenalan pola.

5. Pengorakan Citra (*Image Analysis*)

Jenis operasi ini bertujuan menghitung besaran kuantitif dari citra untuk menghasilkan deskripsinya. Teknik pengolahan citra mengekstraksi ciri-ciri tertentu yang membantu dalam identifikasi objek. Proses segmentasi kadang kala diperlukan untuk melokalisasi objek yang diinginkan dari sekelilingnya.

Contoh-contoh operasi pengorakan citra :

a. Pendeteksian tepian objek (*edge detection*)

b. Ekstraksi batas (*boundary*)

c. Representasi Daerah (*region*)

6. Rekonstruksi Citra (*Image Reconstruction*)

Jenis operasi ini bertujuan untuk membentuk ulang objek dari beberapa citra hasil proyeksi. Operasi rekonstruksi citra banyak digunakan dalam bidang medis.

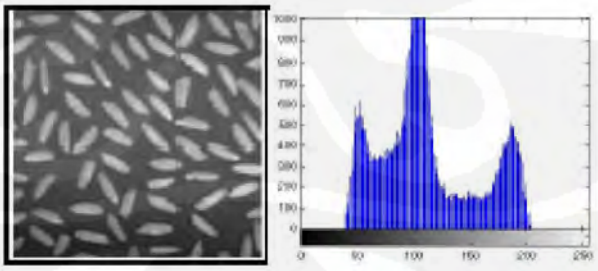
2.3.1 Pengaturan Intensitas

Intensity adjustment bekerja dengan cara melakukan pemetaan linear terhadap nilai intensitas pada histogram awal menjadi nilai intensitas pada histogram yang baru. Perintah umum untuk melakukan pemetaan linear tersebut adalah:

Intensity = atur_intensitas (Image, [(low_in, high_in), (low_out, high_out)])

Low_in merupakan nilai intensitas yang akan dipetakan sebagai *low_out*, *high_in* merupakan nilai intensitas yang akan dipetakan sebagai *high_out*.

Pada citra seperti pada gambar 2.10, memiliki nilai kontras yang rendah. Berdasarkan histogramnya dapat diketahui bahwa citra ini tidak memiliki *pixel* dengan intensitas di bawah 40 dan di atas 225. Untuk memperbaikinya, kita dapat memetakan histogram secara linear sehingga diperoleh sebuah citra baru yang memiliki rentang histogram antara 0 hingga 255.



Gambar 2.10 Contoh tampilan sebuah citra dan histogramnya [9]

2.3.2 EKSTRAKSI NILAI *PIXEL RED, GREEN* dan *BLUE* (RGB)

Warna menjadi elemen panting pada gambar/citra yang akan ditampilkan. Warna dibagi menjadi 3 dasar warna dasar yang sering disebut RGB, yaitu *Red*

(merah), *Green* (hijau), dan *Blue* (biru). Dengan ketiga warna inilah pengolahan warna dapat dilakukan. Pengolahan citra secara intensif memanfaatkan warna RGB ini.

2.3.3 *Grayscale*

Proses mengubah citra warna menjadi citra *grayscale* digunakan dalam *image processing* untuk menyederhanakan model citra. Karena citra yang sebelumnya berukuran besar, maka dari itu sederhanakan dengan teknik *grayscale* ini. Dimana citra hasil *grayscale* memiliki format *grayscale* 8 bit (2^8 = 256 derajat keabuan) yang memiliki level intensitas yang sama untuk tiap-tiap layernya.

2.3.4 Binerisasi

Thresholding digunakan untuk mengatur derajat keabuan yang ada pada citra. Dengan *thresholding* maka derajat keabuan bisa diubah sesuai dengan keinginan. Teknik ini memisahkan bagian gambar yang sesuai dengan obyek (*foreground*) dan latar belakangnya (*background*). Metode ini juga digunakan untuk mengkonversi data image menjadi data biner dengan tujuan agar proses selanjutnya menjadi mudah. Hasil dari proses ini nantinya hanya warna hitam (*pixel* 0) dan warna putih (*pixel* 1) yang akan digunakan untuk memisahkan iris dari mata.

2.3.5 Morphologi

Operasi morphologi adalah teknik pengolahan citra yang didasarkan pada bentuk segmen atau *region* dalam citra. Karena difokuskan pada bentuk obyek, maka operasi ini biasanya diterapkan pada citra biner.
Segmentasi dilakukan dengan membedakan antara obyek dan latar, antara lain dengan memanfaatkan operasi yang mengubah citra warna dan skala keabuan menjadi citra biner.

Dalam aplikasi *image processing* ada yang disebut dilasi (*dilation*) dan erosi (*erosion*), biasanya digunakan dalam berbagai kombinasi. Gambar akan melewati dilasi atau erosi menggunakan struktur elemen yang sama atau kadang berbeda. Pada bagian ini kombinasi ini membagi 3 atau yang memungkinkan

yaitu *opening*, *closing*, dan *hit-or-miss transformation* (berhasil tidaknya transformasi *image*). Beberapa operasi morfologi yang dapat dilakukan adalah:

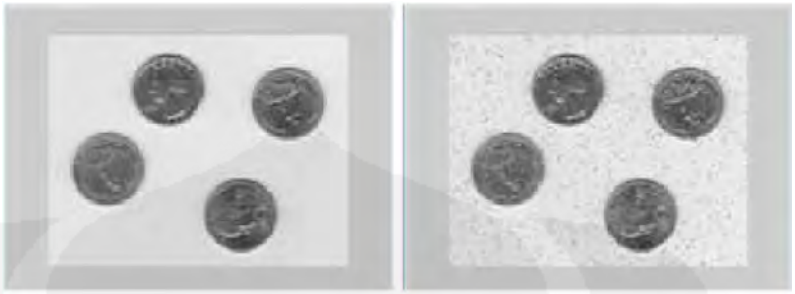
1. Erosi : operasi pengurangan *pixel* pada pinggiran tiap obyek biner yaitu daerah yang memiliki nilai 1.
2. Dilasi : operasi penambahan *pixel* pada pinggiran tiap obyek biner yaitu daerah yang memiliki nilai 1.
3. *Opening* : kombinasi operasi erosi yang diikuti dengan operasi dilasi dengan struktur elemen yang sama.
4. *Closing* : operasi kebalikan dari opening yaitu kombinasi operasi dilasi yang diikuti dengan operasi erosi dengan struktur elemen yang sama.
5. Skeletonisasi : operasi pembentukan kerangka obyek
6. *Thinning* : operasi mengerosi obyek hingga berbentuk garis-garis
7. *Shrinking* : operasi menghilangkan obyek sampai menjadi berbentuk titik-titik.

2.3.6 Rekonstruksi / Memisahkan Hanya Citra Iris Saja

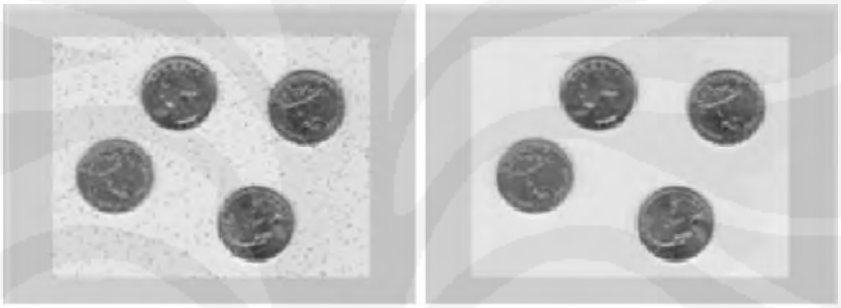
Dari teknik sebelumnya didapatkan bulatan hitam dan layar putih. Disini akan dilakukan pemisahan citra iris berdasarkan perbedaan nilai putih dan hitam tersebut. Sehingga akan direkonstruksi ulang citra iris asli yang hanya bagian irisnya saja. Langkah selanjutnya adalah untuk mengambil hanya bagian iris dengan melakukan rekonstruksi ulang citra mata asli.

2.3.7 *Filter Median*

Filter median sangat bermanfaat untuk menghilangkan *outliers*, yaitu nilai-nilai yang ekstrim. *Filter median* menggunakan *sliding neighborhoods* untuk memproses suatu citra, suatu operasi dimana filter ini akan menentukan nilai masing-masing *pixel* keluaran dengan menentukan tetangga m x n disekitar *pixel* masukan yang bersangkutan, *Filter median* mengatur nilai-nilai *pixel* dalam satu tetangga dan memilih nilai tengah atau median sebagai hasil. Pada gambar 2.11 dan gambar 2.12 merupakan contoh menambahkan *salt and pepper* pada citra koin lalu menghilangkanya dengan menggunakan *filter median*.



Gambar 2.11 Penambahan *noise* (a) Citra Asli ;(b) Citra yang diberi *noise*[9]



Gambar 2.12 Penghilangan *noise* dengan *filter median* [9]

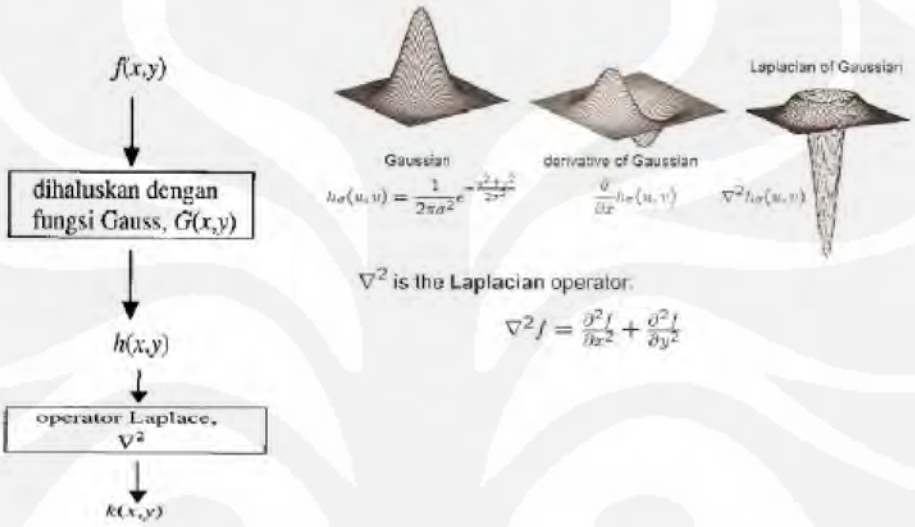
2.3.8 Tahap Transformasi

a. Deteksi Tepi *Canny*

Setelah melewati tahap pemisahan iris dan dilanjutkan perbaikan mutu citra. Tapi untuk pengenalan garis-garis irisnya sangat menyulitkan, oleh sebab itu diperlukan teknik deteksi tepi. Dan deteksi yang dipakai adalah deteksi *canny*. Ada beberapa teori deteksi tepi, yaitu:

1. Metoda *canny*
2. Metoda *sobel*
3. Metoda *Roberts*
4. Metoda *Log*
5. Metoda *Prewitt*
6. Metoda *Zerocross*

Yang akan dipilih adalah metoda *canny*, karena memiliki beberapa kelebihan dalam mengekstrak tepian. Operator *Canny* merupakan deteksi tepi yang paling optimal, dikarenakan operator *Canny* menggunakan *Gaussian Derivative Kernel* untuk menyaring kegaduhan dari citra awal untuk mendapatkan hasil deteksi tepi yang halus.



Gambar 2.13 *Gaussian Derivative Kernel*

Tahapan dalam metoda *canny* dapat dilihat pada gambar 2.14.

Gambar 2.14 Tahapan metoda canny [2]

Pendekatatan metoda *canny* digunakan dengan konvolusi fungsi *image* dengan operator *Gaussian* dan turunannya.

$$G(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-x^2}{2\sigma^2}} \quad \text{.....(2.1)}$$

turunan pertamanya :

$$G'(x) = \frac{-x}{\sqrt{2\pi}\sigma^3} e^{\frac{-x^2}{2\sigma^2}} \quad \text{.....(2.2)}$$

dan turunan keduanya adalah :

$$G''(x) = -\frac{1}{\sqrt{2\pi}\sigma^3} e^{\frac{-x^2}{2\sigma^2}} \left[1 - \frac{x^2}{\sigma^2}\right] \quad \text{.....(2.3)}$$

b. Transformasi Koordinat Polar [2]

Bentuk iris yang berupa lingkaran akan sangat menyulitkan untuk dianalisis dan diolah lebih lanjut. Pola susunan *pixel* yang dianalisa harus mengikuti algoritma tertentu yang memungkinkan pengambilan *pixel* dengan bentuk geometri lingkaran. Hal ini akan sangat merepotkan dan tidak efisien. Untuk mengatasi hal tersebut kita harus terlebih dahulu mengubah citra iris ke dalam bentuk antara yang sesuai. Pengubahan bentuk ini dapat dilakukan dengan melakukan transformasi koordinat polar dari citra iris. Untuk dapat melakukan transformasi tersebut, pertama harus dilakukan deteksi tepian, perhitungan parameter koordinat polar, baru kemudian dilakukan pembentukan citra koordinat polar itu sendiri.

1. Perhitungan Parameter Koordinat Polar [2]

Parameter lingkaran dibutuhkan untuk pembentukan citra polar. Titik pusat dan jari-jari diperlukan sebagai referensi dalam mentransformasikan citra berbentuk lingkaran ke dalam bentuk koordinat polar.

Untuk dapat menghitung parameter-parameter tersebut, terlebih dahulu diambil beberapa titik sampel pada tepian citra yang membentuk pola lingkaran. Jumlah titik sampel yang diambil sangat berpengaruh pada ketelitian pengukuran. Semakin banyak titik sampel yang diambil maka perhitungan yang dilakukan akan semakin teliti. Namun di sisi lain perhitungan akan berjalan sangat rumit dan memakan waktu cukup lama.

Untuk itu, pemilihan titik sampel harus dipertimbangkan secara optimum karena akan sangat berpengaruh pada unjuk kerja sistem nantinya.

2. Perhitungan Titik Pusat [2]

Kita dapat menggunakan rumus dasar lingkaran yang telah diketahui untuk menghitung titik pusat dari lingkaran.

$$R^2 = (x-a)^2 + (y-b)^2 \quad \text{.....(2.4)}$$

$$R^2 = (x^2 - a^2) - (2x)a - (2y)b + a^2 + b^2 \quad \text{.....(2.5)}$$

Dimana: R = Jari-jari yang di cari
a dan b = Titik pusat citra

Dari persamaan di atas untuk mencari 2 (dua) buah variabel absis dan ordinat titik pusat dibutuhkan sedikitnya minimal 3 (tiga) buah persamaan yang berasal dari 3 (tiga) titik sampel. Oleh karena itu pemilihan sampel harus lebih dari 3 (tiga) buah titik pada tepian lingkaran. Jumlah perhitungan yang akan dilakukan merupakan hasil kombinasi semua titik sampel pada tiap perhitungan 3 (tiga) titik yang dibutuhkan. Jumlah perhitungan yang dilakukan adalah sebanyak 3xN proses dengan N adalah banyaknya titik sampel yang diambil.

3. Perhitungan Jari-jari Lingkaran [2]

Setelah mengetahui titik pusat, jari-jari lingkaran dapat dengan mudah dihitung menurut persamaan berikut :

$$R_{nn}^2 = \left((x_n^2 + y_n^2) - (2x_n)a - (2y_n)b + a^2 + b^2\right)$$

n = 0,1,….N..... (2.6)

Hasil perhitungan akhir diperoleh dengan mengambil nilai rata-rata dari N buah perhitungan jari-jari yang diperoleh dari N buah titik sampel yang berbeda.

4. Pembentukan Citra Polar [2]

Parameter-parameter yang telah dihitung dapat kita pakai untuk mengubah bentuk citra iris yang berbentuk lingkaran ke dalam koordinat polar. Citra ditransformasikan ke dalam bentuk polar dengan titik pusat sebagai acuannya. Pada gambar di bawah, lebar data citra hasil transformasi adalah sebesar a-b dan panjang data citra sebesar c-d. Lebar citra sangat tergantung kepada besar jari-jari dalam dan jari-jari luar lingkaran. Sedangkan panjang data citra tergantung kepada besarnya pengambilan *pixel* tiap derajat lingkaran. Proses transformasi dapat diilustrasikan pada Gambar 2.15.

Gambar 2.15 Ilustrasi Transformasi Koordinat Polar [2]

2.4 *Discrete Fourier Transform* dan *Fast Fourier Transform*

Discrete Fourier Transform (DFT) digunakan untuk mengubah *frame-frame* dari domain spasial ke domain frekuensi. Transformasi diskrit merupakan transformasi dimana input dan output bernilai diskrit yang digunakan untuk manipulasi di komputer. Rumus DFT untuk mengubah N data dari domain spasial ke domain frekuensi adalah sebagai berikut :

$$F(u) = \frac{1}{N}\sum_{x=0}^{N-1} f(x) e^{-j2\pi ux/N} \quad \text{.....(2.7)}$$

dimana : $F(u)$: transformasi *fourier*

$f(x)$: fungsi *image* dalam domain spasial

N : jumlah data

Fast Fourier Transform (FFT) merupakan algoritma yang lebih cepat dari *Discrete Fourier transform* (DFT). FFT dapat mereduksi jumlah perhitungan untuk setiap N data yang sama pada perhitungan DFT sehingga perhitungan yang ada menjadi lebih cepat khususnya ketika nilai N yang digunakan cukup besar [2] menggunakan persamaan:

$$F(u)=\frac{1}{2}\left[\frac{1}{M}\sum_{x=0}^{M-1}f(2x)W_M^{ux}+\frac{1}{M}\sum_{x=0}^{M-1}f(2x+1)W_M^{ux}W_{2M}^{u}\right] \quad \text{.....(2.8)}$$

W_M^{ux} dapat dituliskan sebagai

$$W_M^{ux}=e^{-j2ux\pi/M} \quad \text{.....(2.9)}$$

W_{2M}^{u} dapat dituliskan sebagai

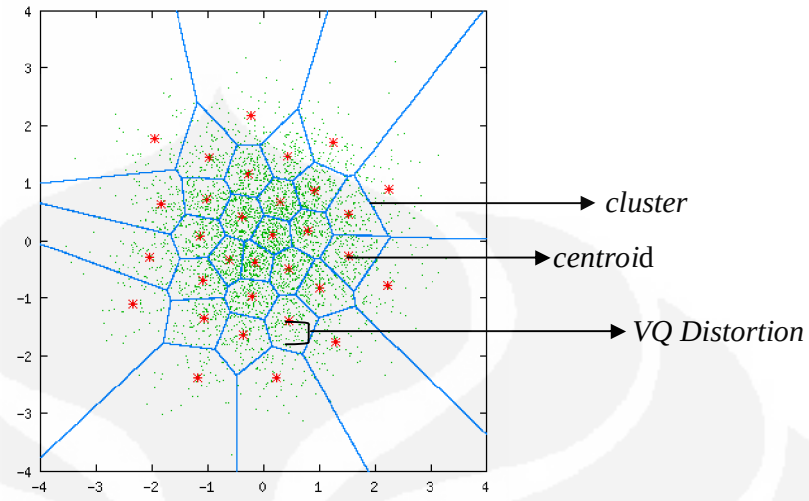
$$W_{2M}^{u}=e^{-j\pi u/M} \quad \text{.....(2.10)}$$

Dimana : $M=\frac{1}{2}N$

Baik DFT maupun FFT akan menghasilkan spektrum frekuensi berupa kumpulan titik-titik dimana masing-masing titik terdiri dari komponen *real* (fungsi *Cosinus*) dan komponen *imaginer* (fungsi *sinusoidal*) disebabkan adanya bilangan eksponensial. Kumpulan titik-titik ini kemudian akan digunakan dalam *Vector quantization* yang akan dijelaskan dalam sub-bab berikut ini.

2.5. *Vector Quantization*

Vector quantization (VQ) adalah proses pemetaan vektor data yang merupakan titik-titik hasil dari proses FFT ke dalam sebuah wilayah yang terbatas dalam grafik dua dimensi (x-y) dimana sumbu x merupakan komponen *real* dari masing-masing titik dan sumbu y merupakan komponen *imajiner* dari masing-masing titik. Pemetaan titik-titik tersebut ditunjukkan oleh Gambar 2.16.



Gambar 2.16 Pemetaan pada proses vektor kuantisasi [3]

Tujuan dari proses vektor kuantisasi adalah untuk menyederhanakan panjang data masukan agar proses selanjutnya menjadi lebih mudah. Tiap komponen dari spketrum frekuensi yang merupakan hasil FFT memiliki beberapa titik yang masing-masing memiliki komponen *real* dan *imaginer*. Kumpulan dari titik-titik yang memiliki jarak berdekatan membentuk suatu *cluster* dan setiap *cluster* yang terbentuk dapat direpresentasikan dengan *centroid* yang disebut *codeword*. Koleksi dari semua *codeword* disebut *codebook* . Jarak antara satu titik dengan titik lain dalam sebuah *cluster* disebut *VQ Distortion*. Semakin kecil *VQ Distortion*-nya, maka *cluster* yang terbentuk menjadi lebih akurat.

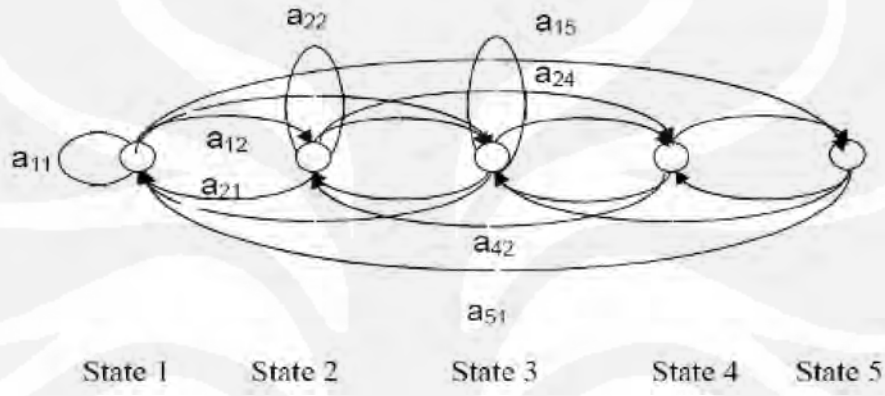
Luas daerah *cluster* ditentukan oleh ukuran *codebook* dimana semakin besar ukuran *codebook*nya, maka luas daerah masing-masing *cluster* menjadi lebih kecil dan jumlah *cluster* yang terbentuk menjadi lebih banyak disertai nilai *VQ distortion* yang semakin kecil sehingga *codeword* yang terbentuk akan semakin mewakili informasi dari masukannya.

2.6. *Hidden Markov Model*

Hidden Markov Model (HMM) adalah pemodelan probabilitas suatu sistem dengan mencari parameter-parameter yang tidak diketahui untuk mempermudah proses analisis sistem tersebut. HMM memiliki 3 parameter utama

yang harus dicari nilainya terlebih dahulu. Ketiga parameter itu adalah sebagai berikut :

1. Parameter A : disebut sebagai probabilitas transisi, merupakan probabilitas kedudukan suatu *state* terhadap semua state yang ada, termasuk kedudukan terhadap *state* itu sendiri. Contoh dari matriks transisi dapat dilihat pada Gambar 2.17.



Gambar 2.17 Contoh matriks transisi [4]

Parameter A pada HMM dinyatakan dalam sebuah matriks dengan ukuran M x M dengan M adalah jumlah *state* yang ada. Matriks transisi pada Gambar 2.17 terdiri dari 5 state sehingga setiap state memiliki 5 hubungan transisi, maka parameter A dapat dituliskan dalam bentuk matriks seperti pada persamaan 2.11

$$A = aij = \begin{pmatrix} a11 & a12 & a13 & a14 & a15 \\ a21 & a22 & a23 & a24 & a25 \\ a31 & a32 & a33 & a34 & a35 \\ a41 & a42 & a43 & a44 & a45 \\ a51 & a52 & a53 & a54 & a55 \end{pmatrix} \quad \ldots\ldots (2.11)$$

2. Parameter B : disebut sebagai probabilitas *state*, merupakan probabilitas kemunculan suatu state dalam deretan seluruh *state* yang ada.

Parameter B dalam HMM dituliskan dalam bentuk matriks kolom dengan ukuran M x 1 dimana M merupakan jumlah seluruh state yang ada. Sebagai contoh, jika terdapat 5 buah *state* dalam suatu kondisi, maka matriks B yang terbentuk ditunjukkan oleh persamaan 2.10.

$$B = \begin{pmatrix} b1 \\ b2 \\ b3 \\ b4 \\ b5 \end{pmatrix} \quad \ldots\ldots (2.12)$$

3. Parameter π : disebut sebagai probabilitas awal, merupakan probabilitas kemunculan suatu *state* di awal.

Sama halnya dengan parameter B, parameter π juga dituliskan dalam bentuk matriks kolom dengan ukuran M x 1 dimana M adalah jumlah *state*nya. Jadi jika terdapat 5 *state*, maka parameter π yang dihasilkan akan ditunjukkan seperti pada persamaan 2.11.

$$\Pi = \begin{pmatrix} c1 \\ c2 \\ c3 \\ c4 \\ c5 \end{pmatrix} \quad \ldots\ldots (2.13)$$

Dari ketiga parameter utama maka HMM dapat dituliskan dalam bentuk $\lambda = (A, B, \pi)$. Dari kesemua parameter yang ada maka bisa diperoleh suatu probabilitas observasi(O). Fungsi untuk probabilitas O ditunjukkan oleh persamaan 2.12.

$$P(O) = \sum_{i=1}^{N} P(A_{ij}) * P(B_i) \quad \ldots\ldots (2.14)$$

Berikut adalah contoh perhitungan untuk mencari probabilitas observasi:

Citra 1 à (w1, w1, w2, w1, w2) à P(O) citra1 = c_1*a_{11}* b_1* a_{12} *b_2* a_{21}*b_2* a_{12}*b_1

Citra 2 à (w2, w1, w1, w3, w2) à P(O) citra2 = c_2* a_{21}* b_2* a_{11} *b_1* a_{13}*b_1* a_{32}*b

BAB III

PERANCANGAN SISTEM

3.1 Pra-Pengolahan

Proses pra-pengolahan atau lebih dikenal *pre-processing* adalah langkah memperbaiki citra untuk menonjolkan karakter citra yang ingin diekstraksi. Pada proses ini terdiri dari beberapa sub-proses yang meliputi proses akuisisi citra, pengaturan intensitas, proses ekstraksi warna RGB, binerisasi, morphologi, proses *autocrop,* transformasi *canny* dan polarisasi.

3.2 Algoritma Pra-Pengolahan

Urutan algoritma pra-pengolahan adalah sebagai berikut :

3.2.1 Pengambilan Citra

Citra iris ini berupa citra mata bagian mata sebelah kanan dari berbagai macam ras dan etnik yang diambil dari Department of Computer Science of the University of Beira Interior [8].

Sampel 10 citra mata dari berbagai macam ras dan etnik ini bisa dilihat di lampiran.

3.2.2 Segmentasi Iris Mata

1. Akuisisi citra mata

   Pada langkah ini ukuran citra mata yang didapat berukuran 400 x 300 *pixel*. Dengan ukuran ini dapat mengurangi beban kerja komputer sehingga waktu komputasinya lebih cepat. Algoritma pengambilan citra mata adalah

   ```
   citra_rgb = baca citra 'mata.jpg'
   tampilkan citra_rgb
   ```

   Tampilan citra mata RGB dapat dilihat pada gambar 3.1 di bawah ini.

Gambar 3.1 Citra Mata

2. Proses pengaturan intensitas

Kurang sempurnanya teknik pengambilan gambar kadangkala menyebabkan kurang meratanya penyebaran cahaya pada citra iris mata. Untuk memudahkan dalam proses morphologi dan dihasilkan *cropping* yang rapi. Maka citra iris dibuat memiliki nilai kontras lebih tinggi dibanding dengan citra di belakangnya. Maka dilakukan *Intensity adjustment*. Algoritma pengaturan intensitas adalah

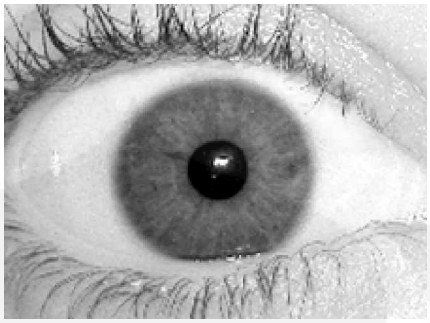
```
panggil citra_rgb
citra_intensitas = pengontrasan citra_rgb
tampilkan citra_Intensitas
```

Tampilan citra hasil *intensity adjustment* dapat dilihat pada gambar 3.2.

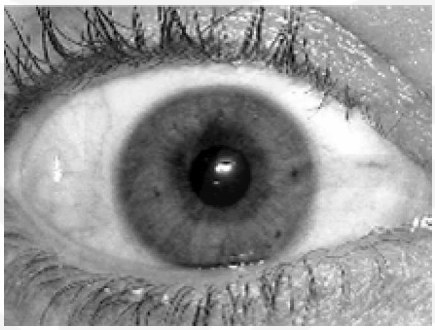
Gambar 3.2 Citra hasil pengaturan intensitas

3. Ekstraksi warna

Pada langkah ini citra mata yang telah melalui proses pengaturan intensitas, warna penyusun RGB nya dipecah menjadi tiga buah citra *monochrome Red* (merah), *Green* (hijau), dan *Blue* (biru), seperti dapat dilihat pada gambar 3.3, gambar 3.4, dan gambar 3.5 dibawah ini.



Gambar 3.3 Elemen warna merah (*Red*)



Gambar 3.4 Elemen warna hijau (*Green*)

Gambar 3.5 Elemen warna biru (*Blue*)

Algoritma ekstraksi warna adalah

```
panggil citra_hasil intensitas
citra_red = elemen red citra_hasil intensitas
citra_green = elemen green citra_hasil intensitas
citra_blue =  elemen blue citra_hasil intensitas
tampilkan citra_red
```

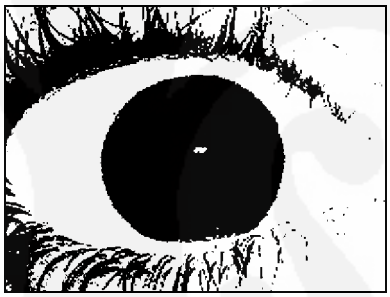
Dalam hal ini citra yang diambil adalah citra *monochrome red* dikarenakan bagian kulitnya terlihat lebih terang, sehingga lebih mudah untuk dipisahkan dari iris untuk proses selanjutnya.

4. Binerisasi

Hasil dari proses ini nantinya hanya warna hitam (atau *pixel* 0) dan warna putih (atau *pixel* 1), agar memudahkan dalam memisahkan iris dari mata. Algoritma proses binerisasi adalah

```
panggil citra_hasil ekstraksi warna
citra_biner = binerisasi citra_hasil ekstraksi warna
              dengan threshold 0.7
tampilkan citra_biner
```

Hasil proses binerisasi ini dapat dilihat pada gambar 3.6 di bawah ini.



Gambar 3.6 Citra hasil binerisasi dengan *threshold* 0.7

5. Morphologi

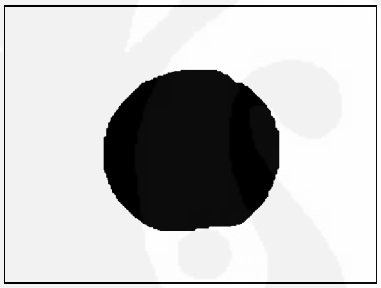
Saat citra mata diambil terkadang ada cahaya yang dipantulkan ke mata. Pantulan itu akan menimbulkan bercak putih pada citra mata. Akibatnya saat citra tersebut diubah ke bentuk biner, pantulan cahaya tersebut akan berwarna putih dan akan menggangu dalam proses *cropping* yang baik. Untuk mengatasi hal tersebut maka dilakukan operasi morphologi yaitu operasi *closing* yang mengkombinasikan operasi dilasi yang diikuti dengan operasi erosi dengan struktur elemen yang sama. Algoritma operasi morphologi adalah

```
Panggil citra_hasil binerisasi
--Proses Morphologi Closing--
Struktur elemen("circle",35)
citra_closing = closing(citra_hasil binerisasi, struktur
elemen)

--Proses Morphologi Dilasi--
struktur elemen("circle",35)
citra_dilasi = dilasi(citra_hasil binerisasi, struktur
elemen)

--Proses Morphologi Erosi--
struktur elemen("circle",35)
citra_erosi = erosi citra_hasil binerisasi dengan
struktur elemen
tampilkan citra_erosi
```

Hasil operasi morphologi ini dapat dilihat pada gambar 3.7 di bawah ini.



Gambar 3.7 Citra hasil morphologi

6. *Autocrop*

Untuk memisahkan bagian iris dari bulu mata dan kelopak mata. Maka dilakukan *cropping*. Pada langkah ini proses pemotongan citra pada ROI (*Region of Interest*) dilakukan secara otomatis, dimana otomatisasi ini mengacu pada bagian iris mata yang telah dibentuk dalam proses sebelumnya. Algoritma proses *autocrop* adalah

```
panggil citra_rgb
panggil citra_hasil morphologi

[baris kolom] = ukuran citra_hasil morphologi
```

```
 x1=1
 y1=1
 x2=kolom
 y2=baris

 --crop dari kiri ke kanan--
 titik_tengah=1
 while jumlah baris pada 1 kolom citra_hasil morphologi =
 baris
     x1=x1+1
     titik_tengah=titik_tengah+1
 end

 --crop dari bawah ke atas--
 titik_tengah=1
 while jumlah kolom pada 1 baris citra_hasil morphologi =
    kolom
    y1=y1+1
    titik_tengah=titik_tengah+1
 end

 --crop dari kanan ke kiri--
 titik_tengah=kolom
 while jumlah baris pada 1 kolom citra_hasil morphologi =
    baris
    x2=x2-1
    titik_tengah=titik_tengah-1
 end

--crop dari atas ke bawah--
titik_tengah=baris
while jumlah kolom pada 1 baris citra_hasil morphologi =
    kolom
    y2=y2-1
    titik_tengah=titik_tengah-1
end

citra_hasil crop = potong citra_rgb pada
                   koordinat(x1,y1),(x2-x1),(y2-y1)
tampilkan citra_hasil crop
```

Hasil *autocrop* ini dapat dilihat pada gambar 3.8 di bawah ini.

Gambar 3.8 Citra hasil *autocrop*

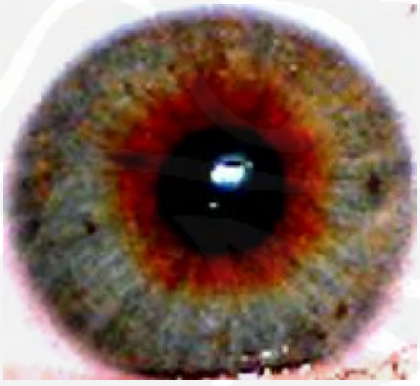
Setelah ini dilakukan lagi proses pengaturan intensitas, seperti pada awal proses segmentasi.

6. Pengaturan Intensitas

Agar garis-garis iris lebih kelihatan dan lebih tampak. Maka dilakukan lagi *intensity adjustment.* Algoritma pengaturan intensitas adalah

```
panggil citra_hasil crop
citra_intensitas2 = pengontrasan citra_hasil crop
tampilkan citra_Intensitas2
```

Tampilan citra hasil *intensity adjustment* ini dapat dilihat pada gambar 3.9



Gambar 3.9 Citra hasil *intensity adjustment*

7. *Grayscale*

Untuk menyederhanakan model citra. Karena citra yang sebelumnya berukuran besar, maka dilakukan proses *grayscale*. Selain itu dapat mengurangi beban kerja komputer sehingga untuk proses selanjutnya waktu komputasinya bisa lebih cepat. Algoritma proses *grayscale* adalah

```
Panggil citra_intensitas2
citra_grayscale = rubah citra_intensitas2 menjadi
grayscale
tampilkan citra_grayscale
```

Citra *grayscale* terlihat pada gambar 3.10 di bawah ini.

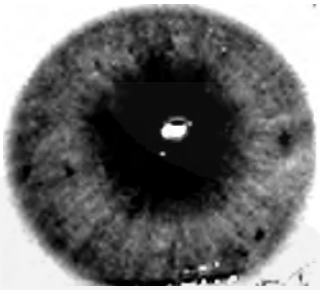
Gambar 3.10 Citra hasil *grayscale*

8. Pengaturan Intensitas

Untuk menghilangkan bagian pupil dari iris. Maka dilakukan lagi *intensity adjustment*. Karena proses ini dilakukan pada citra *grayscale*, maka hasil pengontrasan mengahasilkan citra pada pupil menjadi lebih gelap. Yang akan memudahkan memisahkan garis-garis iris pada saat dilakukan proses deteksi tepi. Algoritma pengaturan intensitas adalah

```
panggil citra_grayscale
citra_intensitas3 = pengontrasan citra_graysclae
tampilkan citra_intensitas3
```

Tampilan citra hasil *intensity adjustment* ini dapat dilihat pada gambar 3.11.



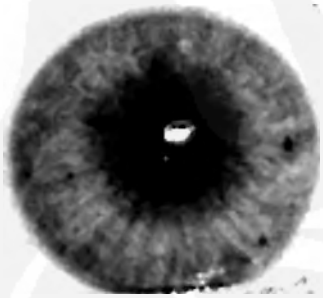
Gambar 3.11 Citra hasil *intensity adjustment*

9. Penapisan citra

Penapisan citra ini bertujuan untuk mengurangi *noise* pada citra. Dalam prosesnya akan menggunakan *filter median*. Algoritma proses penapisan citra adalah

```
Panggil citra_intensitas3
citra_filter median = filter median citra_intensitas3
tampilkan citra_filter median
```

Tampilan citra hasil proses penapisan dapat dilihat pada gambar 3.12 di bawah ini.



Gambar 3.12 Citra hasil proses penapisan

10. Deteksi Tepi *Canny*

Deteksi tepian dilakukan untuk mengekstrak garis tepi dari bentuk iris mata yang dibatasi pupil dibagian dalamnya. Aplikasi deteksi tepian pada citra iris mata akan menghasilkan sebuah citra yang akan membatasi lingkup iris dari bagian lain mata yang tidak diperlukan sehingga mudah untuk dianalisis. Salah satu algoritma deteksi tepi modern adalah deteksi

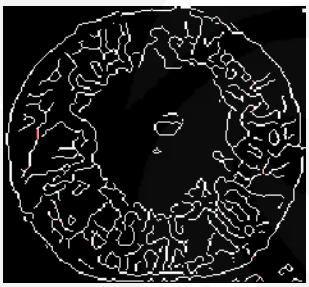
tepi dengan menggunakan metoda *Canny*. Gambar 3.13 di bawah ini adalah diagram blok algoritma *Canny* :

Gambar 3.13 Diagram blok deteksi tepi *Canny*

Algoritma proses deteksi tepi pada citra iris adalah

```
panggil cita_filter median
citra_hasil canny = deteksi tepian citra_filter median
dengan metode canny
tampilkan citra_hasil canny
```

Hasil deteksi tepi *canny* dapat dilihat pada gambar 3.14 di bawah ini.



Gambar 3.14 Citra hasil deteksi tepi *Canny*

7. Polarisasi

Pada langkah ini citra iris yang berbentuk lingkaran akan dibentuk menjadi bentuk polar dengan maksud untuk memudahkan pembacaan garis-garis iris mata dari bagian mata yang lain, seperti pupil dan bulu mata. Agar garis irisnya terlihat lebih jelas, maka dibalikkan warna citra dengan *inverter*. Sehingga garis-garis iris sebagai garis warna hitam (*pixel* 0) dan latarnya berwarna putih (*pixel* 1). Algoritma yang digunakan dalam proses polarisasi adalah.

```
panggil citra_hasil canny
citra_polar = rubah citra_hasil canny ke tipe double
citra_hasil polar = rubah citra_polar ke koordinat polar
```

```
citra_hasil polar2 = inverter citra_hasil polar
tampilkan citra_hasil polar2
```

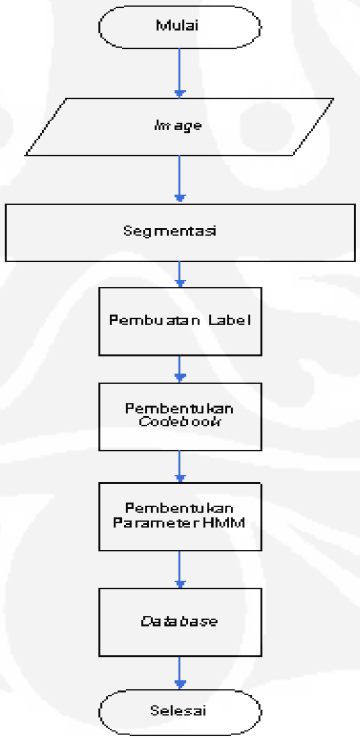
Citra hasil polarisasi ini dapat dilihat pada gambar 3.15 di bawah ini.

Gambar 3.15 Citra hasil polarisasi

Citra hasil dari proses segmentasi ini akan disimpan untuk diproses dalam proses pembentukan basis data (*data base*).

3.3. Pembentukan Basis Data (*Data Base*)

Pada proses pembentukan basis data ini terdapat tiga tahapan, yakni tahap pelabelan, tahap pembuatan *codebook*, dan tahap pembentukan *Hidden Markov Model* (HMM). Seluruh proses yang dilakukan dibuat menggunakan perangkat lunak untuk memecahkan masalah matematisnya.Gambar 3.16 memperlihatkan diagram alir pembentukan basis data.



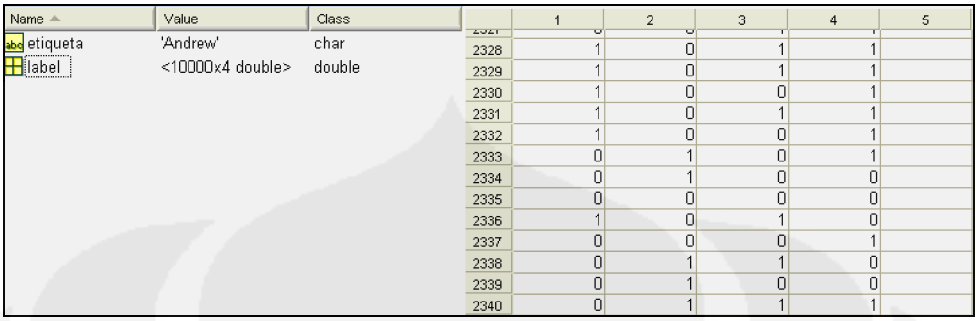
Gambar 3.16 Diagram alir pembentukan basis data

3.3.1 Proses Pelabelan

Pada proses pelabelan ini, masing-masing gambar mata yang akan didaftarkan pada *data base* diberi label sesuai dengan nama mata tersebut. Sebagai contoh, mata ”Andrew1” diberi label1, mata ”Brian1” diberi label 2 dan seterusnya. Nama Label inilah yang nantinya akan menjadi keluaran pada proses pengenalan iris mata.

Pada program pelabelan terdapat tiga masukan, yakni *index* label, jumlah *training*, dan nama label. *Index* label menentukan urutan iris pada keseluruhan *data base* yang ada. Jumlah *training* diisi dengan angka yang diinginkan dan nama label diisi sesuai dengan nama iris yang dimasukkan dalam *data base*. Proses pelabelan ini dilakukan dengan menekan tombol *execute* pada tampilan program proses pelabelan. Tombol *execute* ini kemudian akan memanggil fungsi pelabelan.

Untuk melakukan proses pelabelan dibutuhkan *file-file* gambar dari masing-masing iris yang telah melalui proses segmentasi dan polarisasi dalam format “`*.bmp`” dimana setiap nama iris harus memiliki *file* sebanyak jumlah *training*. Misalnya, akan dibentuk label1 untuk mata ”Andrew1” dengan jumlah *training* 3, maka harus tersedia *file* “`Andrew1.bmp`” sampai *file* “`Andrew3.bmp`”. Pada proses labelisasi, *image-image* dari tiap iris diubah ke dalam bentuk matriks N x M (ukuran matriks N x M akan sama dengan ukuran *image* yang dimasukkan). Keluaran dari pelabelan ini adalah kumpulan matriks-matriks kolom dari tiap mata dengan jumlah kolom sebanyak jumlah *training*. Dapa dilihat pada gambar 3.17 di bawah ini bahwa ukuran matrik dari label “Andrew” adalah 10000 x 4 menunjukan besarnya data sebesar 10000 hasil *reshape MN* x 1 dengan jumlah *training* sebanyak 4 buah.



Name	Value	Class
etiqueta	'Andrew'	char
label	<10000x4 double>	double

	1	2	3	4	5
2328	1	0	1	1	
2329	1	0	1	1	
2330	1	0	0	1	
2331	1	0	1	1	
2332	1	0	0	1	
2333	0	1	0	1	
2334	0	1	0	0	
2335	0	0	0	0	
2336	1	0	1	0	
2337	0	0	0	1	
2338	0	1	1	0	
2339	0	1	0	0	
2340	0	1	1	1	

Gambar 3.17 Tampilan matrik untuk label “Andrew” pada identifikasi iris mata

Dalam proses pelabelan ini akan dibuat 10 buah label yang terdiri dari label 1 sampai dengan 10.

Algoritma dari proses pelabelan adalah:

```
Mulai
   Untuk I=1 sampai 10
   Input nama mata
   Tulis nama mata
   Ubah nama mata menjadi bentuk matriks
   kolom
   Nama label mata[i] = nama mata
   Kembali
Selesai
```

Tampilan program proses pelabelan dapat dilihat pada gambar 3.15 di bawah ini.

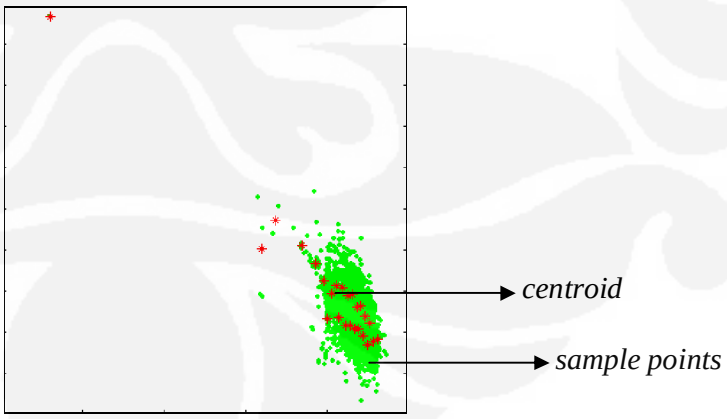
Gambar 3.18 Tampilan Program Pelabelan

3.3.2 Pembuatan *Codebook*

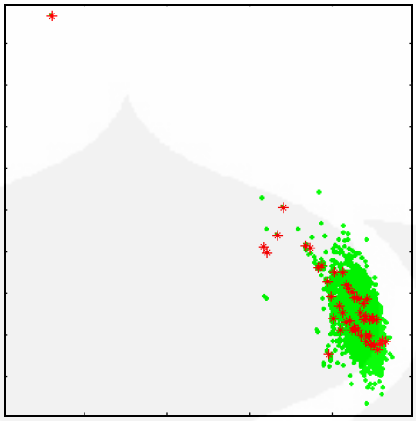
Setelah dilakukan proses pelabelan, maka langkah selanjutnya adalah melakukan proses penggabungan dari semua label yang telah dibuat ke dalam sebuah *file codebook*. Pada pembuatan *codebook* ini juga dilakukan proses *Vector Quantization* yang telah dijelaskan dalam bab sebelumnya.

Pada program ini terdapat empat buah *input*, yakni nama *file*, ukuran *codebook*, jumlah iterasi, dan jumlah label.

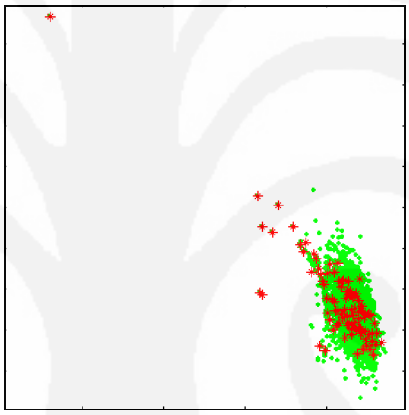
Ukuran *codebook* yang tersedia dalam program ini adalah 32, 64, 128 dan 256. Dimana keempat ukuran *codebook* ini akan dijadikan bahan perbandingan untuk dilihat berapa nilai *codebook* yang paling sesuai pada proses pengenalan citra iris. Jumlah iterasi merupakan banyaknya proses pengulangan yang dilakukan dalam menentukan *centroid* guna mendapatkan *centroid* yang cukup presisi. Semakin besar jumlah iterasinya, maka akan semakin presisi letak *centroid* yang didapat, namun dengan mengambil iterasi yang sangat tinggi proses pembuatan *codebook* akan berjalan sangat lambat, oleh karena itu iterasi yang dilakukan juga tidak perlu teralu besar. Dalam skripsi ini ditentukan *default* untuk besarnya iterasi adalah 10 dengan harapan letak *centroid* yang diperoleh cukup presisi dan waktu proses relatif cepat. Pada gambar 3.19, gambar 3.20, gambar 3.21, dan gambar 3.22 di bawah ini memperlihatkan ukuran *codebook* yang berbeda-beda.



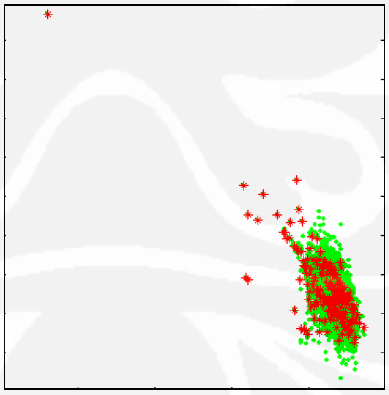
Gambar 3.19 Hasil tampilan VQ *training* map ukuran 32



Gambar 3.20 Hasil tampilan VQ *training* map ukuran 64



Gambar 3.21 Hasil tampilan VQ *training* map ukuran 128

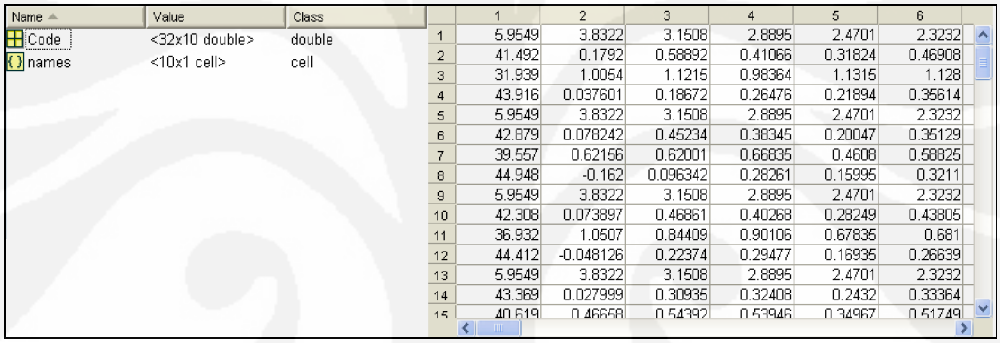


Gambar 3.22 Hasil tampilan VQ *training* map ukuran 256

Berdasarkan keempat gambar di atas terdapat dua buah titik, dimana titik yang berwarna hijau menunjukan interpretasi data hasil proses segmentasi

(*sample points*) data dan titik yang berwarna merah merupakan data hasil *Vector Quantization* (VQ) atau *centroid*, yang akan mewakili dari sekian banyak jumlah data. Terlihat dari gambar semakin besar ukuran *codebook* nya, maka terlihat semakin banyak jumlah *centroid* dan semakin berhimpitan letaknya. Letak *centroid* ini akan menentukan tingkat kepresisian dalam proses identifikasi.

Gambar 3.23 menunjukkan pada *file codebook* "c4_32" berisi 2 buah matrik yang terdiri dari hasil VQ dan matrik yang berisi 10 nama label. Ukuran matrik "*Code*" adalah 32x10 dimana 32 menunjukkan ukuran *codebook* dan 10 merupakan dimensi dari *codebook*. Matrik "*names"* berukuran 10x1 yang menunjukkan nama dari masing-masing label yang terdiri dari 10 buah.



	1	2	3	4	5	6
1	5.9549	3.8322	3.1508	2.8895	2.4701	2.3232
2	41.492	0.1792	0.58892	0.41066	0.31824	0.46908
3	31.939	1.0054	1.1215	0.98364	1.1315	1.128
4	43.916	0.037601	0.18672	0.26476	0.21894	0.35614
5	5.9549	3.8322	3.1508	2.8895	2.4701	2.3232
6	42.879	0.078242	0.45234	0.38345	0.20047	0.35129
7	39.557	0.62156	0.62001	0.66835	0.4608	0.58825
8	44.948	-0.162	0.096342	0.28261	0.15995	0.3211
9	5.9549	3.8322	3.1508	2.8895	2.4701	2.3232
10	42.308	0.073897	0.46861	0.40268	0.28249	0.43805
11	36.932	1.0507	0.84409	0.90106	0.67835	0.681
12	44.412	-0.048126	0.22374	0.29477	0.16935	0.26639
13	5.9549	3.8322	3.1508	2.8895	2.4701	2.3232
14	43.369	0.027999	0.30935	0.32408	0.2432	0.33364
15	40.619	0.46668	0.54392	0.53946	0.34967	0.51749

Gambar 3.23 Tampilan hasil matrik untuk *file* codebook "c4_32"

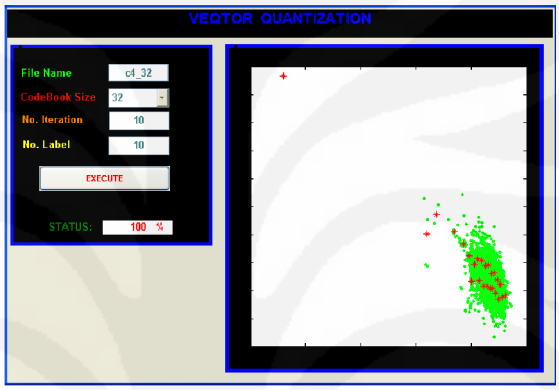
Pada pembuatan *codebook*, jumlah label dimasukan sebanyak 10 label karena terdapat 10 sample iris mata yang dimasukkan dalam *data base*. Program *codebook* ini dilengkapi dengan fasilitas status proses berupa persentase dari jalannya program guna memudahkan user untuk mengetahui sejauh mana proses pembuatan *codebook* sudah berjalan.

Algoritma dari proses pembuatan *codebook* adalah sebagai berikut:

```
Mulai
Tentukan besar nilai N
Untuk I = 1 sampai M
      Hitung FFT untuk setiap sample[i]
      Sample point[i] = nilai FFT
kembali
Tentukan cluster
```

```
Untuk j =1 sampai cluster
      Hitung centroid
      Simpan centroid[j] berdasarkan urutan
      labelnya
Kembali
Selesai
```

Tampilan program pembuatan *codebook* dapat dilihat pada gambar 3.24 di bawah ini.

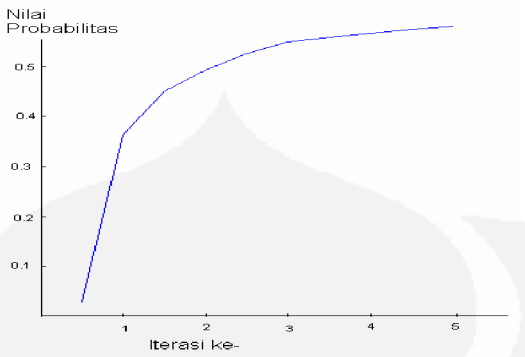


Gambar 3.24 Tampilan Program Pembuatan *codebook*

3.3.3 Pembentukan Parameter HMM

Pada program untuk pembentukan parameter HMM ini terdapat tiga input, yakni *file* hmm, *file codebook*, dan jumlah iterasi. *File* hmm dimasukkan sesuai dengan keinginan dan hasil dari proses parameter hmm akan disimpan ke dalam *folder* untuk pengujian. *File codebook* dimasukkan sesuai dengan nama *file codebook* yang telah diisi sebelumnya pada proses pembuatan *codebook*.

Jumlah iterasi merupakan banyaknya proses pengulangan yang dilakukan dalam mengolah parameter HMM guna mendapatkan probabilitas yang paling baik. Semakin besar jumlah iterasinya, maka akan semakin baik probabilitas yang didapat, namun terdapat suatu nilai iterasi dimana nilai probabilitas yang didapat akan konstan atau relatif sangat kecil perubahannya. Seperti terlihat pada gambar 3.25 merupakan tampilan grafik probabilitas dalam pembuatan model HMM.



Gambar 3.25 Probabilitas 1 label huruf dengan 10 iterasi

Dengan demikian tidak perlu diambil nilai iterasi yang sangat tinggi karena dengan mengambil iterasi yang sangat tinggi proses akan berjalan sangat lambat. Sama halnya pada proses pembuatan *codebook*, dalam pembentukan hmm juga ditentukan *default* untuk besarnya iterasi sebesar 10 dengan harapan probabilitas yang didapat bernilai tinggi dan waktu proses relatif cepat. Seperti pada gambar 3.26 di bawah ini menunjukkan hasil pembentukkan *data base* HMM pada identifikasi iris mata dengan ukuran 32.

Name	Value	Class
A1	<7x7 double>	double
A10	<7x7 double>	double
A2	<7x7 double>	double
A3	<7x7 double>	double
A4	<7x7 double>	double
A5	<7x7 double>	double
A6	<7x7 double>	double
A7	<7x7 double>	double
A8	<7x7 double>	double
A9	<7x7 double>	double
B1	<7x32 double>	double
B10	<7x32 double>	double
B2	<7x32 double>	double
B3	<7x32 double>	double
B4	<7x32 double>	double
B5	<7x32 double>	double
B6	<7x32 double>	double
B7	<7x32 double>	double
B8	<7x32 double>	double
B9	<7x32 double>	double
M	32	double
N	7	double
available	0	double
p1	[1;0;0;0;0;0;0]	double
p10	[1;0;0;0;0;0;0]	double
p2	[1;0;0;0;0;0;0]	double
p3	[1;0;0;0;0;0;0]	double
p4	[1;0;0;0;0;0;0]	double
p5	[1;0;0;0;0;0;0]	double
p6	[1;0;0;0;0;0;0]	double
p7	[1;0;0;0;0;0;0]	double
p8	[1;0;0;0;0;0;0]	double
p9	[1;0;0;0;0;0;0]	double

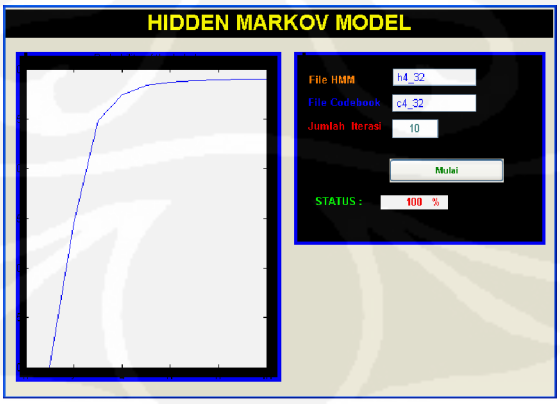
Gambar 3.26 Hasil pembentukkan *data base* HMM “h4_32”

Dimana *data base* tersebut terdiri dari 5 buah variabel utama. Variabel A dan B yang merupakan matriks transisi dan probabilitas matrik dari observasi, variabel M menunjukkan jumlah simbol observasi yang berbeda dari masing-masing *state*, yang tergantung dari ukuran *codebook*, variabel N menunjukkan jumlah *state*, dan variabel *p* yang merupakan vektor dari *initial probability* dengan jumlah 10 buah sesuai dengan jumlah label. Dalam program ini jumlah *state* yang digunakan sebanyak 7 buah.

Algoritma dari proses pembentukan HMM yang bertujuan untuk memperoleh nilai parameter dari setiap label berdasarkan *file codebook* yang telah dibuat adalah sebagai berikut :

```
Mulai
Ambil file codebook dari codebook yang telah dibuat
HMM iris = file codebook iris
Tentukan letak centroid dari tiap iris
Inisialisasi matriks A, B dan π dengan nilai   acak;
Pelatihan dengan memasukkan semua data image
sampai nilai matriks tidak berubah
Hitung probabilitas observasi HMM untuk setiap
image
Simpan hasil dalam folder yang sama
Selesai
```

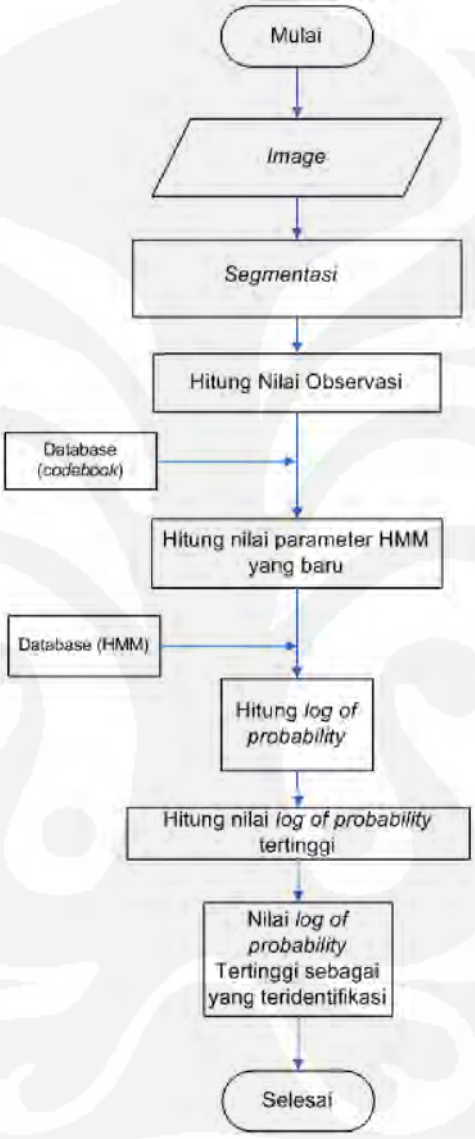
Tampilan program pembuatan *hmm* dapat dilihat pada gambar 3.27 di bawah ini.



Gambar 3.27 Tampilan Program Pembuatan HMM

3.4 Proses Pengenalan Iris

Secara keseluruhan diagram alir proses pengenalan iris terlihat pada Gambar 3.28.



Gambar 3.28 Diagram alir proses pengenalan

Jadi setelah semua data disimpan dalam *data base* yang terdiri *file* iris mata, *file* hasil labelisasi, *file* codebook, dan *file* HMM. Maka proses pengenalan dapat dilakukan dengan cara sebagai berikut:

1. Memasukan *file* input *mata.jpg* kemudian program akan membacanya dengan fungsi *baca_citra RGB.*
2. Hasil *baca_citra RGB* akan dilakukan proses segmentasi
3. Hasil dari proses segmentasi diubah ke dalam domain frekuensi dengan menggunakan FFT. Spektrum frekuensi yang merupakan hasil dari FFT tersebut akan membentuk nilai vektor *rea*l dan *imajiner* yang akan dipetakan dalam *codebook* dalam bentuk *sample points*.
4. Selanjutnya beberapa *sample point* yang terdekat dikuantisasikan ke satu titik vektor yang dinamakan *centroid*. Letak dari *centroid* ini kemudian dicocokkan dengan letak *centroid* yang ada pada *codeword* dalam basis data
5. Hasil dari perbandingan ini adalah urutan kode observasi. Kemudian matriks dari nilai observasi yang didapat akan dicocokkan dengan matriks-matriks dari parameter-parameter HMM dalam basis data
6. Hitung besar *Log of Probability* (LoP) untuk semua iris mata yang akan dikenali.
7. Tampilkan hasil yang memiliki *Log of Probability* (LoP) yang tertinggi sebagai hasil dari proses identifikasi.

Algoritma dari proses pengenalan dengan HMM dapat ditulis sebagai berikut:

```
Mulai
Baca Citra mata.jpg
Dilakukan proses segmentasi iris
Hitung FFT
iris = matriks codeword iris
Cari centroid codeword_iris di basis data
Tentukan nilai observasi
Tentukan parameter HMM iris
untuk h = 1 sampai  jml_label
Baca data parameter HMM untuk jenis iris
lainnya dari basis data
Hitung log of probability (LoP) untuk semua label
iris
LoP[jml_label] = LoP
```

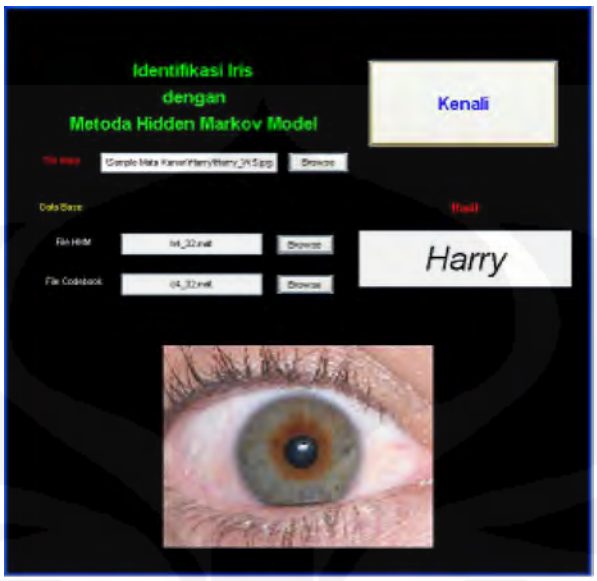
```
Cari LoP[jumlah_label] = tertinggi
Tentukan nama label untuk LoP tertinggi
Selesai
```

Tampilan program pengenalan iris mata ini dapat dilihat pada gambar 3.29 di bawah ini.

Gambar 3.29 Tampilan program identifikasi iris mata

Program pengenalan iris diatas memiliki 3 buah variabel utama yang terdiri dari *file* input mata, *file* HMM, dan *file codebook*. Untuk variabel *file* HMM dan *file codebook* diambil dari *data base* yang telah dibuat sebelumnya. Inputan keduanya harus memiliki ukuran yang sama, misalnya untuk *file* HMM "h4_32" maka variabel *file codebook* nya diisi dengan "c4_32". Pada variabel *file* input mata diisi dengan *file .jpg* yang berupa citra mata digital bagian mata sebelah kanan dari berbagai macam ras dan etnik yang diambil dari Department of Computer Science of the University of Beira Interior [8].

Ukuran citra mata digital ini sebesar 400x300 *pixel*, dengan ukuran ini menurut Penulis cukup. Sehingga pada waktu komputasinya tidak akan terlalu berat. Proses identifikasi akan dimulai setelah tombol "Kenali" ditekan. Dan hasilnya seperti pada gambar 3.30 dibawah ini.



Gambar 3.30 Tampilan program identifikasi iris mata setelah eksekusi

Hasil dari proses identifikasi ini berupa nilai probabilitas pada setiap nilai citra iris mata. Jumlah nilai probabilitas yang dihasilkan sesuai dengan banyaknya label dalam *data base.* Probabilitas paling tinggi akan diambil sebagai identifikasi. Seperti terlihat pada gambar 3.31 dimana variable *P* yang berisi kumpulan probabilitas, memiliki nilai tertinggi -280.5737 dengan nama label "Harry" yang berarti citra mata itu adalah matanya Harry.

```
nam =

Harry


P =

  Columns 1 through 5

 -316.6188 -380.6096 -350.1901 -284.8029 -532.0274

  Columns 6 through 10

 -471.1264 -334.5347 -280.5737 -293.4298 -282.9471
```

Gambar 3.31 Nilai probabilitas pada variabel P

BAB IV

UJI COBA DAN ANALISA SISTEM

4.1 Sistem Identifikasi

Sistem ini memanfaatkan metode *Hidden Markov Model* untuk mengenali identitas mata setiap individu. Data mata yang digunakan sebagai masukan sistem ini berupa citra digital mata manusia yang diambil dari Department of Computer Science of the University of Beira Interior [8]. Dalam uji coba ini digunakan 10 citra mata bagian mata sebelah kanan yang masing- masing mewakili individu yang berbeda. Dari 10 individu ini masing-masing disediakan 5 citra mata yang berbeda. Jadi citra mata keseluruhan tersedia 50 buah yang bisa dilihat pada lampiran.

Uji coba dilakukan dengan memvariasikan ukuran *codebook* dan jumlah *training* masing-masing citra mata, yang kemudian akan dihitung tingkat keberhasilannya. Tingkat akurasi dihitung dengan membandingkan jumlah hasil identifikasi yang benar dengan jumlah total inputan yang ada. Sebagai contoh setiap individu memiliki 5 citra mata yang berbeda, akan dilakukan *training* sebanyak 3 buah dan *codebook* 32. Maka 3 citra mata diantaranya akan dilakukan sebagai data latih (dimasukkan ke *data base*) dan 2 citra mata lainnya untuk data uji. Jadi jumlah keseluruhan dari 10 citra mata untuk semua individu yang berbeda sebanyak 30 buah untuk *data base* dan 20 buah citra mata untuk uji coba. Pada saat ujicoba ternyata hanya dapat mengidentifikasi 12 inputan. Maka tingkat akurasinya dihitung dengan cara (17/20) * 100 % = 85 %. Waktu yang diperlukan untuk satu kali proses pengenalan dapat bervariasi tergantung spesifikasi komputer yang digunakan. Dalam percobaan ini diperlukan waktu ± 4 detik mengunakan *Notebook Computer* dengan spesifikasi umum sebagai berikut:

Sistem Operasi : Windows XP SP 2
Processor : Mobile Intel(R) Pentium(R) 4 - M CPU 1.80 GHz
RAM : 512 MB

4.1.1 Uji Coba dengan Variasi Jumlah *Training* dan Ukuran Codebook

Uji coba yang dilakukan dengan variasi jumlah *training* sebanyak 2 kali, yang terdiri dari jumlah *training* 3, dimana mata1, mata 2, mata 3 sebagai data latih dan mata 4, mata 5 sebagai data uji. Sedangkan jumlah *training* 4, mata1, mata 2, mata 3, mata 4 sebagai data latih dan mata 5 sebagai data uji. Uji coba jumlah *training* akan membandingkan pengaruh ukuran *codebook* terhadap tingkat akurasi proses identifikasi masing-masing tahap. Untuk perbandingan secara keseluruhan pada identifikasi iris mata hasil uji coba dengan jumlah *training* 3 ditunjukkan oleh Tabel IV.1, hasil uji coba dengan jumlah *training* 4 ditunjukkan oleh Tabel IV.2.

Tabel IV.1 Hasil uji coba identifikasi iris mata dengan jumlah *training* 3

Nama	***Training* 3 (mata 1, mata 2 , mata 3)**				
	Citra	***Codebook* 32**	***Codebook* 64**	***Codebook* 128**	***Codebook* 256**
Andrew	**mata 4**	OK	OK	-	-
	mata 5	OK	OK	-	-
Brian	**mata 4**	OK	OK	-	-
	mata 5	OK	OK	-	-
Charlie	**mata 4**	OK	OK	-	-
	mata 5	OK	OK	-	-
David	**mata 4**	OK	OK	-	-
	mata 5	OK	OK	-	-
Edward	**mata 4**	OK	OK	-	-
	mata 5	OK	OK	-	-
Frank	**mata 4**	OK	OK	-	-
	mata 5	OK	OK	-	-
Gilbert	**mata 4**	OK	OK	-	-
	mata 5	OK	OK	-	-
Harry	**mata 4**	OK	OK	-	-
	mata 5	GAGAL	OK	-	-
Inka	**mata 4**	OK	OK	-	-
	mata 5	OK	OK	-	-
Jason	**mata 4**	OK	OK	-	-
	mata 5	OK	OK	-	-
Total Error		1	0	-	-
Akurasi (%)		95 %	100 %	-	-

Tabel IV.2 Hasil uji coba identifikasi iris mata dengan jumlah *training* 4

Nama	*Training* 4 (mata 1, mata 2 , mata 3, mata 4)				
	Citra	*Codebook* 32	*Codebook* 64	*Codebook* 128	*Codebook* 256
Andrew	mata 5	OK	OK	-	-
Brian	mata 5	OK	OK	-	-
Charlie	mata 5	OK	OK	-	-
David	mata 5	OK	OK	-	-
Edward	mata 5	OK	OK	-	-
Frank	mata 5	OK	OK	-	-
Gilbert	mata 5	OK	OK	-	-
Harry	mata 5	OK	OK	-	-
Inka	mata 5	OK	OK	-	-
Jason	mata 5	OK	OK	-	
Total Error		0	0	-	-
Akurasi (%)		100 %	100 %	-	-

4.2 Analisa Hasil Uji Coba

Berdasarkan data dari hasil uji coba yang telah dilakukan sebelumnya, maka analisa terhadap perangkat lunak identifikasi iris ini dapat terdiri dari 2 aspek, yaitu analisa berdasarkan variasi ukuran *codebook* dan jumlah *training*.

4.2.1 Analisa Berdasarkan Variasi Ukuran Codebook

Pengaruh dari perubahan ukuran *codebook* dapat dilihat dari hasil uji coba yang telah dilakukan seperti pada tabel IV.1 dan IV.2. Berdasarkan dari kedua tabel tersebut dapat dilihat bahwa semakin besar ukuran dari *codebook* akan meningkatkan akurasi. Peningkatan besarnya persen akurasi ini disebabkan karena ukuran *codebook* yang lebih besar membuat jumlah *codeword* (*centroid*) semakin banyak. Apabila ukuran dari *codebook* diperbesar maka luas daerah *cluster* akan semakin kecil. Hal ini menyebabkan semakin banyaknya daerah *cluster* yang terbentuk, sehingga jumlah *codeword* yang mewakili setiap daerah tersebut juga semakin bertambah. Jumlah *codeword* yang banyak akan dapat

mempresentasikan data sesungguhnya. Akan tetapi dengan ukuran *codebook* yang besar akan memperlambat proses komputasi dan identifikasi, serta membutuhkan memori yang besar pula. Penentuan dari ukuran *codebook* yang tepat dapat dilakukan dengan cara melihat hasil uji coba. Apabila dengan ukuran *codebook* tertentu tingkat akurasinya sudah baik maka penambahan dari ukuran *codebook* tidak diperlukan lagi. Seperti yang terjadi pada hasil tabel IV.1 dan tabel IV.2 diatas, dimana perangkat lunak identifikasi iris telah dapat mengenal 100 % pada ukuran *codebook* 64.

4.2.2 Analisa Berdasarkan Variasi Jumlah *Training*

Pengaruh dari jumlah *training* dapat membandingkan dari kedua tabel diatas. Terlihat pada tabel IV.1 yang merupakan identifikasi iris dengan jumlah *training* 3, perangkat lunak mampu mengenal dengan tingkat akurasi 95% pada ukuran *codebook* 32. Tetapi dengan jumlah *training* 4 seperti ditunjukkan pada tabel IV.2, pada ukuran *codebook* 32 perangkat lunak sudah mampu mengenal dengan tingkat akurasi 100%. Hal ini menunjukkan bahwa selain ukuran *codebook*, tingkat akurasi dipengaruhi juga oleh jumlah *training*. *Training* yang semakin banyak akan melatih perangkat lunak untuk mengenali inputan citra mata baru dengan kondisi yang beragam. Dikarenakan jumlah *data base* yang dimiliki perangkat lunak akan semakin banyak dengan kondisi data yang berbeda-beda pula.

BAB V

KESIMPULAN

Berdasarkan hasil ujicoba dan analisa dari sistem yang telah dibuat dapat disimpulkan bahwa :

1. Hasil identifikasi iris mata dengan menggunakan metode *Hidden Markov Model* ini diperoleh tingkat akurasi sebesar 95 % pada jumlah *training* 3 ukuran *codebook* 32 dan 100 % pada ukuran *codebook* 64. Sedangkan pada jumlah *training* 4 untuk ukuran *codebook* 32 dan 64 diperoleh tingkat akurasi sebesar 100 %. Hal ini menandakan bahwa sistem identifikasi sudah berjalan dengan baik.
2. Ukuran *codebook* yang semakin besar akan meningkatkan kemampuan dalam mengidentifikasi citra iris mata.
3. Penambahan jumlah *training* juga dapat meningkatkan tingkat akurasi. Dikarenakan jumlah *data base* yang dimiliki perangkat lunak akan semakin banyak dengan kondisi data yang berbeda-beda pula.

DAFTAR ACUAN

[1] R.C. Gonzalez, R.E. Woods, S.L. Eddins, *Digital Image Processing*, 2nd Ed., (New Jersey : Prentice-Hall, 2004)

[2] Fahmi, "Perancangan Algoritma Pengolahan Citra Mata Menjadi Citra Polar Iris Sebagai Bentuk Antara Sistem Biometrik." Karya Ilmiah, Fakultas Teknik Universitas Sumatra Utara, Medan, 2007

[3] Ahmad Mujadid Amin. "Pengenalan Suara Manusia Menggunakan *Hidden Markov Model*". Skripsi. Program Sarjana Fakultas Teknik Universitas Indonesia. 2007.

[4] Michael Hasudungan Siahaan. "Identifikasi Pola *Schooling* Ikan Dengan *Analisis Echogram Fish Finder* Menggunakan *Hidden Markov Model*". Skripsi. Program Sarjana Fakultas Teknik Universitas Indonesia. 2007.

[5] Anil K. Jain, Arun Ross and Salil Prabhakar, "An Introduction to Biometric Recognition", (January, 2004)

[6] L. Carlos Junquerira, Jose Carneiro, Robert O. Kelley, "*Basic Histology* 8th edition".ISBN: 0-8385-0567-8.

[7] Ferdinand Simanjuntak. "*Design and Implementation of Barcode Encoder and Decoder Based on Digital ImageProcessing"*. IT TELKOM 2009

[8] Data Sample Iris diambil dari Department of Computer Science of the University of Beira Interior, http://www.di.ubi.pt/~hugomcp/investigacao.htm

[9] Image Processing Toolbox User Guide Version 7.3, (The Math Works, Inc, 2006)

DAFTAR PUSTAKA

Diponegoro, A.D., ”Penentuan jenis ikan dengan menggunakan *Hidden Markov Model* dari penditeksian fase penerimaan sinyal akustik.” Disertasi, Program Studi Teknologi Kelautan Program Pasca Sarjana Insitut Pertanian Bogor, 2004.

Emmanuel C, Ifeachor dan Barri W, Jervis. *Digital Signal Processing*. *A Practical Approach, Second Edition*. Prentice Hall.

Gonzalez, Rafael, Woods, Richard, “ *Digital Image Processing*”, Addison Wesley Publishing co., USA, 1992

Gonzalez, Rafael, Woods, Richard, “ *Digital Image Processing Using MATLAB*”, Prentince Hall , New Jersey, 2004

Hasudungan, Michael, ”Identifikasi species ikan dengan analisis echogram *fish finder* menggunakan *Hidden Markov Model*.” Skripsi, Departemen Teknik Elektro, Fakultas Teknik, Universitas Indonesia, 2006.

M Ch. Wijaya, Agus Prijono, *Pengolahan Citra Dijital Menggunakan MATLAB* (Bandung : Informatika, 2007)

Sugiarto, Ferry, “Pengenalan Plat Nomor Mobil Dengan Skeletonisasi Dan *Hidden Markov Model*.”Skripsi, Departemen Teknik Elektro, Fakultas Teknik, Universitas Indonesia, 2007

LAMPIRAN

Data mata yang diambil dari Department of Computer Science of the University of Beira Interior.

Citra mata "**Andrew**"

1

2

3

4

5

Citra mata "**Brian**"

1

2

3

4

5

Citra mata "**Charlie**"

 **1**	 **2**	 **3**
 **4**	 **5**	

Citra mata "**David**"

 **1**	 **2**	 **3**
 **4**	 **5**	

Citra mata "**Edward**"

 **1**	 **2**	 **3**
 **4**	 **5**	

Citra mata "**Frank**"

 **1**	 **2**	 **3**
 **4**	 **5**	

Citra mata "**Gilbert**"

 1	 2	 3
 4	 5	

Citra mata "**Harry"**

 1	 2	 3
 4	 5	

Citra mata “**Inka”**

 **1**	 **2**	 **3**
 **4**	 **5**	

Citra mata “**Jason”**

 **1**	 **2**	 **3**
 **4**	 **5**	