



**TUGAS AKHIR - KI141502**

# **IMPLEMENTASI PENGENALAN IRIS MATA MENGGUNAKAN METODE SUPPORT VECTOR MACHINES DAN HAMMING DISTANCE**

**AFDHAL BASITH ANUGRAH**  
**NRP 5112100153**

**Dosen Pembimbing I**  
**Dr.Eng. Nanik Suciati, S.Kom, M.Kom**

**Dosen Pembimbing II**  
**Dr.Eng. Chastine Fatichah, S.Kom, M.Kom**

**JURUSAN TEKNIK INFORMATIKA**  
**Fakultas Teknologi Informasi**  
**Institut Teknologi Sepuluh Nopember**  
**Surabaya 2016**





**TUGAS AKHIR - KI141502**

# **IMPLEMENTASI PENGENALAN IRIS MATA MENGUNAKAN METODE *SUPPORT VECTOR MACHINES* DAN *HAMMING DISTANCE***

**AFDHAL BASITH ANUGRAH  
NRP 5112100153**

**Dosen Pembimbing I  
Dr.Eng. Nanik Suciati, S.Kom, M.Kom**

**Dosen Pembimbing II  
Dr.Eng. Chastine Fatichah, S.Kom, M.Kom**

**JURUSAN TEKNIK INFORMATIKA  
Fakultas Teknologi Informasi  
Institut Teknologi Sepuluh Nopember  
Surabaya 2016**

***[Halaman ini sengaja dikosongkan]***



**FINAL PROJECT - KI141502**

# **IMPLEMENTATION OF IRIS RECOGNITION USING SUPPORT VECTOR MACHINES AND HAMMING DISTANCE**

**AFDHAL BASITH ANUGRAH  
NRP 5112100153**

**Supervisor I  
Dr.Eng. Nanik Suciati, S.Kom, M.Kom**

**Supervisor II  
Dr.Eng. Chastine Fatichah, S.Kom, M.Kom**

**DEPARTMENT OF INFORMATICS  
FACULTY OF INFORMATION TECHNOLOGY  
INSTITUT TEKNOLOGI SEPULUH NOPEMBER  
SURABAYA 2016**

***[Halaman ini sengaja dikosongkan]***

## **LEMBAR PENGESAHAN**

### **IMPLEMENTASI PENGENALAN IRIS MATA MENGUNAKAN METODE SUPPORT VECTOR MACHINES DAN HAMMING DISTANCE**

#### **TUGAS AKHIR**

Diajukan Untuk Memenuhi Salah Satu Syarat  
Memperoleh Gelar Sarjana Komputer  
pada  
Bidang Studi Komputasi Cerdas dan Visualisasi  
Program Studi S-1 Jurusan Teknik Informatika  
Fakultas Teknologi Informasi  
Institut Teknologi Sepuluh Nopember

Oleh  
**AFDHAL BASITH ANUGRAH**  
**NRP : 5112 100 153**

Disetujui oleh Dosen Pembimbing Tugas Akhir:

1. Dr.Eng. Nanik Suciati, S.Kom, M.Kom. ....  
(NIP 197104281994122001) (Pembimbing 1)
2. Dr.Eng. Chastine Fatichah, S.Kom, M.Kom .....  
(NIP 197512202001122002) (Pembimbing 2)

**SURABAYA**  
**JUNI, 2016**

*[Halaman ini sengaja dikosongkan]*



# **IMPLEMENTASI PENGENALAN IRIS MATA MENGUNAKAN METODE SUPPORT VECTOR MACHINES DAN HAMMING DISTANCE**

**Nama Mahasiswa** : Afdhal Basith Anugrah  
**NRP** : 5112100153  
**Jurusan** : Teknik Informatika FTIF-ITS  
**Dosen Pembimbing 1** : Dr.Eng. Nanik Suciati, S.Kom,  
M.Kom.  
**Dosen Pembimbing 2** : Dr.Eng. Chastine Fatichah, S.Kom,  
M.Kom.

## ***Abstrak***

*Pengenalan iris mata adalah teknik pengenalan pola yang digunakan dalam aplikasi biometrik. Teknik biometrik ini menghasilkan pola acak yang unik secara statistik. Dengan kata lain, tekstur iris mata adalah bagian yang unik pada masing-masing orang.*

*Pada Tugas Akhir ini diimplementasikan metode klasifikasi Support Vector Machine (SVM) dan metode Hamming distance untuk pengenalan iris mata dengan Wavelet Haar dan Log-Gabor Filter sebagai ekstraksi fiturnya. Sebelum dilakukan ekstraksi fitur, terlebih dahulu dilakukan praproses untuk identifikasi dan normalisasi bagian iris mata.*

*Hasil uji coba pada dataset mata CASIA versi 1.0 dengan menggunakan klasifikasi SVM dan Hamming distance terbukti cukup efektif dalam proses pengenalan iris. Metode klasifikasi SVM menghasilkan akurasi terbaik 92,28% dengan ekstraksi fitur Wavelet Haar dekomposisi tingkat 2. Metode Hamming distance menghasilkan akurasi terbaik 91,67 % dengan menggunakan Log-Gabor Filter sebagai ekstraksi fiturnya.*

***Kata Kunci: Hamming Distance, Log-Gabor Filter,  
Pengenalan iris, Support Vector Machine, Wavelet Haar.***

*[Halaman ini sengaja dikosongkan]*

# **IMPLEMENTATION OF IRIS RECOGNITION USING SUPPORT VECTOR MACHINES AND HAMMING DISTANCE**

**Student's Name** : Afdhal Basith Anugrah  
**Student's ID** : 5112100153  
**Department** : Informatics Department FTIF-ITS  
**First Advisor** : Dr.Eng. Nanik Suciati, S.Kom,  
M.Kom.  
**Second Advisor** : Dr.Eng. Chastine Fatichah, S.Kom,  
M.Kom.

## ***Abstract***

*Iris recognition is a pattern recognition technique used in biometric applications. This biometric technique generates a unique random pattern statistically. In other words, the texture of the iris is the part that is unique to each person.*

*This final project implements classification methods Support Vector Machine (SVM) and Hamming Distance method for iris recognition with Haar Wavelet and Log-Gabor Filter as their feature extraction. Before feature extraction, first performed preprocessing for identification and normalization of iris.*

*The test result with CASIA eye dataset using SVM classification and Hamming distance proved to be quite effective in the process of iris recognition. SVM method showed the best accuracy of 92,28% with Haar Wavelet feature extraction at decomposition level 2. Hamming distance method showed the best accuracy of 91,67% with Log-Gabor Filter as its feature extraction.*

***Keywords: Haar Wavelet, Hamming Distance, Iris Recognition, Log-Gabor Filter, Support Vector Machine.***

***[Halaman ini sengaja dikosongkan]***

## KATA PENGANTAR

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

Puji syukur kehadiran Allah *subhanallahu wa ta'ala* karena berkat rahmat dan karunia-NYA penulis dapat menyelesaikan Tugas Akhir yang berjudul

### **IMPLEMENTASI PENGENALAN IRIS MATA MENGUNAKAN METODE SUPPORT VECTOR MACHINES DAN HAMMING DISTANCE**

Tugas Akhir ini merupakan salah satu syarat untuk memperoleh gelar Sarjana Komputer di Jurusan Teknik Informatika, Fakultas Teknologi Informasi, Institut Teknologi Sepuluh Nopember Surabaya.

Penulis ingin menyampaikan terima kasih yang sebesar-besarnya atas dukungan dan semangat yang diberikan dan membantu penulis baik secara langsung ataupun tidak dalam menyelesaikan Tugas Akhir ini. Penulis ingin mengucapkan terima kasih kepada

1. Ibu, bapak, dan kakak-kakak tercinta yang telah membantu doa, moral dan material selama penulis belajar di Teknik Informatika ITS.
2. Bapak Dr. Darlis Herumurti, S.Kom., M.Kom., selaku ketua jurusan, Bapak Radityo Anggoro, S.Kom., M.Sc. selaku Koordinator Tugas Akhir, dan segenap dosen Teknik Informatika yang telah memberikan ilmunya selama penulis berkuliah di Teknik Informatika ITS.
3. Ibu Dr.Eng. Nanik Suciati, S.Kom, M.Kom selaku Dosen Pembimbing I Tugas Akhir yang telah memberikan bimbingan dan dukungan selama penulis menyelesaikan Tugas Akhir sampai selesai.
4. Ibu Dr.Eng. Chastine Fatichah, S.Kom, M.Kom selaku pembimbing II Tugas Akhir yang telah memberikan

bimbingan dan dukungan selama penulis menyelesaikan Tugas Akhir sampai selesai.

5. Para penulis artikel ilmiah yang digunakan pada Tugas akhir ini, pendiri Google, pendiri Mathworks, serta berbagai referensi lain, yang telah membantu mempermudah penyelesaian Tugas Akhir ini.
6. Bu Eva di Ruang Baca dan seluruh Staf dan karyawan Teknik Informatika yang telah memberikan bantuan selama penulis kuliah di Teknik Informatika.
7. Kawan-kawan angkatan 2012 yang menemani, membantu, dan memotivasi selama penulis menjalankan seluruh tugas perkuliahan hingga penulis sampai pada tahap pengerjaan Tugas Akhir ini.
8. Serta semua pihak yang telah turut membantu penulis dalam menyelesaikan Tugas Akhir ini.

Penulis menyadari bahwa Tugas Akhir ini masih memiliki banyak kekurangan. Sehingga dengan kerendahan hati, penulis mengharapkan kritik dan saran dari pembaca untuk perbaikan ke depan. Semoga Tugas Akhir ini dapat memberikan Manfaat yang sebesar besarnya.

Surabaya, Juni 2016

Penulis

## DAFTAR ISI

LEMBAR PENGESAHAN.....	v
<i>Abstrak</i> .....	vii
<i>Abstract</i> .....	ix
KATA PENGANTAR .....	xi
DAFTAR ISI .....	xiii
DAFTAR GAMBAR .....	xvii
DAFTAR TABEL .....	xix
DAFTAR KODE SUMBER .....	xxi
BAB I PENDAHULUAN .....	1
1.1 Latar Belakang .....	1
1.2 Rumusan Masalah .....	3
1.3 Batasan Masalah .....	3
1.4 Tujuan .....	3
1.5 Manfaat .....	3
1.6 Metodologi .....	4
1.7 Sistematika Penulisan .....	5
BAB II TINJAUAN PUSTAKA.....	7
2.1 Anatomi Mata .....	7
2.2 Sistem Pengenalan Iris .....	8
2.3 Representasi Citra Digital .....	9
2.4 Deteksi Tepi <i>Canny</i> .....	10
2.5 Transformasi Hough .....	13
2.6 Transformasi Polar.....	15
2.7 Haar Wavelet.....	16
2.8 Log-Gabor Filter .....	18
2.9 Support Vector Machine .....	19
2.10 Hamming Distance.....	24
2.11 Bahasa Pemrograman Matlab 8.3.....	25
BAB III DESAIN PERANGKAT LUNAK.....	27
3.1 Perancangan Data.....	27
3.1.1 Data Masukan .....	27
3.1.2 Data Proses.....	29
3.1.3 Data Keluaran .....	29

3.2 Desain Umum Sistem .....	30
3.3 Akuisisi Citra Mata .....	32
3.4 Perancangan Praproses .....	33
3.4.1 Deteksi Tepi .....	34
3.4.2 Deteksi Batas Pupil dan Iris .....	35
3.4.3 Pemisahan Bulu dan Kelopak Mata .....	37
3.5 Normalisasi Iris .....	39
3.6 Perancangan Ekstraksi Fitur .....	40
3.6.1 Fitur Koefisien Wavelet Haar .....	40
3.6.2 Fitur Log-Gabor Filter .....	41
3.7 Perancangan Klasifikasi .....	43
3.7.1 Klasifikasi Support Vector Machines Fitur Wavelet .....	43
3.7.2 Klasifikasi Suport Vector Machines Fitur Log- Gabor .....	44
3.7.3 Klasifikasi Hamming Distance Fitur Log-Gabor	45
3.7.4 Klasifikasi Hamming Distance Fitur Wavelet .....	46
3.8 Perancangan Antarmuka .....	47
<b>BAB IV IMPLEMENTASI .....</b>	<b>49</b>
4.1 Lingkungan Implementasi .....	49
4.2 Implementasi Akuisisi Citra Mata .....	50
4.3 Implementasi Preprocessing .....	50
4.3.1 Implementasi Deteksi Tepi .....	51
4.3.2 Implementasi Deteksi Batas Pupil dan Iris .....	51
4.3.3 Implementasi Pemisahan Bulu dan Kelopak Mata .....	53
4.4 Implementasi Normalisasi Iris .....	55
4.5 Implementasi Ekstraksi Fitur .....	56
4.5.1 Implementasi Fitur Koefisien Wavelet Haar .....	57
4.5.2 Implementasi Fitur Log-Gabor Filter .....	57
4.6 Implementasi Klasifikasi .....	60
4.6.1 Implementasi Support Vector Machine Fitur Wavelet .....	60
4.6.2 Implementasi Suport Vector Machin Fitur Log- Gabor .....	62



4.6.3 Implementasi Hamming Distance Fitur Log-Gabor .....	63
4.6.4 Implementasi Hamming Distance Fitur Wavelet .....	64
4.7 Implementasi Antarmuka .....	67
<b>BAB V UJI COBA DAN EVALUASI .....</b>	<b>69</b>
5.1 Lingkungan Uji Coba .....	69
5.2 Data Uji Coba .....	69
5.3 Preprocessing .....	70
5.4 Ekstraksi Fitur .....	71
5.5 Skenario Uji Coba .....	71
5.5.1 Skenario Uji Coba 1 .....	73
5.5.2 Skenario Uji Coba 2 .....	74
5.5.3 Skenario Uji Coba 3 .....	75
5.5.4 Skenario Uji Coba 4 .....	75
5.5.5 Skenario Uji Coba 5 .....	76
5.5.6 Skenario Uji Coba 6 .....	77
5.5.7 Skenario Uji Coba 7 .....	77
5.5.8 Skenario Uji Coba 8 .....	78
5.6 Analisis Hasil Uji Coba .....	79
<b>BAB VI KESIMPULAN DAN SARAN .....</b>	<b>81</b>
6.1 Kesimpulan .....	81
6.2 Saran .....	82
<b>DAFTAR PUSTAKA .....</b>	<b>83</b>
<b>LAMPIRAN A TABEL UJI COBA .....</b>	<b>85</b>
<b>LAMPIRAN B SNAPSHOT PERANGKAT LUNAK .....</b>	<b>109</b>
<b>BIODATA PENULIS .....</b>	<b>113</b>

*[Halaman ini sengaja dikosongkan]*

## DAFTAR GAMBAR

Gambar 2.1 Anatomi Mata.....	7
Gambar 2.2 Sistem Pengenalan Iris secara Umum .....	9
Gambar 2.3 Representasi Citra Digital .....	10
Gambar 2.4 Ilustrasi 8-konektivitas .....	13
Gambar 2.5 Ilustrasi Transformasi Hough Lingkaran.....	14
Gambar 2.6 <i>Daugman's rubber sheet model</i> .....	16
Gambar 2.7 Ilustrasi Dekomposisi <i>Wavelet</i> .....	17
Gambar 2.8 Filter pada <i>Wavelet</i> .....	18
Gambar 2.9 Ilustrasi <i>Support Vector Machine</i> .....	20
Gambar 2.10 Ilustrasi Pembuatan Model <i>SVM</i> Multi Kelas ..	22
Gambar 3.1 Citra Mata CASIA.....	28
Gambar 3.2 Diagram Alir Rancangan Sistem.....	31
Gambar 3.3 Diagram Alir Akuisisi Citra Mata .....	32
Gambar 3.4 Diagram Alir Praproses .....	33
Gambar 3.5 Diagram Alir Proses Deteksi Tepi.....	34
Gambar 3.6 Diagram Alir Proses Deteksi Batas Pupil dan Iris .....	36
Gambar 3.7 Diagram Alir Pemisahan Bulu dan Kelopak Mata .....	38
Gambar 3.8 Diagram Alir Normalisasi Iris.....	39
Gambar 3.9 <i>Pseudocode</i> Dekomposisi <i>Wavelet</i> .....	41
Gambar 3.10 Diagram Alir Pembuatan Fitur Biner .....	42
Gambar 3.11 Diagram Alir Klasifikasi <i>SVM</i> menggunakan Fitur <i>Wavelet Haar</i> .....	44
Gambar 3.12 Diagram Alir Klasifikasi <i>SVM</i> menggunakan Fitur <i>Log-Gabor</i> .....	45
Gambar 3.13 Diagram Alir <i>Hamming distance</i> menggunakan Fitur <i>Log-Gabor</i> .....	46
Gambar 3.14 Diagram Alir <i>Hamming distance</i> menggunakan Fitur <i>Wavelet</i> .....	47
Gambar 3.15 Desain Antarmuka.....	48
Gambar 4.1 Implementasi Antarmuka .....	67

Gambar 5.1 Sampel hasil praproses pada citra mata  
001\_1\_1.bmp.....70

## DAFTAR TABEL

Tabel 2.1 Operasi XOR.....	24
Tabel 3.1 Data Proses.....	29
Tabel 4.1 Lingkungan Perancangan Perangkat Lunak.....	49
Tabel 5.1 Lingkungan Uji Coba Perangkat Lunak.....	69
Tabel 5.3 Sampel fitur pada citra mata 001_1_1.bmp .....	71
Tabel 5.3 Hasil Ujicoba 1.....	73
Tabel 5.4 Hasil Ujicoba 2.....	74
Tabel 5.5 Hasil Ujicoba 3.....	75
Tabel 5.6 Hasil Ujicoba 4.....	78
Tabel 5.7 Hasil Ujicoba 5.....	78
Tabel 5.8 Hasil Ujicoba 6.....	76
Tabel 5.9 Hasil Ujicoba 7.....	76
Tabel 5.10 Hasil Ujicoba 8.....	77
Tabel 5.11 Perbandingan Akurasi Metode Klasifikasi.....	80
Tabel A.1 Hasil Uji Coba Data Uji Citra Iris Fitur <i>Wavelet Haar</i> Dekomposisi level 2 dan Klasifikasi SVM dengan $C = 1$ dan kernel RBF .....	85
Tabel A.2 Hasil Uji Coba Data Uji Citra Iris Fitur <i>Wavelet Haar</i> Dekomposisi level 2 dan Klasifikasi SVM di $C = 30$ dan kernel RBF .....	89
Tabel A.3 Hasil Uji Coba Data Uji Citra Iris Fitur <i>Wavelet Haar</i> Dekomposisi level 2 dan Klasifikasi SVM pada $C = 30$ dan kernel RBF .....	93
Tabel A.4 Hasil Uji Coba Data Uji Citra Iris Fitur <i>Log-Gabor</i> <i>Filter</i> standar deviasi 0.2 dan Pencocokan <i>Hamming distance</i> .....	97
Tabel A.5 Hasil Uji Coba Data Uji Citra Iris Fitur <i>Wavelet Haar</i> Dekomposisi level 1 dan Pencocokan <i>Hamming distance</i> ...	101
Tabel A.6 Hasil Uji Coba Data Uji Citra Iris Fitur <i>Log-Gabor</i> <i>Filter</i> standar deviasi 0.2 dan Klasifikasi SVM pada $C = 1$ dan kernel RBF .....	105

***[Halaman ini sengaja dikosongkan]***

## DAFTAR KODE SUMBER

Kode Sumber 4.1 Implementasi Tahap Akuisisi Citra Mata..	50
Kode Sumber 4.2 Implementasi Deteksi Tepi.....	51
Kode Sumber 4.3 Implementasi Transformasi Hough pada Iris .....	52
Kode Sumber 4.4 Implementasi Transformasi Hough pada Pupil .....	53
Kode Sumber 4.5 Implementasi Penghilangan Kelopak Mata .....	54
Kode Sumber 4.6 Implementasi <i>Thresholding</i> Bulu Mata.....	54
Kode Sumber 4.7 Implementasi Transformasi Polar .....	56
Kode Sumber 4.8 Implementasi Ekstraksi Fitur <i>Wavelet Haar</i> .....	57
Kode Sumber 4.9 Implementasi Pembuatan Fitur Biner.....	58
Kode Sumber 4.10 Implementasi Pembuatan Filter Log-Gabor .....	59
Kode Sumber 4.11 Implementasi Konvulsi Citra Iris dengan Filter Log-Gabor .....	59
Kode Sumber 4.12 Implementasi SVM Fitur Wavelet .....	61
Kode Sumber 4.13 Implementasi SVM Fitur Log-Gabor.....	62
Kode Sumber 4.14 Implementasi Hamming Distance Fitur Log- Gabor.....	64
Kode Sumber 4.15 Implementasi Proses Normalisasi Data...	65
Kode Sumber 4.16 Implementasi Hamming Distance Fitur Wavelet .....	66

***[Halaman ini sengaja dikosongkan]***



# **BAB I**

## **PENDAHULUAN**

Pada bab ini dibahas mengenai latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat, metodologi, dan sistematika laporan tugas akhir. Diharapkan dari penjelasan dalam bab ini gambaran Tugas Akhir secara umum dapat dipahami.

### **1.1 Latar Belakang**

Teknologi Biometrik adalah metode untuk mengenali seseorang melalui ciri-ciri fisik, karakter, dan kebiasaan. Biasanya ciri-ciri fisik khas yang dijadikan indikator untuk mengenali seseorang adalah wajah, sidik jari, telapak tangan, retina, atau iris mata. Dapat pula mengenali seseorang melalui ciri-ciri lainnya seperti cara berjalan, tanda tangan, atau suara. Dengan menggunakan macam-macam ciri yang unik tersebut, dapat dijadikan solusi untuk mengidentifikasi dan mengenali seseorang.

Iris atau selaput pelangi adalah daerah berbentuk gelang pada mata yang dibatasi oleh pupil dan sklera (bagian putih dari mata). Dalam penerapannya pada teknik biometrik, iris memiliki pola yang sangat unik, berbeda pada tiap individu dan pola itu akan tetap stabil. Sebelum kelahiran, degenerasi terjadi sehingga menghasilkan pembukaan pupil dan acak, serta pola-pola unik dari iris. Walaupun genetik serupa, seseorang yang memiliki struktur iris yang unik dan berbeda, dapat memungkinkan untuk digunakan untuk tujuan pengenalan. Iris mata juga mudah untuk dicitrakan pada jarak yang sesuai dari subjek dengan penggunaan alat yang ada, misalnya kamera. Atas dasar inilah iris mata dapat dijadikan dasar bagi pengenalan biometrik [1].

Metode pengklasifikasian yang dapat digunakan adalah *Support Vector Machine* (SVM). Pengklasifikasian dengan metode ini terdiri dari dua modul utama, yaitu pembuatan

model dan klasifikasi data. Model SVM yang dibuat dimaksudkan untuk memisahkan satu kategori dengan satu kategori yang lain dengan menggunakan sebuah bidang *hyperlane*. Selanjutnya klasifikasi akan dilakukan dengan menggunakan model yang dibuat, membuatnya mampu memberikan kinerja yang lebih tinggi dalam hal akurasi dari algoritma klasifikasi lainnya [2].

Salah satu metode lainnya dalam digunakan dalam pengklasifikasian adalah *Hamming distance*. Metode ini merepresentasikan pola iris kedalam bentuk biner, karena lebih mudah untuk menentukan perbedaan antara dua kode biner daripada bilangan biasa. Metode ini pada dasarnya adalah fungsi eksklusif OR (XOR) antara dua pola bit.

Tugas Akhir ini mengimplementasikan pengklasifikasi tekstur iris mata dengan menggunakan masing-masing dua metode klasifikasi tersebut. Sebelum dilakukan klasifikasi, terlebih dahulu dilakukan praproses untuk identifikasi bagian iris pada citra mata. Selanjutnya dilakukan normalisasi iris pada *region of interest*, serta ekstraksi fitur untuk menghasilkan vektor fitur yang digunakan untuk proses klasifikasi.

Beberapa metode telah diaplikasikan dengan menggunakan iris sebagai teknologi biometrik. Pada proses identifikasi iris sampai normalisasi relatif sama. Metode ekstraksi fitur yang sudah pernah diaplikasikan adalah *Gabor filter*, *Bayesian*, dan yang lainnya untuk menghasilkan vektor fitur tekstur iris. Pada tugas akhir ini menggunakan metode *log-Gabor filter* dan *wavelet Haar* sebagai metode ekstraksi fitur untuk menghasilkan vektor fitur tekstur iris. Metode *log-Gabor filter* menutupi kelemahan yang terdapat pada *Gabor filter* yang membuat informasi tekstur iris berkurang banyak [3]. Pada metode *Bayesian* memakai parameter deformasi untuk menghasilkan vektor fitur tekstur iris [4]. Hal-hal tersebutlah yang membedakan antara pengimplementasian yang sudah dilakukan sebelumnya dengan apa yang diimplementasikan pada tugas akhir ini.

## 1.2 Rumusan Masalah

Rumusan masalah yang diangkat dalam Tugas Akhir ini adalah sebagai berikut:

- a) Bagaimana pengklasifikasian dengan *Support Vector Machine* pada pengenalan iris mata ?
- b) Bagaimana pengklasifikasian dengan *Hamming distance* pada pengenalan iris mata ?
- c) Bagaimana melakukan identifikasi bagian iris pada citra mata ?
- d) Manakah representasi klasifikasi yang lebih baik antara *Support Vector Machine* dan *Hamming distance* ?

## 1.3 Batasan Masalah

Permasalahan yang dibahas dalam tugas akhir memiliki beberapa batasan, yakni sebagai berikut.

1. Database yang digunakan adalah data citra mata dari basis data *Chinese Academy of Sciences Institute of Automation* (CASIA).
2. Implementasi menggunakan perangkat lunak MATLAB 8.3 dan *Image Processing Toolbox* didalamnya.
3. Data yang digunakan untuk data latih dan data uji berupa citra mata dari database CASIA.

## 1.4 Tujuan

Tujuan dari pembuatan tugas akhir ini adalah membandingkan dua metode klasifikasi dalam melakukan pengenalan iris mata. Selain itu, membuat sebuah implementasi perangkat lunak yang dapat melakukan pengenalan iris mata.

## 1.5 Manfaat

Pengerjaan tugas akhir ini dilakukan dengan harapan bisa memberikan kontribusi pada teknologi biometrik yang dapat melakukan pengenalan melalui iris mata.

## 1.6 Metodologi

Metodologi yang dipakai pada pengerjaan Tugas Akhir ini adalah sebagai berikut:

1. Penyusunan Proposal Tugas Akhir  
Tahap awal yang dilakukan dalam pengerjaan Tugas Akhir ini adalah penyusunan proposal Tugas Akhir. Di dalam proposal diajukan suatu gagasan untuk melakukan implementasi pengklasifikasian pada iris mata pada metode *Support Vector Machine* dan *Hamming distance*.
2. Studi Literatur  
Pada tahap ini dilakukan pencarian, pengumpulan, pembelajaran dan pemahaman informasi dan literatur yang diperlukan untuk pembuatan implementasi pada tahap praproses citra mata, ekstraksi fitur serta metode-metode klasifikasi. Dasar informasi yang diperlukan pada pembuatan implementasi ini diantaranya mengenai citra secara umum dan algoritma tahap praproses. Serta metode pengklasifikasi pada iris mata. Literatur yang digunakan meliputi: buku referensi, jurnal, dan dokumentasi internet. Selain itu juga dipelajari database citra mata CASIA yang bisa didownload dari <http://biometrics.idealtest.org>.
3. Implementasi perangkat lunak  
Pada tahap ini dilakukan implementasi perangkat lunak sesuai dengan rancangan perangkat lunak yang dibuat pada pengerjaan Tugas Akhir.
4. Uji coba dan Evaluasi  
Pada tahap ini dilakukan uji coba terhadap perangkat lunak yang telah dibuat untuk mengetahui kemampuan algoritma yang dipakai, mengamati kinerja sistem. Ujicoba ini bertujuan untuk mengetahui performa algoritma ekstraksi fitur *wavelet Haar* dan *log-Gabor filter* pada metode klasifikasi *Support Vector Machine* dan *Hamming distance*.

## 5. Penyusunan Laporan Tugas Akhir

Pada tahap ini dilakukan penyusunan laporan pengerjaan Tugas Akhir yang berisi dasar teori, dokumentasi dari perangkat lunak, dan hasil yang diperoleh selama pengerjaan Tugas Akhir.

### 1.7 Sistematika Penulisan

Buku Tugas Akhir ini bertujuan untuk mendapatkan gambaran dari pengerjaan Tugas Akhir ini. Secara garis besar, buku Tugas Akhir terdiri atas beberapa bagian seperti berikut ini:

#### **Bab I    Pendahuluan**

Bab ini berisi latar belakang masalah, tujuan dan manfaat pembuatan Tugas Akhir, permasalahan, batasan masalah, metodologi yang digunakan, dan sistematika penyusunan Tugas Akhir.

#### **Bab II   Dasar Teori**

Bab ini membahas beberapa teori penunjang yang berhubungan dengan pokok pembahasan dan mendasari pembuatan Tugas Akhir ini.

#### **Bab III  Perancangan Perangkat Lunak**

Bab ini berisi tentang desain sistem yang disajikan dalam bentuk *flowchart*. *Flowchart-flowchart* tersebut memperlihatkan diagram alir bagaimana proses-proses berlangsung selama pengimplementasian.

#### **Bab IV  Implementasi**

Bab ini berisi implementasi dari perancangan yang telah dibuat sebelumnya. Penjelasan berupa code yang digunakan untuk proses implementasi.

**Bab V Uji Coba dan Evaluasi**

Bab ini menjelaskan kemampuan perangkat lunak dengan melakukan pengujian kebenaran dan pengujian kinerja dari sistem yang telah dibuat.

**Bab VI Kesimpulan dan Saran**

Bab ini berisi kesimpulan dari hasil pengujian yang dilakukan. Bab ini membahas saran-saran untuk pengembangan sistem lebih lanjut.

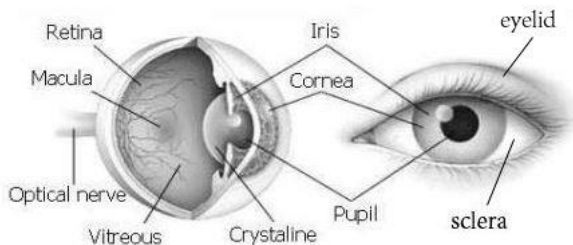
## BAB II TINJAUAN PUSTAKA

Pada bab ini dibahas mengenai teori-teori yang menjadi dasar dari pembuatan Tugas Akhir.

### 2.1 Anatomi Mata

Bagian organ mata yang akan diteliti dalam tugas akhir ini adalah iris atau selaput pelangi. Bagian-bagian mata seperti yang tampak pada Gambar 2.1 dijelaskan sebagai berikut [5]:

1. Retina adalah bagian yang berfungsi untuk menangkap bayangan benda
2. Iris adalah bagian yang mempunyai pigmen untuk memberi warna pada mata. Bagian inilah yang akan diteliti pada Tugas Akhir ini
3. Kornea berfungsi untuk meneruskan cahaya yang masuk ke mata
4. Pupil merupakan bagian mata yang berbentuk lingkaran yang mengatur banyaknya cahaya yang masuk ke mata. Bagian ini akan diteliti juga untuk mendapatkan bagian iris
5. *Sclera* adalah bagian pelindung mata yang bewarna putih pada mata bagian luar
6. *Optical Nerve* adalah saraf mata yang meneruskan atau membelokkan sinar menuju ke otak.



Gambar 2.1 Anatomi Mata

## 2.2 Sistem Pengenalan Iris

Pengenalan iris adalah suatu proses untuk mengenal seseorang dengan menganalisa pola acak dari iris. Dimana pengenalan iris bertujuan untuk mengenali suatu objek dengan cara mengekstraksi informasi yang terdapat dalam suatu citra tersebut.

Dalam pengaplikasiannya, sudah banyak metode yang digunakan. Pada proses identifikasi iris sampai normalisasi relatif sama. Metode dengan menggunakan *wavelet* [1], *Laplacian of Gaussian Filter* [6], dan berbagai macam metode lainnya. Metode-metode tersebut digunakan pada proses ekstraksi fitur. Sedangkan pada proses pengenalan, ada pula klasifikasi *Support Vector Machine* [7], *Hamming distance* [8], dan metode-metode yang lainnya.

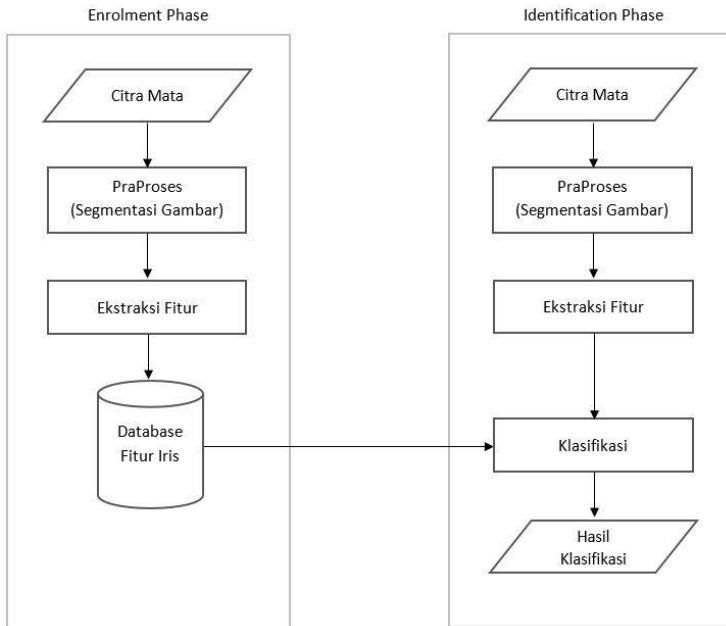
Secara garis besar, proses-proses tersebut dikelompokkan pada empat proses utama yaitu:

1. Akuisisi citra
2. Praproses citra (identifikasi objek iris dan normalisasi)
3. Ekstraksi fitur iris
4. Pencocokan iris.

Akuisisi citra adalah proses mendapatkan citra mata yang akan dilakukan pengenalan. Citra mata yang digunakan pada tugas akhir ini yaitu citra mata CASIA [9]. Proses selanjutnya adalah praproses citra atau melakukan pengolahan awal pada citra mata yang bertujuan untuk mengambil karakteristik tekstur iris mata dan melakukan normalisasi iris untuk mengatasi kondisi citra mata yang bervariasi pada setiap orang. Tekstur iris yang sudah terambil akan diekstraksi ciri atau fitur iris. Setiap fitur akan disimpan didalam sebuah database yang nantinya akan dibandingkan dengan fitur masukan dalam proses pencocokan iris. Proses pencocokan fitur masukan dengan seluruh fitur yang terdapat didalam database menghasilkan bobot atau tingkat kemiripan. Nilai tingkat kemiripan yang paling tinggi adalah citra yang dikenali



paling mirip dengan citra masukan. Gambar 2.2 menjelaskan pengenalan iris secara umum.



Gambar 2.2 Sistem Pengenalan Iris secara Umum

## 2.3 Representasi Citra Digital

Citra dapat didefinisikan sebagai  $f(x,y)$ , yaitu fungsi dua dimensi, dengan  $x$  dan  $y$  menyatakan koordinat spasial dan nilai  $f$  pada sembarang titik  $(x,y)$  disebut dengan intensitas atau tingkat keabuan dari citra pada koordinat tersebut. Jika nilai  $x,y$  dan nilai  $f$  adalah *finite*, bernilai diskrit atau terbatas, maka dapat disebut bahwa citra tersebut merupakan citra digital [10].

Sebuah citra digital  $f(x,y)$  dapat direpresentasikan sebagai sebuah matriks yang indeks baris dan kolomnya mengidentifikasi sebuah titik pada citra dan nilai dari

elemen matriks yang bersangkutan merupakan tingkat warna pada titik tersebut, seperti yang ditunjukkan pada Gambar 2.3. Elemen tersebut disebut elemen citra, elemen gambar, pixels, atau pels. *Picture elements* atau pixel dapat didefinisikan sebagai elemen terkecil dari sebuah citra digital yang menentukan resolusi citra tersebut.

$$f(x,y) \approx \begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,M) \\ f(1,0) & f(1,1) & \cdots & f(1,M) \\ \vdots & \vdots & \vdots & \vdots \\ f(N-1,0) & f(N-1,1) & \cdots & f(N-1,M-1) \end{bmatrix}$$

Gambar 2.3 Representasi Citra Digital

## 2.4 Deteksi Tepi *Canny*

Deteksi Tepi adalah langkah awal dalam pemrosesan citra digital untuk mendapatkan informasi pada citra tersebut. Mendapatkan jenis informasi berbeda dari citra dengan jenis yang berbeda memerlukan metode yang berbeda pula. Tepi adalah perubahan nilai intensitas tingkat keabuan yang besar yang memperlihatkan rincian pada gambar. Tujuan deteksi tepi adalah untuk mengekstraksi fitur penting pada suatu citra, misalnya garis, lingkaran, lengkungan, ujung. Salah satu metode deteksi tepi yang terkenal adalah deteksi tepi *Canny*.

Deteksi tepi *Canny* sangat cocok digunakan untuk pengolahan awal citra digital. Deteksi tepi ini sangat banyak digunakan oleh peneliti karena memiliki keuntungan sebagai berikut:

1. Tepi yang dihasilkan akurat
2. Banyak tepi yang dihasilkan

Pada citra iris mata, deteksi tepi ini dilakukan dengan untuk mendeteksi citra iris mata dengan langkah-langkah berikut [11].

- a. Mengurangi noise dengan menggunakan filter Gaussian sebesar 13x13, standar deviasi iris dengan sclera, dan iris dengan pupil sebesar 2 [12].
- b. Menentukan intensitas gradien dari citra yaitu penentuan arah gradien ke suatu arah tertentu. Arah tersebut yaitu vertikal, horizontal, diagonal kiri&kanan. Perhitungan intensitas gradien dilakukan menggunakan persamaan (2.1).

$$G = \sqrt{G_x^2 + G_y^2} \quad (2.1)$$

dimana  $G_x$  = gradien ke arah sumbu x  
 $G_y$  = gradien ke arah sumbu y  
 $G$  = intensitas gradien

selanjutnya mencari arah gradien dengan persamaan (2.2)

$$\theta = \arctan \frac{G_y}{G_x} \quad (2.2)$$

dimana  $\theta$  adalah arah gradien

Kemudian dilakukan pengelompokkan para arah gradien yang diperoleh dengan kondisi berikut.

- Jika arah tepi yang berkisar antara  $0^\circ$ - $22.5^\circ$  serta  $157.5^\circ$ - $180^\circ$  , arah gradien diubah menjadi  $0^\circ$  (horizontal).
- Jika arah tepi yang berkisar antara  $22.5^\circ$ - $67.5^\circ$ , arah gradien diubah menjadi  $45^\circ$  (diagonal kanan).
- Jika arah tepi yang berkisar antara  $67.5^\circ$ - $112.5^\circ$ , arah gradien diubah menjadi  $90^\circ$  (vertikal).
- Jika arah tepi yang berkisar antara  $112.5^\circ$ - $157.5^\circ$ , arah gradien diubah menjadi  $135^\circ$  (diagonal kiri).

Karena bagian irirs berada diantara tepi sclera dan tepi iris, serta tepi iris dan tepi pupil. Maka dari itu dilakukan pendeteksian secara berturut-turut dengan menggunakan arah gradien vertikal dan horizontal.

- c. Mengatur Gamma atau pengontraskan terhadap arah yang telah terdeteksi. Pengontraskan dilakukan menggunakan persamaan (2.3)

$$M = \frac{I_{max} - I_{min}}{I_{max} + I_{min}} \quad (2.3)$$

dimana  $I_{max}$  = nilai citra intensitas maksimum

$I_{min}$  = nilai citra intensitas minimum

$M$  = nilai citra intensitas rata-rata

Kemudian mendapatkan hasil akhir pengontraskan dengan persamaan (2.4)

$$C = M^{1/g} \quad (2.4)$$

dimana  $C$  = citra gamma

$g$  = nilai gamma

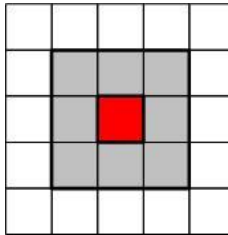
$M$  = nilai citra intensitas rata-rata

Citra akan semakin gelap jika nilai gamma berada diantara 0-1, namun semakin terang jika nilai gamma diatas 1. Pengontraskan dilakukan untuk mempermudah dalam penelusuran tepi citra iris mata.

- d. Proses *non-maximum suppression* atau memperkecil garis tepi yang muncul sehingga menghasilkan garis tepian yang lebih ramping. Proses ini menjadikan nilai piksel yang dianggap tidak layak menjadi sebuah tepi dengan nilai nol dari citra tepi pengontraskan.

- e. Langkah terakhir adalah proses *hystheresis*, proses ini menerapkan dua buah bilangan threshold, yaitu *high threshold* (T1) dan *low threshold* (T2). Kondisi penelusuran tepi pada proses ini adalah sebagai berikut:
1. Semua piksel diatas T1 ditandai sebagai tepi.
  2. Semua piksel yang berdekatan dengan titik yang telah ditandai sebagai tepi dan mempunyai nilai diatas T2 juga ditandai sebagai tepi.

Menggunakan 8-konektivitas piksel untuk mengetahui titik yang berdekatan dengan titik yang telah ditandai sebagai tepi. Dalam Gambar 2.4 memperlihatkan ilustrasi 8-konektivitas.



Gambar 2.4 Ilustrasi 8-konektivitas

Warna merah pada gambar diatas menunjukan titik yang telah ditandai sebagai tepi jika memenuhi syarat satu, sedangkan warnal abu merupakan koordinat piksel yang akan ditandai sebagai tepi jika memenuhi syarat dua.

## 2.5 Transformasi Hough

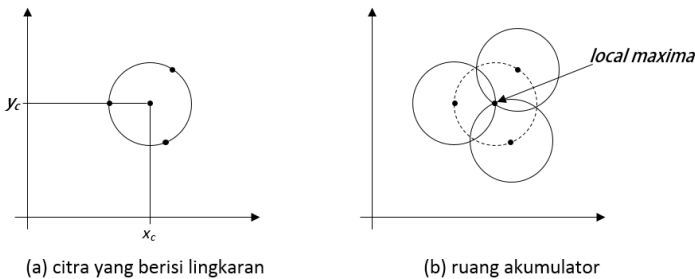
Transformasi Hough adalah suatu metode untuk mencari bentuk fitur tertentu seperti garis, lingkaran, atau bentuk sederhana lain dalam suatu citra. Pada umumnya, transformasi Hough diterapkan pada banyak permasalahan pada visi komputer karena kebanyakan citra mempunyai batas tepi yang membentuk sebuah garis. Implementasi transformasi

Hough menjelaskan sebuah pemetaan pada ruang akumulator. Dalam hal ini, transformasi Hough membahas pada pencarian bentuk lingkaran atau biasa disebut Transformasi Hough Lingkaran (*Circular Hough Transform*). Pencarian lingkaran ini menggunakan persamaan (2.5) untuk mencari titik pusat dan jari-jari pada lingkaran iris dan lingkaran pupil [13].

$$(x_c - a)^2 + (y_c - b)^2 = r^2 \quad (2.5)$$

dimana  $(x_c, y_c)$  = koordinat titik-titik lingkaran  
 $(a, b)$  = koordinat pusat lingkaran  
 $r$  = jari-jari lingkaran

Pada proses pendeteksian iris, transformasi Hough lingkaran digunakan untuk mendeteksi area iris dengan cara mencari radius dan titik pusat lingkaran pupil dan iris. Tahap awal dari proses ini adalah mendeteksi tepi citra. Tujuan dari deteksi tepi adalah untuk menurunkan jumlah titik dalam pencarian ruang bagi objek. Ketika titik tepi sudah ditemukan, transformasi Hough akan bekerja pada titik tersebut [11].



Gambar 2.5 Ilustrasi Transformasi Hough Lingkaran

Transformasi Hough lingkaran membentuk lingkaran sepanjang titik yang ditemukan dengan jari-jari sebesar  $r$ . Kemudian mencari *voting* untuk mencari titik yang sering dilewati (*local maxima*) dari lingkaran yang telah dibentuk dan

titik tersebut diasumsikan sebagai titik pusat lingkaran. Pada Gambar 2.5 adalah ilustrasi Transformasi Hough lingkaran. Hasil akhirnya adalah sebuah citra iris melalui transformasi Hough.

## 2.6 Transformasi Polar

Setelah daerah iris teridentifikasi dari citra mata, langkah berikutnya adalah melakukan transformasi citra iris sehingga memiliki ukuran yang konstan untuk memungkinkan pencocokan. Pada dasarnya, daerah iris akan menciut dan meregang disebabkan oleh pelebaran pupil karena faktor pencahayaan yang beragam. Faktor yang mempengaruhi pelebaran pupil yang lain juga seperti jarak citra, rotasi, dan derajat kemiringan. Proses normalisasi akan menghasilkan daerah iris yang memiliki ukuran yang konstan. Sehingga jika dilakukan pencocokan antara dua daerah iris akan lebih mudah [12].

Metode yang digunakan untuk melakukan normalisasi pada citra iris adalah mengubah citra iris pada bidang Cartesian ke dalam bidang polar dengan metode *Daugman's rubber sheet model* seperti yang terlihat pada Gambar 2.6. Metode ini memetakan daerah iris yang tersegmentasi pada koordinat  $(x, y)$  ke koordinat polar  $(r, \theta)$ . Nilai  $r$  berada pada interval  $[0, 1]$  dan nilai  $\theta$  mempunyai interval derajat sudut  $[0, 2\pi]$ . Pemetaan daerah iris tersebut dimodelkan dengan persamaan 2.6 untuk mencari citra normalisasi, dan didalamnya memakai persamaan (2.7) dan persamaan (2.8) untuk mencari batas tepi pupil dan batas tepi iris.

$$I(x(r, \theta), y(r, \theta)) \rightarrow I(r, \theta) \quad (2.6)$$

dengan

$$x(r, \theta) = (1 - r)x_p(\theta) + rx_i(\theta) \quad (2.7)$$

$$y(r, \theta) = (1 - r)y_p(\theta) + ry_i(\theta) \quad (2.8)$$

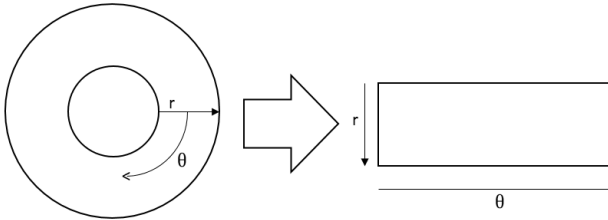
dimana  $I(x, y)$  adalah citra iris

$(x, y)$  adalah titik pada koordinat Cartesian

$(r, \theta)$  adalah koordinat pada polar normalisasi

$x_p, y_p$  adalah koordinat batas tepi pupil

$x_i, y_i$  adalah koordinat batas tepi.



Gambar 2.6 *Daugman's rubber sheet model*

*Rubber sheet model* pada Gambar 2.6 mengatasi pelebaran yang dilakukan pupil dan ketidakkonsistenan ukuran pada citra iris untuk menghasilkan representasi yang ternormalisasi dengan ukuran yang tetap. Dengan begitu, saat melakukan pencocokan dengan berbagai macam iris bisa lebih mudah dan fleksibel dengan batas luar tepi iris dan titik tengah pupil sebagai titik acuan.

## 2.7 Haar Wavelet

*Wavelet* adalah sebuah fungsi matematika yang mampu melakukan dekomposisi terhadap sebuah fungsi. Transformasi *wavelet* akan mengkonversi signal ke dalam sederetan gelombang singkat. Transformasi *wavelet* merupakan sebuah fungsi konversi yang dapat membagi data ke dalam komponen frekuensi yang berbeda-beda dan menganalisis komponen tersebut dengan resolusi tertentu sesuai dengan skalanya. *Wavelet* telah banyak diaplikasikan dalam berbagai bidang, salah satunya adalah pengolahan citra [14]. Pada citra,



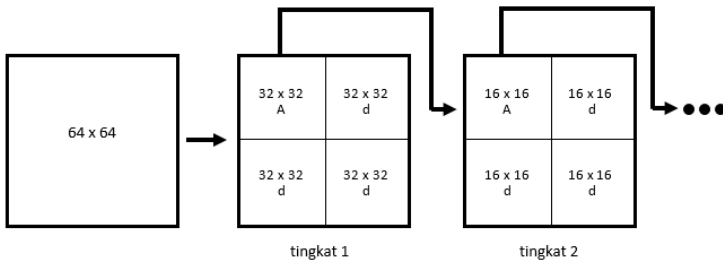
transformasi *wavelet* adalah transformasi yang melakukan filter terhadap suatu masukan. Filter yang digunakan dalam *wavelet Haar* adalah *high pass filter* dan *low pass filter*. Persamaan (2.9) dan persamaan (2.10) adalah *filter coefficient* dari *low pass filter* dan *high pass filter* pada *wavelet Haar* berturut-turut [10].

$$h_0 = h_1 = \frac{1}{\sqrt{2}} \quad (2.9)$$

$$g_0 = \frac{1}{\sqrt{2}} \quad (2.10)$$

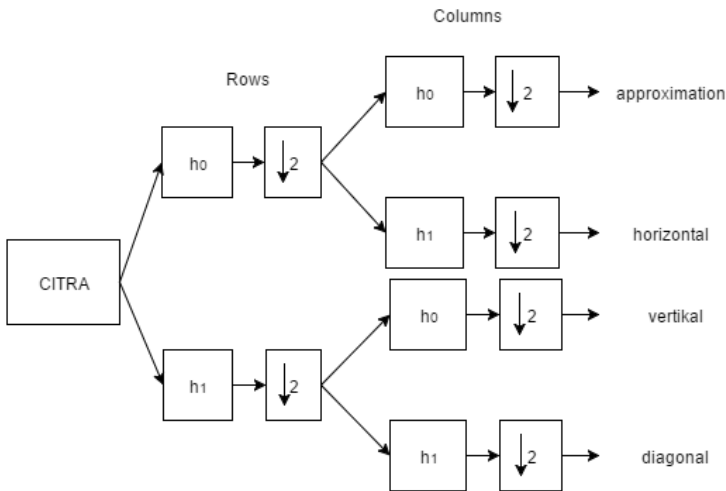
$$g_1 = -\frac{1}{\sqrt{2}}$$

Resolusi citra yang dilakukan *filter* akan ter-reduksi menjadi setengah dari citra awal. Hasil dari proses ini adalah empat subbidang dari citra awal. Keempat subbidang citra ini adalah citra aproksimasi, dan tiga citra detail. Citra detail yang didapat adalah detail horizontal, vertikal dan diagonal. Proses ini disebut dekomposisi, dekomposisi dapat dilanjutkan kembali dengan citra aproksimasi sebagai masukannya untuk mendapatkan tingkat dekomposisi selanjutnya seperti yang ditunjukkan pada Gambar 2.7 [1].



Gambar 2.7 Ilustrasi Dekomposisi *Wavelet*

Tahapan dari proses filter adalah melakukan filter terhadap baris dan kolom citra seperti pada Gambar 2.8. Untuk mendapatkan citra aproksimasi dan citra diagonal dilakukan dengan filter terhadap baris dan kolom. Kemudian melakukan filter terhadap baris saja untuk mendapatkan citra horizontal, dan terakhir melakukan filter terhadap kolom saja untuk mendapatkan citra vertikal [10].



Gambar 2.8 Filter pada *Wavelet*

## 2.8 Log-Gabor Filter

*Log-Gabor filter* adalah sebuah metode yang digunakan untuk memperoleh informasi frekuensi dan dapat menentukan lokasi bagian mana dari sinyal yang menghasilkan frekuensi tertentu. Metode ini banyak digunakan dalam karakteristik tekstur dari suatu citra dengan mencari representasi gabungan optimal pada domain spasial dan frekuensi. Metode ini juga menutupi kelemahan pada metode *Gabor filter* yang memiliki *komponen DC* (*direct current*) ketika bobotnya lebih dari satu oktaf [3]. *Komponen DC* adalah nilai hasil dari transformasi pada frekuensi awal. *Komponen*

DC akan bernilai nol pada bobot berapapun jika menggunakan skala logaritmik yang terdapat pada rumus frekuensi *log-Gabor filter* pada persamaan (2.11).

$$G(f) = \exp\left(-\frac{\left(\log\left(f/f_0\right)^2\right)}{2(\log(\sigma)^2)}\right) \quad (2.11)$$

dengan  $f$  adalah nilai frekuensi,  $f_0$  adalah pusat frekuensi dengan nilai  $1/\lambda$  dan  $\sigma$  adalah bobot filter. Berdasarkan penelitian Masek(2003) pada pengenalan citra iris mata, ditetapkan nilai  $\lambda$  sebesar 18,  $\sigma$  sebesar 0.5, dan interval  $f_0$  sebesar  $[0-0.5]$  [11].

Proses *log-Gabor filter* ini membutuhkan sinyal dari transform Fourier yang berasal dari setiap baris citra hasil normalisasi iris, karena akan direpresentasikan pada domain frekuensi. Kemudian dilakukan konvulsi antara sinyal citra iris dengan filter *log-Gabor* pada setiap bidang satu dimensi. Hasil tersebut akan diubah kembali kedalam bidang spasial dengan inverse transform Fourier. Tiap nilai piksel dari hasil tersebut memiliki komponen bilangan *real*, sehingga perlu dikuantisasi menjadi nilai yang terdiri atas bilangan biner dua bit, dengan ketentuan:

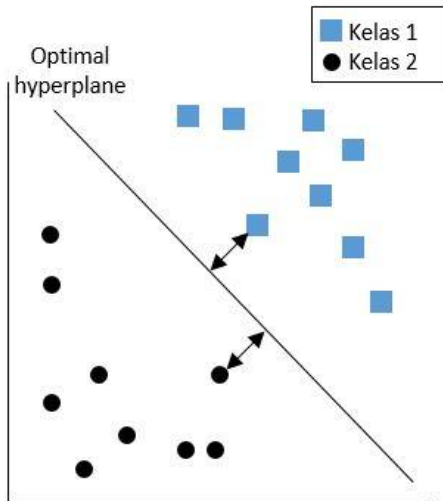
1. jika nilai *real*  $> 0$ , nilai piksel menjadi 1,
2. jika nilai *real*  $< 0$ , nilai piksel menjadi 0.

## 2.9 Support Vector Machine

*Support Vector Machine* (SVM) adalah metode klasifikasi yang hanya memfokuskan pada klasifikasi dua kelas, yaitu kelas +1 dan -1. Pada metode klasifikasi SVM, kedua kelas tersebut digambarkan ditaruh didalam sebuah bidang. Perlu untuk membagi bidang kedalam dua bagian dan masing-masing bagian merepresentasikan satu kelas. Dua kelas tersebut

dipisahkan dengan sebuah *hyperplane*. *Hyperplane* adalah garis pemisah yang memisahkan dua kelas yang berbeda.

Tujuan *hyperplane* adalah mencari fungsi *hyperplane* yang terbaik karena fungsi *hyperplane* itu sendiri mempunyai jumlah kemungkinan yang tak terbatas. Fungsi *hyperplane* dirumuskan dengan persamaan (2.12). Karena tujuan akhirnya adalah mencoba untuk menggunakan *hyperplane* sebagai batas keputusan untuk membedakan dua kelas, sehingga dapat memilih *hyperplane* yang dapat membuat perbedaan yang lebih jelas. Oleh karena itu *hyperplane* yang dicari haruslah yang mempunyai jarak terjauh antara dua kelas tersebut. Jarak atau yang biasa dikenal dengan istilah *margin*. Hal itu dapat dilakukan dengan cara mencari nilai  $w$  dan  $b$  yang optimal, dilakukan dengan persamaan (2.13). Ilustrasi SVM dapat dilihat pada Gambar 2.9 [15].



Gambar 2.9 Ilustrasi *Support Vector Machine*

$$w \cdot x + b = 0 \quad (2.12)$$

$$\min \frac{1}{2} ||w||^2 + C \sum_{i=1}^l t_i \quad (2.13)$$

Pada persamaan diatas, variabel  $C$  adalah konstanta nilai pinalti dari kesalahan klasifikasi. Pencarian nilai  $w$  dan  $b$  dilakukan dengan menggunakan batasan yang ditulis menggunakan persamaan (2.14)

$$y_i(wx_i + b) + t_i \geq 1 \quad (2.14)$$

Persamaan (2.13) dilakukan untuk mencari nilai  $w$  dan  $b$  yang optimum. Fungsi tujuan Persamaan (2.13) berbentuk kuadrat pada variabel  $w$ . Oleh karena itu pengoptimasian pada persamaan (2.13) yang mempunyai nilai pinalti, bentuk tersebut ditransformasi dengan menggunakan pendekatan *Lagrange* atau bentuk *dual space*. Persamaan *dual space* dapat ditulis menggunakan persamaan (2.15).

$$\max \sum_{i=1}^l \alpha_i - \frac{1}{2} \sum_{i,j=1}^l y_i y_j \alpha_i \alpha_j x_i^T x_j \quad (2.15)$$

dengan batasannya terdapat pada persamaan (2.16).

$$\alpha_i \geq 0, \sum_{i=1}^l \alpha_i y_i = 0 \quad (2.16)$$

Untuk mencari nilai  $\alpha_i$ , digunakan *quadratic programming (QR)*. Setelah mendapatkan nilai  $\alpha_i$ , parameter persamaan *hyperplane* menjadi seperti persamaan (2.17).

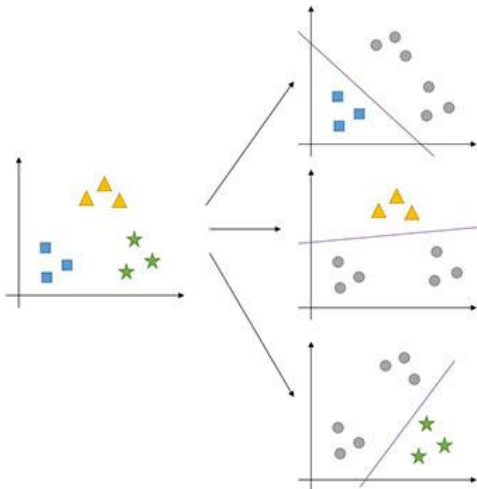
Kemudian hasilnya disubstitusikan ke fungsi *hyperlane* dan dirumuskan kedalam persamaan (2.18).

$$w = \sum_{i=1}^s \alpha_i y_i x_i^T \quad (2.17)$$

$$f = w^T z + b = \sum_{i=1}^s \alpha_i y_i x_i^T z + b \quad (2.18)$$

Dimana  $z$  adalah data training yang dimasukan. Pada banyak kasus, data training yang diklasifikasikan tidak bisa dipisahkan dengan garis yang linear.

Metode klasifikasi ini pula dapat digunakan dalam hal mengklasifikasi multi kelas. Pada klasifikasi multi kelas, output dari set data memiliki lebih dari dua kelas atau kategori. Secara garis besar tahapannya sama dengan cara klasifikasi dua kelas, namun berbeda konsep [16].



Gambar 2.10 Ilustrasi Pembuatan Model SVM Multi Kelas

SVM multi kelas membutuhkan proses pemodelan pada data training yang lebih banyak dibandingkan jika hanya dua kelas. Tahapan pertama metode multi kelas ini yaitu melakukan pembuatan model atau pemisahan setiap kelas dengan kelas yang lain. Hal itu dilakukan sebanyak jumlah kelas. Kemudian saat hendak memprediksi sebuah kelas data masukan, dibandingkan satu per satu terhadap model yang sudah dibuat. Ilustrasi SVM multi kelas dapat dilihat pada Gambar 2.10.

Akurasi model yang akan dihasilkan bergantung pada fungsi kernel serta parameter yang digunakan. Oleh karena itu digunakan metode kernel untuk mencari optimasi terbaik. Dengan metode kernel, suatu data  $x$  di *input space* dimapping ke fitur *space*  $F$  dengan dimensi yang lebih tinggi. Beberapa fungsi kernel yang terkenal digunakan pada model SVM dirumuskan pada persamaan (2.19) [11].

$$k(x, y) = \begin{cases} x \cdot y \\ (\gamma \langle x^T y \rangle + C)^p \\ \exp(-\gamma |x - y|^2) \end{cases} \quad (2.19)$$

Persamaan diatas adalah fungsi kernel linear, fungsi kernel polynomial dan radial basis function (RBF) secara berturut-turut. Persamaan diatas mempunyai parameter kernel yaitu  $p$  dan  $\gamma$ . Dimana variabel  $p$  adalah parameter yang menandakan *order*. Sedangkan variabel  $\gamma$  menandakan *scaling factor*. Dalam fungsi kernel ini, RBF adalah kernel yang paling bagus untuk digunakan karena alasan berikut [17]:

1. RBF kernel memetakan kedalam ruang dimensi yang lebih tinggi daripada linear kernel,
2. RBF kernel memiliki *hyperparameters* lebih sedikit daripada polynomial kernel,
3. RBF kernel memiliki kesulitan lebih sedikit dalam hal numerik.

## 2.10 Hamming Distance

*Hamming distance* adalah metode untuk melakukan pencocokan antara nilai intensitas pada citra yang sudah melalui proses ekstraksi fitur dan menghasilkan vektor fitur. Vektor fitur direpresentasikan dengan bilangan biner dua bit. Penggunaan dengan bilangan biner lebih mudah untuk menentukan perbedaan antara dua kode biner daripada menggunakan bilangan biasa. Selain itu, bilangan boolean lebih mudah untuk dibandingkan dan dimanipulasi [13]. Tabel operasi eksklusif OR (XOR) pada Tabel 2.1 memperlihatkan jika dua bit memberikan nilai biner yang kembar (1 atau 0), maka operasi XOR memberikan nilai 0 pada perbandingan tersebut. Sebaliknya, jika dua bit yang dibandingkan itu berbeda, operasi akan memberikan nilai 1 pada perbandingan tersebut.

Tabel 2.1 Operasi XOR

A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

Pada proses pengenalan iris, *Hamming distance* memberikan ukuran seberapa banyak bit yang sama antara dua vektor fitur. Metode ini pada dasarnya adalah fungsi eksklusif OR (XOR) antara dua pola bit. Setiap bit pada vektor fitur citra masukan dibandingkan dengan vektor fitur dalam database. Dalam melakukan perbandingan vektor fitur bit  $X$  dan  $Y$ , *Hamming distance*,  $HD$ , didefinisikan dengan persamaan (2.20) [8].

$$HD = \frac{1}{N} \sum_{j=1}^N X_j (XOR) Y_j \quad (2.20)$$



dimana  $N$  adalah dimensi dari vektor fitur

$X_j$  adalah komponen ke- $j$  dari vektor fitur

$Y_j$  adalah komponen ke- $j$  dari vektor fitur

Jika dua bit vektor fitur iris yang benar-benar berbeda jauh atau berasal dari iris yang berbeda, maka hasil dari jarak *Hamming* menjauhi nilai 0.0. Sebaliknya jika dua bit vektor iris berasal dari iris yang sama, jarak *Hamming* akan mendekati nilai 0.0 karena saling berkorelasi.

## 2.11 Bahasa Pemrograman Matlab 8.3

Matlab (*Matrix Laboratory*) adalah sebuah bahasa pemrograman komputer generasi keempat dengan performansi yang tinggi. Matlab menggabungkan komputasi, pemrograman, dan visualisasi pada lingkungan yang mudah digunakan. Permasalahan dan solusinya dinyatakan dalam notasi matematika yang sudah dikenal.

Matlab merupakan suatu sistem interaktif yang memiliki elemen data dalam suatu array sehingga tidak memerlukan dimensi. Hal ini memungkinkan untuk memecahkan banyak masalah teknis yang terkait dengan komputasi, khususnya yang berhubungan dengan matriks dan vektor. Matlab menyediakan akses mudah ke aplikasi matriks yang dikembangkan oleh proyek LINPACK dan EISPACK. Saat ini, mesin Matlab menggabungkan *library* LAPACK dan BLAS. *Tool* ini merupakan satu kesatuan dari sebuah seni tersendiri pada aplikasi untuk komputasi matriks.

Matlab berkembang dari tahun ke tahun dengan berbagai masukan dari penggunanya. Bahasa ini adalah *tool* standar untuk memperkenalkan dan mengembangkan penyajian materi matematika, rekayasa dan kelimuan. Sistem Matlab terdiri dari lima bagian utama yaitu:

1. *Development Environment*. *Tool* dan fasilitas yang membantu penggunaan fungsi dan file Matlab. Sebagian

besar tool ini berbasis *graphical user interface*. Bagian ini terdiri dari Matlab *desktop*, *command window*, *command history*, *editor*, *debugger*, *help*, *workspace*, dan file

2. *The Matlab Mathematical Function Library*. Sekumpulan algoritma komputasi mulai dari fungsi-fungsi dasar seperti: *sum*, *sinus*, *cosinus*, dan aritmatika kompleks, sampai dengan fungsi-fungsi tingkat lanjut seperti *matriks eigenvalues* dan *fast Fourier transforms*.
3. *The Matlab Language*. Bahasa pemrograman matriks tingkat tinggi dengan *flow control* fungsi, struktur data, input/output.
4. *Graphics*. Matlab mempunyai fasilitas untuk menampilkan vektor dan matriks dalam bentuk grafik. Fungsi yang disediakan meliputi visualisasi data dua dan tiga dimensi, pengolahan citra, animasi, dan penampilan grafik.
5. *The Matlab Application Program Interface (API)*. Merupakan suatu library yang memungkinkan program dalam bahasa C dan Fortran mampu berinteraksi dengan MATLAB. Ini melibatkan fasilitas untuk pemanggilan MATLAB sebagai sebuah computational engine, dan untuk membaca dan menuliskan file berekstensi .MAT.

## **BAB III**

### **DESAIN PERANGKAT LUNAK**

Pada bab ini dijelaskan mengenai rancangan sistem perangkat lunak yang akan diimplementasikan. Perancangan yang dijelaskan meliputi data dan proses. Data yang dimaksud adalah data yang akan diolah baik digunakan sebagai pembelajaran maupun pengujian sehingga tujuan tugas akhir ini bisa tercapai.

#### **3.1 Perancangan Data**

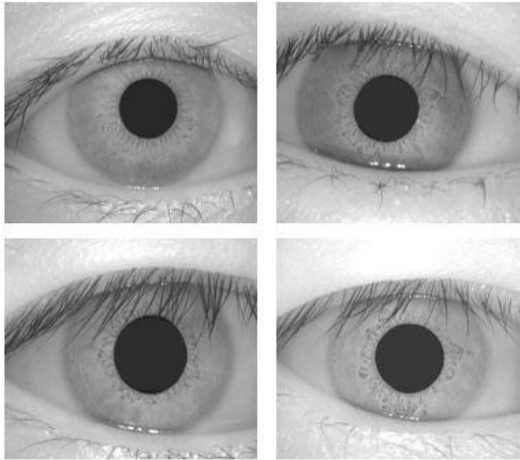
Pada sub bab ini akan dijelaskan mengenai data yang digunakan sebagai masukan untuk selanjutnya diolah dan dilakukan pengujian sehingga menghasilkan data keluaran yang diharapkan.

Data yang digunakan pada proses pengenalan iris dibagi menjadi tiga macam yaitu data masukan, data proses, dan data keluaran. Data masukan merupakan input dari pengguna perangkat lunak. Data proses adalah data ketika tahap-tahap pengenalan iris sedang dilakukan. Terakhir, data keluaran adalah data yang ditampilkan dari hasil proses pengenalan iris. Penjelasan masing-masing jenis data diberikan sebagai berikut.

##### **3.1.1 Data Masukan**

Data masukan adalah data yang digunakan sebagai masukan dari sistem. Data yang digunakan berupa citra mata yang bersumber dari database citra mata *Chinese Academy of Science-Institute of Automation (CASIA)* versi 1.0 seperti yang telah dibahas pada batasan permasalahan tugas akhir. Pada database ini berisi 756 citra mata dengan 108 mata yang berbeda. Masing-masing mata ini dianggap sebagai satu kelas. Didalam satu kelas terdapat tujuh citra mata. Sehingga total keseluruhan citra mata pada database ini yaitu  $108 \times 7 = 756$  citra mata. Semua citra mata ini bertipe *grayscale*.

Tiap kelas pada database citra mata ini diambil dalam dua sesi dengan interval satu bulan pada sesinya. Citra mata ini diambil dengan alat optik khusus yang dikembangkan oleh *National Laboratory of Pattern Recognition*, China, dengan tujuan penelitian pengenalan iris. Pada Gambar 3.1 Citra Mata CASIA adalah contoh citra mata dari database CASIA.



Gambar 3.1 Citra Mata CASIA

Database ini dibagi menjadi dua, yaitu *training data* dan *testing data* atau yang biasa dikenal dengan data latih dan data uji. Training data merupakan data yang digunakan untuk melatih sistem apakah akurasi dan keluarannya sudah sesuai dengan apa yang diharapkan. Sedangkan testing data merupakan data yang digunakan untuk menguji sistem setelah melakukan proses *training*. Dari tiap kelas didalam database akan dibagi dua bagian, yakni empat sebagai data latih dan tiga sebagai data uji.

### 3.1.2 Data Proses

Data proses adalah data yang digunakan selama proses berjalan. Data proses meliputi nama data, tipe data, dan keterangannya. Data proses ditunjukkan dalam Tabel 3.1.

Tabel 3.1 Data Proses

No.	Nama data	Keterangan
1.	Data citra mata	Data dari database yang akan diproses bertipe <i>bitmap</i>
2.	Data fitur	Vektor fitur citra yang telah terdeteksi dan disimpan dalam bentuk <i>.mat</i>
3.	Data citra implementasi	Citra mata yang diolah dalam pengimplementasian yang bertipe data <i>double</i>

Pemrosesan akan dilakukan dengan menggunakan data masukan yang berupa citra mata *grayscale* bertipe *bitmap*. Data citra mata ini akan dikonversi kedalam tipe data *double* untuk memudahkan pengolahan citra. Data latih digunakan sebagai data belajar klasifikasi, sedangkan juga dengan data uji sebagai masukan untuk pencocokan. Data latih dan data uji ini nanti akan diproses sampai menjadikan vektor fitur citra iris. Data vektor fitur juga disimpan dalam tipe data *double*. Pada saat pencocokan, dua data fitur dari data uji dan data latih akan dibandingkan satu sama lain.

### 3.1.3 Data Keluaran

Data keluaran dari sistem ini adalah hasil dari proses yang sudah dilakukan. Data masukan yang sudah diekstrasi fiturnya akan diproses dengan menggunakan klasifikasi SVM dan klasifikasi *Hamming distance* secara terpisah. Hasil dari proses klasifikasi tersebut adalah prediksi kelas pada masing-masing data uji dan akurasi.

### 3.2 Desain Umum Sistem

Secara garis besar ada tiga tahapan utama dalam melakukan pengenalan iris yaitu praproses, ekstraksi fitur, dan klasifikasi. Namun dalam melakukan pengenalan iris ini, tiga tahapan tersebut diperinci menjadi empat tahapan sebagai berikut:

1. Praproses citra mata
2. Normalisasi iris
3. Ekstraksi fitur iris
4. Klasifikasi iris

Tahapan yang pertama merupakan tahap praproses. Namun sebelum masuk ke praproses, citra mata harus didapatkan terlebih dahulu melalui tahapan akuisisi citra mata dari *disk*. Setelah mendapatkan sebuah citra mata, citra mata tersebut akan dipanggil dan diproses, sehingga tahap praproses dapat dilakukan. Tujuan tahap praproses adalah mengidentifikasi bagian iris pada citra mata. Tahap praproses ini dibagi menjadi tiga bagian. Praproses itu sendiri terdiri dari deteksi tepi, deteksi batas pupil dan iris, dan pemisahan bulu dan kelopak. Hasil dari tahap praproses adalah citra iris. Tahap ini juga memisahkan bulu mata dan kelopak mata pada lingkaran iris.

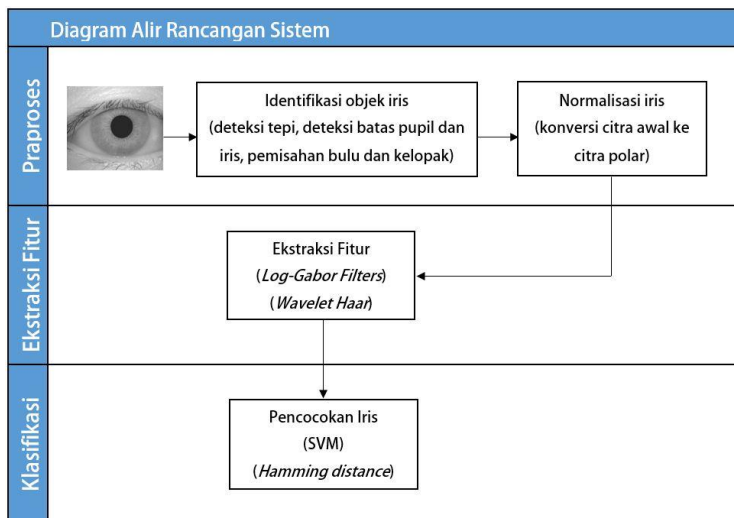
Bagian iris yang sudah teridentifikasi kemudian dilakukan normalisasi. Tujuannya adalah menciptakan dimensi yang konstan pada iris, karena iris luasan iris manusia berubah-ubah tergantung dari berbagai faktor dan juga memudahkan untuk melakukan verifikasi nantinya. Hasil dari tahap praproses diolah kembali dengan menggunakan transformasi polar pada citra iris sehingga menghasilkan sebuah citra iris yang mudah untuk diolah pada tahap selanjutnya.

Setelah melakukan normalisasi, citra iris dimasukan kedalam tahap ekstraksi fitur. Ekstraksi fitur berguna untuk mendapatkan tekstur iris yang penting. Pada tahap ekstraksi fitur ini menggunakan dua buah ekstraksi fitur yang tidak saling

berkorelasi, yaitu *wavelet Haar* dan *log-gabor filter*. Kedua metode tersebut akan diujikan pada dua metode klasifikasi yang berbeda dan tidak saling berkorelasi pula. Hasil tahap ini adalah vektor fitur yang digunakan untuk melakukan verifikasi pada pengenalan iris.

Setelah melalui tahap ekstraksi fitur, vektor-vektor fitur akan di verifikasi menggunakan metode *Support Vector Machine* atau menggunakan metode *Hamming distance* dengan menggunakan fitur yang telah didapatkan. Hasil akhir yang dikeluarkan adalah nama kelas dengan menggunakan masing-masing dua metode klasifikasi tersebut.

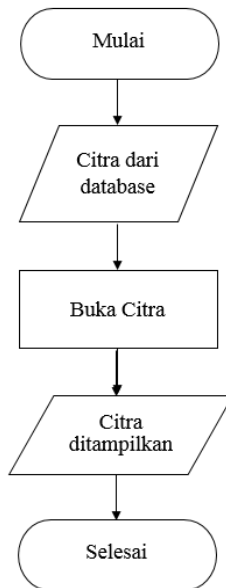
Gambar 3.2 menunjukkan tiga tahapan utama dalam pengenalan iris. Terdapat tahap Praproses, Ekstraksi Fitur, dan Klasifikasi. Tahap Praproses adalah tahap untuk mengidentifikasi iris dari citra input mata. Tahap Ekstraksi Fitur terdiri dari dua metode, *wavelet Haar* dan *log-gabor filter*. Tahap Klasifikasi juga terdiri dari dua metode, SVM dan *Hamming distance*.



Gambar 3.2 Diagram Alir Rancangan Sistem

### 3.3 Akuisisi Citra Mata

Seperti yang telah dijelaskan pada bagian perancangan data, citra mata akan digunakan sebagai masukan untuk keseluruhan proses pengenalan iris. Citra mata diasumsikan telah disimpan didalam *disk* komputer. Dalam tugas akhir ini citra mata yang digunakan berasal dari database CASIA versi 1. Diagram alir proses akuisisi citra mata dapat dilihat pada Gambar 3.3.



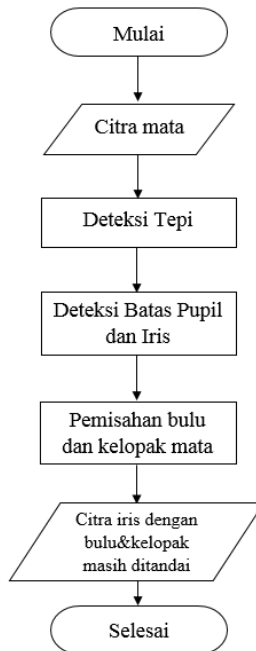
Gambar 3.3 Diagram Alir Akuisisi Citra Mata

Tahap awalnya adalah memilih sebuah citra dari *disk*, kemudian dari *disk* tersebut citra akan dipanggil kedalam aplikasi. Terakhir citra yang akan dikenali irisnya, ditampilkan didalam aplikasi. Citra mata yang ditampilkan mempunyai tipe data uint8.



### 3.4 Perancangan Praproses

Citra mata yang sudah diakuisisi tidak bisa langsung dilakukan verifikasi. Tahap awal yang dilakukan adalah tahap praproses atau nama lainnya *preprocessing*. Praproses bertujuan untuk memisahkan daerah iris dari citra mata dengan mendeteksi batas dalam dan batas luar iris. Praproses dilakukan melalui tiga tahap yaitu deteksi tepi, deteksi batas pupil dan iris, dan pemisahan bulu dan kelopak. Diagram alir tahap praproses ditunjukkan pada Gambar 3.4.



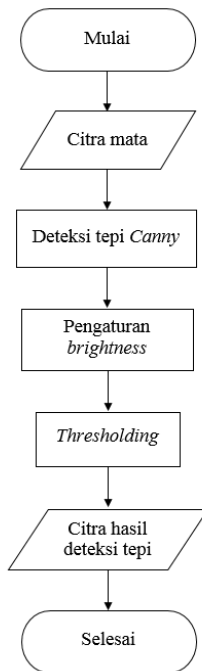
Gambar 3.4 Diagram Alir Praproses

Tahap pertama yaitu deteksi tepi pada citra mata. Kemudian mendeteksi batas antara pupil dan iris untuk mendapatkan citra iris. Setelah itu dipisahkan bagian bulu dan kelopak mata yang merupakan *noise* pada daerah iris. Hasil

akhir dari tahap ini adalah citra iris yang akan dinormalisasi. Masing-masing tahap akan dijelaskan lebih lanjut pada subbab ini.

### 3.4.1 Deteksi Tepi

Tahap deteksi tepi adalah tahap dimana menandai tepi-tepi pada citra mata. Tepi-tepi tersebut memperlihatkan setiap bagian citra menjadi jelas. Tujuan utama tahap ini adalah memperjelas bagian lingkaran iris terlebih dahulu kemudian lingkaran pupil. Proses deteksi tepi ini menggunakan pengaturan *brightness* dan *thresholding* citra sehingga tepi lingkaran menjadi jelas. Diagram alir proses deteksi tepi ditunjukkan pada Gambar 3.5.



Gambar 3.5 Diagram Alir Proses Deteksi Tepi

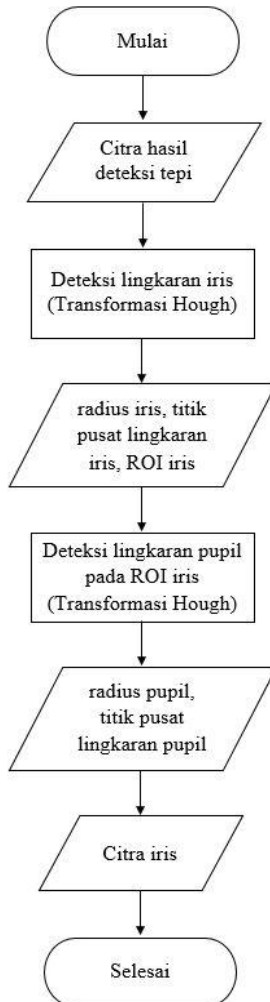
Masukan dari tahap ini adalah citra mata hasil dari akuisisi. Proses yang dilakukan dilakukan pertama kali pada data masukan tersebut adalah mendeteksi tepi. Deteksi tepi yang digunakan adalah deteksi tepi *Canny*. Didalam deteksi tepi tersebut juga sekaligus dilakukan proses pengkontrasan dan thresholding. Hasil keluaran dari tahap ini adalah citra biner yang menandakan tepi-tepi pada citra masukan.

### 3.4.2 Deteksi Batas Pupil dan Iris

Tahapan yang dilakukan disini merupakan proses untuk mendapatkan batas lingkaran iris dan pupil. Masukan dari proses ini adalah citra hasil dari deteksi tepi. Untuk mendapatkan lingkaran iris dan lingkaran pupil, dilakukan dengan metode transformasi Hough pada citra masukan. Transformasi Hough ini dilakukan dua kali. Transformasi Hough yang pertama digunakan untuk mendapatkan lingkaran iris dan menghasilkan *region of interest* (ROI) iris. Selanjutnya dilakukan transformasi Hough yang kedua pada ROI iris untuk mendapatkan lingkaran pupil. Jika menggunakan transformasi Hough sekaligus untuk mencari kedua lingkaran, maka waktu prosesnya akan lama, maka dari itu menggunakan dua kali transformasi Hough sebagai gantinya agar waktu proses bisa direduksi.

Untuk mendapatkan titik pusat lingkaran iris, pada transformasi Hough membutuhkan parameter radius terlebih dahulu. Oleh karena itu dilakukan iterasi untuk radius yang tepat bagi database citra mata CASIA. Radius lingkaran iris minimal dan maksimal yang ditentukan yaitu 80 piksel dan 150 piksel. Sehingga transformasi ini dilakukan secara iterasi dari radius 80 piksel sampai dengan 150 piksel. Jadi terdapat 70 buah radius pada bidang akumulator. Titik pusat lingkaran iris didapatkan dari nilai intensitas maksimal pada tiap bidang akumulator. Radius lingkaran yang dipilih adalah radius yang berkaitan dengan titik pusat yang ditemukan. Hasil output dari

transformasi Hough yang pertama adalah titik pusat lingkaran iris dan radius iris.



Gambar 3.6 Diagram Alir Proses Deteksi Batas Pupil dan Iris

Setelah melakukan transformasi pada lingkaran iris, selanjutnya dilakukan kembali transformasi yang sama terhadap lingkaran pupil dengan menggunakan hasil dari transformasi pertama. Dari hasil transformasi pertama ini bisa ditentukan ROI iris. Setelah itu transformasi Hough kedua dilakukan pada ROI iris tersebut untuk mendeteksi lingkaran pupil. Pada transformasi ini ditentukan radius untuk pupil yaitu 28 piksel sampai dengan 75 piksel sebagai batas bawah dan batas atas secara berturut-turut. Hasil akhir dari transformasi kedua ini adalah radius pupil dan titik pusat pupil.

Proses untuk mendapatkan batas lingkaran iris dan pupil ini dilakukan dengan menggunakan metode yang sama, tetapi menggunakan ukuran citra yang berbeda. Diagram alir deteksi batas pupil dan iris ditunjukkan pada Gambar 3.6. Alhasil proses keseluruhan ini menghasilkan titik pusat dan radius pada masing-masing iris dan pupil yang nantinya akan diproses pada tahap berikutnya yakni normalisasi iris.

### 3.4.3 Pemisahan Bulu dan Kelopak Mata

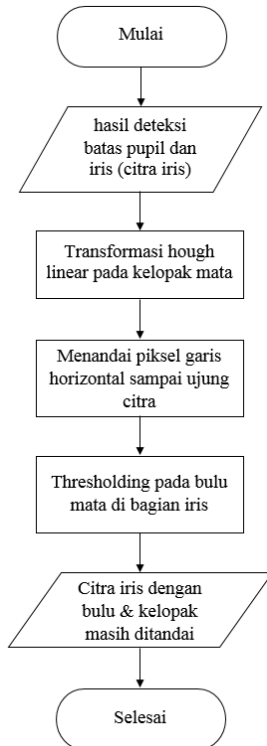
Citra mata yang telah diidentifikasi bagian irisnya masih mengandung bulu dan kelopak mata. Keduanya dianggap sebagai *noise*. Oleh karena itu perlu dilakukan penghilangan bulu dan kelopak yang menghalangi bagian iris.

Untuk memisahkan bagian kelopak mata, caranya dengan membuat sebuah garis pada bagian atas dan bagian bawah kelopak mata dengan transformasi Hough linear. Kemudian, digambar garis horizontal yang berpotongan dengan garis pertama pada tepi iris yang paling dekat dengan pupil. Garis horizontal kedua memungkinkan pemisahan maksimal dari daerah kelopak mata. Intensitas dari batas garis horizontal ini ditarik garis sampai ujung citra diganti dengan notasi NaN. Notasi ini digunakan untuk visualisasi sebagai *noise* yang ditandai.

Pada pemisahan bulu mata, cara yang digunakan adalah operasi *thresholding*. Pada citra iris terdapat bulu mata yang

menutupi bagian iris. Thresholding ini dilakukan pada piksel yang mempunyai intensitas yang hampir serupa dengan bulu mata tersebut. Sehingga saat thresholding dilakukan, semua piksel yang nilainya dibawah threshold akan berubah. Perubahan piksel ini digantikan dengan notasi NaN. Nantinya piksel dengan notasi NaN ini akan diganti nilainya dengan yang lebih baik pada tahap berikutnya yakni normalisasi iris.

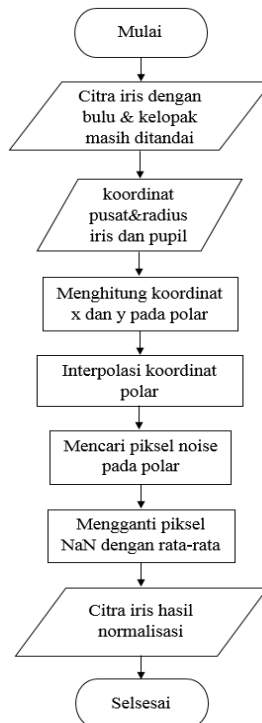
Hasil keluaran dari proses ini merupakan citra yang sudah dapat di normalisasikan dan *noise* sudah berhasil ditandai. Gambar 3.7 menunjukan diagram alir proses pemisahan bulu dan kelopak mata.



Gambar 3.7 Diagram Alir Pemisahan Bulu dan Kelopak Mata

### 3.5 Normalisasi Iris

Pada bab ini akan menjelaskan bagaimana citra iris yang sudah didapat dari hasil praproses dilakukan tahap normalisasi. Citra iris yang berbentuk lingkaran susah untuk dilakukan verifikasi karena mempunyai dimensi yang berbeda-beda, terutama pada radiusnya. Luasan iris mata juga berubah-ubah seiring waktu diakibatkan banyak faktor, salah satunya yaitu pencahayaan yang mengakibatkan pupil mata melebar. Dengan normalisasi, citra iris direpresentasikan lebih baik tanpa memperhatikan penyekalaan atau pembesaran. Diagram alir tahap normalisasi ditunjukan pada Gambar 3.8.



Gambar 3.8 Diagram Alir Normalisasi Iris

Citra iris yang mempunyai radius dan titik pusat pupil dan iris yang telah ditemukan pada tahap praproses digunakan untuk tahap normalisasi. Tahap ini melakukan normalisasi daerah lingkaran iris dari koordinat cartesian ke koordinat polar. Menggunakan transformasi polar seperti pada subbab 2.6, yang didalamnya terdapat persamaan untuk mengubah bidang daerah lingkaran iris dari bentuk lingkaran ke bentuk persegi panjang.

Hasil keluaran dari tahap ini adalah citra iris yang telah dinormalisasi berbentuk persegi panjang dengan dimensi 64x512 piksel. Piksel-piksel pada citra normalisasi yang bernotasi NaN digantikan nilainya dengan rata-rata nilai piksel citra normalisasi.

### 3.6 Perancangan Ekstraksi Fitur

Ekstraksi fitur iris merupakan ciri khas penting pada citra iris ternormalisasi. Masukannya adalah hasil dari tahap normalisasi. Terdapat dua metode ekstraksi fitur yang dipakai. Kedua ekstraksi fitur ini tidak berkaitan satu sama lain. Metode ekstraksi fitur yang dipakai adalah *wavelet Haar* dan *log-Gabor filter*. Ekstraksi fitur yang dipakai *wavelet Haar* adalah koefisien aproksimasi dari hasil dekomposisi pada tingkat-tingkat tertentu. Sedangkan ekstraksi fitur *log-Gabor filter* menghasilkan bilangan bit biner. Masing-masing dari ekstraksi fitur tersebut menghasilkan vektor fitur yang akan digunakan untuk melakukan tahap klasifikasi.

#### 3.6.1 Fitur Koefisien Wavelet Haar

Ekstraksi fitur dimulai dengan dengan melakukan transformasi *wavelet Haar* pada citra iris yang sudah ternormalisasi. Citra iris yang ternormalisasi akan dilakukan proses sesuai dengan penjelasan pada subbab 2.7, yang dimana citra iris akan didekomposisi. Pada sistem ini, citra iris akan didekomposisi pada tingkat tertentu. Artinya, hasil dimensi citra iris akan tereduksi menjadi lebih kecil dari dimensi awal.



Pseudocode penghitungan dekomposisi citra iris ditunjukkan pada Gambar 3.9.

Masukan	Citra iris hasil normalisasi
Keluaran	Vektor fitur koefisien aproksimasi dekomposisi tingkat 3
<pre> 1.  wname ← 'haar' (nama wavelet) 2. 3.  for i=1 to 3 (contoh jika direduksi lvl 3) 4.    [cA,~,~,~] = dwt2(citraNormalisasi,wname) 5.    citraNormalisasi ← cA 6.  end 7.  return cA </pre>	

Gambar 3.9 *Pseudocode* Dekomposisi Wavelet

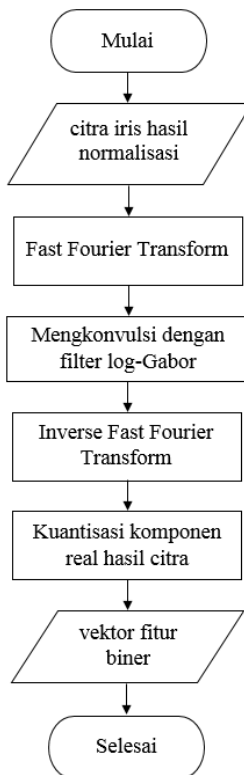
Keluaran dari tahap ini adalah vektor fitur dengan dimensi 8x64 piksel. Koefisien aproksimasi diambil karena berisi informasi yang jelas dibandingkan koefisien detail. Vektor fitur tersebut merupakan representasi dari citra iris yang dibentuk dengan transformasi *wavelet* yang akan digunakan sebagai masukan untuk tahap klasifikasi.

### 3.6.2 Fitur Log-Gabor Filter

Data masukan yang digunakan adalah citra normalisasi dari tahap normalisasi dan citra *noise* dari hasil proses pemisahan bulu dan kelopak mata. Pada dasarnya ekstraksi fitur ini akan mengubah fitur-fitur iris pada citra hasil normalisasi kedalam bentuk bit biner dengan dimensi yang sama, yakni 64x512. Untuk itu digunakan ekstraksi fitur *log-Gabor filter* sebagai suatu batu loncatan untuk memperoleh bilangan biner tersebut. Gambar 3.10 menunjukkan diagram alir dari ekstraksi fitur ini.

Proses ekstraksi fitur ini pada dasarnya menjadikan citra iris menjadi bilang biner seperti yang dijelaskan pada subbab 2.8. Setiap baris pada citra yang telah dinormalisasi, yaitu sebanyak 512 piksel dilakukan proses *Fast Fourier Transform* untuk mempresentasikan citra pada domain

frekuensi. Kemudian setelah itu, nilai tersebut akan dikonvulsi dengan filter log-*Gabor*. Hasil konvulsi akan dilakukan proses *Inverse Fast Fourier Transform* untuk mengembalikan representasi citra ke dalam domain spasial. Tetapi saat mengembalikan citra ke dalam domain spasial, tiap piksel pada citra memiliki komponen bilangan *real*. Selanjutnya akan dilakukan proses kuantisasi di komponen bilangan *real* pada citra.



Gambar 3.10 Diagram Alir Pembuatan Fitur Biner

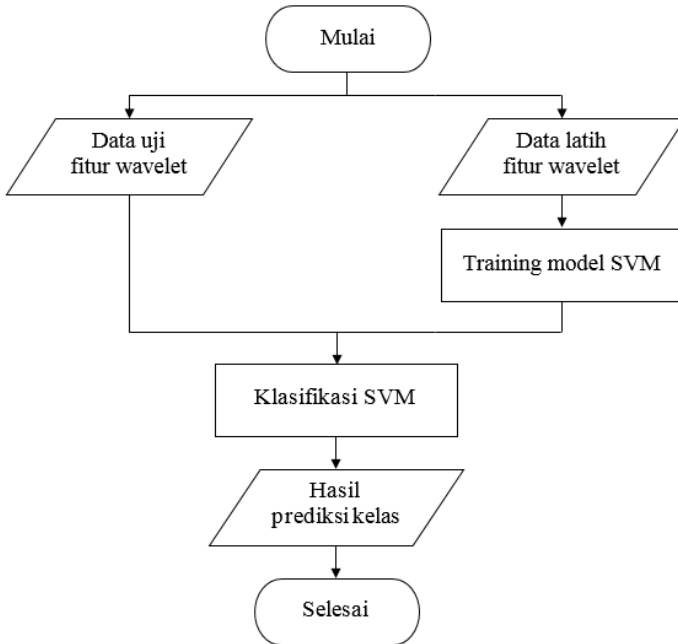
### 3.7 Perancangan Klasifikasi

Klasifikasi dilakukan dengan mengolah data hasil yang sudah didapatkan dari proses ekstraksi fitur. Setelah melalui tahap ekstraksi fitur, citra iris mata direpresentasikan dalam bentuk vektor fitur. Untuk setiap data latih dan data uji citra iris akan dijadikan sebagai vektor fitur. Kemudian dilakukan tahap verifikasi menggunakan metode klasifikasi *Support Vector Machine* (SVM) dan metode *Hamming distance*. Seperti halnya metode ekstraksi fitur, kedua metode klasifikasi ini tidak saling berkaitan.

Tahap ini terlebih dahulu membuat database vektor fitur seluruh citra input dan disimpan dalam sebuah file berekstensi .mat dengan menggunakan Matlab. Jika ada citra masukan yang ingin dikenali irisnya termasuk pada kelas mana, maka akan dilakukan klasifikasi menggunakan salah satu metode klasifikasi. Masukan dari tahap klasifikasi adalah vektor fitur citra iris yang sudah dijelaskan sebelumnya. Hasil akhir dari proses ini adalah prediksi kelas dari hasil tahap klasifikasi.

#### 3.7.1 Klasifikasi Support Vector Machine Fitur Wavelet

Pada subbab ini dijelaskan klasifikasi dengan menggunakan metode SVM, vektor fitur yang didapatkan adalah hasil dari ekstraksi fitur *wavelet Haar*. Sebelum masuk ke dalam proses klasifikasi, data uji dan data latih diproses terlebih dahulu menjadi vektor fitur. Dengan menggunakan SVM, data latih akan dilakukan pembuatan model pada tiap-tiap kelas. Setelah itu data uji akan diprediksi menggunakan SVM termasuk kelas manakah data uji tersebut. Gambar 3.11 menunjukkan diagram alir tahap klasifikasi dengan menggunakan SVM.



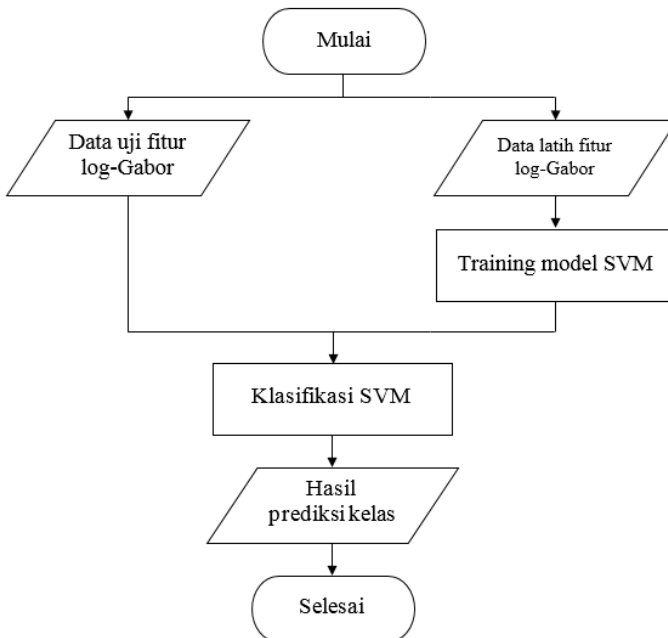
Gambar 3.11 Diagram Alir Klasifikasi SVM menggunakan Fitur *Wavelet Haar*

### 3.7.2 Klasifikasi Support Vector Machine Fitur Log-Gabor

Pada subbab ini dijelaskan klasifikasi dengan menggunakan metode *Support Vector Machine* dengan data masukannya adalah vektor fitur yang didapatkan adalah hasil dari ekstraksi fitur *log-Gabor filter*. Masukan dari proses ini adalah vektor fitur hasil ekstraksi fitur *log-Gabor filter*. Namun dalam hal ini, proses ekstraksi fitur *log-Gabor filter* tidak menggunakan proses kuantisasi elemen matrik menjadi bilangan biner.

Sebelum masuk ke dalam proses klasifikasi, data uji dan data latih diproses terlebih dahulu menjadi vektor fitur. Dengan menggunakan SVM, data latih akan dilakukan pembuatan model pada tiap-tiap kelas. Setelah itu data uji akan

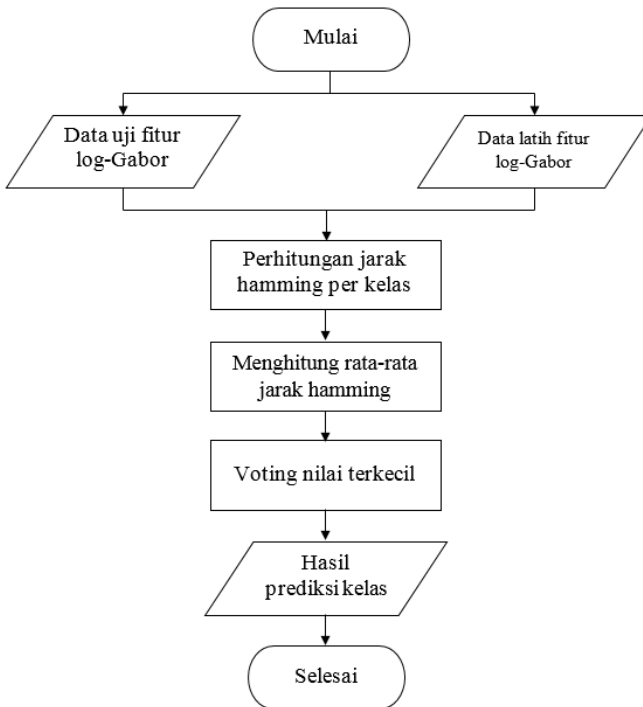
diprediksi menggunakan SVM termasuk kelas manakah data uji tersebut. Gambar 3.12 menunjukan diagram alir tahap klasifikasi ini.



Gambar 3.12 Diagram Alir Klasifikasi SVM menggunakan Fitur *Log-Gabor*

### 3.7.3 Klasifikasi Hamming Distance Fitur Log-Gabor

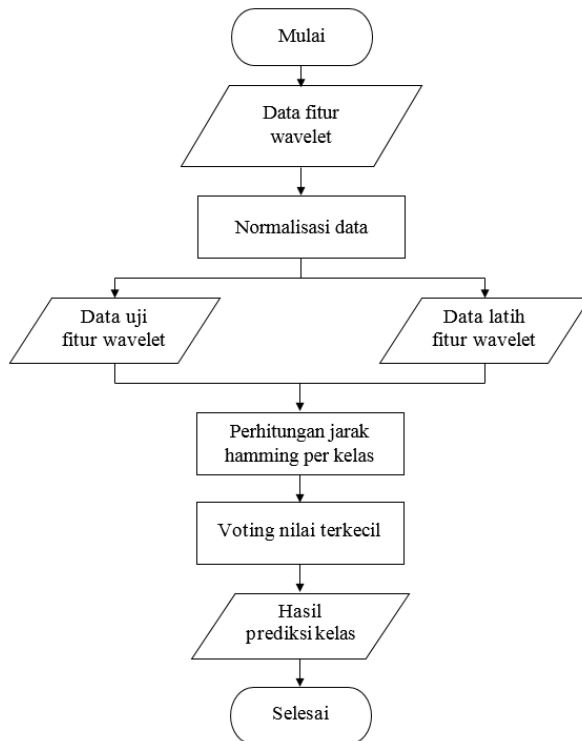
Pada klasifikasi dengan menggunakan metode *hamming distance* pada subbab ini, fitur yang didapatkan adalah hasil dari ekstraksi fitur log-Gabor filters. Sebelum masuk ke dalam proses klasifikasi, data uji dan data latih diproses terlebih dahulu menjadi fitur dalam bentuk biner. Setelah itu, dilakukan perhitungan Hamming. Diagram alir ditunjukkan pada Gambar 3.13, prediksi kelas adalah yang mempunyai nilai paling kecil.



Gambar 3.13 Diagram Alir *Hamming distance* menggunakan Fitur *Log-Gabor*

### 3.7.4 Klasifikasi Hamming Distance Fitur Wavelet

Pada subbab ini, klasifikasi dilakukan dengan menggunakan metode *hamming distance* dan fitur yang dipakai adalah hasil dari ekstraksi fitur *wavelet Haar*. Sebelum masuk ke dalam proses klasifikasi, dataset yang sudah dilakukan proses dekomposisi *wavelet* dilakukan normalisasi data terlebih dahulu. Kemudian dataset tersebut dibagi menjadi data uji dan data latih. Selanjutnya kedua data tersebut diproses menjadi fitur dalam bentuk biner. Setelah itu, dilakukan perhitungan Hamming. Diagram alir ditunjukkan pada Gambar 3.14, prediksi kelas adalah yang mempunyai nilai paling kecil.



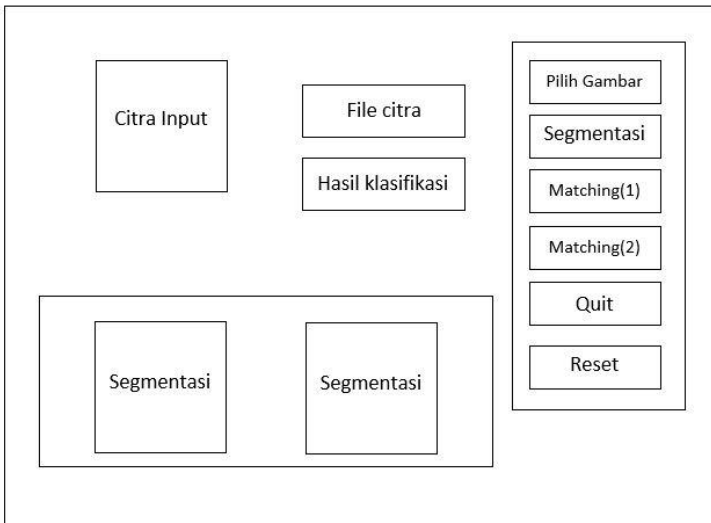
Gambar 3.14 Diagram Alir *Hamming distance* menggunakan Fitur Wavelet

### 3.8 Perancangan Antarmuka

Bagian ini adalah bagian sistem yang memungkinkan user untuk berinteraksi dengan sistem pengenalan iris mata ini. Antarmuka didesain semudah dan sesederhana mungkin untuk digunakan. Gambar 3.15 menunjukkan desain antarmuka sistem pengenalan iris ini. Dalam rancangan antarmuka ini terdapat satu kotak gambar untuk input citra, dua kotak gambar untuk

hasil segmentasi citra, dua kotak teks hasil inpuan, dan enam tombol pada bagian kanan sebagai menu utama.

Kotak gambar citra input akan menunjukkan gambar yang dipilih dari tombol menu *Pilih Gambar*. Nama dari gambar yang dipilih akan ditampilkan pada kotak teks *Filecitra*. Kemudian dua kotak gambar segmentasi akan menghasilkan hasil dari segmentasi mata dari tombol menu *Segmentasi*.



Gambar 3.15 Desain Antarmuka

Tombol menu *Matching(1)* merupakan tombol menu untuk melakukan pencocokan dengan menggunakan klasifikasi *Support Vector Machine*, sedangkan tombol menu *Matching(2)* untuk melakukan pencocokan menggunakan klasifikasi *Hamming distance*. Hasilnya akan ditampilkan pada kotak teks *Hasil klasifikasi*.

Tombol menu *Quit* digunakan untuk keluar dari antarmuka dan tombol menu *Reset* digunakan untuk mendefaultkan ke tampilan awal.



## BAB IV IMPLEMENTASI

Pada sub bab implementasi ini menjelaskan mengenai proses pengenalan iris mata dengan menampilkan kode sumber yang digunakan. Implementasi tiap tahapan meliputi implementasi mendapatkan citra mata, praproses (deteksi tepi, deteksi batas pupil dan iris, serta pemisahan kelopak dan bulu mata), normalisasi iris, ekstraksi fitur iris (*wavelet Haar* dan *log-Gabor filter*), dan klasifikasi iris (*Support Vector Machine* dan *Hamming distance*).

### 4.1 Lingkungan Implementasi

Lingkungan implementasi pengenalan iris ini mencakup perangkat lunak dan perangkat keras yang digunakan. Spesifikasi perangkat keras dan perangkat lunak yang digunakan dalam implementasi ini ditampilkan pada Tabel 4.1.

Tabel 4.1 Lingkungan Perancangan Perangkat Lunak

Perangkat	Spesifikasi
Perangkat keras	Prosesor: Intel® Core™ i5-33170U CPU @ 1.70GHz 1.70GHz Memori: 4.00 GB
Perangkat lunak	Sistem Operasi: Microsoft Windows 8 64-bit Perangkat Pengembang: Matlab 8.3 & <i>Image Processing Toolbox</i> Perangkat Pembantu: Microsoft Excel 2013, <i>libsvm</i>

## 4.2 Implementasi Akuisisi Citra Mata

Tahap paling awal yang dilakukan pada proses pengenalan iris adalah akuisisi citra. Akuisisi citra merupakan proses untuk menyiapkan dan mengambil data citra mata dari *disk*. Implementasi akuisisi citra mata dilakukan dengan menggunakan fungsi yang sudah disediakan pada *Image Processing Toolbox* Matlab seperti yang ditunjukkan pada Kode Sumber 4.1.

1.	% menampilkan citra masukan
2.	[path , user_cancel] = imgetfile();
3.	if user_cancel
4.	msgbox('Silakan Pilih Gambar Kembali', ...
5.	'Cancel PopUp');
6.	im=imread(path);
7.	axes(handles.axes1);imshow(im);
8.	

Kode Sumber 4.1 Implementasi Tahap Akuisisi Citra Mata

Fungsi `imgetfile()` digunakan untuk memilih file yang berada dalam *disk*. Fungsi tersebut mengembalikan *path* dari file yang telah dipilih dan terdapat *error handling* jika user tidak jadi mengambil file. Jika tidak jadi memilih file, maka terdapat pesan yang muncul seperti yang ditunjukkan baris 4 sampai 5. Terakhir *path* file yang telah dipilih akan dibaca dan ditampilkan pada *axes* seperti yang ditunjukkan pada baris 6 sampai 7.

## 4.3 Implementasi Preprocessing

Setelah citra mata ditampilkan dalam aplikasi perangkat lunak, langkah berikutnya adalah melakukan tahap *preprocessing* atau praproses. Tujuan utama tahap ini adalah mengidentifikasi bagian iris pada citra mata. Tahap ini terdiri dari tiga tahap lagi, yaitu deteksi tepi, deteksi batas pupil dan iris, serta pemisahan bulu dan kelopak mata. masing-masing tahap akan dijelaskan pada subbab berikutnya.

### 4.3.1 Implementasi Deteksi Tepi

Tahap ini adalah langkah paling awal yang dilakukan pada tahap praproses. Deteksi tepi dilakukan untuk mengetahui tepi lingkaran pupil dan iris yang terdapat pada citra mata. Implementasi deteksi tepi pada citra mata ditunjukkan oleh Kode Sumber 4.2.

1.	<code>[I2 or] = canny(image, sigma, scaling, ...                   vert, horz);</code>
2.	<code>I3 = adjgamma(I2, 1.9);</code>
3.	<code>I4 = nonmaxsup(I3, or, 1.5);</code>
4.	<code>edgeimage = hysthresh(I4, hithres, lowthres);</code>

Kode Sumber 4.2 Implementasi Deteksi Tepi

Kode sumber diatas merupakan deteksi tepi dengan menggunakan metode *Canny* yang telah dimodifikasi. Pada baris 1, parameter *image* adalah citra mata yang hendak dilakukan deteksi tepi. Sedangkan *sigma* merupakan standar deviasi untuk filter Gaussian untuk dihaluskan. Parameter *scaling* digunakan untuk mengubah ukuran citra dengan tujuan untuk mempercepat proses. Kemudian dua parameter terakhir merupakan bobot untuk gradien deteksi tepi pada arah vertikal dan horizontal. Pada baris ke-3 dan ke-4 melakukan pengaturan kontras dan perampingan pada garis deteksi tepi. Kemudian pada baris terakhir merupakan proses *threshold* pada citra dengan menggunakan dua buah *threshold* atas (*hithres*) dan *threshold* bawah (*lowthres*).

### 4.3.2 Implementasi Deteksi Batas Pupil dan Iris

Sub bab ini membahas lanjutan implementasi tahap praproses setelah melakukan deteksi tepi. Tahap setelah mendapatkan garis tepi adalah melakukan transformasi Hough. Transformasi Hough digunakan pada garis tepi lingkaran iris terlebih dulu dan selanjutnya dilakukan pada lingkaran pupil seperti yang ditunjukkan pada Kode Sumber 4.3 dan Kode

Sumber 4.4 secara berturut-turut dengan radius yang sudah diinisialisasi pada baris ke-2 sampai ke-4.

1.	% menentukan radius pupil & iris
2.	lpupilradius = 28;
2.	upupilradius = 75;
3.	lirisradius = 80;
4.	uirisradius = 150;
5.	
6.	% transformasi Hough pd iris
7.	scaling = 0.4;
8.	[row, col, r] = findcircle(eyeimage, ...
9.	lirisradius, uirisradius, scaling,...
10.	2, 0.20, 0.19, 1.00, 0.00);
11.	circleiris = [row col r];
12.	
13.	% menentukan panjang&lebar ROI iris
14.	irl = double(round(rowd-rd));
15.	iru = double(round(rowd+rd));
16.	icl = double(round(cold-rd));
17.	icu = double(round(cold+rd));
18.	

Kode Sumber 4.3 Implementasi Transformasi Hough pada Iris

Kode Sumber 4.3 menunjukkan transformasi Hough yang dilakukan pada lingkaran iris yang sudah terdeteksi tepinya. Dengan menggunakan parameter *scaling* untuk mengubah ukuran citra deteksi tepi menjadi lebih kecil dengan tujuan untuk mempercepat pemrosesan transformasi Hough. Pada baris ke-8 sampai dengan baris ke-10 adalah transformasi Hough yang diimplementasikan dengan fungsi *findcircle*. Hasilnya dari transformasi Hough seperti yang terlihat pada baris ke-11. Kemudian menentukan *region of interest* iris untuk digunakan sebagai pencarian lingkaran pupil.

Pendeteksian lingkaran pupil dilakukan dengan masukannya yaitu ROI dari iris seperti yang terlihat pada Kode Sumber 4.4 pada baris ke-2. Selanjutnya dilakukan transformasi Hough pada ROI tersebut dan menghasilkan radius dan titik pusat pupil. Titik pusat pupil perlu dijumlahkan dengan panjang

dan lebar ROI iris untuk mendapatkan titik pusat yang tepat. Hasil keluaran dari transformasi Hough ini seperti yang terlihat pada baris ke-14, yakni radius dan titik pusat pupil.

```

1. % mengambil ROI iris
2. imagepupil = eyeimage( irl:iru,icl:icu);
3.
4. % transformasi Hough pd pupil
5. [rowp, colp, r] = findcircle(imagepupil, ...
6.     lpupilradius, upupilradius, scaling,...
7.     0.6, 2, 0.25, 0.25, 1.00, 1.00);
8.
9. % penjumlahan dgn titik pusat untuk
10. % mendapatkan posisi yang tepat
11. row = round(irl + rowp);
12. col = round(icl + colp);
13.
14. % radius dan titik pusat pupil
15. circlepupil = [row col r];

```

Kode Sumber 4.4 Implementasi Transformasi Hough pada Pupil

Hasil akhir dari tahap deteksi batas pupil dan iris ini adalah radius iris, titik pusat iris, radius pupil, dan titik pusat pupil.

#### 4.3.3 Implementasi Pemisahan Bulu dan Kelopak Mata

Pemisahan bulu dan kelopak mata dilakukan karena kedua hal tersebut adalah *noise*. Hal yang dilakukan pertama kali adalah menghilangkan kelompok mata. Implementasinya ditunjukkan pada Kode Sumber 4.5.

Implementasi dilakukan dengan mengkonversi citra mata ke tipe data *double* agar mudah diolah. Selanjutnya, dibuat garis pada bulu mata dengan transformasi Hough linear pada ujung bulu mata yang paling atas. Langkah ini diimplementasikan pada baris 4 sampai 10. Setelah itu pada baris ke-13 dipilih garis (y) yang maksimal dan dari batas ini ditarik garis horizontal. Dari garis horizontal yang telah

didapatkan ini sampai ujung citra ditandai dengan NaN karena dianggap sebagai *noise*.

```

1. % citra mata
2. imagewithnoise = double(eyeimage);
2.
3. % membuat garis pd bulu mata dgn hough
4. % linear
5. topeyelid = imagepupil(1:(rowp-r),:);
6. lines = findline(topeyelid);
7.
8. if size(lines,1) > 0
9.     [x1 y1] =
10.     linecoords(lines,size(topeyelid));
11.     y1 = double(y1) + irl-1;
12.     x1 = double(x1) + icl-1;
13.
14. % menentukan batas garis
15.     yla = max(y1);
16.     y2 = 1:yla;
17.
18. % menandai kelopak(noise) dengan NaN
19.     imagewithnoise(y2, x1) = NaN;
20. end
21.

```

Kode Sumber 4.5 Implementasi Penghilangan Kelopak Mata

Hal yang kedua adalah menghilangkan bulu mata dengan operasi *thresholding*. Implementasinya ditunjukkan pada Kode Sumber 4.6.

```

1. % thresholding pd bulu mata
2. ref = eyeimage < 100;
3. coords = find(ref==1);
4. imagewithnoise(coords) = NaN;

```

Kode Sumber 4.6 Implementasi *Thresholding* Bulu Mata

Langkah ini dilakukan dengan menandai citra dengan intensitas dibawah 100 dengan notasi NaN. Bulu mata yang terdapat pada citra kebanyakan memiliki nilai intensitas dibawah 100. Proses *thresholding* pada citra mata dilakukan

pada baris ke-2, kemudian dicari nilai yang sesuai pada matriks citra dan dilakukan pengubahan intensitas pada baris 3 dan 4.

Hasil akhir dari tahap ini yaitu citra mata dengan bulu dan kelopak mata yang sudah digantikan intensitasnya dengan NaN. Dipilih NaN karena intensitas tersebut memberikan warna hitam sebagai visualisasi.

#### 4.4 Implementasi Normalisasi Iris

Sub bab ini membahas implementasi tahap normalisasi iris. Tujuan utama melakukan tahap ini adalah untuk merepresentasikan citra iris ke dalam bentuk yang lebih baik. Dikarenakan citra mata setiap orang pada saat pengambilan citra untuk dimasukkan dalam database dan ketika dites bisa saja mengalami pergeseran ataupun perbesaran pupil. Masukan dari tahap ini adalah citra mata yang sudah melalui tahap praproses, yang dimana daerah irisnya sudah teridentifikasi dan *noise* masih dinotasikan dengan NaN.

Implementasi tahap normalisasi iris ini menggunakan transformasi polar. Pada dasarnya transformasi polar mengubah citra iris yang berada dalam bidang Kartesian ke dalam bidang polar, seperti yang sudah dijelaskan pada bab sebelumnya. Implementasi transformasi polar ditunjukkan pada Kode Sumber 4.7.

Namun citra iris yang sudah diubah pada bidang polar masih memiliki *noise*. Sehingga perlu dilakukan pengubahan intensitas pada piksel *noise* tersebut. pengubahan nilai intensitas ditunjukkan pada baris 18 sampai 23, yakni dengan mengganti nilai piksel *noise* dengan nilai rata-rata pada citra iris polar. Sehingga pada tahap selanjutnya semua nilai piksel *noise* sudah mempunyai nilai. Variabel `polar_array` merupakan variabel yang berisi hasil dari tahap normalisasi citra mata deraunya sudah diganti nilai intensitasnya, sehingga bisa dilanjutkan ke tahap ekstraksi fitur.

```

1. % setting r dan theta pada citra polar
2. [theta,r]=meshgrid(linspace(0,2*pi,512),...
3.     linspace(0,1,64));
4.
5. % batas lingkaran pd lingkaran dalam (pupil)
6. xp = PupilCenterX + PupilR*cos(theta);
7. yp = PupilCenterY + PupilR*sin(theta);
8. % batas lingkaran pada lingkaran luar (iris)
9. xi = PupilCenterX + IrisR*cos(theta);
10. yi = PupilCenterY + IrisR*sin(theta);
11.
12. % mapping ke koordinat polar
13. x = (1-r).*xp + r.*xi;
14. y = (1-r).*yp + r.*yi;
15. polar_array = interp2(double(citra),x,y);
16.
17. % ganti nilai NaN dgn rata2 piksel citra
18. coords = find(isnan(polar_array));
19. polar_array2 = polar_array;
20. polar_array2(coords) = 0.5;
21. avg = sum(sum(polar_array2)) / ...
22.     (size(polar_array,1)*size(polar_array,2))
23. polar_array(coords) = avg;
24.
25.

```

Kode Sumber 4.7 Implementasi Transformasi Polar

Hasil akhir dari tahap normalisasi iris ini adalah citra yang berbentuk persegi panjang dengan ukuran 64x512 piksel. Didalamnya adalah tekstur iris yang sudah melalui tahap-tahap sebelumnya.

#### 4.5 Implementasi Ekstraksi Fitur

Ekstraksi fitur merupakan tahap untuk pengambilan ciri dari citra iris normalisasi. Dalam tahap ini, digunakan dua buah metode ekstraksi fitur yang tidak saling berkaitan. Metode yang digunakan untuk mengekstrak citra iris normalisasi adalah *wavelet Haar* dan *log-Gabor filter*. Kedua implementasi ini akan dijelaskan pada sub bab 4.5.1 dan 4.5.2.



### 4.5.1 Implementasi Fitur Koefisien Wavelet Haar

Sub bab ini membahas implementasi tahap ekstraksi fitur. Ekstraksi fitur dilakukan dengan mengubah citra iris normalisasi kedalam domain *wavelet*. Pada implementasi, citra iris ternormalisasi akan dilakukan proses dekomposisi. Proses dekomposisi ini membuat dimensi citra iris ternormalisasi akan tereduksi menjadi setengahnya. Implementasi penggunaan *wavelet Haar* pada citra iris ternormalisasi dapat dilihat pada Kode Sumber 4.8.

1.	<code>% inisialisasi awal</code>
2.	<code>wname = 'haar';</code>
3.	<code>sizeEkstrak = [8 64];</code>
4.	
5.	<code>% proses wavelet haar dekomposisi lvl 3</code>
6.	<code>[C,~] = wavedec2(polar_array,level,wname);</code>
7.	<code>HasilWavelet = C(1:(sizeEkstrak(1) * ...</code>
	<code>sizeEkstrak(2)));</code>
8.	

Kode Sumber 4.8 Implementasi Ekstraksi Fitur *Wavelet Haar*

Inisialisasi penggunaan jenis *wavelet* ditunjukan pada baris 2, yang menunjukan bahwa proses dekomposisi *wavelet* ini menggunakan *wavelet Haar*. Diinisialisasikan juga ukuran untuk hasil dari dekomposisi *wavelet* pada baris 3. Kemudian fungsi `wavedec2` digunakan untuk melakukan proses dekomposisi pada `polar_array` (citra normalisasi). Fungsi ini langsung mendekomposisi citra ke tingkat tertentu sesuai dengan nilai `level`, sebagai ganti dari iterasi fungsi `dwt2`. Hasil dekomposisi yang diambil adalah nilai koefisien aproksimasi yang ditunjukan pada baris 7 pada variabel `HasilWavelet`.

### 4.5.2 Implementasi Fitur Log-Gabor Filter

Sub bab ini membahas implementasi tahap fitur bilangan biner. Tahap ini menggunakan ekstraksi fitur *log-Gabor filter*. Masukan dari tahap ini adalah citra iris ternormalisasi yang telah didapatkan dari tahap normalisasi iris.

Pada tahap ini, masukannya untuk ekstraksi fitur adalah citra iris ternormalisasi yang nantinya akan direpresentasikan kedalam bentuk bilangan biner. Implementasi pengubahan citra iris menjadi citra dengan karakteristik biner dapat dilihat pada Kode Sumber 4.9.

```

1.  % konvulsi citra iris dgn filter
2.  [E0] = gaborconvolve(polar_array, ...
3.      minWaveLength, sigmaOnf);
4.
5.  % kuantisasi nilai ke 0 atau 1
6.  E0 = E0{1};
7.  H1 = real(E0) > 0;
8.
9.  % rows dan cols untuk proses konversi
10. nrows = 1:size(polar_array,1);
11. length = size(polar_array,2);
12.
13. % iris template ( var store hasil konversi)
14. templateTest = zeros(size(polar_array));
15.
16. % konversi nilai piksel ke biner
17. for i=0:(length-2)
18.     ja = double((i));
19.     templateTest(nrows,ja+(2)-1)=H1(nrows,i+1);
20. end

```

Kode Sumber 4.9 Implementasi Pembuatan Fitur Biner

Kode Sumber 4.9 melakukan pengubahan citra iris ternormalisasi menjadi citra fitur biner. Hasil konvulsi dari citra iris ternormalisasi dengan filter gabor ditunjukkan pada baris 2, dengan isi fungsi `gaborconvolve` ditunjukkan pada Kode Sumber 4.10. Kemudian hasil dari konvulsi yang mengandung komponen bilangan real akan dikuantisasikan untuk menghasilkan bilangan biner. Terakhir, hasil dari pengkonversian akan disimpan pada variabel `templateTest`.

1.	<code>ndata = size(polar_array,2);</code>
2.	<code>matTemp = [0:fix(ndata/2)];</code>
3.	<code>radius = matTemp/fix(ndata/2)/2;</code>
4.	
5.	<code>% pembuatan filter gabor dgn wavelength = 18</code>
6.	<code>% dan sigma = 0.5</code>
7.	<code>filterlogGabor = zeros(1, ndata);</code>
8.	<code>fo = 1.0/wavelength;</code>
9.	
10.	<code>% persamaan pembuatan filter gabor</code>
11.	<code>filterlogGabor(1:size(radius,2)) = ...</code>
12.	<code>exp((-log(radius/fo)).^2) / ...</code>
13.	<code>(2 * log(sigmaOnf)^2));</code>
14.	

Kode Sumber 4.10 Implementasi Pembuatan Filter Log-Gabor

Pada Kode Sumber 4.10, digunakan parameter `wavelength` yang mengatur frekuensi yang dipakai dan `sigmaOnf` untuk menentukan filter *bandwidth*. Semakin kecil nilai `sigmaOnf`, maka filter *bandwidth* akan semakin besar. Kemudian dengan menggunakan parameter-parameter tersebut membentuk filter gabor. Hasil filter gabor ini membentuk matriks dengan ukuran satu dimensi.

1.	<code>% konvulsi citra normalisasi dgn filter</code>
2.	<code>for r = 1:size(polar_array,1)</code>
3.	<code>    signal = polar_array(r,1:ndata);</code>
4.	<code>    imagefft = fft( signal);</code>
5.	<code>    result(r,:) = ifft(imagefft .* filter);</code>
6.	<code>end</code>
7.	
8.	<code>EO = result;</code>
9.	

Kode Sumber 4.11 Implementasi Konvulsi Citra Iris dengan Filter Log-Gabor

Pada Kode Sumber 4.11 merupakan proses konvulsi citra iris ternormalisasi dengan filter gabor yang sudah terbentuk. Proses konvulsi ini dilakukan dengan melakukan iterasi pada tiap baris citra iris ternormalisasi dengan filter gabor. Namun filter gabor merupakan filter yang berada dalam

domain frekuensi, sehingga citra iris ternormalisasi perlu diubah pula kedalam domain frekuensi. Pengubahan ini menggunakan fungsi pada Matlab `fft` seperti pada baris 4. Hasil dari proses konversi pada domain frekuensi akan dikembalikan ke domain spasial dengan fungsi pada Matlab `ifft` seperti pada baris 5. Hasil akhirnya disimpan dalam variabel `EO`, lalu dikembalikan pada baris 2 Kode Sumber 4.9.

## 4.6 Implementasi Klasifikasi

Sub bab ini membahas implementasi tahap klasifikasi. Dalam tahap ini, digunakan dua buah metode klasifikasi yang tidak saling berkaitan seperti yang dijelaskan pada sub bab 3.2. Hasil dari masing-masing ekstraksi fitur akan digunakan sebagai masukan untuk klasifikasi. Hasil ekstraksi fitur *wavelet Haar* dan ekstraksi fitur *log-Gabor filter* akan digunakan sebagai masukan untuk metode klasifikasi *Support Vector Machine* dan *Hamming distance*. Implementasi keduanya dapat dilihat pada sub bab ini.

### 4.6.1 Implementasi Support Vector Machine Fitur Wavelet

Kode Sumber 4.12 merupakan implementasi klasifikasi *Support Vector Machine*. Sebelum masuk kedalam proses klasifikasi, data citra iris sudah dilakukan ekstraksi fitur dan dibagi menjadi data latih & data uji menggunakan *wavelet Haar* dan disimpan dalam file `dataset_wavelet_haar.mat` yang dapat dilihat pada baris 2. Sehingga dapat dengan mudah diakses isinya. Didalam file tersebut terdapat variabel `TrainingSet` yang dapat dilihat pada baris 3 dan variabel `GroupTrain` pada baris 4. Kedua variabel tersebut adalah data latih dan labelnya yang digunakan sebagai masukan tahap klasifikasi ini.

Sebelum melakukan klasifikasi, data latih dan labelnya terlebih dahulu diproses sebagai model untuk klasifikasi seperti yang terlihat pada baris 7 sampai 11. Pembuatan model ini

bertujuan sebagai representasi tiap kelas. Sehingga akhirnya akan ada model sebanyak jumlah kelas.

```

1. % me-load data latih (hasil haar wavelet)
2. foo = load('dataset_wavelet_haar.mat');
3. TrainingData = foo.TrainingSet;
4. trainLabel = foo.GroupTrain;
5.
6. % train model
7. model = cell(108,1);
8. for k=1:108
9.     model{k} = svmtrain(double(trainLabel==k)
10.        TrainingData, '-c 1 -b 1 -q');
11. end
12.
13. % prediksi kelas
14. probKelas = zeros(1,108);
15. for k=1:108
16.     [~,~,p] = svmpredict(~, HasilWavelet, ...
17.        model{k}, '-b 1 -q');
18.     probKelas(:,k) = p(:,model{k}.Label==1);
19. end
20.
21. % penentuan kelas
22. [~,hasilKelas] = max(probKelas,[],2);
23.

```

Kode Sumber 4.12 Implementasi SVM Fitur Wavelet

Pada baris 14 sampai 23 dilakukan iterasi sebanyak jumlah model yang sudah dibentuk. Sebuah data uji hasilWavelet akan dihitung kemiripannya pada setiap model dengan menggunakan probabilitas. Sehingga pada akhirnya terdapat matriks yang berisi probabilitas pada setiap model seperti yang ditunjukkan pada baris 18. Hasil yang dikeluarkan dari proses ini adalah indeks model dengan nilai terbesar dari matriks tersebut, yang juga dapat dikatakan hasil prediksi kelasnya.

#### 4.6.2 Implementasi Suport Vector Machin Fitur Log-Gabor

Kode Sumber 4.13 merupakan implementasi klasifikasi *Support Vector Machine*. Sebelum masuk kedalam proses klasifikasi, data citra iris sudah dilakukan ekstraksi fitur dan dibagi menjadi data latih & data uji menggunakan *log-Gabor filter* dan disimpan dalam file `dataset_logGabor.mat` yang dapat dilihat pada baris 2. Sehingga dapat dengan mudah diakses isinya. Didalam file tersebut terdapat variabel `TrainingSet` yang dapat dilihat pada baris 3 dan variabel `GroupTrain` pada baris 4. Kedua variabel tersebut adalah data latih dan labelnya yang digunakan sebagai masukan tahap klasifikasi ini.

```

1.  % me-load data latih (hasil haar wavelet)
2.  foo = load('dataset_logGabor.mat');
3.  TrainingData = foo.TrainingSet;
4.  trainLabel = foo.GroupTrain;
5.
6.  % train model
7.  model = cell(108,1);
8.  for k=1:108
9.      model{k} = svmtrain(double(trainLabel==k)
10.      TrainingData, '-c 1 -b 1 -q');
11.  end
12.
13. % prediksi kelas
14. probKelas = zeros(1,108);
15. for k=1:108
16.     [~,~,p] = svmpredict(~, templateTest, ...
17.     model{k}, '-b 1 -q');
18.     probKelas(:,k) = p(:,model{k}.Label==1);
19. end
20.
21. % penentuan kelas
22. [~,hasilKelas] = max(probKelas,[],2);
23.
24.

```

Kode Sumber 4.13 Implementasi SVM Fitur Log-Gabor

Sebelum melakukan klasifikasi, data latih dan labelnya terlebih dahulu diproses sebagai model untuk klasifikasi seperti yang terlihat pada baris 7 sampai 11. Pembuatan model ini bertujuan sebagai representasi tiap kelas. Sehingga akhirnya akan ada model sebanyak jumlah kelas.

Pada baris 14 sampai 23 dilakukan iterasi sebanyak jumlah model yang sudah dibentuk. Sebuah data uji `templateTest` akan dihitung kemiripannya pada setiap model dengan menggunakan probabilitas. Sehingga pada akhirnya terdapat matriks yang berisi probabilitas pada setiap model seperti yang ditunjukkan pada baris 18. Hasil yang dikeluarkan dari proses ini adalah indeks model dengan nilai terbesar dari matriks tersebut, yang juga dapat dikatakan hasil prediksi kelasnya.

#### 4.6.3 Implementasi Hamming Distance Fitur Log-Gabor

Kode Sumber 4.14 merupakan implementasi klasifikasi *Hamming distance*. Sebelum masuk kedalam proses perhitungan jarak, data citra iris sudah dilakukan ekstraksi fitur menggunakan *log-Gabor filter* dan disimpan dalam file `dataset_logGabor.mat` yang dapat dilihat pada baris 2. Didalam file tersebut terdapat variabel `TrainingSet` yang dapat dilihat pada baris 3, yang merupakan kumpulan data latih data citra.

Pada baris 5 terdapat variabel `templateTest` yang merupakan sebuah data uji untuk melakukan klasifikasi. Selanjutnya melakukan proses perhitungan jarak. Karena data latih berjumlah empat dalam setiap kelas, maka data uji akan dihitung jaraknya terhadap keempat data latih tersebut menggunakan fungsi `zHamming` seperti penjelasan subbab 2.10, kemudian dihitung rata-ratanya. Hal tersebut dilakukan iterasi sebanyak sejumlah kelas yang ditunjukkan pada baris 9 sampai 18. Hasil perhitungan rata-rata tersebut disimpan dalam variabel `avgDist` yang merupakan matriks. Hasil yang dikeluarkan dari proses ini adalah indeks kelas dengan nilai

terkecil dari matriks tersebut, yang juga dapat dikatakan hasil prediksi kelasnya.

```

1. % me-load data latih (hasil log-Gabor)
2. foo = load('dataset_logGabor.mat');
3. TempTrainingData = foo.TrainingSet;
4.
5. % variable masukan untuk test
6. [templateTest] = template;
7.
8. % perhitungan jarak
9. for h =1:108 % iterasi sejumlah kelas
10.     distance1 = zHamming(templateTest , ...
11.         TempTrainingData{1,1} );
12.     distance2 = zHamming(templateTest , ...
13.         TempTrainingData{2,1} );
14.     distance3 = zHamming(templateTest , ...
15.         TempTrainingData{3,1} );
16.     distance4 = zHamming(templateTest , ...
17.         TempTrainingData{4,1} );
18.
19. % penyimpanan matriks avg dari tiap kelas
20. avg=distance1+distance2+distance3+distance4;
21. avgDist = [avgDist , avg];
22. end
23.
24. % penentuan kelas
25. [~,hasilKelas] = min(avgDist,[],2);

```

Kode Sumber 4.14 Implementasi Hamming Distance Fitur Log-Gabor

#### 4.6.4 Implementasi Hamming Distance Fitur Wavelet

Subbab ini merupakan implementasi klasifikasi *Hamming distance*. Sebelum masuk kedalam proses perhitungan jarak, data citra iris sudah dilakukan ekstraksi fitur menggunakan *wavelet Haar* dan disimpan dalam file `dataset_wavelet.mat` yang ditunjukkan pada baris 2. Didalam file tersebut terdapat variabel `wavelet_all` yang dapat dilihat pada baris 3, yang merupakan kumpulan dataset seluruh citra



mata yang sudah dilakukan proses dekomposisi wavelet yang ditunjukkan pada Kode Sumber 4.15.

```

1. % normalisasi data wavelet
2. foo = load('dataset_wavelet.mat');
3. AllCitra = foo.wavelet_all;
4.
5. for i = 1:size(AllCitra,2)
6.     ALLCitra(:,i) = ,...
7.         ( AllCitra(:,i) - min(AllCitra(:,i))) ./
8.         ( max(AllCitra(:,i))-min(AllCitra(:,i)))
9. );
10. end
11.
12. AllCitra(AllCitra >= 0.5) = 1;
13. AllCitra(AllCitra < 0.5) = 0;
14.

```

Kode Sumber 4.15 Implementasi Proses Normalisasi Data

Proses normalisasi tersebut dilakukan dengan tujuan untuk mengubah nilai pada seluruh elemen matrik hasil dekomposisi *wavelet* agar bernilai antara rentang 0 sampai 1. Hasil akhirnya adalah keseluruhan data yang sudah dinormalisasi dan tersimpan pada variabel `AllCitra`. Agar menjadi sebuah matrik dengan elemen bilangan biner, dilakukan pengubahan jika intensitas matriks lebih dari 0.5, maka intensitasnya menjadi 1, sebaliknya jika kurang dari 0.5, intensitasnya menjadi 0, seperti yang ditunjukkan pada baris 11 sampai 12. Selanjutnya dengan pengopeasian biasa dilakukan pembagian data uji dan data latih dan disimpan dalam sebuah file ekstensi `mat`.

Setelah dibagi dan ditentukan data uji dan data testing, kemudian melakukan pencarian jarak dengan menggunakan Hamming distance seperti yang ditunjukkan pada Kode Sumber 4.16.

Masukan pada Kode Sumber 4.16 adalah sebuah file `dataset_wavelet_norm.mat` yang berisi data latih dan data uji dari proses normalisasi data dekomposisi *wavelet Haar*. Pada

baris 2, file tersebut di-*load* untuk dibaca dan diolah isi dalamnya.

```

1.  % me-load data latih (hasil log-Gabor)
2.  foo = load('dataset_wavelet_norm.mat');
3.  TempTrainingData = foo.TrainingSet;
4.
5.  % variable masukan untuk test
6.  [templateTest] = templateTest;
7.
8.  % perhitungan jarak
9.  for h =1:108      % iterasi sejumlah kelas
10.     distance1 = zHamming(templateTest , ...
        TempTrainingData{1,1} );
11.     distance2 = zHamming(templateTest , ...
        TempTrainingData{2,1} );
12.     distance3 = zHamming(templateTest , ...
        TempTrainingData{3,1} );
13.     distance4 = zHamming(templateTest , ...
        TempTrainingData{4,1} );
14.
15. % penyimpanan matriks avg dari tiap kelas
16. avg=distance1+distance2+distance3+distance4;
17. avgDist = [avgDist , avg];
18. end
19.
20. % penentuan kelas
21. [~,hasilKelas] = min(avgDist,[],2);
22.

```

Kode Sumber 4.16 Implementasi Hamming Distance Fitur Wavelet

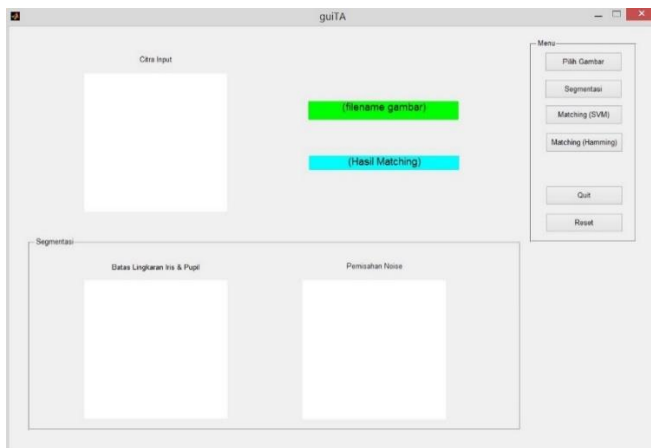
Pada baris 6 terdapat variabel `templateTest` yang merupakan sebuah data uji untuk melakukan klasifikasi. Selanjutnya melakukan proses perhitungan jarak. Karena data latih berjumlah empat dalam setiap kelas, maka data uji akan dihitung jaraknya terhadap keempat data latih tersebut menggunakan fungsi `zHamming` seperti penjelasan subbab 2.10, kemudian dihitung rata-ratanya. Hal tersebut dilakukan iterasi sebanyak sejumlah kelas yang ditunjukkan pada baris 9 sampai 18. Hasil perhitungan rata-rata tersebut disimpan dalam

variabel `avgDist`. Hasil yang dikeluarkan dari proses ini adalah indeks kelas dengan nilai terkecil dari matriks `avgDist` tersebut, yang juga dapat dikatakan hasil prediksi kelasnya.

#### 4.7 Implementasi Antarmuka

Bagian ini adalah bagian sistem yang memungkinkan user untuk berinteraksi dengan sistem pengenalan iris mata ini. Antarmuka didesain semudah dan sesederhana mungkin untuk digunakan. Gambar 4.1 menunjukkan desain antarmuka sistem pengenalan iris ini. Dalam rancangan antarmuka ini terdapat satu kotak gambar untuk input citra, dua kotak gambar untuk hasil segmentasi citra, dua kotak teks hasil inpuan, dan enam tombol pada bagian kanan sebagai menu utama.

Kotak gambar citra input akan menunjukkan gambar yang dipilih dari tombol menu `Pilih Gambar`. Nama dari gambar yang dipilih akan ditampilkan pada kotak teks `Filecitra`. Kemudian dua kotak gambar segmentasi akan menghasilkan hasil dari segmentasi mata dari tombol menu `Segmentasi`.



Gambar 4.1 Implementasi Antarmuka

Tombol menu *Matching (SVM)* merupakan tombol menu untuk melakukan pencocokan dengan menggunakan klasifikasi *Support Vector Machine*, sedangkan tombol menu *Matching (Hamming)* untuk melakukan pencocokan menggunakan klasifikasi *Hamming distance*. Hasilnya akan ditampilkan pada kotak teks Hasil klasifikasi.

Tombol menu *Quit* digunakan untuk keluar dari antarmuka dan tombol menu *Reset* digunakan untuk mendefaultkan ke tampilan awal.

## **BAB V**

### **UJI COBA DAN EVALUASI**

Pada bab ini akan dijelaskan hasil uji coba dan evaluasi program yang telah selesai diimplementasi.

#### **5.1 Lingkungan Uji Coba**

Lingkungan uji coba pengenalan iris ini mencakup perangkat lunak dan perangkat keras yang digunakan. Spesifikasi perangkat keras dan perangkat lunak yang digunakan dalam implementasi ini ditampilkan pada Tabel 5.1.

Tabel 5.1 Lingkungan Uji Coba Perangkat Lunak

Perangkat	Spesifikasi
Perangkat keras	Prosesor: Intel® Core™ i5-33170U CPU @ 1.70GHz 1.70GHz Memori: 4.00 GB
Perangkat lunak	Sistem Operasi: Microsoft Windows 8 64-bit Perangkat Pengembang: Matlab 8.3 & <i>Image Processing Toolbox</i> Perangkat Pembantu: Microsoft Excel 2013, <i>libsvm</i>

#### **5.2 Data Uji Coba**

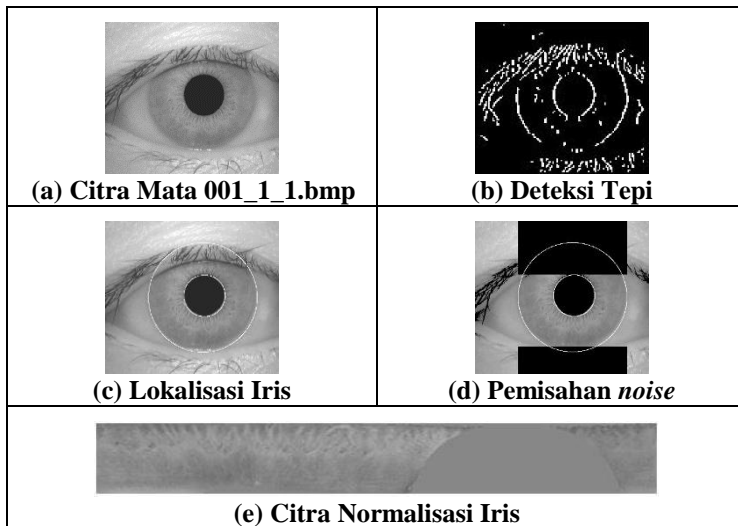
Data yang digunakan untuk uji coba implementasi pengenalan iris mata ini adalah data citra mata yang bersumber dari database citra mata *Chinese Academy of Science-Institute of Automation (CASIA)* versi 1.0. Pada database ini berisi 756 citra mata dengan 108 mata yang berbeda yang masing-masing diambil dua sesi dengan interval satu bulan pada sesinya. Sesi pertama diambil sebanyak 3 mata dan sesi kedua sebanyak 4 mata.

Contoh format nama file citra mata pada database CASIA adalah 001\_1\_1.bmp. Pada format file tersebut, 001 menunjukkan mata/kelas orang pertama, angka 1 dikanannya menunjukkan mata yang diambil pada sesi pertama, dan angka 1 yang terakhir menunjukkan mata yang diambil pada urutan ke-1.

Dalam melakukan uji coba, citra mata perlu dibagi ke dalam data uji dan data latih. Pengambilan citra mata pada sesi pertama akan dijadikan sebagai data uji, sedangkan pada sesi kedua akan dijadikan sebagai data latih. Sehingga jumlah data uji adalah  $3 \times 108 = 324$  buah dan jumlah data latih adalah  $4 \times 108 = 432$  buah.

### 5.3 Preprocessing

Pada tahap preprocessing, citra mata akan diolah untuk mendapatkan bagian irisnya, contoh sampel ditunjukkan pada Gambar 5.1.





Gambar 5.1 Sampel hasil praproses pada citra mata 001\_1\_1.bmp

Gambar 5.1 merupakan sampel gambar dari hasil *preprocessing* citra mata yang terdiri dari proses deteksi tepi, deteksi batas pupil dan iris, serta pemisahan bulu dan kelopak mata, juga proses normalisasi. Gambar (a) merupakan citra mata yang digunakan sebagai masukan, gambar (b) merupakan hasil dari proses deteksi tepi, gambar (c) adalah pengalokasian bagian iris dengan mencari batas lingkaran pupil dan iris, kemudian *noise* yang didalam bagian iris dipisahkan pada gambar (d), dan gambar (e) merupakan tahap normalisasi ukuran citra iris yang sudah tidak mempunyai *noise* di bagian iris.

#### 5.4 Ekstraksi Fitur

Pada tahap ekstraksi fitur, citra normalisasi akan diproses menggunakan dua buah metode, yakni *wavelet Haar* dan *log-gabor filter*. Sampel dari kedua metode tersebut dapat dilihat pada Tabel 5.2.

Tabel 5.2 Sampel fitur pada citra mata 001\_1\_1.bmp

<i>Wavelet Haar</i>	<i>Log-Gabor Filter</i>
	
<b>(a) Fitur Aproksimasi Haar</b>	<b>(b) Fitur biner</b>

Tabel 5.2 merupakan sampel gambar dari hasil ekstraksi fitur citra iris normalisasi. Gambar (a) merupakan fitur *wavelet Haar* bidang aproksimasi pada dekomposisi level 2, sedangkan gambar (b) merupakan fitur biner hasil dari *log-Gabor filter*.

#### 5.5 Skenario Uji Coba

Pada sub bab ini akan dijelaskan mengenai skenario uji coba yang telah dilakukan menggunakan data uji dan data latih. Melalui skenario ini, implementasi algoritma ekstraksi fitur dan klasifikasi akan diuji bagaimana performa pada masing-masing

skenario. Telah dilakukan beberapa skenario uji coba, diantaranya yaitu:

1. Perhitungan performa hasil akurasi menggunakan ekstraksi fitur *wavelet Haar* dan klasifikasi *Support Vector Machine* berdasarkan level dekomposisi *wavelet*. Level dekomposisi yang diujikan adalah 1, 2, 3, dan 4.
2. Perhitungan performa hasil akurasi menggunakan ekstraksi fitur *wavelet Haar* dan klasifikasi *Support Vector Machine* berdasarkan variasi nilai  $C$  (*penalty error term*) pada klasifikasi. Nilai  $C$  yang akan diuji yaitu 1, 10, 20, 30, dan 40.
3. Perhitungan performa hasil akurasi menggunakan ekstraksi fitur *wavelet Haar* dan klasifikasi *Support Vector Machine* berdasarkan variasi kernel yang digunakan pada klasifikasi. Kernel yang akan diuji yaitu linear, RBF, polynomial 2, dan polynomial 3.
4. Perhitungan performa hasil akurasi menggunakan ekstraksi fitur *log-Gabor filter* dan klasifikasi *Support Vector Machine* berdasarkan standar deviasi ekstraksi fitur. Besaran standar deviasi antara lain 0.2, 0.4, 0.6, dan 0.8.
5. Perhitungan performa hasil akurasi menggunakan ekstraksi fitur *log-Gabor filter* dan klasifikasi *Support Vector Machine* berdasarkan variasi nilai  $C$  (*penalty error term*) pada klasifikasi. Nilai  $C$  yang akan diuji yaitu 1, 10, 20, 30, dan 40.
6. Perhitungan performa hasil akurasi menggunakan ekstraksi fitur *log-Gabor filter* dan klasifikasi *Support Vector Machine* berdasarkan variasi kernel yang digunakan pada klasifikasi. Kernel yang akan diuji yaitu linear, RBF, polynomial 2, dan polynomial 3.
7. Perhitungan performa hasil akurasi menggunakan ekstraksi fitur *wavelet Haar* dan metode *Hamming distance* berdasarkan level dekomposisi *wavelet*. Level dekomposisi yang diujikan adalah 1, 2, 3, dan 4.



8. Perhitungan performa hasil akurasi menggunakan ekstraksi fitur *log-Gabor filter* dan metode *Hamming distance* berdasarkan standar deviasi ekstraksi fitur. Besaran standar deviasi antara lain 0.2, 0.4, 0.6, dan 0.8.

Perhitungan akurasi dilakukan dengan cara menghitung jumlah data uji yang benar sesuai dengan label kelas sesungguhnya kemudian dibagi dengan total data uji seperti yang ditunjukkan persamaan (5.1).

$$Akurasi = \frac{\sum \text{data uji benar}}{\text{jumlah data uji}} \quad (5.1)$$

### 5.5.1 Skenario Uji Coba 1

Skenario uji coba 1 adalah perhitungan akurasi dan waktu eksekusi. Uji coba ini dicoba dengan menggunakan beberapa level dekomposisi *wavetlet Haar* pada proses ekstraksi fitur dan SVM sebagai *classifier*. Level dekomposisi *wavelet* yang di uji coba yaitu 1, 2, 3 dan 4. Bidang aproksimasi akan digunakan sebagai fitur karena memuat energi atau informasi yang lebih banyak dari bidang horizontal, vertikal, ataupun diagonal.

Tabel 5.3 Persentase Akurasi dan Waktu Eksekusi masing-masing level dekomposisi *wavelet Haar* (SVM)

Level dekomposisi	Akurasi (%)	Waktu Eksekusi (s)
1	91.98	165.36
2	<b>91.98</b>	<b>39.82</b>
3	89.81	8.60
4	85.19	1.84

Setiap level akan menghasilkan jumlah fitur yang berbeda. Jumlah fitur pada level 1 menghasilkan 8192 fitur,

level 2 akan menghasilkan 2048 fitur, level 3 menghasilkan 512 fitur dan level 4 menghasilkan 128 fitur. Pada skenario 1, jenis *wavelet* yang digunakan adalah *Haar*, sedangkan parameter nilai  $C$  adalah 1 menggunakan kernel RBF. Hasil performa dan waktu eksekusi masing-masing dekomposisi dapat dilihat pada Tabel 5.3.

Berdasarkan hasil performa pada Tabel 5.3., dekomposisi *wavelet Haar* pada level 1 dan level 2 mempunyai akurasi tertinggi yaitu 91.98%, dibandingkan level 3 dan level 4 yang mencapai 89.81% dan 85.19% secara berturut-turut. Namun pada level dekomposisi level 1 dan level 2 memiliki hasil akurasi yang sama, tetapi waktu akurasi yang berbeda. Hal ini dikarenakan dimensi citra tereduksi setiap level dekomposisi yang mengakibatkan pula waktu komputasi menjadi lebih cepat saat masuk ke tahap klasifikasi. Dari uji coba tersebut didapatkan dekomposisi *wavelet Haar* level 2 sebagai yang terbaik. Hasil setiap data uji yang pada dekomposisi level 2 terdapat pada lampiran.

### 5.5.2 Skenario Uji Coba 2

Skenario uji coba 2 bertujuan menghitung nilai akurasi pada variasi nilai  $C$  pada klasifikasi, dimana nilai  $C$  adalah nilai *penalty error term*. Nilai  $C$  yang diuji yaitu 1, 10, 20, dan 40. Fitur yang digunakan adalah hasil dekomposisi *wavelet Haar* level 2 dan menggunakan kernel RBF sebagai klasifikasi.

Tabel 5.4 Persentase Akurasi fitur *wavelet* menggunakan variasi nilai  $C$  (SVM)

Level wavelet	$C$	Akurasi (%)	Waktu Eksekusi (s)
2	1	91.98	39.82
	10	91.98	38.24
	20	91.98	39.02
	<b>30</b>	<b>92.28</b>	<b>39.66</b>
	40	91.98	39.71

Pada Tabel 5.4 Hasil uji coba 2 hasil terbaik didapatkan ketika nilai  $C$  adalah 30, dimana memiliki akurasi yang lebih tinggi dari yang lainnya. Hasil setiap data uji dengan parameter terbaik tersebut terdapat pada lampiran.

### 5.5.3 Skenario Uji Coba 3

Pada skenario uji 3 merupakan uji coba menggunakan variasi kernel SVM untuk melihat persentase akurasi. Variasi kernel yang digunakan yaitu linear, *polynomial 2*, *polynomial 3*, dan RBF. Fitur yang digunakan adalah hasil dekomposisi *wavelet Haar* level 2 dan nilai  $C$  diberikan adalah 30.

Tabel 5.5 Persentase Akurasi fitur *wavelet* menggunakan variasi kernel SVM

Level wavelet	$C$	Kernel	Akurasi (%)	Waktu Eksekusi (s)
2	30	Linear	91.98	38.56
		Polynomial 2	91.98	39.89
		Polynomial 3	91.67	40.23
		<b>RBF</b>	<b>92.28</b>	<b>39.66</b>

Berdasarkan Tabel 5.5, hasil performa yang diperlihatkan diatas, akurasi tertinggi didapatkan menggunakan kernel RBF yaitu 92.28%. Hasil setiap data uji dengan parameter terbaik tersebut terdapat pada lampiran.

### 5.5.4 Skenario Uji Coba 4

Pada skenario uji coba 4 akan dilakukan percobaan untuk menghitung akurasi dan waktu eksekusi menggunakan fitur *log-Gabor filter* dan klasifikasi SVM. Pada uji coba ini akan dilakukan percobaan pada variasi nilai standar deviasi yang terdapat pada ekstraksi fitur *log-Gabor filter*. Standar deviasi yang akan digunakan adalah 0.2, 0.4, 0.6, dan 0.8. Hasil perhitungan performa tiap nilai standar deviasi dapat dilihat

pada Tabel 5.6. Pada skenario 1, parameter nilai  $C$  yang digunakan adalah 1 dan menggunakan kernel RBF.

Tabel 5.6 Persentase Akurasi dan Waktu Eksekusi fitur biner menggunakan variasi standar deviasi (SVM)

Standar deviasi	Akurasi (%)	Waktu Eksekusi (s)
<b>0.2</b>	<b>89.51</b>	<b>45.96</b>
0.4	81.48	46.77
0.6	73.15	45.90
0.8	55.86	47.45

Berdasarkan Tabel 5.6, hasil yang ditunjukkan oleh Tabel 5.6 nilai simpangan 0.2 memiliki hasil akurasi paling tinggi yaitu 89.51%. Hasil setiap data uji dengan parameter terbaik tersebut terdapat pada lampiran.

### 5.5.5 Skenario Uji Coba 5

Skenario uji coba 5 dilakukan dengan menghitung nilai akurasi jika menggunakan variasi nilai  $C$  pada klasifikasi SVM. Nilai  $C$  yang diuji yaitu 1, 10, 20, dan 40. Fitur yang digunakan adalah hasil *log-Gabor filter* pada saat standar deviasi 0.2 dan menggunakan kernel RBF sebagai klasifikasi.

Tabel 5.7 Persentase Akurasi fitur biner menggunakan variasi nilai  $C$  (SVM)

Standar deviasi	$C$	Akurasi (%)	Waktu Eksekusi (s)
0.2	<b>1</b>	<b>89.51</b>	<b>45.96</b>
	10	88.58	45.12
	20	88.27	46.31
	30	88.27	47.95
	40	88.27	45.34

Berdasarkan hasil yang ditunjukkan oleh Tabel 5.7, akurasi tertinggi didapat ketika nilai  $C$  adalah 1 yaitu 89.51%.

### 5.5.6 Skenario Uji Coba 6

Uji coba 6 adalah percobaan menggunakan variasi kernel SVM untuk menghitung persentase akurasi. Variasi kernel yang digunakan yaitu linear, *polynomial 2*, *polynomial 3*, dan RBF. Fitur yang digunakan adalah hasil *log-Gabor filter* dengan standar deviasi 0.2 dan nilai  $C$  diberikan adalah 1. Hasil perhitungan akurasi variasi kernel dapat dilihat pada Tabel 5.8.

Berdasarkan hasil yang ditunjukkan, kernel RBF memiliki hasil akurasi paling tinggi yaitu 89.51%.

Tabel 5.8 Persentase Akurasi fitur biner menggunakan variasi kernel SVM

Standar deviasi	$C$	Kernel	Akurasi (%)	Waktu Eksekusi (s)
0.2	1	Linear	87.96	47.12
		Polynomial 2	88.89	46.61
		Polynomial 3	89.20	45.89
		<b>RBF</b>	<b>89.51</b>	<b>45.96</b>

### 5.5.7 Skenario Uji Coba 7

Skenario uji coba 7 adalah perhitungan akurasi dan waktu eksekusi. Uji coba ini dicoba dengan menggunakan beberapa level dekomposisi *wavetlet Haar* pada proses ekstraksi fitur dan *Hamming distance* sebagai metode untuk pengklasifikasiannya. Level dekomposisi *wavelet* yang di uji coba yaitu 1, 2, 3 dan 4. Bidang aproksimasi akan digunakan sebagai fitur karena memuat energi atau informasi yang lebih banyak dari bidang horizontal, vertikal, ataupun diagonal. Hasil percobaan ditunjukkan pada Tabel 5.9.

Tabel 5.9 Persentase Akurasi dan Waktu Eksekusi masing-masing level dekomposisi *wavelet Haar* (*Hamming distance*)

Level dekomposisi	Akurasi (%)	Waktu Eksekusi (s)
1	<b>83.02</b>	<b>311.15</b>
2	78.09	76.74
3	75.00	17.88
4	64.81	2.52

Dari uji coba 7, hasil terbaik didapatkan pada dekomposisi *wavelet Haar* level 1, dimana memiliki akurasi yang lebih tinggi dari yang lainnya. Hasil setiap data uji dengan parameter terbaik tersebut terdapat pada lampiran.

### 5.5.8 Skenario Uji Coba 8

Pada uji skenario yang terakhir akan dilakukan percobaan untuk melihat akurasi dan waktu eksekusi menggunakan fitur *log-Gabor filter* dan metode *Hamming distance*. Pada uji coba ini akan dilakukan percobaan pada variasi nilai standar deviasi yang terdapat pada ekstraksi fitur *log-Gabor filter*. Standar deviasi yang akan digunakan adalah 0.2, 0.4, 0.6, dan 0.8. Hasil perhitungan performa tiap nilai standar deviasi dapat dilihat pada Tabel 5.10.

Tabel 5.10 Persentase Akurasi dan Waktu Eksekusi fitur biner menggunakan variasi standar deviasi (*Hamming distance*)

Standar deviasi	Akurasi (%)	Waktu Eksekusi (s)
<b>0.2</b>	<b>91.67</b>	<b>279.52</b>
0.4	86.73	280.32
0.6	82.41	278.93
0.8	76.23	281.84

Berdasarkan hasil yang ditunjukkan oleh Tabel 5.10 nilai simpangan 0.2 memiliki hasil akurasi paling tinggi yaitu 91.67%. Hasil setiap data uji dengan parameter terbaik tersebut terdapat pada lampiran.

## 5.6 Analisis Hasil Uji Coba

Berdasarkan skenario-skenario uji coba yang telah dilakukan, dapat diketahui bahwa tingkat ketepatan untuk mengenali iris mata sudah sangat baik. Uji coba 1 sampai uji coba 8 merupakan kombinasi uji coba untuk melihat performa algoritma ekstraksi fitur (*wavelet Haar* dan *log-Gabor filter*) dan algoritma klasifikasi (*support vector machine* dan *Hamming distance*) untuk melakukan pengenalan iris mata.

Dari hasil skenario uji coba yang telah dilakukan, beberapa parameter memberikan pengaruh terhadap hasil akurasi. Pada ekstraksi fitur *wavelet Haar*, parameter yang berpengaruh adalah tingkatan level dekomposisinya, sedangkan pada ekstraksi fitur *log-Gabor filter* parameter yang berpengaruh adalah standar deviasinya. Kemudian pada klasifikasi SVM, parameter yang berpengaruh adalah nilai  $C$ . Kemudian uji coba dilakukan dengan membandingkan akurasi.

Pada skenario ke-1 didapat bahwa level dekomposisi yang menghasilkan akurasi paling baik yaitu *wavelet Haar* level 2. Selanjutnya level dekomposisi tersebut akan diuji pada skenario ke-2 yang menggunakan variasi nilai  $C$  pada SVM untuk pengujiannya dan hasil akurasi menjadi naik saat nilai  $C$  bernilai 30. Skenario ke-3 dengan menggunakan level dekomposisi *wavelet Haar* level 2 dan nilai  $C=30$ , diuji coba menggunakan variasi kernel dan mendapatkan akurasi yang sama seperti skenario sebelumnya yaitu 92.28%.

Skenario ke-4 adalah uji coba dengan menggunakan parameter standar deviasi pada ekstraksi fitur *log-Gabor filter* dan menggunakan klasifikasi *Hamming distance*. Hasilnya baik dan mendapatkan akurasi sebesar 91.67%.

Skenario ke-5 merupakan uji coba dengan menggunakan variasi level dekomposisi *wavelet Haar* dan metode *Hamming distance* sebagai klasifikasi. Hasil terbaik yang didapatkan adalah menggunakan level dekomposisi pada level 1 yaitu 83.02%.

Para skenario ke-6 sampai dengan ke-8 merupakan uji coba menggunakan variasi standar deviasi pada ekstraksi *log-Gabor filter*, variasi nilai  $C$ , dan variasi kernel SVM. Hasil akurasi terbaik yang didapatkan adalah 89.51% dengan standar deviasi adalah 0.2, nilai  $C$  adalah 1, dan kernel RBF.

Perbandingan akurasi dua metode klasifikasi sudah terlihat cukup efektif untuk dapat melakukan pengenalan orang melalui iris mata. Perbandingan dua metode klasifikasi ini dapat dilihat pada Tabel 5.11.

Tabel 5.11 Perbandingan Akurasi Metode Klasifikasi

Ekstraksi Fitur	Klasifikasi	Akurasi (%)
<b>Wavelet Haar</b>	<b>SVM</b>	<b>92.28</b>
Wavelet Haar	Hamming distance	83.02
Log-Gabor Filter	SVM	89.51
<b>Log-Gabor Filter</b>	<b>Hamming distance</b>	<b>91.67</b>

Berdasarkan Tabel 5.11, klasifikasi *Support Vector Machine* dengan menggunakan *Wavelet Haar* dan pencocokan *Hamming distance* dengan menggunakan *log-Gabor filter* memberikan hasil yang cukup baik untuk mengenali iris.



## **BAB VI**

### **KESIMPULAN DAN SARAN**

Dalam bab ini diuraikan kesimpulan dan saran berdasarkan pembahasan yang telah dijabarkan dalam tugas akhir ini. Kesimpulan diambil dari proses dan uji coba dari hasil uji coba, juga saran untuk pengembangan perangkat lunak ini.

#### **6.1 Kesimpulan**

Dari hasil uji coba yang telah dilakukan, dapat diambil kesimpulan sebagai berikut :

1. Metode pengenalan iris mata menggunakan klasifikasi SVM dan klasifikasi *Hamming distance* memberikan hasil yang cukup baik pada keduanya
2. Penggunaan transformasi *wavelet Haar* pada bidang aproksimasi dan *log-Gabor filter* dalam pengekstraksian fitur iris cukup bagus untuk dijadikan sebagai fitur
3. Penggunaan klasifikasi SVM memberikan hasil lebih baik jika menggunakan ekstraksi fitur *wavelet Haar* daripada jika menggunakan fitur *log-Gabor filter*
4. Penggunaan pencocokan *Hamming distance* memberikan hasil lebih baik jika menggunakan ekstraksi fitur *log-Gabor filter* daripada jika menggunakan fitur *wavelet Haar*
5. Variasi kernel pada klasifikasi mempengaruhi hasil akurasi. Kernel yang menghasilkan akurasi terbaik adalah RBF
6. Dari hasil uji coba bisa disimpulkan bahwa akurasi tertinggi yaitu 92.28%, menggunakan ekstraksi fitur *wavelet Haar* dekomposisi level 2 dan menggunakan klasifikasi *Support Vector Machine* dengan nilai  $C = 30$  dengan kernel RBF.

## 6.2 Saran

Saran-saran untuk pengembangan pengembangan perangkat lunak ini lebih lanjut adalah sebagai berikut. :

1. Perhitungan klasifikasi atau pencocokan dengan metode yang lain, tidak hanya menggunakan klasifikasi *Support Vector Machine* atau perhitungan jarak *Hamming* saja
2. Pengembangan aplikasi ke dalam bidang iridologi untuk mendeteksi kelainan organ dalam
3. Mengembangkan sistem yang dapat mengenali iris berdasarkan warna dan tekstur.

Demikian saran yang mungkin menjadi masukan dalam pengembangan selanjutnya.

## DAFTAR PUSTAKA

- [1] A. D. Hartanto, R. Isnanto and A. Hidayatno, "Pengenalan Citra Iris Mata Menggunakan Alihragam Wavelet Daubechies Orde 4," *TRANSMISI*, pp. 145-149, 2010.
- [2] D. K. SRIVASTAVA and L. BHAMBHU, "DATA CLASSIFICATION USING SUPPORT VECTOR," *Journal of Theoretical and Applied Information Technology*, 2009.
- [3] F. D, "Relations between the statistics of natural images and the response properties of cortical cells," *Journal of the Optical Society of America*, vol. 4, pp. 2379-2394, 1987.
- [4] Y. P. Syamsuddin, Aplikasi Pengenalan Iris yang Mengalami Deformasi dengan Metode Bayesian, Surabaya: Buku Tugas Akhir Teknik Informatika FtIf ITS, 2009.
- [5] R. P. Moreno and A. Gonzaga, "Features Vector For Personal Identification Based On Iris Texture," *Departamento de Engenharia Elétrica - EESC – USP*, 2009.
- [6] B. Chouhan and S. Shukla, "Analysis of statistical feature extraction for Iris Recognition System using Laplacian of Gaussian filter," *INTERNATIONAL JOURNAL OF APPLIED ENGINEERING RESEARCH, DINDIGUL*, 2010.
- [7] Hasimah and M. J. Salami, "Iris Recognition System Using Support Vector Machines," *InTech*, 2011.
- [8] H. Rai and A. Yadav, "Iris Recognition using Combined Support Vector Machine and Hamming Distance Approach," *Expert Sytem with Applications*, pp. 588-593, 2013.

- [9] "Chinese Academy of Sciences(CASIA)," National Laboratory of Pattern Recognition (NLPR), February 2003. [Online]. Available: <http://biometrics.idealtest.org/>.
- [10] R. C. Gonzalez and R. E. Woods, Digital Image Processing, New Jersey: Prentice Hall, 2002.
- [11] M. R. Faundra, Aplikasi Filter Log-Gabor pada Sistem Pengenalan Iris Mata, Surabaya: Buku Tugas Akhir Fakultas Matematika dan Ilmu Pengetahuan Alam ITS , 2011.
- [12] L. Masek, Recognition of Human Iris Patterns for Biometric Identification, The University of Western Australia, 2003.
- [13] H. Studiawan, Teknik Pengenalan Iris berbasis Wavelet dan Special Gabor Filter, Surabaya: Buku Tugas Akhir Teknik Informatika FTIf ITS, 2009.
- [14] A. Wicaksono, R. R. Isnanto and A. Hidayatno, "Analisis Transformasi Balik Citra Iris Menggunakan Wavelet Haar Berdasarkan Faktor Retensi Koefisien Wavelet," *TRANSMISI*, p. 130, 2012.
- [15] C.-J. L. Chih-Chung Chang, "A Library for Support Vector Machines," Taipei, Taiwan, 2001.
- [16] X. Hou, "Support Vector Machine," github, 25 April 2015. [Online]. Available: <http://houxianxu.github.io/support-vector-machine/>. [Accessed 23 May 2016].
- [17] V. Vapnik, The Nature of Statistical Learning Theory, NY: Springer-Verlag, 1995.

## LAMPIRAN A

### TABEL UJI COBA

Tabel A.1 Hasil Uji Coba Data Uji Citra Iris Fitur *Wavelet Haar* Dekomposisi level 2 dan Klasifikasi SVM dengan  $C = 1$  dan kernel RBF

No Subyek	UJI CASIA IRIS DATABASE V.1	
	Data Uji	Data Uji Benar
1	3	3
2	3	2
3	3	3
4	3	3
5	3	3
6	3	2
7	3	3
8	3	3
9	3	3
10	3	3
11	3	2
12	3	3
13	3	3
14	3	3
15	3	3
16	3	3
17	3	3
18	3	3
19	3	1
20	3	3
21	3	3
22	3	3

23	3	3
24	3	3
25	3	3
26	3	3
27	3	3
28	3	3
29	3	3
30	3	3
31	3	3
32	3	3
33	3	3
34	3	3
35	3	3
36	3	3
37	3	3
38	3	3
39	3	3
40	3	3
41	3	2
42	3	3
43	3	3
44	3	3
45	3	2
46	3	2
47	3	3
48	3	2
49	3	3
50	3	3
51	3	3
52	3	3

53	3	3
54	3	3
55	3	3
56	3	2
57	3	3
58	3	1
59	3	3
60	3	3
61	3	3
62	3	3
63	3	3
64	3	3
65	3	3
66	3	2
67	3	2
68	3	3
69	3	3
70	3	3
71	3	3
72	3	3
73	3	2
74	3	3
75	3	3
76	3	3
77	3	3
78	3	3
79	3	2
80	3	3
81	3	2
82	3	3

83	3	3
84	3	3
85	3	3
86	3	2
87	3	3
88	3	3
89	3	3
90	3	3
91	3	2
92	3	2
93	3	3
94	3	2
95	3	3
96	3	3
97	3	3
98	3	3
99	3	3
100	3	3
101	3	3
102	3	3
103	3	3
104	3	0
105	3	2
106	3	2
107	3	3
108	3	3
<b>JUMLAH</b>	<b>324</b>	<b>298</b>
<b>PERSENTASE</b>		<b>91.98%</b>



Tabel A.2 Hasil Uji Coba Data Uji Citra Iris Fitur *Wavelet Haar* Dekomposisi level 2 dan Klasifikasi SVM di  $C = 30$  dan kernel RBF

No Subyek	UJI CASIA IRIS DATABASE V.1	
	Data Uji	Data Uji Benar
1	3	3
2	3	2
3	3	3
4	3	3
5	3	3
6	3	2
7	3	3
8	3	3
9	3	3
10	3	3
11	3	2
12	3	3
13	3	3
14	3	3
15	3	3
16	3	3
17	3	3
18	3	3
19	3	1
20	3	3
21	3	3
22	3	3
23	3	3
24	3	3

25	3	3
26	3	3
27	3	3
28	3	3
29	3	3
30	3	3
31	3	3
32	3	3
33	3	3
34	3	3
35	3	3
36	3	3
37	3	3
38	3	3
39	3	3
40	3	3
41	3	2
42	3	3
43	3	3
44	3	3
45	3	2
46	3	2
47	3	3
48	3	2
49	3	3
50	3	3
51	3	3
52	3	3
53	3	3
54	3	3

55	3	3
56	3	2
57	3	3
58	3	1
59	3	3
60	3	3
61	3	3
62	3	3
63	3	3
64	3	3
65	3	3
66	3	2
67	3	2
68	3	3
69	3	3
70	3	3
71	3	3
72	3	3
73	3	2
74	3	3
75	3	3
76	3	3
77	3	3
78	3	3
79	3	2
80	3	3
81	3	2
82	3	3
83	3	3
84	3	3

85	3	3
86	3	3
87	3	3
88	3	3
89	3	3
90	3	3
91	3	2
92	3	2
93	3	3
94	3	2
95	3	3
96	3	3
97	3	3
98	3	3
99	3	3
100	3	3
101	3	3
102	3	3
103	3	3
104	3	0
105	3	2
106	3	2
107	3	3
108	3	3
<b>JUMLAH</b>	<b>324</b>	<b>299</b>
<b>PERSENTASE</b>		<b>92.28%</b>

Tabel A.3 Hasil Uji Coba Data Uji Citra Iris Fitur *Wavelet Haar* Dekomposisi level 2 dan Klasifikasi SVM pada  $C = 30$  dan kernel RBF

No Subyek	UJI CASIA IRIS DATABASE V.1	
	Data Uji	Data Uji Benar
1	3	3
2	3	2
3	3	3
4	3	3
5	3	3
6	3	2
7	3	3
8	3	3
9	3	3
10	3	3
11	3	2
12	3	3
13	3	3
14	3	3
15	3	3
16	3	3
17	3	3
18	3	3
19	3	1
20	3	3
21	3	3
22	3	3
23	3	3
24	3	3
25	3	3

26	3	3
27	3	3
28	3	3
29	3	3
30	3	3
31	3	3
32	3	3
33	3	3
34	3	3
35	3	3
36	3	3
37	3	3
38	3	3
39	3	3
40	3	3
41	3	2
42	3	3
43	3	3
44	3	3
45	3	2
46	3	2
47	3	3
48	3	2
49	3	3
50	3	3
51	3	3
52	3	3
53	3	3
54	3	3
55	3	3

56	3	2
57	3	3
58	3	1
59	3	3
60	3	3
61	3	3
62	3	3
63	3	3
64	3	3
65	3	3
66	3	2
67	3	2
68	3	3
69	3	3
70	3	3
71	3	3
72	3	3
73	3	2
74	3	3
75	3	3
76	3	3
77	3	3
78	3	3
79	3	2
80	3	3
81	3	2
82	3	3
83	3	3
84	3	3
85	3	3

86	3	3
87	3	3
88	3	3
89	3	3
90	3	3
91	3	2
92	3	2
93	3	3
94	3	2
95	3	3
96	3	3
97	3	3
98	3	3
99	3	3
100	3	3
101	3	3
102	3	3
103	3	3
104	3	0
105	3	2
106	3	2
107	3	3
108	3	3
<b>JUMLAH</b>	<b>324</b>	<b>299</b>
<b>PERSENTASE</b>		<b>92.28%</b>



Tabel A.4 Hasil Uji Coba Data Uji Citra Iris Fitur *Log-Gabor Filter* standar deviasi 0.2 dan Pencocokan *Hamming distance*

No Subyek	UJI CASIA IRIS DATABASE V.1	
	Data Uji	Data Uji Benar
1	3	3
2	3	3
3	3	3
4	3	3
5	3	3
6	3	3
7	3	3
8	3	3
9	3	2
10	3	3
11	3	3
12	3	3
13	3	3
14	3	3
15	3	3
16	3	3
17	3	3
18	3	2
19	3	3
20	3	3
21	3	3
22	3	3
23	3	3
24	3	0
25	3	2
26	3	3

27	3	3
28	3	3
29	3	3
30	3	2
31	3	3
32	3	3
33	3	3
34	3	3
35	3	3
36	3	3
37	3	3
38	3	3
39	3	3
40	3	3
41	3	3
42	3	3
43	3	2
44	3	3
45	3	2
46	3	2
47	3	3
48	3	3
49	3	3
50	3	2
51	3	3
52	3	3
53	3	3
54	3	3
55	3	3
56	3	3

57	3	3
58	3	1
59	3	3
60	3	3
61	3	3
62	3	3
63	3	3
64	3	3
65	3	3
66	3	3
67	3	2
68	3	3
69	3	3
70	3	3
71	3	3
72	3	3
73	3	2
74	3	3
75	3	3
76	3	3
77	3	3
78	3	3
79	3	2
80	3	3
81	3	2
82	3	3
83	3	3
84	3	3
85	3	2
86	3	3

87	3	3
88	3	2
89	3	3
90	3	3
91	3	2
92	3	2
93	3	3
94	3	2
95	3	3
96	3	3
97	3	3
98	3	3
99	3	3
100	3	3
101	3	3
102	3	2
103	3	3
104	3	2
105	3	1
106	3	2
107	3	3
108	3	3
<b>JUMLAH</b>	<b>324</b>	<b>297</b>
<b>PERSENTASE</b>		<b>91.67%</b>

Tabel A.5 Hasil Uji Coba Data Uji Citra Iris Fitur *Wavelet Haar* Dekomposisi level 1 dan Pencocokan *Hamming distance*

No Subyek	UJI CASIA IRIS DATABASE V.1	
	Data Uji	Data Uji Benar
1	3	3
2	3	2
3	3	3
4	3	3
5	3	3
6	3	3
7	3	2
8	3	3
9	3	3
10	3	3
11	3	3
12	3	3
13	3	2
14	3	3
15	3	2
16	3	3
17	3	3
18	3	2
19	3	1
20	3	3
21	3	3
22	3	3
23	3	3
24	3	3
25	3	3
26	3	3

27	3	3
28	3	3
29	3	3
30	3	2
31	3	3
32	3	3
33	3	3
34	3	3
35	3	3
36	3	3
37	3	3
38	3	3
39	3	3
40	3	3
41	3	2
42	3	2
43	3	3
44	3	3
45	3	2
46	3	2
47	3	3
48	3	1
49	3	1
50	3	3
51	3	3
52	3	1
53	3	3
54	3	3
55	3	0
56	3	3

57	3	3
58	3	1
59	3	3
60	3	3
61	3	2
62	3	3
63	3	3
64	3	3
65	3	3
66	3	2
67	3	2
68	3	0
69	3	3
70	3	0
71	3	3
72	3	3
73	3	2
74	3	3
75	3	3
76	3	3
77	3	3
78	3	0
79	3	1
80	3	3
81	3	2
82	3	3
83	3	3
84	3	3
85	3	3
86	3	2

87	3	3
88	3	2
89	3	3
90	3	3
91	3	1
92	3	2
93	3	3
94	3	3
95	3	2
96	3	3
97	3	3
98	3	3
99	3	3
100	3	3
101	3	2
102	3	3
103	3	2
104	3	0
105	3	2
106	3	2
107	3	0
108	3	3
<b>JUMLAH</b>	<b>324</b>	<b>269</b>
<b>PERSentase</b>		<b>83.02%</b>



Tabel A.6 Hasil Uji Coba Data Uji Citra Iris Fitur *Log-Gabor Filter* standar deviasi 0.2 dan Klasifikasi SVM pada  $C = 1$  dan kernel RBF

No Subyek	UJI CASIA IRIS DATABASE V.1	
	Data Uji	Data Uji Benar
1	3	3
2	3	2
3	3	3
4	3	3
5	3	3
6	3	2
7	3	3
8	3	3
9	3	3
10	3	3
11	3	3
12	3	3
13	3	3
14	3	3
15	3	3
16	3	2
17	3	3
18	3	2
19	3	1
20	3	3
21	3	3
22	3	3
23	3	3
24	3	2
25	3	1

26	3	3
27	3	3
28	3	3
29	3	3
30	3	3
31	3	3
32	3	3
33	3	3
34	3	3
35	3	3
36	3	3
37	3	3
38	3	2
39	3	3
40	3	3
41	3	2
42	3	3
43	3	3
44	3	3
45	3	2
46	3	3
47	3	3
48	3	3
49	3	3
50	3	2
51	3	3
52	3	3
53	3	3
54	3	3
55	3	3

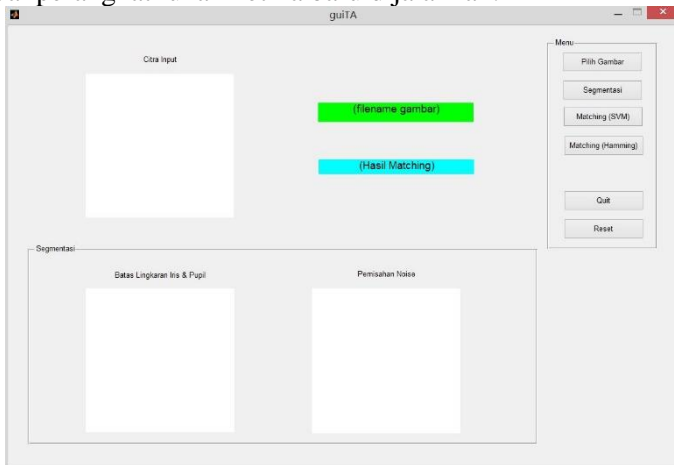
56	3	3
57	3	3
58	3	1
59	3	3
60	3	3
61	3	3
62	3	3
63	3	3
64	3	3
65	3	3
66	3	3
67	3	2
68	3	3
69	3	3
70	3	3
71	3	3
72	3	3
73	3	2
74	3	3
75	3	3
76	3	3
77	3	3
78	3	3
79	3	2
80	3	3
81	3	2
82	3	3
83	3	3
84	3	3
85	3	2

86	3	3
87	3	3
88	3	2
89	3	3
90	3	2
91	3	0
92	3	2
93	3	3
94	3	2
95	3	3
96	3	3
97	3	3
98	3	3
99	3	3
100	3	3
101	3	0
102	3	2
103	3	3
104	3	3
105	3	1
106	3	2
107	3	3
108	3	3
<b>JUMLAH</b>	<b>324</b>	<b>290</b>
<b>PERSENTASE</b>		<b>89.51%</b>

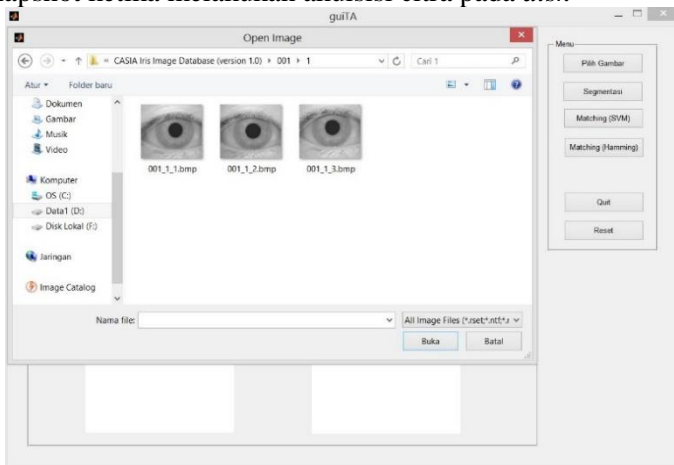
## LAMPIRAN B

### SNAPSHOT PERANGKAT LUNAK

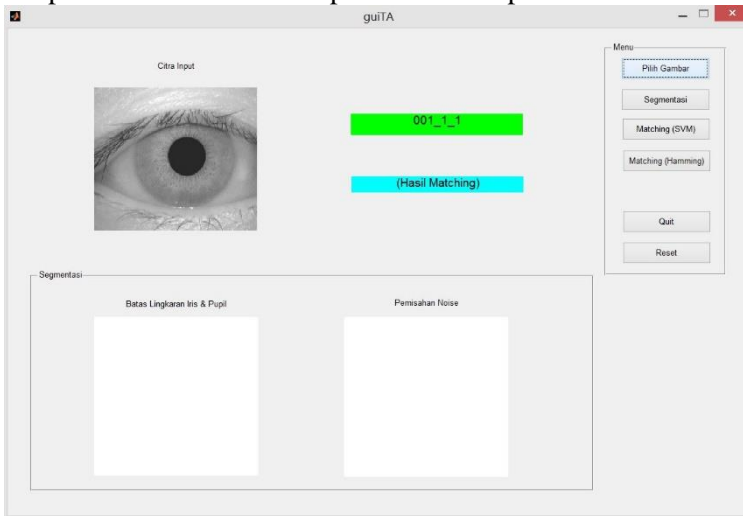
Pada lampiran ini akan ditampilkan snapshot perangkat lunak yang sudah diimplementasikan. Berikut adalah snapshot awal perangkat lunak ketika baru dijalankan.



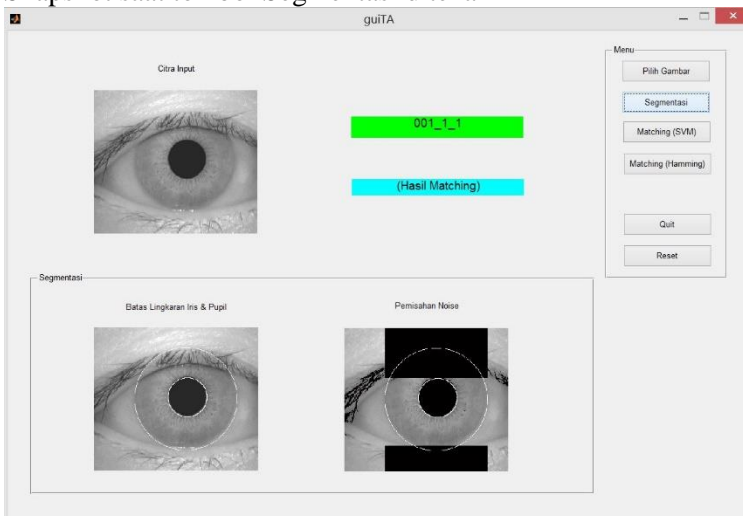
Snapshot ketika melakukan akuisisi citra pada *disk*



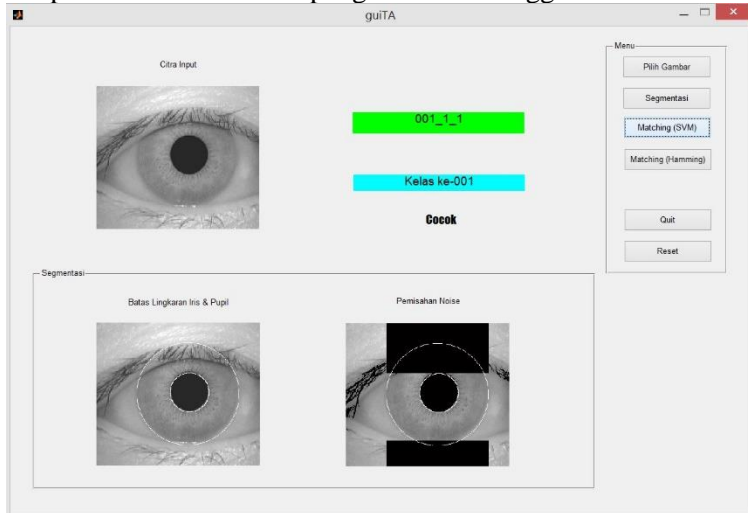
Snapshot saat citra sudah dipilih dan ditampilkan



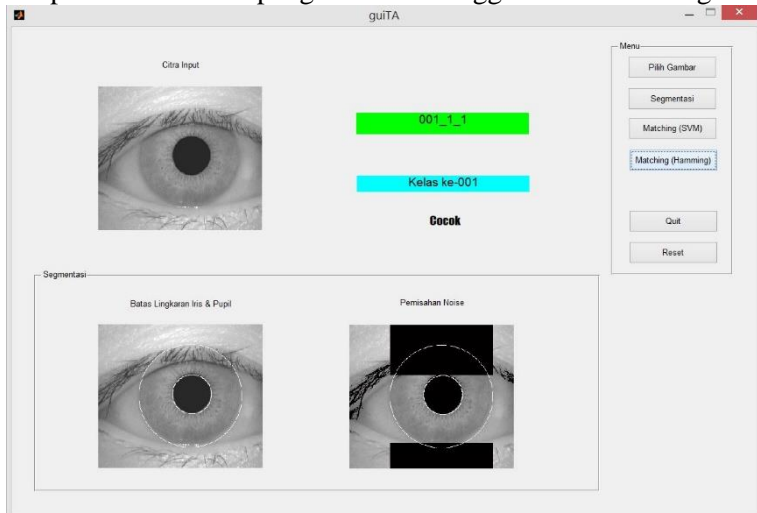
Snapshot saat tombol Segmentasi ditekan



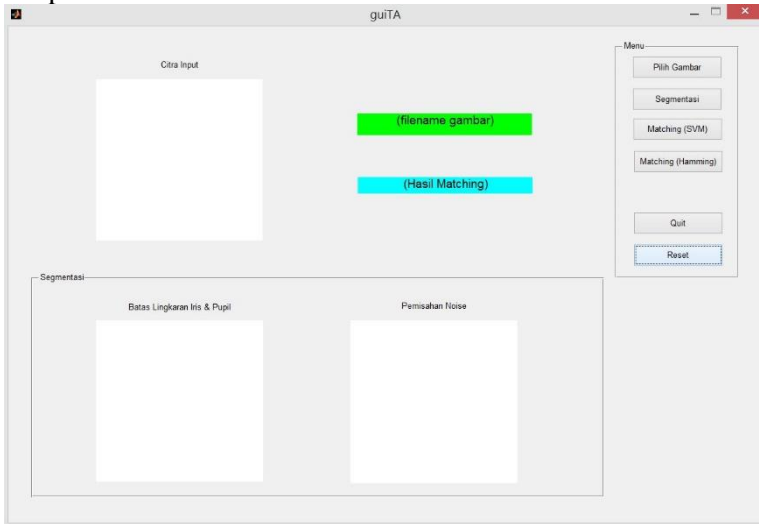
## Snapshot saat melakukan pengklasifian menggunakan SVM



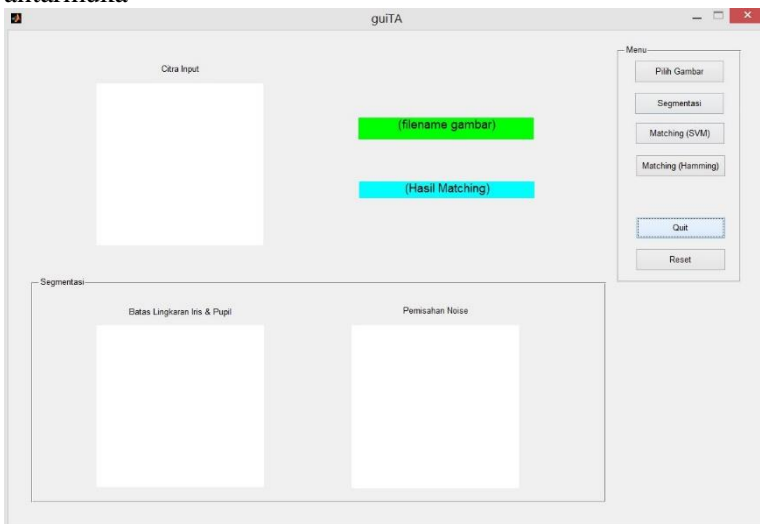
## Snapshot melakukan pengklasifian menggunakan Hamming



## Snapshot ketika menekan tombol Reset



## Snapshot saat hendak menekan tombol Quit dan keluar dari antarmuka





## BIODATA PENULIS



Lahir pada tahun 1994 di pinggiran kota Bandung, tepatnya pada 13 Agustus penulis memulai awal kehidupan di dunia. Penulis adalah anak ketiga dari tiga bersaudara dari pasangan suami istri Bapak AMA Suyanto dan Ibu Restu Kemala. Penulis bertempat tinggal di Komplek Ujungberung Indah blok 17 no 3 Kecamatan Cigending Kota Bandung.

Penulis menempuh pendidikan formalnya di SD Islam Zakaria Bandung (2000-2006), SMP N 44 Bandung (2006-2009), dan SMA Darul Hikam Bandung (2009-2012). Setelah lulus SMA penulis melanjutkan ke jenjang perkuliahan di Jurusan Teknik Informatika, Fakultas Teknologi Informasi, Institut Teknologi Sepuluh Nopember Surabaya. Bidang Studi yang diambil oleh penulis pada saat kuliah di Teknik Informatika ITS adalah Komputasi Cerdas dan Visualisasi.

Penulis yang mempunyai nama lengkap Afdhal Basith Anugrah mempunyai hobi membaca. Dengan nilai kesederhanaan, penulis ingin berbagi pengalaman dan ilmu pengetahuan dengan pembaca yang berminat pada penulis atau hal lain dengan menghubungi melalui *email* [afdhalbasith@gmail.com](mailto:afdhalbasith@gmail.com) karena pengalaman dan ilmu pengetahuan dapat menjadikan seseorang lebih maju dari yang sekarang.

Semoga para pembaca dapat memberikan saran dan kritik yang membangun penulis agar bisa menjadi pribadi yang unik, tahan banting, dan bersahaja.