taW

;/--n

**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**

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

**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**

****

***[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**

# LEMBAR PENGESAHAN

**IMPLEMENTASI PENGENALAN IRIS MATA MENGGUNAKAN 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)

1. Dr.Eng. Chastine Fatichah, S.Kom, M.Kom .....................

(NIP *197512202001122002*) (Pembimbing 2)

**SURABAYA**

**JUNI, 2016**

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

**IMPLEMENTASI PENGENALAN IRIS MATA MENGGUNAKAN METODE SUPPORT VECTOR MACHINE 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.*

*Dalam Tugas Akhir ini mengimplementasikan metode ekstraksi fitur Wavelet dan Log-Gabor Filter. Kemudian dilanjutkan dengan metode klasifikasi Support Vector Machines dan Hamming Distance. Sebelum dilakukan ekstraksi fitur, terlebih dahulu dilakukan praproses untuk identifikasi bagian iris mata dan melakukan proses normalisasi.*

*Hasil uji coba pada dataset mata CASIA dengan menggunakan klasifikasi SVM dan Hamming distance terbukti cukup efektif dalam proses pengenalan iris. Pada proses menggunakan metode klasifikasi SVM saja menghasilkan akurasi 94.44%, dengan klasifikasi Hamming distance saja menghasilkan akurasi 94.14%, dan menghasilkan 98.45% dengan menggunakan kombinasi dari SVM dan Hamming distance.*

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

***[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** | **:** | **Teknik Informatika 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.*

*In this final project implements iris recognition using Wavelet and Log-Gabor Filter as feature extraction. Then continued with Support Vector Machines and Hamming Distance as the classification method. Before feature extraction, first performed preprocessing for part identification iris of the eye and the process of normalization.*

*The test result with CASIA eye dataset using SVM classification and Hamming distance proved to be quite effective in the process of iris recognition. In the process using SVM classification method only achieved accuracy 94.44%, with a Hamming distance classification accuracy alone achieved 94.14%, and achieved 98.45% by using a combination of SVM and Hamming distance*.

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

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

# KATA PENGANTAR



Segala puji syukur ditujukan kepada Allah SWT yang telah melimpahkan rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan Tugas Akhir yang berjudul

**IMPLEMENTASI PENGENALAN IRIS MATA MENGGUNAKAN 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 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](#_Toc441189682)

[*Abstrak* vii](#_Toc441189683)

[*Abstract* ix](#_Toc441189684)

[KATA PENGANTAR xi](#_Toc441189685)

[DAFTAR ISI xiii](#_Toc441189686)

[DAFTAR GAMBAR xvii](#_Toc441189687)

[DAFTAR TABEL xix](#_Toc441189688)

[DAFTAR KODE SUMBER xxi](#_Toc441189689)

[BAB I PENDAHULUAN 1](#_Toc441189690)

[1.1 Latar Belakang 1](#_Toc441189691)

[1.2 Rumusan Masalah 2](#_Toc441189692)

[1.3 Batasan Masalah 2](#_Toc441189693)

[1.4 Tujuan 2](#_Toc441189694)

[1.5 Manfaat 3](#_Toc441189695)

[1.6 Metodologi 3](#_Toc441189696)

[BAB II TINJAUAN PUSTAKA 5](#_Toc441189697)

[2.1 Gaussian Mixture Models 5](#_Toc441189698)

[2.2 Gaussian Pyramid 7](#_Toc441189699)

[2.3 Probabilitas Distribusi Gaussian 8](#_Toc441189700)

[2.4 Region Growing 9](#_Toc441189701)

[2.5 Daubachies 4 Wavelet 10](#_Toc441189702)

[2.6 Normalisasi Min-Max 13](#_Toc441189703)

[2.7 Support Vector Machines 14](#_Toc441189704)

[BAB III DESAIN PERANGKAT LUNAK 19](#_Toc441189705)

[3.1 Data 19](#_Toc441189706)

[3.1.1 Data Masukan 19](#_Toc441189707)

[3.1.2 Data Pembelajaran 19](#_Toc441189708)

[3.1.3 Data Keluaran 20](#_Toc441189709)

[3.2 Desain Sistem Secara Umum 21](#_Toc441189710)

[3.3 Preprocessing 22](#_Toc441189711)

[3.3.1 Reduksi Size Frame 23](#_Toc441189712)

[3.3.2 Deteksi Gerak 25](#_Toc441189713)

[3.3.3 Deteksi Warna Piksel 25](#_Toc441189714)

[3.3.4 Region Growing 27](#_Toc441189715)

[3.3.5 Perhitungan Luasan Region 28](#_Toc441189716)

[3.4 Verifikasi 28](#_Toc441189717)

[3.4.1 Ekstraksi Fitur dengan Wavelet 30](#_Toc441189718)

[3.4.2 Klasifikasi 31](#_Toc441189719)

[3.5 Menandai Region Api 32](#_Toc441189720)

[BAB IV IMPLEMENTASI 35](#_Toc441189721)

[4.1 Lingkungan Implementasi 35](#_Toc441189722)

[4.2 Implementasi 35](#_Toc441189723)

[4.2.1 Implementasi Tahap Reduksi Size Frame 35](#_Toc441189724)

[4.2.2 Implementasi Tahap Deteksi Gerak 36](#_Toc441189725)

[4.2.3 Implementasi Tahap Deteksi Warna Piksel 37](#_Toc441189726)

[4.2.4 Implementasi Tahap Region Growing 41](#_Toc441189727)

[4.2.5 Implementasi Tahap Perhitungan Luasan Region 44](#_Toc441189728)

[4.2.6 Implementasi Tahap Ekstraksi Fitur dengan Wavelet 45](#_Toc441189729)

[4.2.7 Implementasi Tahap Klasifikasi 45](#_Toc441189730)

[4.2.8 Implementasi Tahap Menandai Region Api 47](#_Toc441189731)

[BAB V UJI COBA DAN EVALUASI 49](#_Toc441189732)

[5.1 Lingkungan Uji Coba 49](#_Toc441189733)

[5.2 Data Uji Coba 49](#_Toc441189734)

[5.3 Alur Uji Coba 49](#_Toc441189735)

[5.3.1 Preprocessing 50](#_Toc441189736)

[5.3.2 Verifikasi 52](#_Toc441189737)

[5.4 Skenario Uji Coba 54](#_Toc441189738)

[5.4.1 Skenario Uji Coba 1 55](#_Toc441189739)

[5.4.2 Skenario Uji Coba 2 56](#_Toc441189740)

[5.4.3 Skenario Uji Coba 3 56](#_Toc441189741)

[5.4.4 Skenario Uji Coba 4 57](#_Toc441189742)

[5.4.5 Skenario Uji Coba 5 57](#_Toc441189743)

[5.4.6 Skenario Uji Coba 6 58](#_Toc441189744)

[5.5 Analisis Hasil Uji Coba 58](#_Toc441189745)

[BAB VI KESIMPULAN DAN SARAN 61](#_Toc441189746)

[6.1 Kesimpulan 61](#_Toc441189747)

[6.2 Saran 62](#_Toc441189748)

[DAFTAR PUSTAKA 63](#_Toc441189749)

[LAMPIRAN A 65](#_Toc441189750)

[BIODATA PENULIS 107](#_Toc441189751)

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

# DAFTAR GAMBAR

[Gambar 2.1 Ilustrasi *Gaussian Pyramid* [3] 7](#_Toc441189752)

[Gambar 2.2 *Seed* Awal *Region Growing* 9](#_Toc441189753)

[Gambar 2.3 Piksel yang diamati dan *Region* 10](#_Toc441189754)

[Gambar 2.4 Ilustrasi *Low Pass Filter* 12](#_Toc441189755)

[Gambar 2.5 Ilustrasi *Filter* yang dilakukan 12](#_Toc441189756)

[Gambar 2.6 Citra Masukan 13](#_Toc441189757)

[Gambar 2.7 Citra Keluaran Hasil *Wavelet* 13](#_Toc441189758)

[Gambar 2.8 Ilustrasi *Support Vector Machines* 15](#_Toc441189759)

[Gambar 3.1 Contoh Data Masukan 20](#_Toc441189760)

[Gambar 3.2 Contoh Data Keluaran 20](#_Toc441189761)

[Gambar 3.3 Diagram Alir Rancangan Perangkat Lunak Secara Umum 21](#_Toc441189762)

[Gambar 3.4 Digram Alir Reduksi *Size Frame* 23](#_Toc441189763)

[Gambar 3.5 Diagram Alir *Preprocessing* 24](#_Toc441189764)

[Gambar 3.6 Contoh Deteksi Gerak 25](#_Toc441189765)

[Gambar 3.7 Contoh *Dataset* Gambar Api 26](#_Toc441189766)

[Gambar 3.8 *Pseudocode* Deteksi Warna Piksel 26](#_Toc441189767)

[Gambar 3.9 *Pseudocode Regon Growing* 27](#_Toc441189768)

[Gambar 3.10 Diagram Alir Perhitungan Luasan *Region* 29](#_Toc441189769)

[Gambar 3.11 Diagram Alir Verifikasi 30](#_Toc441189770)

[Gambar 3.12 Diagram Alir Ekstraksi Fitur 32](#_Toc441189771)

[Gambar 3.13 *Pseudocode* fungsi menandai *region* 33](#_Toc441189772)

[Gambar 3.14 Menandai *region* api 33](#_Toc441189773)

[Gambar 5.1 Contoh Video Kejadian 50](#_Toc441189774)

[Gambar 5.2 Tahap *Preprocessing* 52](#_Toc441189775)

[Gambar 5.3 Tahap Verifikasi 54](#_Toc441189776)

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

# DAFTAR TABEL

[Tabel 4.1 Lingkungan Perancangan Perangkat Lunak 35](#_Toc441189777)

[Tabel 5.1 Hasil Uji Coba 1 55](#_Toc441189778)

[Tabel 5.2 Hasil Uji Coba 2 56](#_Toc441189779)

[Tabel 5.3 Hasil Uji Coba 3 57](#_Toc441189780)

[Tabel 5.4 Hasil Uji Coba 4 57](#_Toc441189781)

[Tabel 5.5 Hasil Uji Coba 5 58](#_Toc441189782)

[Tabel 5.6 Hasil Uji Coba 6 58](#_Toc441189783)

[Tabel A.1 Hasil Uji Coba Menggunakan Parameter *Threshold*  = 10-7, = 5 dan kernel RBF 65](#_Toc441189784)

[Tabel A.2 Hasil Uji Coba Menggunakan Parameter *Threshold*  = 10-8, = 5 dan kernel RBF 68](#_Toc441189785)

[Tabel A.3 Hasil Uji Coba Menggunakan Parameter *Threshold*  = 5x10-9, = 5 dan kernel RBF 71](#_Toc441189786)

[Tabel A.4 Hasil Uji Coba Menggunakan Parameter *Threshold*  = 10-9, = 5 dan kernel RBF 74](#_Toc441189787)

[Tabel A.5 Hasil Uji Coba Menggunakan Parameter *Threshold*  = 5x10-9, = 1 dan kernel RBF 77](#_Toc441189788)

[Tabel A.6 Hasil Uji Coba Menggunakan Parameter *Threshold*  = 5x10-9, = 3.5 dan kernel RBF 80](#_Toc441189789)

[Tabel A.7 Hasil Uji Coba Menggunakan Parameter *Threshold*  = 5x10-9, = 7 dan kernel RBF 83](#_Toc441189790)

[Tabel A.8 Hasil Uji Coba Menggunakan Parameter *Threshold*  = 5x10-9, = 5 dan kernel *Polynomial 2* 86](#_Toc441189791)

[Tabel A.9 Hasil Uji Coba Menggunakan Parameter *Threshold*  = 5x10-9, = 5 dan kernel *Polynomial 3* 89](#_Toc441189792)

[Tabel A.10 Hasil Uji Coba Menggunakan Parameter konstanta *region* = 5%, *Threshold*  = 5x10-9, = 5 dan kernel RBF 92](#_Toc441189793)

[Tabel A.11 Hasil Uji Coba Menggunakan Parameter konstanta *region* = 10%, *Threshold*  = 5x10-9, = 5 dan kernel RBF 95](#_Toc441189794)

[Tabel A.12 Hasil Uji Coba Menggunakan Parameter *size frame* = 240 x 320, *Threshold*  = 5x10-9, = 5 dan kernel RBF 98](#_Toc441189795)

[Tabel A.13 Hasil Uji Coba Menggunakan Parameter *size frame* = 60 x 80, *Threshold*  = 5x10-9, = 5 dan kernel RBF 101](#_Toc441189796)

[Tabel A.14 Hasil Uji Coba Tanpa Menggunakan *region growing* dan perhitungan luasan *region* 104](#_Toc441189797)

# DAFTAR KODE SUMBER

[Kode Sumber 4.1 Implementasi Tahap Reduksi *Size Frame* 36](#_Toc441189798)

[Kode Sumber 4.2 Penggunaan Fungsi pyrDown() 36](#_Toc441189799)

[Kode Sumber 4.3 Implementasi Tahap Deteksi Gerak 36](#_Toc441189800)

[Kode Sumber 4.4 Implementasi Penyimpanan Piksel 37](#_Toc441189801)

[Kode Sumber 4.5 Implementasi Menghitung Nilai Standar Deviasi dan Rata-Rata Setiap *Channel* 38](#_Toc441189802)

[Kode Sumber 4.6 *Generate list* piksel api 39](#_Toc441189803)

[Kode Sumber 4.7 Fungsi Menghitung Nilai Probabilitas Distribusi Gaussian 40](#_Toc441189804)

[Kode Sumber 4.8 Mendapatkan Threshold 40](#_Toc441189805)

[Kode Sumber 4.9 Membaca *List* Piksel Api 40](#_Toc441189806)

[Kode Sumber 4.10 Implementasi Tahap Deteksi Warna Piksel 41](#_Toc441189807)

[Kode Sumber 4.11 Implementasi Tahap *Region Growing* 42](#_Toc441189808)

[Kode Sumber 4.12 Implementasi *Growing* 43](#_Toc441189809)

[Kode Sumber 4.13 Implementasi Tahap *Clock Wise* 43](#_Toc441189810)

[Kode Sumber 4.14 Implementasi Tahap Variasi Warna *Region* 44](#_Toc441189811)

[Kode Sumber 4.15 Implementasi Memasukan Nilai *Wavelet* Kedalam *List* 45](#_Toc441189812)

[Kode Sumber 4.16 Pemanggilan Fungsi *Wavelet* 45](#_Toc441189813)

[Kode Sumber 4.17 *Training* Klasifikasi 46](#_Toc441189814)

[Kode Sumber 4.18 Implementasi Tahap Klasifikasi 47](#_Toc441189815)

[Kode Sumber 4.19 Implementasi Tahap Menandai Region Api 48](#_Toc441189816)

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

# PENDAHULUAN

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

## 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 Machines*(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 klasifiskasi 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.

## Rumusan Masalah

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

1. Bagaimana penggunaan *Support Vector Machines* pada pengenalan tekstur iris mata.
2. Bagaimana penggunaan *Hamming distance* pada pengenalan tekstur iris mata.
3. Menyusun uji coba pengklasifikasi tekstur iris mata menggunakan *kernel* pada SVM.
4. Bagaimana melakukan identifikasi bagian iris pada citra mata.

## 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.

## Tujuan

Tujuan dari pembuatan tugas akhir ini adalah untuk membuat sebuah implementasi perangkat lunak yang dapat melakukan pengenalan iris mata.

## Manfaat

Pengerjaan tugas akhir ini dilakukan dengan harapan bisa memberikan kontribusi pada teknologi biometrik yang dapat melakukan pengenalan melalui iris mata. Harapannya, dengan pembuatan perangkat lunak ini, akan dapat melakukan pendeteksian identitas seseorang melalui iris mata. Sehingga dapat melakukan proses verifikasi menjadi lebih cepat.

## 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 Machines dan Hamming distance.

1. Studi Literatur

Pada tahap ini dilakukan pencarian, pengumpulan, pembelajaran dan pemahaman informasi dan literatur yang diperlukan untuk pembuatan implementasi pada tahap praproses citra mata, sehingga menghasilkan daerah iris, 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.

1. Implementasi perangkat lunak

Pada tahap ini dilakukan implementasi perangkat lunak sesuai dengan rancangan perangkat lunak yang dibuat pada pengerjaan Tugas Akhir.

1. 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, serta mengidentifikasi kendala yang mungkin timbul. Parameter yang diujicobakan adalah fungsi kernel pada metode klasifikasi *support vector machines*.

1. 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.

## Sistematika Penulisan

Buku Tugas Akhir ini bertujuan untuk mendapatkan gambaran dari pengerjaan Tugas Akhir ini. Selain itu, diharapkan dapat berguna untuk pembaca yang tertarik untuk melakukan pengembangan lebih lanjut.

Secara garis besar, buku Tugas Akhir terdiri atas beberapa bagian seperti berikut ini:

1. **Pendahuluan**

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

1. **Dasar Teori**

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

1. **Perancangan Perangkat Lunak**

Bab ini berisi tentang desain sistem yang disajikan dalam bentuk *flowchart*.

1. **Implementasi**

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

1. **Uji Coba dan Evaluasi**

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

1. **Kesimpulan dan Saran**

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

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

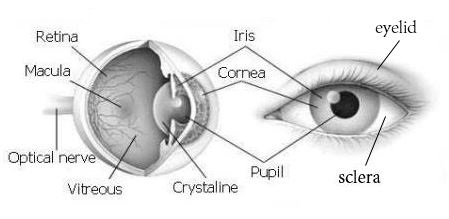
# TINJAUAN PUSTAKA

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

## 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 [3]:

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 membelokan sinar menuju ke otak.



Gambar 2.1 Anatomi Mata

## Sistem Pengenalan Iris

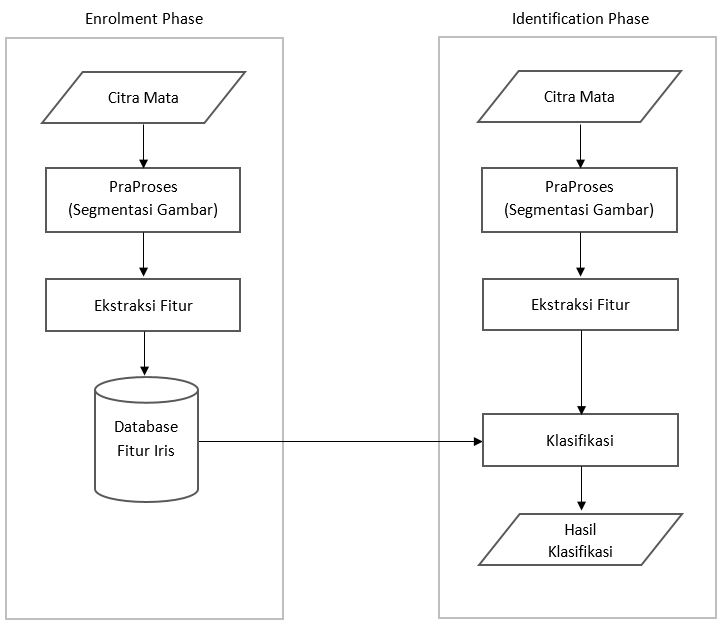
Pengenalan iris adalah suatu proses untuk mengenal seseorang dengan menganalisa pola acak dari iris. Dimana pengenalan iris bertujuan untuk mengenali suatu objek dengan cara mengekstraksi informasi yang terdapat dalam suatu citra tersebut.

Dalam pengaplikasiannya, sudah banyak metode yang digunakan. Pada proses identifikasi iris sampai normalisasi relatif sama. Metode dengan menggunakan *wavelet* [1], *Laplacian of Gaussian Filter* [4], dan berbagai macam metode lainnya. Metode-metode tersebut digumakan pada proses ekstraksi fitur. Sedangkan pada proses pengenalan, ada pula klasifikasi *Support Vector Machines* [5], *Hamming distance* [6], 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 [7]. 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 citi atau fitur iris. Setiap fitur akan disimpan didalam sebuah database yang nantinya akan dibandingkan dengan fitur masukan dalam proses pencocokan iris. Proses tersebut mencocokan fitur masukan dengan seluruh fitur yang terdapat didalam database dan menghasikan 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.

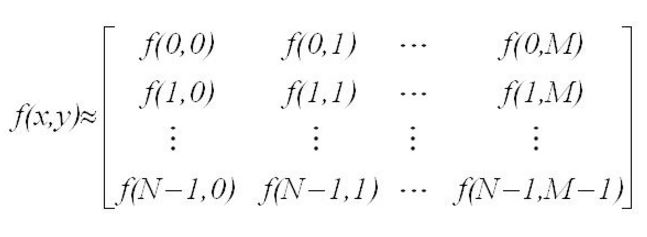


Gambar 2.2 Sistem pengenalan iris secara umum

## 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 [8].

Sebuah citra digital f(x,y) dapat direpresentasikan sebagai sebuah matriks yang indeks baris dan kolomnya mengidentifikasikan sebuah titik pada citra dan nilai dari elemen matriks yang bersangkutan merupakan tingkat warna pada titik tersebut, seperti yang ditunjukan 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.



Gambar 2.3 Representasi citra digital

## 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 [9].

1. Mengurangi noise dengan menggunakan filter Gaussian sebesar 13x13, standar deviasi iris dengan sclera, dan iris dengan pupil sebesar 2 [10].
2. 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

|  |  |
| --- | --- |
|  | (2.1) |

dimana *Gx*= gradien ke arah sumbu x

*G*y = gradien ke arah sumbu y

*G*= intensitas gradien

selanjutnya mencari arah gradien dengan persamaan 2.2

|  |  |
| --- | --- |
|  | (2.2) |

dimana *θ* adalah arah gradien

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

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

1. Mengatur Gamma atau pengontrasan terhadap arah yang telah terdeteksi. Pengontrasan dilakukan menggunakan persamaan 2.3

|  |  |
| --- | --- |
|  | (2.3) |

dimana *Imax* = nilai citra intensitas maksimum

*Imin* = nilai citra intensitas minimum

*M =* nilai citra intensitas rata-rata

Kemudian mendapatkan hasil akir pengontrasan dengan persamaan 2.4

|  |  |
| --- | --- |
|  | (2.4) |

dimana *C* = citra gamma

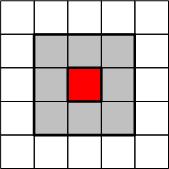
*g* = nilai gamma

*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. Pengontrasan dilakukan untuk mempermudah dalam penelusuran tepi citra iris mata.

1. 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 pengontrasan.
2. 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. Gambar 2.4 menunjukan 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.

## 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 mempunyaibatas 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 jaro-jari pada lingkaran iris dan lingkaran pupil.

|  |  |
| --- | --- |
|  | (2.5) |

dimana (xc,yc) = 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, trasformasi Hough akan bekerja pada titik tersebut.



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.

## 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 mempengaruihi pelebaran pupil yang lain juga seperti jarak citra, rotasi, dan derajar kemiringan. Proses normalisasi akan menghasilkan daerah iris yang memiliki ukrang yang konstan. Sehingga jika dilakukan pencocokan antara dua daerah iris akan lebih mudah [10].

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, θ*). Nilai *r* berada pada interval [0,1] dan nilai *θ* mempunyai interval derajat sudut [0, 2π]. 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.

|  |  |
| --- | --- |
|  | (2.6) |

dengan

|  |  |
| --- | --- |
|  | (2.7) |
|  | (2.8) |

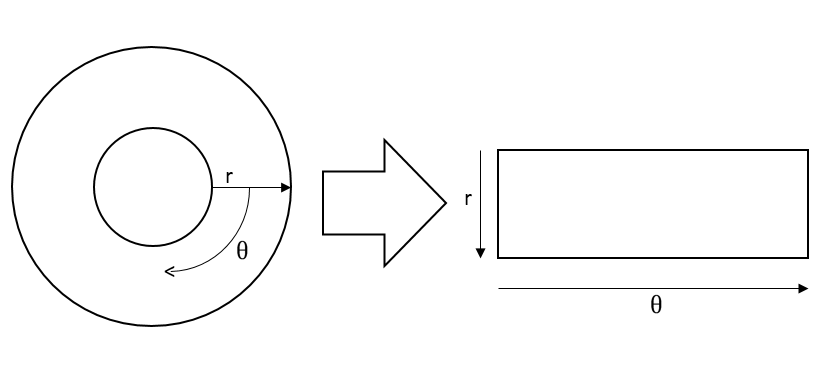
dimana *I*(*x*, *y*)adalah citra iris

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

(*r*, *θ*) adalah koordinat pada polar normalisasi

*xp*,*yp* adalah koordinat batas tepi pupil

*xi*,*yi* adalah koordinat batas tepi.



Gambar 2.6 *Daugman’s rubber sheet model*

*Rubber sheet model* mengatasi pelebaran yang dilakukan pupil dan ketidakkonsistenan ukuran pada 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.

## Haar Wavelet

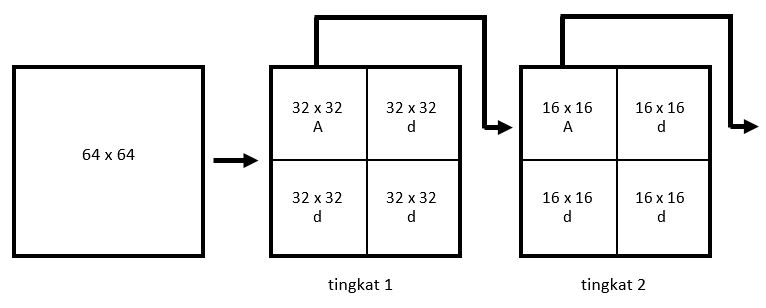
Wavelet adalah sebuah fungsi matematika mampu melakukan dekomposisi terhadap sebuah fungsi. Transformasi wavelet akan mengkonversi signal ke dalam sederetan gelombang singkat. Transformasi wavelet merupakan sebuah fungsi konversi yang dapat membagi data ke dalam komponen frekuensi yang berbeda-beda dan menganalis komponen tersebut dengan resolusi tertentu sesuai dengan skalanya. Wavelet telah banyak diaplikasikan dalam berbagai bidang, salah satunya adalah pengolahan citra [11].

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* secara berturut-turut.

|  |  |
| --- | --- |
|  | (2.9) |

|  |  |
| --- | --- |
|  | (2.10) |

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 approksimasi, dan tiga citra detail. Citra detail yang didapat adalah detail horizontal, vertikal dan diagonal. Proses ini disebut dekomposisi, dekomposisi dapat dilanjutkan kembali dengan citra approksimasi sebagai masukannya untuk mendapatkan tingkat dekomposisi selanjutnya seperti yang ditunjukan pada Gambar 2.7.



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 approksimasi dan citra diagonal dilakukan dengan filter terdahap baris&kolom. Kemudian melakukan filter terhadap baris saja untuk mendapatkan citra horizontal, dan terakhir melakukan filter terhadap kolom saja untuk mendapatkan citra vertikal.

F:\KULIAH\TA\bukuTA\source\wavelt.png

Gambar 2.8 Filter pada wavelet

## Log-Gabor Filter

*Log-Gabor* 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 [12]. 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.

|  |  |
| --- | --- |
|  | (2.11) |

dengan f adalah nilai frekuensi, f0 adalah pusat frekuensi dengan nilai 1/λ dan σ adalah bobot filter. Berdasarkan penelitian Masek(2003) pada pengenalan citra iris mata, ditetapkan nilai λ sebesar 18, σ sebesar 0.5, dan interval f0 sebesar [0-0.5]. Ilustrasi penggungaan log-Gabor dapat dilihat pada lampiran.

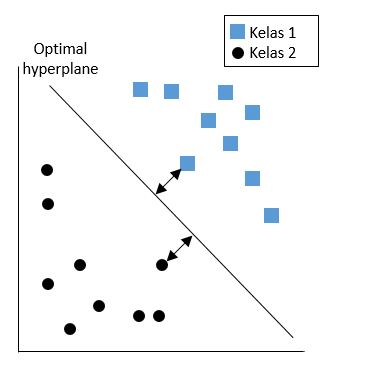
Proses log-Gabor filter ini membutuhkan sinyal dari transform Fourier yang berasal dari setiap baris citra hasil normalisasi iris, karena akan direpresentasikan pada domain frekuensi. Kemudian dilakukan konvulsi antara sinyal citra iris dengan filter log-Gabor pada setiap bidang satu dimensi. Hasil tersebut 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.

## Support Vector Machines

*Support vector machines* adalah metode klasifikasi yang hanya memfokuskan pada klasifikasi dua kelas, yaitu kelas +1 dan -1. Pada metode klasifikasi *support vector machines*, 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 *support vector machines* dapat dilihat pada Gambar 2.9



Gambar 2.9 Ilustrasi *Support Vector Machines*

|  |  |
| --- | --- |
|  | (2.12) |

|  |  |
| --- | --- |
|  | (2.13) |

Pada persaman diatas, variabel *C* adalah konstanta nilai pinalti dari kesalahan klasifikasi. Pencarian nilai dan dilakukan dengan menggunakan batasan yang ditulis menggnakan Persamaan 2.14

|  |  |
| --- | --- |
|  | (2.14) |

Persamaan 2.13 dilakukan untuk mencari nilai dan 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.

|  |  |
| --- | --- |
|  | (2.15) |

dengan batasannya terdapat pada Persamaan 2.16.

|  |  |
| --- | --- |
|  | (2.16) |

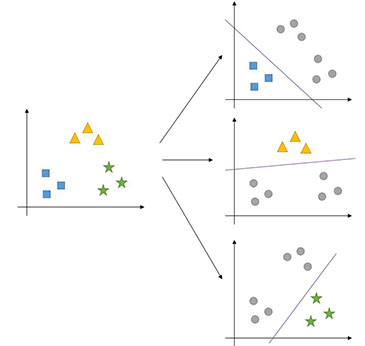
Untuk mencari nilai , digunakan *quadratic programming* (*QR*) seperti contohnya *CVXOPT* untuk Python. Setelah mendapatkan nilai , parameter persamaan *hyperplane* menjadi seperti persamaan 2.17. Kemudian hasilnya disubtitusikan ke fungsi *hyperlane* dan dirumuskan kedalam Persamaan 2.18.

|  |  |
| --- | --- |
|  | (2.17) |

|  |  |
| --- | --- |
|  | (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.



Gambar 2.10 Ilustrasi *svm* multi kelas

SVM multi kelas membutuhkan proses pemodelan pada data training yang lebih banyak dibandingan 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, dibandingan satu per satu terhadap model yang sudah dibuat. Ilustrasi *support vector machines* 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 di *input space* dimapping ke fitur *space*  dengan dimensi yang lebih tinggi. Beberapa fungsi kernel yang terkenal digunakan pada model *support vector machines* dirumuskan pada persamaan 2.19.

|  |  |
| --- | --- |
|  | (2.19) |

Persamaan diatas adalah fungsi kernel polynomial dan radial basis function (RBF) secara berturut-turut. Persamaan diatas mempunyai parameter kernel yaitu *p* dan. Dimana variabel *p* adalah parameter yang menandakan *order*. Sedangkan variabel menandakan *scaling factor*. Dalam fungsi kernel ini, RBF adalah kernel yang paling bagus untuk digunakan karena alasan berikut [13]:

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

## Hamming Distance

*Hamming distance* adalah metode untuk melakukan pencocokan antara fitur-fitur iris mata yang sudah melalui proses ekstraksi 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.

|  |  |
| --- | --- |
|  | (2.20) |

dimana *N* adalah dimensi dari vektor fitur

*Xj* adalah komponen ke-j dari vektor fitur

*Yj* 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.

## Bahasa Pemograman Matlab 8.3

Matlab (*Matrix Laboratory*) adalah sebuah bahasa pemograman komputer generasi keempat dengan performansi yang tinggi. Matlab menggabungkan komputasi, pemograman, 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, kususnya yang berhubungan dengan matriks dan vektor. Matlab menyediakanakses mudah ke aplikasi matriks yang dikembangkan oleh projek 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*,danfile
2. *The Matlab Mathematical Function Library*. Sekumpulan algoritma komputasi mulai dari fungsi-fungsi dasar sepertri: sum, sinus, cosinus, dan aritmatika kompleks, sampai dengan fungsi-fungsi tingkat lanjut seperti *matriks eigenvalues* dan *fast Fourier transforms*.
3. *The Matlab Language*. Bahasa pemograman 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 berinterakasi dengan MATLAB. Ini melibatkan fasilitas untuk pemanggilan MATLAB sebagai sebuah computational engine, dan untuk membaca dan menuliskan file berekstensi .MAT.

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

# 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.

## Perancangan Data

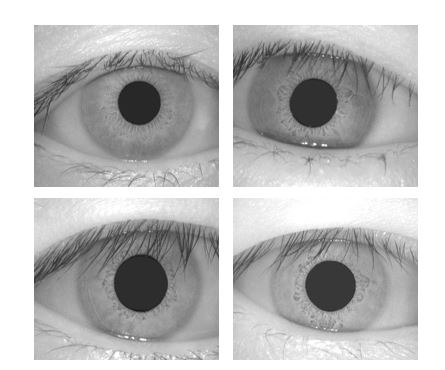
Pada sub bab ini akan dijelaskan mengenai data yang digunakan sebagai masukan untuk selanjutnya diolah dan dilakukan pengujian sehingga menghasilkan data keluaran yang diharapkan.

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.

### 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 x 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 adalah contoh citra mata dari database CASIA.



Gambar 3.11 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 merupakandata 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.

### Data Proses

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.

### 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 *support vector machines* dan klasifikasi *Hamming distance* secara terpisah. Hasil dari proses klasifikasi tersebut adalah nama kelas pada masing-masing data uji dan akurasinya.

## Desain Umum Sistem

Secara garis besar ada tiga tahapan utama dalam melakukan pengenalan iris yaitu praposes, 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 praposes 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 kelompak. 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-rubah 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 filters*. Kedua metode tersebut akan digunakan pada dua metode klasifikasi yang berbeda dan tidak saling berkorelasi pula. Hasil tahap ini adalah vektor fitur yan digunakan ntuk melakukan verifikasi pada pengenalan iris.

Setelah melalui tahap ekstraksi fitur, vektor-vektor fitur akan di verifikasi menggunakan metode *support vector machines* yang jika fitur didapatkan dari nilai konstanta *wavelet haar*. Lalu di verifikasi menggunakan metode *Hamming distance* yang jika fitur didapatkan dari hasil *log-Gabor filters*. Hasil akhir yang dikeluarkan adalah nama kelas dengan menggunakan masing-masing dua metode klasifikasi yang berbeda tersebut.

Gambar 3.2 menunjukan tiga tahapan utama dalam pengenalan iris ditunjukan dengan menggunakan ekstraksi fitur *wavelet haar* dan klasifikasi *support vector machines*, juga ekstraksi fitur *log-Gabor filters* dan klasifikasi *Hamming distance*.

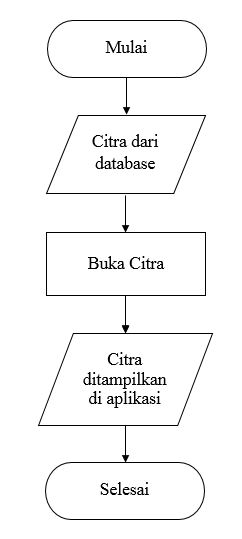


Gambar 3.12 Diagram Alir Rancangan Sistem

Masing-masing proses ini akan dijelaskan lebih detail pada bagian selanjutnya dari subbab berikutnya.

## Akuisisi Citra Mata

Seperti yang telah dijelaskan pada bagian perancangan data, citra mata akan digunakan sebagai masukan untuk keseluruh 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.

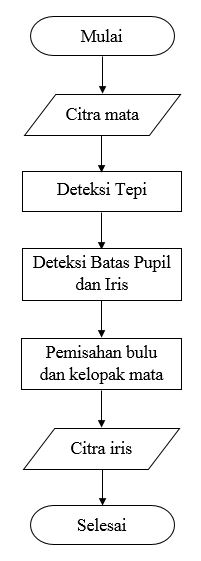


Gambar 3.13 Diagram Alir Akuisisi Citra Mata

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

## Perancangan Preprocessing

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

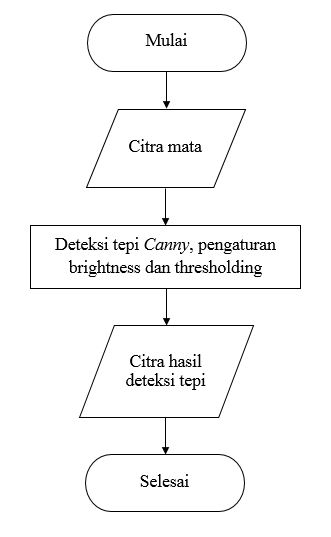


Gambar 3.14 Diagram Alir *Preprocessing*

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 lanjutpada subbab ini.

### Deteksi Tepi

Tahap deteksi tepiadalah tahap dimanamenandai 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 *brighness* dan *thresholding* citra sehingga tepi lingkaran menjadi jelas. Diagram alir proses deteksi tepi ditunjukan pada gambar 3.5.



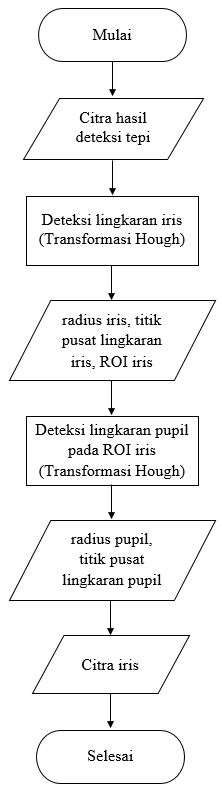
Gambar 3.15 Diagram Alir Proses Deteksi Tepi

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.

### 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 metodetransformasi Houghpada 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 bidangakumulator. 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 3.16 Diagram Alir Proses deteksi batas pupil dan iris

Setelah melakukan transformasi pada lingkaran iris, selanjutnya diakukan 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 ditunjukan 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.

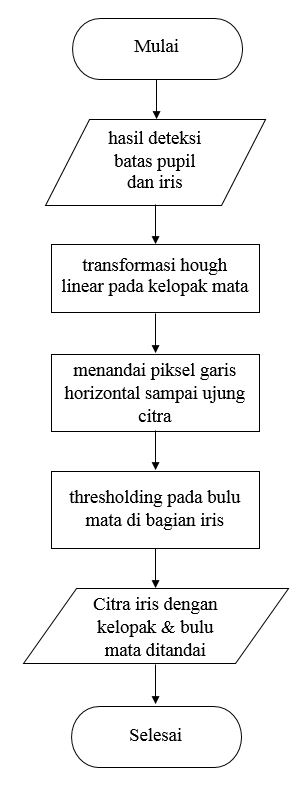
### Pemisahan Bulu dan Kelopak Mata

Citra mata yang telah diidentifikasi bagian irisnya masih mengandung bulu dan kelopak mata. Keduanya diaangap 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.

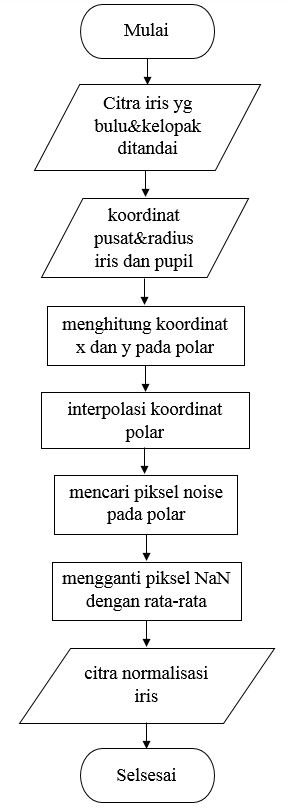
Hasil keluaran dari proses ini meruapakan citra yang sudah dapat di normalisasikan dan *noise* sudah berhasil ditandai. Gambar 3.7 menunjukan diagram alir proses pemisahan bulu dan kelopak mata.



Gambar 3.17 Diagram Alir Pemisahan Bulu dan Kelopak Mata

## 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-rubah 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 ditujunkan pada gambar 3.8.



Gambar 3.18 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 lingkar iris dari koordinat cartesian ke koordinat polar. Menggunakan transformasi polar seperti pada subbab 2.6, yang didalamnya terdapat persamaan untuk mengubah bidang daerah lingkar 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.

## 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 filters. Ekstraksi fitur yang dipakai wavelet haar adalah koefisien aproksimasi. Sedangakan ekstraksi fitur log-Gabor filters menghasilkan bilangan bit biner. Masing-masing dari ekstraksi fitur tersebut menghasilkan vektor fitur yang akan digunakan untuk melakukan tahap klasifikasi.

### Fitur Koefisien Wavelet Haar

Ektraksi fitur dimulai dengan dengan melakukan transformasi wavelet haar pada citra iris yang sudah ternormalisasi. Citra iris yang ternomalisasi akan dilakukan proses sesuai dengan penjelasan pada subbab 2.7, yang pada dimana citra iris akan didekomposisi. Pada sitem ini, citra iris akan didekomposisi sampai dengan tingkat ketiga. Artinya, hasil dimensi citra iris akan tereduksi menjadi seperdelapan dari dimensi awal. Pseudocode penghitungan dekomposisi citra iris ditunjukan pada gambar 3.9.

|  |  |  |
| --- | --- | --- |
| Masukan | | Citra iris hasil normalisasi |
| Keluaran | | Vektor fitur koefisien aproksimasi dekomposisi tingkat 3 |
| 1.  2.  3.  4.  5.  6.  7. | wname 🡨 “haar“ (nama wavelet)  for i=1 to 3  [cA,~,~,~] = dwt2(citraNorm,wname)  citraNormalisasi 🡨 cA  end  return cA | |

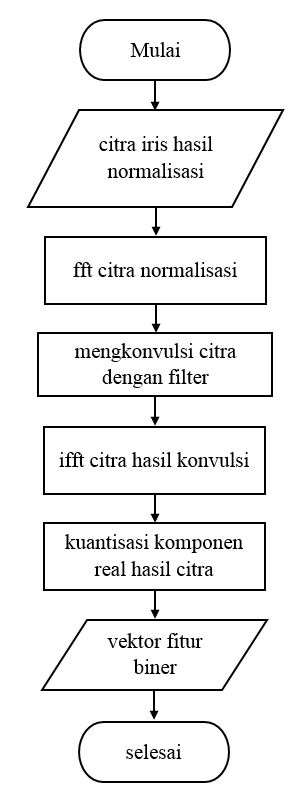
Gambar 3.19 *Pseudocode* Dekomposisi Wavelet

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

### Fitur Bilangan Biner

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 normalisasi kedalam bentuk bit biner dengan dimensi yang sama, yakni 64x512. Untuk itu digunakan ekstraksi fitur log-Gabor filters sebagai suatu batu loncatan untuk memperoleh bilangan biner tersebut. Gambar 3.10 menunjukan 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 sebanyakak 512 piksel dilakukan proses fft untuk mempresentasikan citra pada domain frekuensi. Kemudian setelah dilakukan proses fft, nilai tersebut akan dikonvulsi dengan filter log-Gabor. Hasil konvulsi akan dilakukan proses ifft untuk mengembalikan representasi citra ke dalam domain spasial menggunakan ifft. 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. Pada citra *noise* dari hasil normalisasi juga akan diubah kedalam bentuk biner dengan melakukan proses kuantisasi yang sama.



Gambar 3.20 Diagram Alir pembuatan Fitur Biner

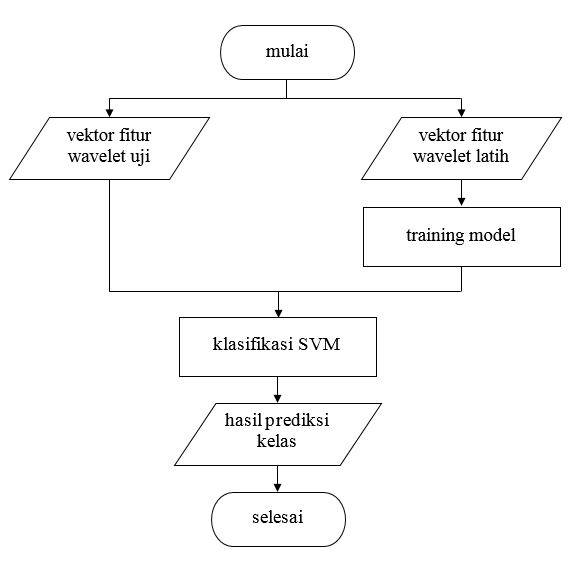
## Perancangan Klasifikasi

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 machines* untuk metode ekstraksi fitur wavelet haar dan metode klasifikasi *hamming distance* untuk metode ekstraksi fitur *log-Gabor filters*. Seperti halnya metode ekstraksi fitur, kedua metode klasifikasi ini tidak saling berkaitan.

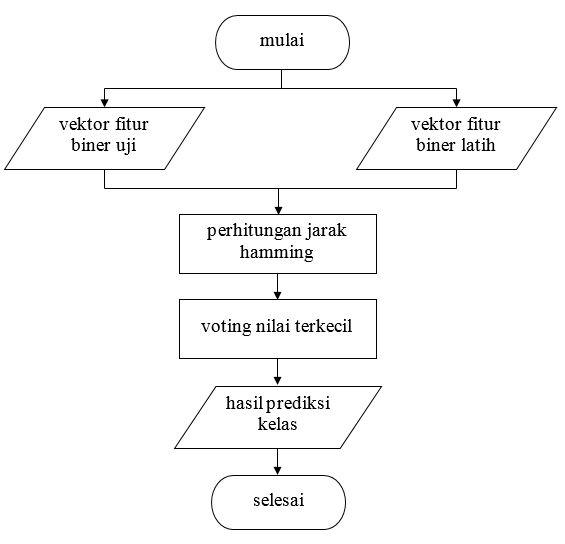
Tahap ini terlebih dahulu membuat database vektor fitur dan disimpan dalam sebuah file berekstensi .mat dengan menggunakan Matlab. Jika ada citra masukan yang ingin dikenali irisnya termasuk pada kelas mana, maka akan dilakukan klasifikasi menggunakan salah satu metode klasifikasi. Masukan dari tahap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

Pada klasifikasi dengan menggunakan metode *support vector machines* (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 menunjukan diagram alir tahap klasifikasi dengan menggunakan svm.

Pada klasifikasi dengan menggunakan metode *hamming distance*, 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 ditunjukan pada Gambar 3.12, prediksi kelas adalah yang mempunyai nilai paling kecil.



Gambar 3.11 Diagram Alir Klasifikasi SVM



Gambar 3.12 Diagram Alir Klasifikasi Hamming Distance

# 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 machines* dan *Hamming distance*).

## 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 4.2 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 & *Image Processing Toolbox*  Perangkat Pembantu:  Microsoft Excel 2013 |

## 

## 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 ditunjukan pada Kode Sumber 4.1.

FCK FKC

## Implementasi Preprocessing

Setelah citra mata ditampilkan dalam aplikasi perangkat lunak, langkah berikutnya adalah 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.

### 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 ditunjukan oleh Kode Sumber 4.2.

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

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).

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

Kode sumber 4.3 menunjukan 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.  2.  2.  3.  4.  5.  6.  7.  8.  9.  10.  11.  12.  13.  14.  15. | % mengambil ROI iris  imagepupil = eyeimage( irl:iru,icl:icu);  % transformasi Hough pd pupil  [rowp, colp, r] = findcircle(imagepupil, ...  lpupilradius, upupilradius, scaling,...  0.6, 2, 0.25, 0.25, 1.00, 1.00);  % penjumlahan dgn titik pusat untuk  % mendapatkan posisi yang tepat  row = round(irl + rowp);  col = round(icl + colp);  % radius dan titik pusat pupil  circlepupil = [row col r]; |

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

### 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 kelompak mata. Implementasinya ditunjukan pada Kode Sumber 4.5.

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

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*.

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

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

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.

## 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 mengealami 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 ditunjukan pada Kode Sumber 4.7.

|  |  |
| --- | --- |
| 1.  2.  3.  4.  5.  6.  7.  8.  9.  10.  11.  12.  13.  14.  15.  16.  17.  18.  19.  20.  21.  22.  23.  24.  25. | % setting r dan theta pada citra polar  [theta,r]=meshgrid(linspace(0,2\*pi,512),...  linspace(0,1,64));  % batas lingkaran pd lingkaran dalam (pupil)  xp = PupilCenterX + PupilR\*cos(theta);  yp = PupilCenterY + PupilR\*sin(theta);  % batas lingkaran pada lingkaran luar (iris)  xi = PupilCenterX + IrisR\*cos(theta);  yi = PupilCenterY + IrisR\*sin(theta);  % mapping ke koordinat polar  x = (1-r).\*xp + r.\*xi;  y = (1-r).\*yp + r.\*yi;  polar\_array = interp2(double(citra),x,y);  % ganti nilai NaN dgn rata2 piksel citra  coords = find(isnan(polar\_array));  polar\_array2 = polar\_array;  polar\_array2(coords) = 0.5;  avg = sum(sum(polar\_array2)) / ...  (size(polar\_array,1)\*size(polar\_array,2))  polar\_array(coords) = avg; |

Pada Kode Sumber 4.7, tahap normalisasi diawali dengan pembentukan ukuran bidang pada polar. Dengan menentukan resolusi radial (*r*) dan resolusi angular (*θ*) seperti pada baris 2. Selanjutnya pada baris 6 samapai 10 dilakukan perhitungan batas lingkaran pupil dan batas lingkaran iris sepanjang sudut angular. Batas-batas lingkaran yang sudah didapat akan digunakan untuk pembentukan koordinat pada bidang polar, lalu menginterpolasikan koordinat polar pada citra.

Namun citra iris yang sudah diubah pada bidang polar masih memiliki *noise*. Sehingga perlu dilakukan pengubahan intensitas pada piksel *noise* tersebut. pengubahan nilai intensitas ditunjukan 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.

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.

## 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 dam 4.5.2.

### Implementasi Fitur Koefisien Wavelet Haar

Sub bab ini membahas implementasi tahap ekstraksi fitur. Ekstraksi fitur dilakukan dengan mengubah citra iris normalisasikedalam 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.  2.  3.  4.  5.  6.  7.  8.  9. | % inisialisasi awal  wname = 'haar';  sizeEkstrak = [8 64];  % proses wavelet haar dekomposisi lvl 3  [C,~] = wavedec2(polar\_array,3,wname);  HasilWavelet = C(1:(sizeEkstrak(1) \* ...  sizeEkstrak(2))); |

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 tingkat tiga pada baris 3. Kemudian fungsi wavedec2 digunakan untuk melakukan proses dekomposisi pada polar\_array (citra normalisasi). Fungsi ini langsung mendekomposisi citra ke tingkat 3 sebagai ganti dari iterasi fungsi dwt2. Hasil dekomposisi yang diambil adalah nilai koefisien aproksimasi yang ditunjukan pada baris 7.

### Implementasi Fitur Bilangan Biner

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.

Kode Sumber 4.9 melakukan pengubahan citra iris ternormalisasi menjadi citra fitur biner. Hasil konvulsi dari citra iris ternomalisasi dengan filter gabor ditunjukan pada baris 2, dengan isi fungsi gaborconvolve ditunjukan 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 template.

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

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

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, sehinggan citra iris ternormalisasi perlu diubah pula kedalam domain frekuensi. Pengubahan ini menggunakan fungsi pada Matlab fft seperti pada baris 4. Hasil dari proses konvulsi 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.

|  |  |
| --- | --- |
| 1.  2.  3.  4.  5.  6.  7.  8.  9. | % konvulsi citra normalisasi dgn filter  for r = 1:size(polar\_array,1)  signal = polar\_array(r,1:ndata);  imagefft = fft( signal);  result(r,:) = ifft(imagefft .\* filter);  end  EO = result; |

## 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 tersendiri. Hasil ekstraksi fitur wavelet haar akan digunakan sebagai masukan untuk metode klasifikasi *support vector machine*. Sedangkan ekstraksi fitur bilangan biner akan digunakan sebagai masukan untuk metode klasifikasi *Hamming distance*. Implementasi keduanya dapat dilihat pada Kode Sumber 4.10 dan Kode Sumber 4.11.

Kode Sumber 4.10 merupakan implementasi klasifikasi *support vector machines*. Sebelum masuk kedalam proses klasifikasi, data citra mata 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.

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

Sebelum melakukan klasifikasi, data latih dan labelnya terlebih dahulu diproses sebagai modeluntuk 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. Setiap data uji hasilEkstrak akan dihitung kemiripannya pada setiap model dengan menggunakan probabilitas. Sehingga pada akhirnya terdapat matriks yang berisi probabilitas pada setiap model seperti yang ditunjukan 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.

Kode Sumber 4.11 merupakan implementasi klasifikasi *Hamming distance*. Sebelum masuk kedalam proses perhitungan jarak, data citra mata sudah dilakukan ekstraksi fitur menggunakan *log-Gabor filter* dan disimpan dalam file dataset\_log-gabor.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.

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

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 menghitung keempat data latih tersebut menggunakan fungsi zHamming seperti penjelasan subbab 2.10 , kemudian dihitung rata-ratanya. Hal tersebut dilakukan iterasi sebanyak sejumlah kelas yang ditunjukan 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.

## Implementasi Klasifikasi

Sub bab ini membahas implementasi tahap *region growing* yang menggunakan kandidat piksel api pada tahap deteksi warna api sebagai masukan. *Region growing* dilakukan menggunakan *list* kandidat piksel api sebagai masukan awal dari piksel yang dicari *region*nya. Titik piksel kandidat api dilakukan pengecekan probabilitas warna api terhadap delapan tetangga piksel tersebut. Jika piksel tetangga termasuk piksel api, maka piksel tersebut akan ditandai sebagai region dari piksel awalan tersebut. *Region* akan diberi nomor sesuai dengan titik piksel awal dari *region* tersebut. Implementasi *region growing* dapat dilihat pada Kode Sumber 4.11.

|  |  |
| --- | --- |
| 1.  2.  3.  4.  5.  6.  7.  8.  9.  10.  11.  12.  13.  14.  16. | def getRegionGrowing(self, list\_candidate, images, color\_dataset,counter):  gray\_image = ImageProcessing.getRGBtoGray(self,images)  is\_visit = gray\_image\*0  result\_image = copy.copy(gray\_image)  region\_number = 0  for x in list\_candidate:  coor\_y = x[0]  coor\_x = x[1]  if is\_visit[coor\_y][coor\_x] == 0:  stack = []  region\_number+=1  stack.append([coor\_y,coor\_x])  is\_visit[coor\_y][coor\_x] = region\_number  result\_image, is\_visit = self.doFloodFill( gray\_image, result\_image, is\_visit, stack, region\_number, color\_dataset, images)  return is\_visit |

Kode Sumber 4.11 Implementasi Tahap *Region Growing*

Kode Sumber 4.11 melakukan inisialisasi *region* dengan nilai 0 pada variabel *is\_visit*. Selanjutnya melakukan iterasi sebanyak kandidat piksel api yang masuk kedalam tahap ini. Setiap kandidat piksel api dilakukan pengecekan, apakah piksel tersebut sudah dilakukan *region growing* atau belum. Jika belum (nilai variabel indeks yang sedang dicek bernilai 0) , maka koordinat dari titik tersebut akan dijadikan sebagai *seed* untuk dimasukan kedalam *stack* dan dilakukan *growing*. Implementasi *growing* dapat dilihat pada Kode Sumber 4.12.

Kode Sumber 4.12 melakukan pegecekan terhadap *stack* piksel yang akan dicek. Jika stack masih memiliki nilai, maka nilai tersebut akan digunakan sebagai *seed* untuk melakukan cek terhadap tetangga piksel. Jika tetangga piksel belum mempunyai *region* dan masuk kedalam *region* tersebut, maka nilai dari variabel *is\_visit* diganti sesuai dengan nilai *region* dan koordinat piksel tersebut dimasukan kedalam *stack* untuk melakukan pengecekan tetangga selanjutnya. Pada fungsi *growing()* melakukan pemenggilan fungsi *getClockWise()*. Implementasi *getClockWise()* dapat dilihat pada Kode Sumber 4.12.

|  |  |
| --- | --- |
| 1.  2.  3.  4.  5.  6.  7.  8.  9.  10.  11.  12.  13.  14.  15.  16.  17. | def growing(self, gray\_image ,result\_image ,is\_visit, stack, region\_number, color\_dataset, original\_image):  clocks = Data.getClockwise(self)  while len(stack) != 0:  coory,coorx = stack[0]  stack.pop(0)  result\_image[coory][coorx] = 255  data = original\_image[coory][coorx]  B,G,R = data[0],data[1],data[2]  for x in clocks:  try :  if is\_visit[coory+x[0]][coorx+x[1]] == 0 and color\_dataset[B][G][R] == True :  is\_visit[coory+x[0]] [coorx+x[1]] = region\_number  stack.append([coory+x[0], coorx+x[1]]  except :  pass  return result\_image,is\_visit |

Kode Sumber 4.12 Implementasi *Growing*

|  |  |
| --- | --- |
| 1.  2.  3.  4.  5.  6.  7.  8.  9.  10.  11. | def getClockwise(self):  clocks = []  clocks.append([-1,-1])  clocks.append([-1,0])  clocks.append([-1,1])  clocks.append([0,-1])  clocks.append([0,1])  clocks.append([1,-1])  clocks.append([1,0])  clocks.append([1,1])  return clocks |

Kode Sumber 4.13 Implementasi Tahap *Clock Wise*

Iterasi yang dilakukan pada *getClockWise()* akan melakukan pengecekan delapan tetangga searah jarum jam. Hasil keluaran dair ptoses ini adalah *region* yang mempunyai nomor *region* yang berbeda antar *region*.

### Implementasi Tahap Perhitungan Luasan Region

Sub bab ini membahas implementasi tahap perhitungan luasan *region*. Masukan dari tahap ini adalah kandidat piksel api yang telah didapatkan dari tahap probabilitas warna api, dan *region*. Pada tahap perhitungan luasan *region* dihitung banyaknya piksel yang ada pada *region* tersebut. Jika luasan piksel *region* tersebut melebihi batas, maka kandidat piksel yang ada pada *region* tersebut masuk sebagai kandidat piksel api . Implementasi dapat dilihat pada Kode Sumber 4.14.

|  |  |
| --- | --- |
| 1.  2.  3.  4.  5.  6.  7.  8.  9.  10.  11.  12.  13.  14.  15.  16.  17.  18.  19. | def getFilterSizeRegion (self,list\_candidate,region):  true\_piksel = []  false\_piksel = []  list\_region = np.unique(region)  threshold = dict()  for x in range(1,len(list\_region)):  lists = np.where(region == x)  if len(lists[0]) > 1\*len(region)\* len(region[0])/100:  threshold[x] = True  else :  threshold[x] = False  for x in list\_candidate:  coor\_y = x[0]  coor\_x = x[1]  if threshold[region[coor\_y][coor\_x]] == True:  true\_piksel.append([coor\_y,coor\_x])  else :  false\_piksel.append([coor\_y,coor\_x])  return true\_piksel,false\_piksel |

Kode Sumber 4.14 Implementasi Tahap Variasi Warna *Region*

Setiap *region* akan dilakukan iterasi dan dilakukan pengecekan. Jika luasan dari *region* tersebut memenuhi syarat, maka seluruh piksel kandidat api masuk kedalam proses berikutnya. Hasil keluaran dari fungsi ini adalah *list* piksel yang yang masuk kedalam kandidat piksel api.

### Implementasi Tahap Ekstraksi Fitur dengan Wavelet

Sub bab ini membahas implementasi tahap ekstraksi fitur. Ekstraksi fitur dilakukan dengan mengubah *frame* kedalam domain *wavelet*. Digunakan sepuluh buah *frame* yang berurutan untuk diubah kedalam domain *wavelet*. Pada implementasi setiap *frame* diubah kedalam domain *wavelet* dan disimpan kedalam *list*. Implementasi mengubah dan manyimpan *frame* domain *wavelet* dapat dilihat pada Kode Sumber 4.15.

|  |  |
| --- | --- |
| 1.  2. | LL,(HL,LH,HH) = wv.toWavelet(copy.copy(grayImage))  list\_wavelet.append([HL,LH,HH]) |

Kode Sumber 4.15 Implementasi Memasukan Nilai *Wavelet* Kedalam *List*

Pada Kode Sumber 4.16, fungsi *toWavelet()* digunakan untuk mengubah *frame* kedalam domain *wavelet*. Implementasi mengubah *frame* kedalam domain *wavelet* dapat dilihat pada Kode Sumber 4.16.

|  |  |
| --- | --- |
| 1.  2. | def toWavelet(image):  return pywt.dwt2(image,'db2') |

Kode Sumber 4.16 Pemanggilan Fungsi *Wavelet*

### Implementasi Tahap Klasifikasi

Sub bab ini membahas implementasi tahap klasifikasi. Pada tahap ini sistem diberi masukan kandidat piksel api dan sepuluh buah *frame* domain *wavelet*, dimana masing-masing *frame* berisi tiga buah sub *frame* seperti penjelasan sub bab 3.4.1 . Sebelum melakukan klasifikasi, data training terlebih dahulu diproses sebagai data *training* untuk klasifikasi. Implementasi *training* klasifikasi dapat dilihat pada Kode Sumber 4.17.

|  |  |
| --- | --- |
| 1.  2.  3. | def getClassifier(datatraining):  x,y = readDataSet(datatraining)  clf = svm.SVC(kernel = 'rbf',C = 3.5)  clf.fit(x,y)  return clf |

Kode Sumber 4.17 *Training* Klasifikasi

Setiap piksel pada kandidat piksel api dilakukan perhitungan nilai fitur seperti yang sudah dijelaskan pada sub bab 3.4.1. Implementasi klasifikasi dapat dilihat pada Kode Sumber 4.18.

|  |  |
| --- | --- |
| 1.  2.  3.  4.  5.  6.  7.  8.  9.  10.  11.  12.  13.  14.  15.  16.  17.  18.  19.  20.  21.  22.  23.  24.  25.  26.  27.  28. | def doClassification(classifier, list, wavelet):  truePiksel = []  falsePiksel = []  listMax = []  listMin = []  cpyWavelet = np.int\_(copy.copy(wavelet))  cpyWavelet = np.power(cpyWavelet,2)  for x in cpyWavelet:  lists = np.add(np.add(x[0],x[1]) , x[2])  listMax.append(np.max(lists))  listMin.append(np.min(lists))  for x in list:  data = []  cnt= 0  for y in wavelet:  res = pow(y[0][x[0]][x[1]],2) + pow(y[1][x[0]][x[1]],2) + pow(y[2][x[0]][x[1]],2)  res = (float(res)-float(listMin[cnt]))/(float(listMax[cnt])-float(listMin[cnt]))  res = float('%.2f' % res)  cnt+=1  data.append(res)  data = np.sort(data)  classes = classifier.predict(data)  if classes == 'Api':  truePiksel.append([x[0],x[1]])  else :  falsePiksel.append([x[0],x[1]])  return truePiksel,falsePiksel |

Kode Sumber 4.18 Implementasi Tahap Klasifikasi

Pada Kode Sumber 4.18 dilakukan iterasi sebanyak kandidat piksel api yang lolos ketahap verifikasi. Setiap piksel akan dihitung nilai fitur *wavelet*. Dilakukan normalisasi nilai fitur yang dilakukan pada *line* 18. Jika hasil klasifikasi suatu fitur adalah api, maka nilai indeks dari piksel tersebut akan dimasukan kedalam *list*. Hasil yang dikeluarkan dari proses ini adalah *list* piksel api yang lolos tahap verifikasi.

### Implementasi Tahap Menandai Region Api

Sub bab ini membahas implementasi menandai *region* api. Piksel api yang sudah lolos tahap verifikasi selenjutnya diproses sebagai data keluaran yang ditampilkan. Masukan dari tahap ini adalah kandidat piksel api dan *frame* dari variabel *currentFrame2*. Karena perbedaan ukuran antara indeks piksel api, dilakukan normalisasi indeks. Dilakukan penyesuaian indeks-indeks piksel dengan *frame* keluaran. Implementasi *training* klasifikasi dapat dilihat pada Kode Sumber 4.19.

|  |  |
| --- | --- |
| 1.  2.  3.  4.  5.  6.  7.  8.  9.  10.  11.  12.  13.  14.  15.  16.  17. | def markingFire(self, list\_fire, image, constanta):  if len(list\_fire) == 0:  return image  list = np.array(list\_fire)  min\_y,max\_y = min(list[:,0]),max(list[:,0])  min\_x,max\_x = min(list[:,1]),max(list[:,1])  distance\_y = int((max\_y-min\_y)/2)\*constanta  distance\_x = int((max\_x-min\_x)/2)\*constanta  center\_point = [int((max\_y+min\_y)\*constanta/2) , int((max\_x+min\_x)\*constanta/2)]  min\_y,min\_x,max\_y,max\_x = center\_point[0]-distance\_y, center\_point[1] - distance\_x, center\_point[0] + distance\_y, center\_point[1] +distance\_x  for y in range(min\_y,max\_y+1):  image[y][min\_x] = [255,191,0]  image[y][max\_x] = [255,191,0]  for x in range(min\_x,max\_x+1):  image[min\_y][x] = [255,191,0]  image[max\_y][x] = [255,191,0]  return image |

Kode Sumber 4.19 Implementasi Tahap Menandai Region Api

Hasil keluaran dari Kode Sumber 4.19 adalah *frame* dengan tanda persegi jika terdapat piksel api pada *frame* yang diproses.

# UJI COBA DAN EVALUASI

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

## Lingkungan Uji Coba

Lingkungan uji coba yang akan digunakan adalah,

* 1. Perangkat Keras

Prosesor Intel® Core™ i3-2350M CPU @ 2.30GHz 2.30GHz RAM 4 GB.

Sistem Operasi 64-bit .

* 1. Perangkat Lunak

Sistem Operasi Microsoft Windows 8 64-bit Pro.

Perangkat Pengembang PyCharm.

## Data Uji Coba

Data yang digunakan untuk uji coba implementasi deteksi api berbasis sensor visual menggunakan metode *support vector machines* adalah potongan video yang didapatkan dari berbagai sumber. Kualitas video yang digunakan adalah video dengan *size* 240x320 piksel dan memiliki *channel* R,G,B. Data video yang digunakan diambil dari beberapa kejadian. Data video yang digunakan meliputi dua buah jenis video. Video dengan objek api dan video dengan objek bukan api. Jumlah video yang diuji berjumlah enam puluh tujuh video dengan jumlah video api sejumlah tiga puluh empat dan video bukan api berjumlah tiga puluh tiga. Contoh video kejadian dapat dilihat pada Gambar 5.1.

## Alur Uji Coba

Pada sub bab ini akan dijelaskan mengenai alur kerja dari sistem deteksi api. Dimulai dari *preprocessing* hingga verifikasi.

|  |  |  |  |
| --- | --- | --- | --- |
| Movie 1 | Movie 2 | Movie 3 | Movie 4 |
| Movie 5 | Movie 6 | Movie 7 | Movie 8 |
| Movie 9 | Movie 10 | Movie 11 | Movie 12 |
| Movie 13 | Movie 14 | Movie 15 | Movie 16 |

Gambar 5.28 Contoh Video Kejadian

### Preprocessing

Tahap *preprocessing* akan dijelaskan bagaimana alur setiap *frame* masuk hingga menghasilkan kandidat api yang selanjutnya akan di proses pada tahap verifikasi. Ilustrasi tahap *proprocessing* dapat dilihat pada Gambar 5.2.

|  |  |
| --- | --- |
| F:\KULIAH\TA\bukuTA\source\original_91.png | *Frame* Masukan |
|  |  |
| F:\KULIAH\TA\bukuTA\source\original_91.png | Reduksi *Size Frame* |
|  |  |
| F:\KULIAH\TA\bukuTA\source\moving_91.png | Deteksi Gerak |
|  |  |
| F:\KULIAH\TA\bukuTA\source\color_91.png | Deteksi Warna Piksel |
|  |  |
| F:\KULIAH\TA\bukuTA\source\region_91.png | *Region Growing* |
|  |  |
| F:\KULIAH\TA\bukuTA\source\size_91.png | Perhitungan Luasan *Region* |

Gambar 5.29 Tahap *Preprocessing*

### Verifikasi

Tahap verifikasi akan dijelaskan bagaimana alur verifikasi dilakukan. Masukan dari tahap ini adalah hasil akhir dari tahap *preprocessing*. Hasil akhir dari proses verifikasi adalah *region* yang masuk kedalam objek api. Ilustrasi proses verifikasi dapat dilihat pada Gambar 5.3.

|  |  |
| --- | --- |
| F:\KULIAH\TA\bukuTA\source\size_91.png | *Frame* Masukan dari proses *preprocessing* |
|  |  |
| |  | | --- | |  | |  | |  | | Ekstraksi Fitur, detail gambar vertikal, horizontal, digonal |
|  |  |
| F:\KULIAH\TA\firedetection\originial\-laporan- gambar\final_91.png | *Support Vector Machines* |
|  |  |
| F:\KULIAH\TA\bukuTA\source\fire_91.png | Menandai *Region* Api |

Gambar 5.30 Tahap Verifikasi

## Skenario Uji Coba

Pada sub bab ini akan dijelaskan mengenai skenario uji coba yang telah dilakukan. Telah dilakukan beberapa skenario uji coba, diantaranya yaitu:

1. Perbandingan hasil *true positif*, *false positif*, dan *missing rate* berdasarkan variasi nilai *threshold* pada deteksi warna api. *Threshold* yang akan diuji yaitu 10-7, 10-8, 5 x 10-9, 10-9.
2. Perbandingan hasil *true positif*, *false positif*, dan *missing rate* berdasarkan variasi nilai (*penalty error term*) pada klasifikasi dengan kernel tetap yaitu RBF. Nilai yang akan diuji yaitu 1, 3.5, 5,dan 7.
3. Perbandingan hasi *true positif*, *false positif*, dan *missing rate* berdasarkan variasi kernel yang digunakan pada klasifikasi. Kernel yang akan diuji yaitu *polynomial 2, polynomial 3*, dan RBF.
4. Perbandingan hasil *true positif*, *false positif*, dan *missing rate* berdasarkan variasi besarnya *region* objek. Variasi yang digunakan adalah 1%, 5%, dan 10%.
5. Perbandingan kecepatan deteksi dengan variasi *size frame* yang telah direduksi. *Size frame* yang diuji adalah 240 x 320, 120 x 160, 60 x 80.
6. Perbandingan hasil *true positif*, *false positif*, dan *missing rate* dengan menghilangkan tahap *region growing* dan perhitungan luasan *region*.

### Skenario Uji Coba 1

Skenario uji coba 1 adalah perhitungan *true positif*, *false positif*, dan *missing rate*. Dimana *true positif* adalah kondisi suatu *frame* mengandung gambar api dan terdeteksi api atau *frame* tidak mengandung api dan tidak terdeteksi api. *False positif* adalah kondisi dimana *frame* tidak mengandung gambar api, namun terdeteksi api dan *missing rate* adalah keadaan dimana suatu *frame* yang mempunyai gambar api namun tidak terdeteksi api. Pada skenario uji coba 1 dilakukan uji coba pada tahap probabilitas warna api dengan mengubah nilai *threshold* probabilitas piksel api. Nilai *threshold* yang diuji yaitu 10-7, 10-8, 5 x 10-9, 10-9. Untuk parameter nilai pada uji coba 1 diberikan nilai 5 menggunakan kernel RBF.

Tabel 5.3 Hasil Uji Coba 1

|  |  |  |  |
| --- | --- | --- | --- |
| *Threshold* | *True Positif (%)* | *False Positif (%)* | *Missing Rate (%)* |
| 10-7 | 77.58 | 0.37 | 22.05 |
| 10-8 | 94.58 | 1.22 | 4.19 |
| 5x10-9 | 96.32 | 1.46 | 2.23 |
| 10-9 | 91.95 | 7.85 | 0.20 |

Dari hasil uji yang dilakukan, semakin kecil nilai *threshold* yang digunakan, hasil dari *true positif* akan semakin besar. Begitu juga untuk *false positif*, dimana makin kecil nilai *threshold* makin besar nilai *false positif*. Hal ini dikarenakan piksel yang dianggap piksel api sudah melewati batas warna kuning hingga merah. Dari uji coba tersebut didapatkan nilai *threshold* 5x10-9 sebagai nilai terbaik, karena *False positif* yang dihasilkan tidak terlalu besar dan *True Positif*  bernilai besar. Hasil uji coba 1 lebih lengkap terdapat pada lampiran.

### Skenario Uji Coba 2

Skenario uji coba 2 dilakukan dengan menghitung nilai *true positif, false positif,* dan *missing rate*. Pada skenario uji 2 dilakukan uji coba variasi nilai pada klasifikasi, dimana nilai adalah nilai *penalty error term*. Nilai yang diuji yaitu 1, 3.5, 5, dan 7. Untuk parameter *threshold* warna piksel diberikan nilai 5 x 10-9 menggunakan kernel RBF sebagai klasifikasi. Hasil uji coba 2 lebih lengkap terdapat pada lampiran.

Tabel 5.4 Hasil Uji Coba 2

|  |  |  |  |
| --- | --- | --- | --- |
|  | *True Positif (%)* | *False Positif (%)* | *Missing Rate (%)* |
| 1 | 96.30 | 1.43 | 2.27 |
| 3.5 | 96.32 | 1.46 | 2.23 |
| 5 | 96.32 | 1.46 | 2.23 |
| 7 | 96.32 | 1.46 | 2.23 |

Hasil uji coba tahap 2 hasil terbaik didapatkan ketika nilai = 7, dimana memiliki nilai *true positif* yang lebih tinggi dari yang lainnya.

### Skenario Uji Coba 3

Pada skenario uji 3 dilakukan uji coba variasi kernel klasifikasi. Variasi kernel yang digunakan yaitu *polynomial 2, polynomial 3*, dan RBF. Untuk parameter *threshold* warna piksel diberikan nilai 5 x 10-9 dan nilai diberikan nilai 5.

Tabel 5.5 Hasil Uji Coba 3

|  |  |  |  |
| --- | --- | --- | --- |
| Kernel | *True Positif (%)* | *False Positif (%)* | *Missing Rate (%)* |
| *Polynomial 2* | 87.93 | 0.95 | 11.12 |
| *Polynomial 3* | 65.78 | 0.05 | 34.17 |
| RBF | 96.32 | 1.46 | 2.23 |

Hasil uji coba tahap 3 didapatkan nilai *true positif*  terbaik didapatkan dengan menggunakan kernel RBF. Hasil uji coba 3 lebih lengkap terdapat pada lampiran.

### Skenario Uji Coba 4

Pada skenario uji 4 dilakukan uji coba variasi besarnya *region*. Variasi *region* yang digunakan yaitu 1%, 5%, 10%. Untuk parameter *threshold* warna piksel diberikan nilai 5 x 10-9 , nilai diberikan nilai 5 dan menggunakan kernel RBF.

Tabel 5.6 Hasil Uji Coba 4

|  |  |  |  |
| --- | --- | --- | --- |
| Konstanta *Region* | *True Positif (%)* | *False Positif (%)* | *Missing Rate (%)* |
| 1% | 96.32 | 1.46 | 2.23 |
| 5% | 62.92 | 0.47 | 36.61 |
| 10% | 53.99 | 0.41 | 45.60 |

### Skenario Uji Coba 5

Pada skenario uji 5 dilakukan uji coba variasi *size frame*. Variasi *size frame* yang digunakan yaitu 240 x 320, 120 x 160, 60 x 80. Untuk parameter *threshold* warna piksel diberikan nilai 5 x 10-9 , nilai diberikan nilai 5 dan menggunakan kernel RBF.

Tabel 5.7 Hasil Uji Coba 5

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Piksel | *True Positif (%)* | *False Positif (%)* | *Missing Rate (%)* | *Execution Time (s)* |
| 240 x 320 | 92.96 | 0.79 | 6.25 | 123.03 |
| 120 x 160 | 96.32 | 1.46 | 2.23 | 35.89 |
| 60 x 80 | 96.40 | 3.22 | 0.37 | 11.75 |

### Skenario Uji Coba 6

Pada skenario uji coba 6, dilakukan Perbandingan hasil *true positif*, *false positif*, dan *missing rate* dengan menghilangkan tahap *region growing* dan perhitungan luasan *region*.

Tabel 5.8 Hasil Uji Coba 6

|  |  |  |  |
| --- | --- | --- | --- |
|  | *True Positif (%)* | *False Positif (%)* | *Missing Rate (%)* |
| Tanpa Menggunakan *region growing* dan perhitungan luasan *region* | 88.62 | 11.38 | 0.00 |
| Menggunakan *region growing* dan perhitungan luasan *region* | 96.32 | 1.46 | 2.23 |

## Analisis Hasil Uji Coba

Dari hasil skenario uji coba yang telah dilakukan, beberapa parameter memberikan pengaruh terhadap hasil deteksi. Parameter yang digunakan antara lain nilai *threshold* pada deteksi warna api dan nilai pada klasifikasi. Uji coba dilakukan dengan membandingkan nilai *true positif*, *false pisitif*, dan *missing rate*.

Dari uji coba 1, parameter yang di uji adalah *threshold* pada deteksi warna piksel. Hasil percobaan menunjukkan semakin tinggi nilai *threshold* yang digunakan, semakin besar *missing rate* yang dihasilkan. Hal ini dikarenakan semakin kecil kombinasi warna piksel yang dianggap sebagai warna api. Sebaliknya jika nilai *threshold* yang digunakan terlalu kecil, maka banyak piksel yang tidak termasuk warna api lolos sebagai kandidat piksel berwarna api. Nilai terbaik yang didapatkan dari hasil percobaan adalah nilai *threshold* 5 x 10-9.

Uji coba 2, dapat diambil kesimpulan bahwa nilai tidak berpengaruh besar pada klasifikasi. Pada uji coba 3, variasi kernel yang digunakan adalah *polynomial 3*, dan RBF. Hasil uji coba menunjukkan kernel terbaik dari variasi kernel yang digunakan adalah kernel RBF. Uji coba 4, dapat diambil kesimpulan jika konstanta yang digunakan terlalu besar banyak *region* yang dianggap *noise*. Pada uji coba 5, semakin kecil *size frame* yang diproses, waktu eksekusi yang diperlukan semakin kecil. Pada uji coba 6, penghilangan proses *region growing* dan perhitungan luasan *region* menurunkan hasil yang dikeluarkan. Hal ini disebabkan banyaknya piksel-piksel *noise* yang masuk kedalam piksel api.

Dari keseluruhan uji coba yang dilakukan, parameter-parameter tersebut menghasilkan presentase terbaik ketika *threshold* yang digunakan 5 x 10-9 dan yang digunakan sebesar 7 dan menggunakan kernel RBF.

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

# KESIMPULAN DAN SARAN

Pada bab ini akan dijelaskan mengenai kesimpulan dari proses dan uji coba dari program dan saran untuk pengembangan dari program itu sendiri.

## Kesimpulan

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

1. Reduksi *size frame* mempercepat proses deteksi. Dari hasil Skenario Uji Coba 5, reduksi *size frame* mempengaruhi hasil deteksi namun tidak terlalu besar.
2. Metode deteksi gerak menggunakan *gaussian mixture model* menyaring piksel-piksel pada *frame* dengan *thereshold* yang berbeda setiap piksel dan dapat beradaptasi dengan waktu. Sehingga piksel-piksel yang bergerak dapat disaring dengan baik.
3. Deteksi warna menyaring piksel-piksel yang tidak masuk kedalam *range* warna api menggunakan probabilitas distribusi gaussian menyaring warna piksel api dengan baik. *Threshold* terbaik didapatkan sebesar 5 x 10-9, dapat dilihat pada Skenario Uji Coba 1.
4. Metode perhitungan luasan *region* dapat menghilangkan *noise* dengan baik. Pada Skenario Uji Coba 6, dapat disimpulkan bahwa penggunaan perhitungan luasan *region* meningkatkan hasil deteksi.
5. Penggunaan kernel pada klasifikasi mempengaruhi hasil dari verifikasi piksel, dapat dilihat pada Skenario Uji Coba 3. Kernel terbaik pada Skenario Uji Coba 3 adalah RBF.
6. Hasil terbaik pada uji coba adalah menggunakan nilai *threshold =* 5 x 10-9 dan nilai = 7. Menghasilkan nilai *true positif* sebesar 96.32, *false positif* sebesar 1.46 dan *missing rate* sebesar 2.23.

## Saran

Saran yang diberikan untuk pengembangan perangkat lunak ini adalah :

1. Analisa fitur pada pross verifikasi perlu dilakukan analisa lebih lanjut.

# DAFTAR PUSTAKA

|  |  |
| --- | --- |
| [1] | K.-H. C. J.-Y. N. ByoungChul Ko, "Early fire detection algorithm based on irregular patterns of flames and," *Fire Safety Journal,* pp. 262-270, 2007. |
| [2] | R. B. P. KaewTraKulPong, "An Improved Adaptive Background Mixture Model for Real- time Tracking with Shadow Detection," Kluwer Academic Publishers, 2001. |
| [3] | "Images Pyramid Open CV," 2011-2014. [Online]. Available: http://docs.opencv.org/2.4/doc/tutorials/imgproc/pyramids/pyramids.html. [Accessed 6 1 2016]. |
| [4] | E. H. A. PETER J. BURT, "The Laplacian Pyramid as a Compact Image Code," *IEEE ,* Vols. COM-31, pp. 522-540, 1983. |
| [5] | I. S. T. Maria Isabel Ribeiro, "Gaussian Probability Density Functions: Properties and Error Characterization," 2004. |
| [6] | R. E. W. Refael C. Gonzalez, Digital Image Processing third edition, p. 785. |
| [7] | R. S. F. D. R. S. Lee A. Barford, "An Introduction to Wavelets," 1992. |
| [8] | B. K. HYERAN BYUN, "ROBUST FACE DETECTION AND TRACKING FOR REAL-LIFE APPLICATIONS," *International Journal of Pattern Recognition,* vol. 17, pp. 1035-1055, 2003. |
| [9] | C.-J. L. Chih-Chung Chang, "A Library for Support Vector Machines," Taipei, Taiwan, 2001. |
| [10] | S. T. Punam Patel, "Flame Detection using Image Processing Techniques," *International Journal of Computer Applications,* vol. 58, pp. 13-16, 2012. |

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

# LAMPIRAN A

Tabel A.9 Hasil Uji Coba Menggunakan Parameter *Threshold*  = 10-7, = 5 dan kernel RBF

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Nama File | True Positif | False Positif | Missing Rate |
| 1 | api-bakar\_sampah.avi | 6.25 | 0.00 | 93.75 |
| 2 | api-boneka\_dora.avi | 6.37 | 0.00 | 93.63 |
| 3 | api-kayu.avi | 66.18 | 0.00 | 33.82 |
| 4 | api-kayu2.avi | 77.94 | 0.00 | 22.06 |
| 5 | api-kebakaran\_barang.avi | 97.55 | 0.00 | 2.45 |
| 6 | api-kebakaran\_barang2.avi | 100.00 | 0.00 | 0.00 |
| 7 | api-kebakaran\_hutan.avi | 100.00 | 0.00 | 0.00 |
| 8 | api-kebakaran\_hutan2.avi | 100.00 | 0.00 | 0.00 |
| 9 | api-kebakaran\_hutan3.avi | 100.00 | 0.00 | 0.00 |
| 10 | api-kebakaran\_hutan4.avi | 0.48 | 0.00 | 99.52 |
| 11 | api-kebakaran\_ladang.avi | 98.22 | 0.00 | 1.78 |
| 12 | api-kebakaran\_mobil.avi | 98.04 | 0.00 | 1.96 |
| 13 | api-kebakaran\_mobil2.avi | 100.00 | 0.00 | 0.00 |
| 14 | api-kebakaran\_mobil3.avi | 95.59 | 0.00 | 4.41 |
| 15 | api-kebakaran\_mobil4.avi | 97.06 | 0.00 | 2.94 |
| 16 | api-kebakaran\_mobil5.avi | 47.55 | 0.00 | 52.45 |
| 17 | api-kebakaran\_mobil6.avi | 98.53 | 0.00 | 1.47 |
| 18 | api-kebakaran\_rumah.avi | 56.86 | 0.00 | 43.14 |
| 19 | api-kebakaran\_rumah2.avi | 60.29 | 0.00 | 39.71 |
| 20 | api-kebakaran\_rumah3.avi | 16.18 | 0.00 | 83.82 |
| 21 | api-kebakaran\_rumah4.avi | 1.47 | 0.00 | 98.53 |
| 22 | api-kebakaran\_tol.avi | 60.78 | 0.00 | 39.22 |
| 23 | api-kebakaran-truck.avi | 50.49 | 0.00 | 49.51 |
| 24 | api-kebakaran-truck2.avi | 62.75 | 0.00 | 37.25 |
| 25 | api-kebakaran-truck3.avi | 16.18 | 0.00 | 83.82 |
| 26 | api-kertas.avi | 0.00 | 0.00 | 100.00 |
| 27 | api-kertas2.avi | 4.94 | 0.00 | 95.06 |
| 28 | api-miniatur\_mainan.avi | 10.06 | 0.00 | 89.94 |
| 29 | api-mobil\_mainan.avi | 0.00 | 0.00 | 100.00 |
| 30 | api-orang\_terjun.avi | 46.08 | 0.00 | 53.92 |
| 31 | api-pesawat\_mainan.avi | 100.00 | 0.00 | 0.00 |
| 32 | api-pesawat\_mainan2.avi | 47.93 | 0.00 | 52.07 |
| 33 | api-ruang\_tamu.avi | 0.00 | 0.00 | 100.00 |
| 34 | api-rumah\_mainan.avi | 99.02 | 0.00 | 0.98 |
| 35 | non\_api-anak\_kecil.avi | 100.00 | 0.00 | 0.00 |
| 36 | non\_api-anak\_kecil2.avi | 100.00 | 0.00 | 0.00 |
| 37 | non\_api-anak\_kecil3.avi | 100.00 | 0.00 | 0.00 |
| 38 | non\_api-bertemu.avi | 100.00 | 0.00 | 0.00 |
| 39 | non\_api-jaket\_merah.avi | 100.00 | 0.00 | 0.00 |
| 40 | non\_api-jalan\_malam.avi | 100.00 | 0.00 | 0.00 |
| 41 | non\_api-jalan\_malam2.avi | 100.00 | 0.00 | 0.00 |
| 42 | non\_api-jalan\_raya.avi | 100.00 | 0.00 | 0.00 |
| 43 | non\_api-jalan\_raya2.avi | 100.00 | 0.00 | 0.00 |
| 44 | non\_api-jalan\_raya3.avi | 100.00 | 0.00 | 0.00 |
| 45 | non\_api-kantor.avi | 100.00 | 0.00 | 0.00 |
| 46 | non\_api-kecelakaan.avi | 100.00 | 0.00 | 0.00 |
| 47 | non\_api-kecelakaan2.avi | 100.00 | 0.00 | 0.00 |
| 48 | non\_api-kecelakaan3.avi | 100.00 | 0.00 | 0.00 |
| 49 | non\_api-kerusuhan.avi | 100.00 | 0.00 | 0.00 |
| 50 | non\_api-kerusuhan2.avi | 100.00 | 0.00 | 0.00 |
| 51 | non\_api-kerusuhan3.avi | 100.00 | 0.00 | 0.00 |
| 52 | non\_api-las\_vegas.avi | 96.91 | 3.09 | 0.00 |
| 53 | non\_api-manuju\_mobil.avi | 100.00 | 0.00 | 0.00 |
| 54 | non\_api-manuju\_mobil2.avi | 100.00 | 0.00 | 0.00 |
| 55 | non\_api-manuju\_mobil3.avi | 100.00 | 0.00 | 0.00 |
| 56 | non\_api-parkiran.avi | 100.00 | 0.00 | 0.00 |
| 57 | non\_api-parkiran2.avi | 100.00 | 0.00 | 0.00 |
| 58 | non\_api-parkiran3.avi | 100.00 | 0.00 | 0.00 |
| 59 | non\_api-pencuri.avi | 100.00 | 0.00 | 0.00 |
| 60 | non\_api-pencuri2.avi | 100.00 | 0.00 | 0.00 |
| 61 | non\_api-penembakan.avi | 100.00 | 0.00 | 0.00 |
| 62 | non\_api-serbet.avi | 78.45 | 21.55 | 0.00 |
| 63 | non\_api-tas.avi | 100.00 | 0.00 | 0.00 |
| 64 | non\_api-tas2.avi | 100.00 | 0.00 | 0.00 |
| 65 | non\_api-tas3.avi | 100.00 | 0.00 | 0.00 |
| 66 | non\_api-televisi.avi | 100.00 | 0.00 | 0.00 |
| 67 | non\_api-televisi2.avi | 100.00 | 0.00 | 0.00 |

Tabel A.10 Hasil Uji Coba Menggunakan Parameter *Threshold*  = 10-8, = 5 dan kernel RBF

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Nama File | True Positif | False Positif | Missing Rate |
| 1 | api-bakar\_sampah.avi | 89.06 | 0.00 | 10.94 |
| 2 | api-boneka\_dora.avi | 58.82 | 0.00 | 41.18 |
| 3 | api-kayu.avi | 88.73 | 0.00 | 11.27 |
| 4 | api-kayu2.avi | 100.00 | 0.00 | 0.00 |
| 5 | api-kebakaran\_barang.avi | 100.00 | 0.00 | 0.00 |
| 6 | api-kebakaran\_barang2.avi | 100.00 | 0.00 | 0.00 |
| 7 | api-kebakaran\_hutan.avi | 100.00 | 0.00 | 0.00 |
| 8 | api-kebakaran\_hutan2.avi | 100.00 | 0.00 | 0.00 |
| 9 | api-kebakaran\_hutan3.avi | 100.00 | 0.00 | 0.00 |
| 10 | api-kebakaran\_hutan4.avi | 100.00 | 0.00 | 0.00 |
| 11 | api-kebakaran\_ladang.avi | 100.00 | 0.00 | 0.00 |
| 12 | api-kebakaran\_mobil.avi | 100.00 | 0.00 | 0.00 |
| 13 | api-kebakaran\_mobil2.avi | 100.00 | 0.00 | 0.00 |
| 14 | api-kebakaran\_mobil3.avi | 100.00 | 0.00 | 0.00 |
| 15 | api-kebakaran\_mobil4.avi | 100.00 | 0.00 | 0.00 |
| 16 | api-kebakaran\_mobil5.avi | 100.00 | 0.00 | 0.00 |
| 17 | api-kebakaran\_mobil6.avi | 100.00 | 0.00 | 0.00 |
| 18 | api-kebakaran\_rumah.avi | 97.55 | 0.00 | 2.45 |
| 19 | api-kebakaran\_rumah2.avi | 100.00 | 0.00 | 0.00 |
| 20 | api-kebakaran\_rumah3.avi | 100.00 | 0.00 | 0.00 |
| 21 | api-kebakaran\_rumah4.avi | 71.57 | 0.00 | 28.43 |
| 22 | api-kebakaran\_tol.avi | 90.20 | 0.00 | 9.80 |
| 23 | api-kebakaran-truck.avi | 100.00 | 0.00 | 0.00 |
| 24 | api-kebakaran-truck2.avi | 100.00 | 0.00 | 0.00 |
| 25 | api-kebakaran-truck3.avi | 99.51 | 0.00 | 0.49 |
| 26 | api-kertas.avi | 59.88 | 0.00 | 40.12 |
| 27 | api-kertas2.avi | 72.22 | 0.00 | 27.78 |
| 28 | api-miniatur\_mainan.avi | 76.92 | 0.00 | 23.08 |
| 29 | api-mobil\_mainan.avi | 88.73 | 0.00 | 11.27 |
| 30 | api-orang\_terjun.avi | 74.51 | 0.00 | 25.49 |
| 31 | api-pesawat\_mainan.avi | 100.00 | 0.00 | 0.00 |
| 32 | api-pesawat\_mainan2.avi | 100.00 | 0.00 | 0.00 |
| 33 | api-ruang\_tamu.avi | 51.48 | 0.00 | 48.52 |
| 34 | api-rumah\_mainan.avi | 100.00 | 0.00 | 0.00 |
| 35 | non\_api-anak\_kecil.avi | 100.00 | 0.00 | 0.00 |
| 36 | non\_api-anak\_kecil2.avi | 100.00 | 0.00 | 0.00 |
| 37 | non\_api-anak\_kecil3.avi | 100.00 | 0.00 | 0.00 |
| 38 | non\_api-bertemu.avi | 100.00 | 0.00 | 0.00 |
| 39 | non\_api-jaket\_merah.avi | 100.00 | 0.00 | 0.00 |
| 40 | non\_api-jalan\_malam.avi | 100.00 | 0.00 | 0.00 |
| 41 | non\_api-jalan\_malam2.avi | 100.00 | 0.00 | 0.00 |
| 42 | non\_api-jalan\_raya.avi | 100.00 | 0.00 | 0.00 |
| 43 | non\_api-jalan\_raya2.avi | 100.00 | 0.00 | 0.00 |
| 44 | non\_api-jalan\_raya3.avi | 100.00 | 0.00 | 0.00 |
| 45 | non\_api-kantor.avi | 100.00 | 0.00 | 0.00 |
| 46 | non\_api-kecelakaan.avi | 100.00 | 0.00 | 0.00 |
| 47 | non\_api-kecelakaan2.avi | 100.00 | 0.00 | 0.00 |
| 48 | non\_api-kecelakaan3.avi | 100.00 | 0.00 | 0.00 |
| 49 | non\_api-kerusuhan.avi | 100.00 | 0.00 | 0.00 |
| 50 | non\_api-kerusuhan2.avi | 100.00 | 0.00 | 0.00 |
| 51 | non\_api-kerusuhan3.avi | 100.00 | 0.00 | 0.00 |
| 52 | non\_api-las\_vegas.avi | 51.23 | 48.77 | 0.00 |
| 53 | non\_api-manuju\_mobil.avi | 100.00 | 0.00 | 0.00 |
| 54 | non\_api-manuju\_mobil2.avi | 100.00 | 0.00 | 0.00 |
| 55 | non\_api-manuju\_mobil3.avi | 100.00 | 0.00 | 0.00 |
| 56 | non\_api-parkiran.avi | 100.00 | 0.00 | 0.00 |
| 57 | non\_api-parkiran2.avi | 100.00 | 0.00 | 0.00 |
| 58 | non\_api-parkiran3.avi | 100.00 | 0.00 | 0.00 |
| 59 | non\_api-pencuri.avi | 100.00 | 0.00 | 0.00 |
| 60 | non\_api-pencuri2.avi | 97.63 | 2.37 | 0.00 |
| 61 | non\_api-penembakan.avi | 100.00 | 0.00 | 0.00 |
| 62 | non\_api-serbet.avi | 69.06 | 30.94 | 0.00 |
| 63 | non\_api-tas.avi | 100.00 | 0.00 | 0.00 |
| 64 | non\_api-tas2.avi | 100.00 | 0.00 | 0.00 |
| 65 | non\_api-tas3.avi | 100.00 | 0.00 | 0.00 |
| 66 | non\_api-televisi.avi | 100.00 | 0.00 | 0.00 |
| 67 | non\_api-televisi2.avi | 100.00 | 0.00 | 0.00 |

Tabel A.11 Hasil Uji Coba Menggunakan Parameter *Threshold*  = 5x10-9, = 5 dan kernel RBF

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Nama File | True Positif | False Positif | Missing Rate |
| 1 | api-bakar\_sampah.avi | 93.75 | 0.00 | 6.25 |
| 2 | api-boneka\_dora.avi | 80.39 | 0.00 | 19.61 |
| 3 | api-kayu.avi | 93.63 | 0.00 | 6.37 |
| 4 | api-kayu2.avi | 100.00 | 0.00 | 0.00 |
| 5 | api-kebakaran\_barang.avi | 100.00 | 0.00 | 0.00 |
| 6 | api-kebakaran\_barang2.avi | 100.00 | 0.00 | 0.00 |
| 7 | api-kebakaran\_hutan.avi | 100.00 | 0.00 | 0.00 |
| 8 | api-kebakaran\_hutan2.avi | 100.00 | 0.00 | 0.00 |
| 9 | api-kebakaran\_hutan3.avi | 100.00 | 0.00 | 0.00 |
| 10 | api-kebakaran\_hutan4.avi | 100.00 | 0.00 | 0.00 |
| 11 | api-kebakaran\_ladang.avi | 100.00 | 0.00 | 0.00 |
| 12 | api-kebakaran\_mobil.avi | 100.00 | 0.00 | 0.00 |
| 13 | api-kebakaran\_mobil2.avi | 100.00 | 0.00 | 0.00 |
| 14 | api-kebakaran\_mobil3.avi | 100.00 | 0.00 | 0.00 |
| 15 | api-kebakaran\_mobil4.avi | 100.00 | 0.00 | 0.00 |
| 16 | api-kebakaran\_mobil5.avi | 100.00 | 0.00 | 0.00 |
| 17 | api-kebakaran\_mobil6.avi | 100.00 | 0.00 | 0.00 |
| 18 | api-kebakaran\_rumah.avi | 100.00 | 0.00 | 0.00 |
| 19 | api-kebakaran\_rumah2.avi | 100.00 | 0.00 | 0.00 |
| 20 | api-kebakaran\_rumah3.avi | 100.00 | 0.00 | 0.00 |
| 21 | api-kebakaran\_rumah4.avi | 79.90 | 0.00 | 20.10 |
| 22 | api-kebakaran\_tol.avi | 90.69 | 0.00 | 9.31 |
| 23 | api-kebakaran-truck.avi | 100.00 | 0.00 | 0.00 |
| 24 | api-kebakaran-truck2.avi | 100.00 | 0.00 | 0.00 |
| 25 | api-kebakaran-truck3.avi | 100.00 | 0.00 | 0.00 |
| 26 | api-kertas.avi | 67.28 | 0.00 | 32.72 |
| 27 | api-kertas2.avi | 98.77 | 0.00 | 1.23 |
| 28 | api-miniatur\_mainan.avi | 89.35 | 0.00 | 10.65 |
| 29 | api-mobil\_mainan.avi | 100.00 | 0.00 | 0.00 |
| 30 | api-orang\_terjun.avi | 79.41 | 0.00 | 20.59 |
| 31 | api-pesawat\_mainan.avi | 100.00 | 0.00 | 0.00 |
| 32 | api-pesawat\_mainan2.avi | 100.00 | 0.00 | 0.00 |
| 33 | api-ruang\_tamu.avi | 77.51 | 0.00 | 22.49 |
| 34 | api-rumah\_mainan.avi | 100.00 | 0.00 | 0.00 |
| 35 | non\_api-anak\_kecil.avi | 100.00 | 0.00 | 0.00 |
| 36 | non\_api-anak\_kecil2.avi | 100.00 | 0.00 | 0.00 |
| 37 | non\_api-anak\_kecil3.avi | 100.00 | 0.00 | 0.00 |
| 38 | non\_api-bertemu.avi | 100.00 | 0.00 | 0.00 |
| 39 | non\_api-jaket\_merah.avi | 100.00 | 0.00 | 0.00 |
| 40 | non\_api-jalan\_malam.avi | 100.00 | 0.00 | 0.00 |
| 41 | non\_api-jalan\_malam2.avi | 100.00 | 0.00 | 0.00 |
| 42 | non\_api-jalan\_raya.avi | 100.00 | 0.00 | 0.00 |
| 43 | non\_api-jalan\_raya2.avi | 100.00 | 0.00 | 0.00 |
| 44 | non\_api-jalan\_raya3.avi | 100.00 | 0.00 | 0.00 |
| 45 | non\_api-kantor.avi | 100.00 | 0.00 | 0.00 |
| 46 | non\_api-kecelakaan.avi | 100.00 | 0.00 | 0.00 |
| 47 | non\_api-kecelakaan2.avi | 100.00 | 0.00 | 0.00 |
| 48 | non\_api-kecelakaan3.avi | 100.00 | 0.00 | 0.00 |
| 49 | non\_api-kerusuhan.avi | 100.00 | 0.00 | 0.00 |
| 50 | non\_api-kerusuhan2.avi | 100.00 | 0.00 | 0.00 |
| 51 | non\_api-kerusuhan3.avi | 100.00 | 0.00 | 0.00 |
| 52 | non\_api-las\_vegas.avi | 46.30 | 53.70 | 0.00 |
| 53 | non\_api-manuju\_mobil.avi | 100.00 | 0.00 | 0.00 |
| 54 | non\_api-manuju\_mobil2.avi | 100.00 | 0.00 | 0.00 |
| 55 | non\_api-manuju\_mobil3.avi | 100.00 | 0.00 | 0.00 |
| 56 | non\_api-parkiran.avi | 100.00 | 0.00 | 0.00 |
| 57 | non\_api-parkiran2.avi | 100.00 | 0.00 | 0.00 |
| 58 | non\_api-parkiran3.avi | 100.00 | 0.00 | 0.00 |
| 59 | non\_api-pencuri.avi | 92.31 | 7.69 | 0.00 |
| 60 | non\_api-pencuri2.avi | 97.04 | 2.96 | 0.00 |
| 61 | non\_api-penembakan.avi | 100.00 | 0.00 | 0.00 |
| 62 | non\_api-serbet.avi | 66.85 | 33.15 | 0.00 |
| 63 | non\_api-tas.avi | 100.00 | 0.00 | 0.00 |
| 64 | non\_api-tas2.avi | 100.00 | 0.00 | 0.00 |
| 65 | non\_api-tas3.avi | 100.00 | 0.00 | 0.00 |
| 66 | non\_api-televisi.avi | 100.00 | 0.00 | 0.00 |
| 67 | non\_api-televisi2.avi | 100.00 | 0.00 | 0.00 |

Tabel A.12 Hasil Uji Coba Menggunakan Parameter *Threshold*  = 10-9, = 5 dan kernel RBF

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Nama File | True Positif | False Positif | Missing Rate |
| 1 | api-bakar\_sampah.avi | 100.00 | 0.00 | 0.00 |
| 2 | api-boneka\_dora.avi | 99.02 | 0.00 | 0.98 |
| 3 | api-kayu.avi | 100.00 | 0.00 | 0.00 |
| 4 | api-kayu2.avi | 100.00 | 0.00 | 0.00 |
| 5 | api-kebakaran\_barang.avi | 100.00 | 0.00 | 0.00 |
| 6 | api-kebakaran\_barang2.avi | 100.00 | 0.00 | 0.00 |
| 7 | api-kebakaran\_hutan.avi | 100.00 | 0.00 | 0.00 |
| 8 | api-kebakaran\_hutan2.avi | 100.00 | 0.00 | 0.00 |
| 9 | api-kebakaran\_hutan3.avi | 100.00 | 0.00 | 0.00 |
| 10 | api-kebakaran\_hutan4.avi | 100.00 | 0.00 | 0.00 |
| 11 | api-kebakaran\_ladang.avi | 100.00 | 0.00 | 0.00 |
| 12 | api-kebakaran\_mobil.avi | 100.00 | 0.00 | 0.00 |
| 13 | api-kebakaran\_mobil2.avi | 100.00 | 0.00 | 0.00 |
| 14 | api-kebakaran\_mobil3.avi | 100.00 | 0.00 | 0.00 |
| 15 | api-kebakaran\_mobil4.avi | 100.00 | 0.00 | 0.00 |
| 16 | api-kebakaran\_mobil5.avi | 100.00 | 0.00 | 0.00 |
| 17 | api-kebakaran\_mobil6.avi | 100.00 | 0.00 | 0.00 |
| 18 | api-kebakaran\_rumah.avi | 100.00 | 0.00 | 0.00 |
| 19 | api-kebakaran\_rumah2.avi | 100.00 | 0.00 | 0.00 |
| 20 | api-kebakaran\_rumah3.avi | 100.00 | 0.00 | 0.00 |
| 21 | api-kebakaran\_rumah4.avi | 100.00 | 0.00 | 0.00 |
| 22 | api-kebakaran\_tol.avi | 99.02 | 0.00 | 0.98 |
| 23 | api-kebakaran-truck.avi | 100.00 | 0.00 | 0.00 |
| 24 | api-kebakaran-truck2.avi | 100.00 | 0.00 | 0.00 |
| 25 | api-kebakaran-truck3.avi | 100.00 | 0.00 | 0.00 |
| 26 | api-kertas.avi | 96.30 | 0.00 | 3.70 |
| 27 | api-kertas2.avi | 100.00 | 0.00 | 0.00 |
| 28 | api-miniatur\_mainan.avi | 99.41 | 0.00 | 0.59 |
| 29 | api-mobil\_mainan.avi | 100.00 | 0.00 | 0.00 |
| 30 | api-orang\_terjun.avi | 94.61 | 0.00 | 5.39 |
| 31 | api-pesawat\_mainan.avi | 100.00 | 0.00 | 0.00 |
| 32 | api-pesawat\_mainan2.avi | 100.00 | 0.00 | 0.00 |
| 33 | api-ruang\_tamu.avi | 98.22 | 0.00 | 1.78 |
| 34 | api-rumah\_mainan.avi | 100.00 | 0.00 | 0.00 |
| 35 | non\_api-anak\_kecil.avi | 100.00 | 0.00 | 0.00 |
| 36 | non\_api-anak\_kecil2.avi | 100.00 | 0.00 | 0.00 |
| 37 | non\_api-anak\_kecil3.avi | 100.00 | 0.00 | 0.00 |
| 38 | non\_api-bertemu.avi | 100.00 | 0.00 | 0.00 |
| 39 | non\_api-jaket\_merah.avi | 100.00 | 0.00 | 0.00 |
| 40 | non\_api-jalan\_malam.avi | 100.00 | 0.00 | 0.00 |
| 41 | non\_api-jalan\_malam2.avi | 100.00 | 0.00 | 0.00 |
| 42 | non\_api-jalan\_raya.avi | 5.33 | 94.67 | 0.00 |
| 43 | non\_api-jalan\_raya2.avi | 87.57 | 12.43 | 0.00 |
| 44 | non\_api-jalan\_raya3.avi | 11.24 | 88.76 | 0.00 |
| 45 | non\_api-kantor.avi | 100.00 | 0.00 | 0.00 |
| 46 | non\_api-kecelakaan.avi | 100.00 | 0.00 | 0.00 |
| 47 | non\_api-kecelakaan2.avi | 100.00 | 0.00 | 0.00 |
| 48 | non\_api-kecelakaan3.avi | 100.00 | 0.00 | 0.00 |
| 49 | non\_api-kerusuhan.avi | 25.44 | 74.56 | 0.00 |
| 50 | non\_api-kerusuhan2.avi | 81.07 | 18.93 | 0.00 |
| 51 | non\_api-kerusuhan3.avi | 100.00 | 0.00 | 0.00 |
| 52 | non\_api-las\_vegas.avi | 43.21 | 56.79 | 0.00 |
| 53 | non\_api-manuju\_mobil.avi | 100.00 | 0.00 | 0.00 |
| 54 | non\_api-manuju\_mobil2.avi | 100.00 | 0.00 | 0.00 |
| 55 | non\_api-manuju\_mobil3.avi | 100.00 | 0.00 | 0.00 |
| 56 | non\_api-parkiran.avi | 100.00 | 0.00 | 0.00 |
| 57 | non\_api-parkiran2.avi | 100.00 | 0.00 | 0.00 |
| 58 | non\_api-parkiran3.avi | 100.00 | 0.00 | 0.00 |
| 59 | non\_api-pencuri.avi | 8.28 | 91.72 | 0.00 |
| 60 | non\_api-pencuri2.avi | 93.49 | 6.51 | 0.00 |
| 61 | non\_api-penembakan.avi | 97.06 | 2.94 | 0.00 |
| 62 | non\_api-serbet.avi | 21.55 | 78.45 | 0.00 |
| 63 | non\_api-tas.avi | 100.00 | 0.00 | 0.00 |
| 64 | non\_api-tas2.avi | 100.00 | 0.00 | 0.00 |
| 65 | non\_api-tas3.avi | 100.00 | 0.00 | 0.00 |
| 66 | non\_api-televisi.avi | 100.00 | 0.00 | 0.00 |
| 67 | non\_api-televisi2.avi | 100.00 | 0.00 | 0.00 |

Tabel A.13 Hasil Uji Coba Menggunakan Parameter *Threshold*  = 5x10-9, = 1 dan kernel RBF

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Nama File | True Positif | False Positif | Missing Rate |
| 1 | api-bakar\_sampah.avi | 92.19 | 0.00 | 7.81 |
| 2 | api-boneka\_dora.avi | 80.39 | 0.00 | 19.61 |
| 3 | api-kayu.avi | 93.63 | 0.00 | 6.37 |
| 4 | api-kayu2.avi | 100.00 | 0.00 | 0.00 |
| 5 | api-kebakaran\_barang.avi | 100.00 | 0.00 | 0.00 |
| 6 | api-kebakaran\_barang2.avi | 100.00 | 0.00 | 0.00 |
| 7 | api-kebakaran\_hutan.avi | 100.00 | 0.00 | 0.00 |
| 8 | api-kebakaran\_hutan2.avi | 100.00 | 0.00 | 0.00 |
| 9 | api-kebakaran\_hutan3.avi | 100.00 | 0.00 | 0.00 |
| 10 | api-kebakaran\_hutan4.avi | 100.00 | 0.00 | 0.00 |
| 11 | api-kebakaran\_ladang.avi | 100.00 | 0.00 | 0.00 |
| 12 | api-kebakaran\_mobil.avi | 100.00 | 0.00 | 0.00 |
| 13 | api-kebakaran\_mobil2.avi | 100.00 | 0.00 | 0.00 |
| 14 | api-kebakaran\_mobil3.avi | 100.00 | 0.00 | 0.00 |
| 15 | api-kebakaran\_mobil4.avi | 100.00 | 0.00 | 0.00 |
| 16 | api-kebakaran\_mobil5.avi | 100.00 | 0.00 | 0.00 |
| 17 | api-kebakaran\_mobil6.avi | 100.00 | 0.00 | 0.00 |
| 18 | api-kebakaran\_rumah.avi | 100.00 | 0.00 | 0.00 |
| 19 | api-kebakaran\_rumah2.avi | 100.00 | 0.00 | 0.00 |
| 20 | api-kebakaran\_rumah3.avi | 100.00 | 0.00 | 0.00 |
| 21 | api-kebakaran\_rumah4.avi | 78.92 | 0.00 | 21.08 |
| 22 | api-kebakaran\_tol.avi | 90.69 | 0.00 | 9.31 |
| 23 | api-kebakaran-truck.avi | 100.00 | 0.00 | 0.00 |
| 24 | api-kebakaran-truck2.avi | 100.00 | 0.00 | 0.00 |
| 25 | api-kebakaran-truck3.avi | 100.00 | 0.00 | 0.00 |
| 26 | api-kertas.avi | 67.28 | 0.00 | 32.72 |
| 27 | api-kertas2.avi | 98.77 | 0.00 | 1.23 |
| 28 | api-miniatur\_mainan.avi | 89.35 | 0.00 | 10.65 |
| 29 | api-mobil\_mainan.avi | 100.00 | 0.00 | 0.00 |
| 30 | api-orang\_terjun.avi | 79.41 | 0.00 | 20.59 |
| 31 | api-pesawat\_mainan.avi | 100.00 | 0.00 | 0.00 |
| 32 | api-pesawat\_mainan2.avi | 100.00 | 0.00 | 0.00 |
| 33 | api-ruang\_tamu.avi | 77.51 | 0.00 | 22.49 |
| 34 | api-rumah\_mainan.avi | 100.00 | 0.00 | 0.00 |
| 35 | non\_api-anak\_kecil.avi | 100.00 | 0.00 | 0.00 |
| 36 | non\_api-anak\_kecil2.avi | 100.00 | 0.00 | 0.00 |
| 37 | non\_api-anak\_kecil3.avi | 100.00 | 0.00 | 0.00 |
| 38 | non\_api-bertemu.avi | 100.00 | 0.00 | 0.00 |
| 39 | non\_api-jaket\_merah.avi | 100.00 | 0.00 | 0.00 |
| 40 | non\_api-jalan\_malam.avi | 100.00 | 0.00 | 0.00 |
| 41 | non\_api-jalan\_malam2.avi | 100.00 | 0.00 | 0.00 |
| 42 | non\_api-jalan\_raya.avi | 100.00 | 0.00 | 0.00 |
| 43 | non\_api-jalan\_raya2.avi | 100.00 | 0.00 | 0.00 |
| 44 | non\_api-jalan\_raya3.avi | 100.00 | 0.00 | 0.00 |
| 45 | non\_api-kantor.avi | 100.00 | 0.00 | 0.00 |
| 46 | non\_api-kecelakaan.avi | 100.00 | 0.00 | 0.00 |
| 47 | non\_api-kecelakaan2.avi | 100.00 | 0.00 | 0.00 |
| 48 | non\_api-kecelakaan3.avi | 100.00 | 0.00 | 0.00 |
| 49 | non\_api-kerusuhan.avi | 100.00 | 0.00 | 0.00 |
| 50 | non\_api-kerusuhan2.avi | 100.00 | 0.00 | 0.00 |
| 51 | non\_api-kerusuhan3.avi | 100.00 | 0.00 | 0.00 |
| 52 | non\_api-las\_vegas.avi | 46.30 | 53.70 | 0.00 |
| 53 | non\_api-manuju\_mobil.avi | 100.00 | 0.00 | 0.00 |
| 54 | non\_api-manuju\_mobil2.avi | 100.00 | 0.00 | 0.00 |
| 55 | non\_api-manuju\_mobil3.avi | 100.00 | 0.00 | 0.00 |
| 56 | non\_api-parkiran.avi | 100.00 | 0.00 | 0.00 |
| 57 | non\_api-parkiran2.avi | 100.00 | 0.00 | 0.00 |
| 58 | non\_api-parkiran3.avi | 100.00 | 0.00 | 0.00 |
| 59 | non\_api-pencuri.avi | 92.31 | 7.69 | 0.00 |
| 60 | non\_api-pencuri2.avi | 97.04 | 2.96 | 0.00 |
| 61 | non\_api-penembakan.avi | 100.00 | 0.00 | 0.00 |
| 62 | non\_api-serbet.avi | 68.51 | 31.49 | 0.00 |
| 63 | non\_api-tas.avi | 100.00 | 0.00 | 0.00 |
| 64 | non\_api-tas2.avi | 100.00 | 0.00 | 0.00 |
| 65 | non\_api-tas3.avi | 100.00 | 0.00 | 0.00 |
| 66 | non\_api-televisi.avi | 100.00 | 0.00 | 0.00 |
| 67 | non\_api-televisi2.avi | 100.00 | 0.00 | 0.00 |

Tabel A.14 Hasil Uji Coba Menggunakan Parameter *Threshold*  = 5x10-9, = 3.5 dan kernel RBF

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Nama File | True Positif | False Positif | Missing Rate |
| 1 | api-bakar\_sampah.avi | 93.75 | 0.00 | 6.25 |
| 2 | api-boneka\_dora.avi | 80.39 | 0.00 | 19.61 |
| 3 | api-kayu.avi | 93.63 | 0.00 | 6.37 |
| 4 | api-kayu2.avi | 100.00 | 0.00 | 0.00 |
| 5 | api-kebakaran\_barang.avi | 100.00 | 0.00 | 0.00 |
| 6 | api-kebakaran\_barang2.avi | 100.00 | 0.00 | 0.00 |
| 7 | api-kebakaran\_hutan.avi | 100.00 | 0.00 | 0.00 |
| 8 | api-kebakaran\_hutan2.avi | 100.00 | 0.00 | 0.00 |
| 9 | api-kebakaran\_hutan3.avi | 100.00 | 0.00 | 0.00 |
| 10 | api-kebakaran\_hutan4.avi | 100.00 | 0.00 | 0.00 |
| 11 | api-kebakaran\_ladang.avi | 100.00 | 0.00 | 0.00 |
| 12 | api-kebakaran\_mobil.avi | 100.00 | 0.00 | 0.00 |
| 13 | api-kebakaran\_mobil2.avi | 100.00 | 0.00 | 0.00 |
| 14 | api-kebakaran\_mobil3.avi | 100.00 | 0.00 | 0.00 |
| 15 | api-kebakaran\_mobil4.avi | 100.00 | 0.00 | 0.00 |
| 16 | api-kebakaran\_mobil5.avi | 100.00 | 0.00 | 0.00 |
| 17 | api-kebakaran\_mobil6.avi | 100.00 | 0.00 | 0.00 |
| 18 | api-kebakaran\_rumah.avi | 100.00 | 0.00 | 0.00 |
| 19 | api-kebakaran\_rumah2.avi | 100.00 | 0.00 | 0.00 |
| 20 | api-kebakaran\_rumah3.avi | 100.00 | 0.00 | 0.00 |
| 21 | api-kebakaran\_rumah4.avi | 79.90 | 0.00 | 20.10 |
| 22 | api-kebakaran\_tol.avi | 90.69 | 0.00 | 9.31 |
| 23 | api-kebakaran-truck.avi | 100.00 | 0.00 | 0.00 |
| 24 | api-kebakaran-truck2.avi | 100.00 | 0.00 | 0.00 |
| 25 | api-kebakaran-truck3.avi | 100.00 | 0.00 | 0.00 |
| 26 | api-kertas.avi | 67.28 | 0.00 | 32.72 |
| 27 | api-kertas2.avi | 98.77 | 0.00 | 1.23 |
| 28 | api-miniatur\_mainan.avi | 89.35 | 0.00 | 10.65 |
| 29 | api-mobil\_mainan.avi | 100.00 | 0.00 | 0.00 |
| 30 | api-orang\_terjun.avi | 79.41 | 0.00 | 20.59 |
| 31 | api-pesawat\_mainan.avi | 100.00 | 0.00 | 0.00 |
| 32 | api-pesawat\_mainan2.avi | 100.00 | 0.00 | 0.00 |
| 33 | api-ruang\_tamu.avi | 77.51 | 0.00 | 22.49 |
| 34 | api-rumah\_mainan.avi | 100.00 | 0.00 | 0.00 |
| 35 | non\_api-anak\_kecil.avi | 100.00 | 0.00 | 0.00 |
| 36 | non\_api-anak\_kecil2.avi | 100.00 | 0.00 | 0.00 |
| 37 | non\_api-anak\_kecil3.avi | 100.00 | 0.00 | 0.00 |
| 38 | non\_api-bertemu.avi | 100.00 | 0.00 | 0.00 |
| 39 | non\_api-jaket\_merah.avi | 100.00 | 0.00 | 0.00 |
| 40 | non\_api-jalan\_malam.avi | 100.00 | 0.00 | 0.00 |
| 41 | non\_api-jalan\_malam2.avi | 100.00 | 0.00 | 0.00 |
| 42 | non\_api-jalan\_raya.avi | 100.00 | 0.00 | 0.00 |
| 43 | non\_api-jalan\_raya2.avi | 100.00 | 0.00 | 0.00 |
| 44 | non\_api-jalan\_raya3.avi | 100.00 | 0.00 | 0.00 |
| 45 | non\_api-kantor.avi | 100.00 | 0.00 | 0.00 |
| 46 | non\_api-kecelakaan.avi | 100.00 | 0.00 | 0.00 |
| 47 | non\_api-kecelakaan2.avi | 100.00 | 0.00 | 0.00 |
| 48 | non\_api-kecelakaan3.avi | 100.00 | 0.00 | 0.00 |
| 49 | non\_api-kerusuhan.avi | 100.00 | 0.00 | 0.00 |
| 50 | non\_api-kerusuhan2.avi | 100.00 | 0.00 | 0.00 |
| 51 | non\_api-kerusuhan3.avi | 100.00 | 0.00 | 0.00 |
| 52 | non\_api-las\_vegas.avi | 46.30 | 53.70 | 0.00 |
| 53 | non\_api-manuju\_mobil.avi | 100.00 | 0.00 | 0.00 |
| 54 | non\_api-manuju\_mobil2.avi | 100.00 | 0.00 | 0.00 |
| 55 | non\_api-manuju\_mobil3.avi | 100.00 | 0.00 | 0.00 |
| 56 | non\_api-parkiran.avi | 100.00 | 0.00 | 0.00 |
| 57 | non\_api-parkiran2.avi | 100.00 | 0.00 | 0.00 |
| 58 | non\_api-parkiran3.avi | 100.00 | 0.00 | 0.00 |
| 59 | non\_api-pencuri.avi | 92.31 | 7.69 | 0.00 |
| 60 | non\_api-pencuri2.avi | 97.04 | 2.96 | 0.00 |
| 61 | non\_api-penembakan.avi | 100.00 | 0.00 | 0.00 |
| 62 | non\_api-serbet.avi | 66.85 | 33.15 | 0.00 |
| 63 | non\_api-tas.avi | 100.00 | 0.00 | 0.00 |
| 64 | non\_api-tas2.avi | 100.00 | 0.00 | 0.00 |
| 65 | non\_api-tas3.avi | 100.00 | 0.00 | 0.00 |
| 66 | non\_api-televisi.avi | 100.00 | 0.00 | 0.00 |
| 67 | non\_api-televisi2.avi | 100.00 | 0.00 | 0.00 |

Tabel A.15 Hasil Uji Coba Menggunakan Parameter *Threshold*  = 5x10-9, = 7 dan kernel RBF

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Nama File | True Positif | False Positif | Missing Rate |
| 1 | api-bakar\_sampah.avi | 93.75 | 0.00 | 6.25 |
| 2 | api-boneka\_dora.avi | 79.90 | 0.00 | 20.10 |
| 3 | api-kayu.avi | 93.63 | 0.00 | 6.37 |
| 4 | api-kayu2.avi | 100.00 | 0.00 | 0.00 |
| 5 | api-kebakaran\_barang.avi | 100.00 | 0.00 | 0.00 |
| 6 | api-kebakaran\_barang2.avi | 100.00 | 0.00 | 0.00 |
| 7 | api-kebakaran\_hutan.avi | 100.00 | 0.00 | 0.00 |
| 8 | api-kebakaran\_hutan2.avi | 100.00 | 0.00 | 0.00 |
| 9 | api-kebakaran\_hutan3.avi | 100.00 | 0.00 | 0.00 |
| 10 | api-kebakaran\_hutan4.avi | 100.00 | 0.00 | 0.00 |
| 11 | api-kebakaran\_ladang.avi | 100.00 | 0.00 | 0.00 |
| 12 | api-kebakaran\_mobil.avi | 100.00 | 0.00 | 0.00 |
| 13 | api-kebakaran\_mobil2.avi | 100.00 | 0.00 | 0.00 |
| 14 | api-kebakaran\_mobil3.avi | 100.00 | 0.00 | 0.00 |
| 15 | api-kebakaran\_mobil4.avi | 100.00 | 0.00 | 0.00 |
| 16 | api-kebakaran\_mobil5.avi | 100.00 | 0.00 | 0.00 |
| 17 | api-kebakaran\_mobil6.avi | 100.00 | 0.00 | 0.00 |
| 18 | api-kebakaran\_rumah.avi | 100.00 | 0.00 | 0.00 |
| 19 | api-kebakaran\_rumah2.avi | 100.00 | 0.00 | 0.00 |
| 20 | api-kebakaran\_rumah3.avi | 100.00 | 0.00 | 0.00 |
| 21 | api-kebakaran\_rumah4.avi | 80.39 | 0.00 | 19.61 |
| 22 | api-kebakaran\_tol.avi | 90.69 | 0.00 | 9.31 |
| 23 | api-kebakaran-truck.avi | 100.00 | 0.00 | 0.00 |
| 24 | api-kebakaran-truck2.avi | 100.00 | 0.00 | 0.00 |
| 25 | api-kebakaran-truck3.avi | 100.00 | 0.00 | 0.00 |
| 26 | api-kertas.avi | 67.28 | 0.00 | 32.72 |
| 27 | api-kertas2.avi | 98.77 | 0.00 | 1.23 |
| 28 | api-miniatur\_mainan.avi | 89.35 | 0.00 | 10.65 |
| 29 | api-mobil\_mainan.avi | 100.00 | 0.00 | 0.00 |
| 30 | api-orang\_terjun.avi | 79.41 | 0.00 | 20.59 |
| 31 | api-pesawat\_mainan.avi | 100.00 | 0.00 | 0.00 |
| 32 | api-pesawat\_mainan2.avi | 100.00 | 0.00 | 0.00 |
| 33 | api-ruang\_tamu.avi | 77.51 | 0.00 | 22.49 |
| 34 | api-rumah\_mainan.avi | 100.00 | 0.00 | 0.00 |
| 35 | non\_api-anak\_kecil.avi | 100.00 | 0.00 | 0.00 |
| 36 | non\_api-anak\_kecil2.avi | 100.00 | 0.00 | 0.00 |
| 37 | non\_api-anak\_kecil3.avi | 100.00 | 0.00 | 0.00 |
| 38 | non\_api-bertemu.avi | 100.00 | 0.00 | 0.00 |
| 39 | non\_api-jaket\_merah.avi | 100.00 | 0.00 | 0.00 |
| 40 | non\_api-jalan\_malam.avi | 100.00 | 0.00 | 0.00 |
| 41 | non\_api-jalan\_malam2.avi | 100.00 | 0.00 | 0.00 |
| 42 | non\_api-jalan\_raya.avi | 100.00 | 0.00 | 0.00 |
| 43 | non\_api-jalan\_raya2.avi | 100.00 | 0.00 | 0.00 |
| 44 | non\_api-jalan\_raya3.avi | 100.00 | 0.00 | 0.00 |
| 45 | non\_api-kantor.avi | 100.00 | 0.00 | 0.00 |
| 46 | non\_api-kecelakaan.avi | 100.00 | 0.00 | 0.00 |
| 47 | non\_api-kecelakaan2.avi | 100.00 | 0.00 | 0.00 |
| 48 | non\_api-kecelakaan3.avi | 100.00 | 0.00 | 0.00 |
| 49 | non\_api-kerusuhan.avi | 100.00 | 0.00 | 0.00 |
| 50 | non\_api-kerusuhan2.avi | 100.00 | 0.00 | 0.00 |
| 51 | non\_api-kerusuhan3.avi | 100.00 | 0.00 | 0.00 |
| 52 | non\_api-las\_vegas.avi | 46.30 | 53.70 | 0.00 |
| 53 | non\_api-manuju\_mobil.avi | 100.00 | 0.00 | 0.00 |
| 54 | non\_api-manuju\_mobil2.avi | 100.00 | 0.00 | 0.00 |
| 55 | non\_api-manuju\_mobil3.avi | 100.00 | 0.00 | 0.00 |
| 56 | non\_api-parkiran.avi | 100.00 | 0.00 | 0.00 |
| 57 | non\_api-parkiran2.avi | 100.00 | 0.00 | 0.00 |
| 58 | non\_api-parkiran3.avi | 100.00 | 0.00 | 0.00 |
| 59 | non\_api-pencuri.avi | 92.31 | 7.69 | 0.00 |
| 60 | non\_api-pencuri2.avi | 97.04 | 2.96 | 0.00 |
| 61 | non\_api-penembakan.avi | 100.00 | 0.00 | 0.00 |
| 62 | non\_api-serbet.avi | 66.85 | 33.15 | 0.00 |
| 63 | non\_api-tas.avi | 100.00 | 0.00 | 0.00 |
| 64 | non\_api-tas2.avi | 100.00 | 0.00 | 0.00 |
| 65 | non\_api-tas3.avi | 100.00 | 0.00 | 0.00 |
| 66 | non\_api-televisi.avi | 100.00 | 0.00 | 0.00 |
| 67 | non\_api-televisi2.avi | 100.00 | 0.00 | 0.00 |

Tabel A.16 Hasil Uji Coba Menggunakan Parameter *Threshold*  = 5x10-9, = 5 dan kernel *Polynomial 2*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Nama File | True Positif | False Positif | Missing Rate |
| 1 | api-bakar\_sampah.avi | 12.50 | 0.00 | 87.50 |
| 2 | api-boneka\_dora.avi | 58.33 | 0.00 | 41.67 |
| 3 | api-kayu.avi | 90.69 | 0.00 | 9.31 |
| 4 | api-kayu2.avi | 98.04 | 0.00 | 1.96 |
| 5 | api-kebakaran\_barang.avi | 68.14 | 0.00 | 31.86 |
| 6 | api-kebakaran\_barang2.avi | 88.73 | 0.00 | 11.27 |
| 7 | api-kebakaran\_hutan.avi | 96.97 | 0.00 | 3.03 |
| 8 | api-kebakaran\_hutan2.avi | 100.00 | 0.00 | 0.00 |
| 9 | api-kebakaran\_hutan3.avi | 100.00 | 0.00 | 0.00 |
| 10 | api-kebakaran\_hutan4.avi | 100.00 | 0.00 | 0.00 |
| 11 | api-kebakaran\_ladang.avi | 100.00 | 0.00 | 0.00 |
| 12 | api-kebakaran\_mobil.avi | 100.00 | 0.00 | 0.00 |
| 13 | api-kebakaran\_mobil2.avi | 53.43 | 0.00 | 46.57 |
| 14 | api-kebakaran\_mobil3.avi | 50.00 | 0.00 | 50.00 |
| 15 | api-kebakaran\_mobil4.avi | 94.61 | 0.00 | 5.39 |
| 16 | api-kebakaran\_mobil5.avi | 100.00 | 0.00 | 0.00 |
| 17 | api-kebakaran\_mobil6.avi | 100.00 | 0.00 | 0.00 |
| 18 | api-kebakaran\_rumah.avi | 50.00 | 0.00 | 50.00 |
| 19 | api-kebakaran\_rumah2.avi | 41.67 | 0.00 | 58.33 |
| 20 | api-kebakaran\_rumah3.avi | 6.86 | 0.00 | 93.14 |
| 21 | api-kebakaran\_rumah4.avi | 65.20 | 0.00 | 34.80 |
| 22 | api-kebakaran\_tol.avi | 47.55 | 0.00 | 52.45 |
| 23 | api-kebakaran-truck.avi | 100.00 | 0.00 | 0.00 |
| 24 | api-kebakaran-truck2.avi | 100.00 | 0.00 | 0.00 |
| 25 | api-kebakaran-truck3.avi | 99.51 | 0.00 | 0.49 |
| 26 | api-kertas.avi | 64.20 | 0.00 | 35.80 |
| 27 | api-kertas2.avi | 98.77 | 0.00 | 1.23 |
| 28 | api-miniatur\_mainan.avi | 72.19 | 0.00 | 27.81 |
| 29 | api-mobil\_mainan.avi | 99.02 | 0.00 | 0.98 |
| 30 | api-orang\_terjun.avi | 77.45 | 0.00 | 22.55 |
| 31 | api-pesawat\_mainan.avi | 92.36 | 0.00 | 7.64 |
| 32 | api-pesawat\_mainan2.avi | 98.82 | 0.00 | 1.18 |
| 33 | api-ruang\_tamu.avi | 30.18 | 0.00 | 69.82 |
| 34 | api-rumah\_mainan.avi | 100.00 | 0.00 | 0.00 |
| 35 | non\_api-anak\_kecil.avi | 100.00 | 0.00 | 0.00 |
| 36 | non\_api-anak\_kecil2.avi | 100.00 | 0.00 | 0.00 |
| 37 | non\_api-anak\_kecil3.avi | 100.00 | 0.00 | 0.00 |
| 38 | non\_api-bertemu.avi | 100.00 | 0.00 | 0.00 |
| 39 | non\_api-jaket\_merah.avi | 100.00 | 0.00 | 0.00 |
| 40 | non\_api-jalan\_malam.avi | 100.00 | 0.00 | 0.00 |
| 41 | non\_api-jalan\_malam2.avi | 100.00 | 0.00 | 0.00 |
| 42 | non\_api-jalan\_raya.avi | 100.00 | 0.00 | 0.00 |
| 43 | non\_api-jalan\_raya2.avi | 100.00 | 0.00 | 0.00 |
| 44 | non\_api-jalan\_raya3.avi | 100.00 | 0.00 | 0.00 |
| 45 | non\_api-kantor.avi | 100.00 | 0.00 | 0.00 |
| 46 | non\_api-kecelakaan.avi | 100.00 | 0.00 | 0.00 |
| 47 | non\_api-kecelakaan2.avi | 100.00 | 0.00 | 0.00 |
| 48 | non\_api-kecelakaan3.avi | 100.00 | 0.00 | 0.00 |
| 49 | non\_api-kerusuhan.avi | 100.00 | 0.00 | 0.00 |
| 50 | non\_api-kerusuhan2.avi | 100.00 | 0.00 | 0.00 |
| 51 | non\_api-kerusuhan3.avi | 100.00 | 0.00 | 0.00 |
| 52 | non\_api-las\_vegas.avi | 48.15 | 51.85 | 0.00 |
| 53 | non\_api-manuju\_mobil.avi | 100.00 | 0.00 | 0.00 |
| 54 | non\_api-manuju\_mobil2.avi | 100.00 | 0.00 | 0.00 |
| 55 | non\_api-manuju\_mobil3.avi | 100.00 | 0.00 | 0.00 |
| 56 | non\_api-parkiran.avi | 100.00 | 0.00 | 0.00 |
| 57 | non\_api-parkiran2.avi | 100.00 | 0.00 | 0.00 |
| 58 | non\_api-parkiran3.avi | 100.00 | 0.00 | 0.00 |
| 59 | non\_api-pencuri.avi | 95.86 | 4.14 | 0.00 |
| 60 | non\_api-pencuri2.avi | 100.00 | 0.00 | 0.00 |
| 61 | non\_api-penembakan.avi | 100.00 | 0.00 | 0.00 |
| 62 | non\_api-serbet.avi | 92.27 | 7.73 | 0.00 |
| 63 | non\_api-tas.avi | 100.00 | 0.00 | 0.00 |
| 64 | non\_api-tas2.avi | 100.00 | 0.00 | 0.00 |
| 65 | non\_api-tas3.avi | 100.00 | 0.00 | 0.00 |
| 66 | non\_api-televisi.avi | 100.00 | 0.00 | 0.00 |
| 67 | non\_api-televisi2.avi | 100.00 | 0.00 | 0.00 |

Tabel A.17 Hasil Uji Coba Menggunakan Parameter *Threshold*  = 5x10-9, = 5 dan kernel *Polynomial 3*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Nama File | True Positif | False Positif | Missing Rate |
| 1 | api-bakar\_sampah.avi | 0.00 | 0.00 | 100.00 |
| 2 | api-boneka\_dora.avi | 1.47 | 0.00 | 98.53 |
| 3 | api-kayu.avi | 8.33 | 0.00 | 91.67 |
| 4 | api-kayu2.avi | 10.78 | 0.00 | 89.22 |
| 5 | api-kebakaran\_barang.avi | 0.00 | 0.00 | 100.00 |
| 6 | api-kebakaran\_barang2.avi | 0.00 | 0.00 | 100.00 |
| 7 | api-kebakaran\_hutan.avi | 20.20 | 0.00 | 79.80 |
| 8 | api-kebakaran\_hutan2.avi | 86.87 | 0.00 | 13.13 |
| 9 | api-kebakaran\_hutan3.avi | 32.32 | 0.00 | 67.68 |
| 10 | api-kebakaran\_hutan4.avi | 25.00 | 0.00 | 75.00 |
| 11 | api-kebakaran\_ladang.avi | 100.00 | 0.00 | 0.00 |
| 12 | api-kebakaran\_mobil.avi | 24.02 | 0.00 | 75.98 |
| 13 | api-kebakaran\_mobil2.avi | 2.45 | 0.00 | 97.55 |
| 14 | api-kebakaran\_mobil3.avi | 4.90 | 0.00 | 95.10 |
| 15 | api-kebakaran\_mobil4.avi | 5.88 | 0.00 | 94.12 |
| 16 | api-kebakaran\_mobil5.avi | 91.18 | 0.00 | 8.82 |
| 17 | api-kebakaran\_mobil6.avi | 97.55 | 0.00 | 2.45 |
| 18 | api-kebakaran\_rumah.avi | 2.45 | 0.00 | 97.55 |
| 19 | api-kebakaran\_rumah2.avi | 0.00 | 0.00 | 100.00 |
| 20 | api-kebakaran\_rumah3.avi | 0.00 | 0.00 | 100.00 |
| 21 | api-kebakaran\_rumah4.avi | 54.41 | 0.00 | 45.59 |
| 22 | api-kebakaran\_tol.avi | 0.00 | 0.00 | 100.00 |
| 23 | api-kebakaran-truck.avi | 79.90 | 0.00 | 20.10 |
| 24 | api-kebakaran-truck2.avi | 90.20 | 0.00 | 9.80 |
| 25 | api-kebakaran-truck3.avi | 81.37 | 0.00 | 18.63 |
| 26 | api-kertas.avi | 58.64 | 0.00 | 41.36 |
| 27 | api-kertas2.avi | 55.56 | 0.00 | 44.44 |
| 28 | api-miniatur\_mainan.avi | 2.96 | 0.00 | 97.04 |
| 29 | api-mobil\_mainan.avi | 38.24 | 0.00 | 61.76 |
| 30 | api-orang\_terjun.avi | 2.45 | 0.00 | 97.55 |
| 31 | api-pesawat\_mainan.avi | 21.53 | 0.00 | 78.47 |
| 32 | api-pesawat\_mainan2.avi | 40.24 | 0.00 | 59.76 |
| 33 | api-ruang\_tamu.avi | 2.37 | 0.00 | 97.63 |
| 34 | api-rumah\_mainan.avi | 69.61 | 0.00 | 30.39 |
| 35 | non\_api-anak\_kecil.avi | 100.00 | 0.00 | 0.00 |
| 36 | non\_api-anak\_kecil2.avi | 100.00 | 0.00 | 0.00 |
| 37 | non\_api-anak\_kecil3.avi | 100.00 | 0.00 | 0.00 |
| 38 | non\_api-bertemu.avi | 100.00 | 0.00 | 0.00 |
| 39 | non\_api-jaket\_merah.avi | 100.00 | 0.00 | 0.00 |
| 40 | non\_api-jalan\_malam.avi | 100.00 | 0.00 | 0.00 |
| 41 | non\_api-jalan\_malam2.avi | 100.00 | 0.00 | 0.00 |
| 42 | non\_api-jalan\_raya.avi | 100.00 | 0.00 | 0.00 |
| 43 | non\_api-jalan\_raya2.avi | 100.00 | 0.00 | 0.00 |
| 44 | non\_api-jalan\_raya3.avi | 100.00 | 0.00 | 0.00 |
| 45 | non\_api-kantor.avi | 100.00 | 0.00 | 0.00 |
| 46 | non\_api-kecelakaan.avi | 100.00 | 0.00 | 0.00 |
| 47 | non\_api-kecelakaan2.avi | 100.00 | 0.00 | 0.00 |
| 48 | non\_api-kecelakaan3.avi | 100.00 | 0.00 | 0.00 |
| 49 | non\_api-kerusuhan.avi | 100.00 | 0.00 | 0.00 |
| 50 | non\_api-kerusuhan2.avi | 100.00 | 0.00 | 0.00 |
| 51 | non\_api-kerusuhan3.avi | 100.00 | 0.00 | 0.00 |
| 52 | non\_api-las\_vegas.avi | 98.15 | 1.85 | 0.00 |
| 53 | non\_api-manuju\_mobil.avi | 100.00 | 0.00 | 0.00 |
| 54 | non\_api-manuju\_mobil2.avi | 100.00 | 0.00 | 0.00 |
| 55 | non\_api-manuju\_mobil3.avi | 100.00 | 0.00 | 0.00 |
| 56 | non\_api-parkiran.avi | 100.00 | 0.00 | 0.00 |
| 57 | non\_api-parkiran2.avi | 100.00 | 0.00 | 0.00 |
| 58 | non\_api-parkiran3.avi | 100.00 | 0.00 | 0.00 |
| 59 | non\_api-pencuri.avi | 100.00 | 0.00 | 0.00 |
| 60 | non\_api-pencuri2.avi | 100.00 | 0.00 | 0.00 |
| 61 | non\_api-penembakan.avi | 100.00 | 0.00 | 0.00 |
| 62 | non\_api-serbet.avi | 98.34 | 1.66 | 0.00 |
| 63 | non\_api-tas.avi | 100.00 | 0.00 | 0.00 |
| 64 | non\_api-tas2.avi | 100.00 | 0.00 | 0.00 |
| 65 | non\_api-tas3.avi | 100.00 | 0.00 | 0.00 |
| 66 | non\_api-televisi.avi | 100.00 | 0.00 | 0.00 |
| 67 | non\_api-televisi2.avi | 100.00 | 0.00 | 0.00 |

Tabel A.18 Hasil Uji Coba Menggunakan Parameter konstanta *region* = 5%, *Threshold*  = 5x10-9, = 5 dan kernel RBF

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Nama File | True Positif | False Positif | Missing Rate |
| 1 | api-bakar\_sampah.avi | 0.00 | 0.00 | 100.00 |
| 2 | api-boneka\_dora.avi | 0.00 | 0.00 | 100.00 |
| 3 | api-kayu.avi | 0.00 | 0.00 | 100.00 |
| 4 | api-kayu2.avi | 0.00 | 0.00 | 100.00 |
| 5 | api-kebakaran\_barang.avi | 0.00 | 0.00 | 100.00 |
| 6 | api-kebakaran\_barang2.avi | 0.00 | 0.00 | 100.00 |
| 7 | api-kebakaran\_hutan.avi | 100.00 | 0.00 | 0.00 |
| 8 | api-kebakaran\_hutan2.avi | 100.00 | 0.00 | 0.00 |
| 9 | api-kebakaran\_hutan3.avi | 100.00 | 0.00 | 0.00 |
| 10 | api-kebakaran\_hutan4.avi | 0.00 | 0.00 | 100.00 |
| 11 | api-kebakaran\_ladang.avi | 100.00 | 0.00 | 0.00 |
| 12 | api-kebakaran\_mobil.avi | 91.67 | 0.00 | 8.33 |
| 13 | api-kebakaran\_mobil2.avi | 32.35 | 0.00 | 67.65 |
| 14 | api-kebakaran\_mobil3.avi | 60.29 | 0.00 | 39.71 |
| 15 | api-kebakaran\_mobil4.avi | 0.00 | 0.00 | 100.00 |
| 16 | api-kebakaran\_mobil5.avi | 14.71 | 0.00 | 85.29 |
| 17 | api-kebakaran\_mobil6.avi | 96.57 | 0.00 | 3.43 |
| 18 | api-kebakaran\_rumah.avi | 0.00 | 0.00 | 100.00 |
| 19 | api-kebakaran\_rumah2.avi | 0.00 | 0.00 | 100.00 |
| 20 | api-kebakaran\_rumah3.avi | 15.20 | 0.00 | 84.80 |
| 21 | api-kebakaran\_rumah4.avi | 2.45 | 0.00 | 97.55 |
| 22 | api-kebakaran\_tol.avi | 21.08 | 0.00 | 78.92 |
| 23 | api-kebakaran-truck.avi | 0.00 | 0.00 | 100.00 |
| 24 | api-kebakaran-truck2.avi | 0.49 | 0.00 | 99.51 |
| 25 | api-kebakaran-truck3.avi | 0.00 | 0.00 | 100.00 |
| 26 | api-kertas.avi | 0.00 | 0.00 | 100.00 |
| 27 | api-kertas2.avi | 0.00 | 0.00 | 100.00 |
| 28 | api-miniatur\_mainan.avi | 0.59 | 0.00 | 99.41 |
| 29 | api-mobil\_mainan.avi | 0.00 | 0.00 | 100.00 |
| 30 | api-orang\_terjun.avi | 0.00 | 0.00 | 100.00 |
| 31 | api-pesawat\_mainan.avi | 95.83 | 0.00 | 4.17 |
| 32 | api-pesawat\_mainan2.avi | 15.98 | 0.00 | 84.02 |
| 33 | api-ruang\_tamu.avi | 0.00 | 0.00 | 100.00 |
| 34 | api-rumah\_mainan.avi | 100.00 | 0.00 | 0.00 |
| 35 | non\_api-anak\_kecil.avi | 100.00 | 0.00 | 0.00 |
| 36 | non\_api-anak\_kecil2.avi | 100.00 | 0.00 | 0.00 |
| 37 | non\_api-anak\_kecil3.avi | 100.00 | 0.00 | 0.00 |
| 38 | non\_api-bertemu.avi | 100.00 | 0.00 | 0.00 |
| 39 | non\_api-jaket\_merah.avi | 100.00 | 0.00 | 0.00 |
| 40 | non\_api-jalan\_malam.avi | 100.00 | 0.00 | 0.00 |
| 41 | non\_api-jalan\_malam2.avi | 100.00 | 0.00 | 0.00 |
| 42 | non\_api-jalan\_raya.avi | 100.00 | 0.00 | 0.00 |
| 43 | non\_api-jalan\_raya2.avi | 100.00 | 0.00 | 0.00 |
| 44 | non\_api-jalan\_raya3.avi | 100.00 | 0.00 | 0.00 |
| 45 | non\_api-kantor.avi | 100.00 | 0.00 | 0.00 |
| 46 | non\_api-kecelakaan.avi | 100.00 | 0.00 | 0.00 |
| 47 | non\_api-kecelakaan2.avi | 100.00 | 0.00 | 0.00 |
| 48 | non\_api-kecelakaan3.avi | 100.00 | 0.00 | 0.00 |
| 49 | non\_api-kerusuhan.avi | 100.00 | 0.00 | 0.00 |
| 50 | non\_api-kerusuhan2.avi | 100.00 | 0.00 | 0.00 |
| 51 | non\_api-kerusuhan3.avi | 100.00 | 0.00 | 0.00 |
| 52 | non\_api-las\_vegas.avi | 100.00 | 0.00 | 0.00 |
| 53 | non\_api-manuju\_mobil.avi | 100.00 | 0.00 | 0.00 |
| 54 | non\_api-manuju\_mobil2.avi | 100.00 | 0.00 | 0.00 |
| 55 | non\_api-manuju\_mobil3.avi | 100.00 | 0.00 | 0.00 |
| 56 | non\_api-parkiran.avi | 100.00 | 0.00 | 0.00 |
| 57 | non\_api-parkiran2.avi | 100.00 | 0.00 | 0.00 |
| 58 | non\_api-parkiran3.avi | 100.00 | 0.00 | 0.00 |
| 59 | non\_api-pencuri.avi | 100.00 | 0.00 | 0.00 |
| 60 | non\_api-pencuri2.avi | 100.00 | 0.00 | 0.00 |
| 61 | non\_api-penembakan.avi | 100.00 | 0.00 | 0.00 |
| 62 | non\_api-serbet.avi | 68.51 | 31.49 | 0.00 |
| 63 | non\_api-tas.avi | 100.00 | 0.00 | 0.00 |
| 64 | non\_api-tas2.avi | 100.00 | 0.00 | 0.00 |
| 65 | non\_api-tas3.avi | 100.00 | 0.00 | 0.00 |
| 66 | non\_api-televisi.avi | 100.00 | 0.00 | 0.00 |
| 67 | non\_api-televisi2.avi | 100.00 | 0.00 | 0.00 |

Tabel A.19 Hasil Uji Coba Menggunakan Parameter konstanta *region* = 10%, *Threshold*  = 5x10-9, = 5 dan kernel RBF

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Nama File | True Positif | False Positif | Missing Rate |
| 1 | api-bakar\_sampah.avi | 0.00 | 0.00 | 100.00 |
| 2 | api-boneka\_dora.avi | 0.00 | 0.00 | 100.00 |
| 3 | api-kayu.avi | 0.00 | 0.00 | 100.00 |
| 4 | api-kayu2.avi | 0.00 | 0.00 | 100.00 |
| 5 | api-kebakaran\_barang.avi | 0.00 | 0.00 | 100.00 |
| 6 | api-kebakaran\_barang2.avi | 0.00 | 0.00 | 100.00 |
| 7 | api-kebakaran\_hutan.avi | 27.27 | 0.00 | 72.73 |
| 8 | api-kebakaran\_hutan2.avi | 81.82 | 0.00 | 18.18 |
| 9 | api-kebakaran\_hutan3.avi | 10.10 | 0.00 | 89.90 |
| 10 | api-kebakaran\_hutan4.avi | 0.00 | 0.00 | 100.00 |
| 11 | api-kebakaran\_ladang.avi | 98.22 | 0.00 | 1.78 |
| 12 | api-kebakaran\_mobil.avi | 0.00 | 0.00 | 100.00 |
| 13 | api-kebakaran\_mobil2.avi | 0.00 | 0.00 | 100.00 |
| 14 | api-kebakaran\_mobil3.avi | 0.49 | 0.00 | 99.51 |
| 15 | api-kebakaran\_mobil4.avi | 0.00 | 0.00 | 100.00 |
| 16 | api-kebakaran\_mobil5.avi | 0.00 | 0.00 | 100.00 |
| 17 | api-kebakaran\_mobil6.avi | 0.00 | 0.00 | 100.00 |
| 18 | api-kebakaran\_rumah.avi | 0.00 | 0.00 | 100.00 |
| 19 | api-kebakaran\_rumah2.avi | 0.00 | 0.00 | 100.00 |
| 20 | api-kebakaran\_rumah3.avi | 0.00 | 0.00 | 100.00 |
| 21 | api-kebakaran\_rumah4.avi | 0.00 | 0.00 | 100.00 |
| 22 | api-kebakaran\_tol.avi | 9.80 | 0.00 | 90.20 |
| 23 | api-kebakaran-truck.avi | 0.00 | 0.00 | 100.00 |
| 24 | api-kebakaran-truck2.avi | 0.00 | 0.00 | 100.00 |
| 25 | api-kebakaran-truck3.avi | 0.00 | 0.00 | 100.00 |
| 26 | api-kertas.avi | 0.00 | 0.00 | 100.00 |
| 27 | api-kertas2.avi | 0.00 | 0.00 | 100.00 |
| 28 | api-miniatur\_mainan.avi | 0.00 | 0.00 | 100.00 |
| 29 | api-mobil\_mainan.avi | 0.00 | 0.00 | 100.00 |
| 30 | api-orang\_terjun.avi | 0.00 | 0.00 | 100.00 |
| 31 | api-pesawat\_mainan.avi | 21.53 | 0.00 | 78.47 |
| 32 | api-pesawat\_mainan2.avi | 0.00 | 0.00 | 100.00 |
| 33 | api-ruang\_tamu.avi | 0.00 | 0.00 | 100.00 |
| 34 | api-rumah\_mainan.avi | 95.59 | 0.00 | 4.41 |
| 35 | non\_api-anak\_kecil.avi | 100.00 | 0.00 | 0.00 |
| 36 | non\_api-anak\_kecil2.avi | 100.00 | 0.00 | 0.00 |
| 37 | non\_api-anak\_kecil3.avi | 100.00 | 0.00 | 0.00 |
| 38 | non\_api-bertemu.avi | 100.00 | 0.00 | 0.00 |
| 39 | non\_api-jaket\_merah.avi | 100.00 | 0.00 | 0.00 |
| 40 | non\_api-jalan\_malam.avi | 100.00 | 0.00 | 0.00 |
| 41 | non\_api-jalan\_malam2.avi | 100.00 | 0.00 | 0.00 |
| 42 | non\_api-jalan\_raya.avi | 100.00 | 0.00 | 0.00 |
| 43 | non\_api-jalan\_raya2.avi | 100.00 | 0.00 | 0.00 |
| 44 | non\_api-jalan\_raya3.avi | 100.00 | 0.00 | 0.00 |
| 45 | non\_api-kantor.avi | 100.00 | 0.00 | 0.00 |
| 46 | non\_api-kecelakaan.avi | 100.00 | 0.00 | 0.00 |
| 47 | non\_api-kecelakaan2.avi | 100.00 | 0.00 | 0.00 |
| 48 | non\_api-kecelakaan3.avi | 100.00 | 0.00 | 0.00 |
| 49 | non\_api-kerusuhan.avi | 100.00 | 0.00 | 0.00 |
| 50 | non\_api-kerusuhan2.avi | 100.00 | 0.00 | 0.00 |
| 51 | non\_api-kerusuhan3.avi | 100.00 | 0.00 | 0.00 |
| 52 | non\_api-las\_vegas.avi | 100.00 | 0.00 | 0.00 |
| 53 | non\_api-manuju\_mobil.avi | 100.00 | 0.00 | 0.00 |
| 54 | non\_api-manuju\_mobil2.avi | 100.00 | 0.00 | 0.00 |
| 55 | non\_api-manuju\_mobil3.avi | 100.00 | 0.00 | 0.00 |
| 56 | non\_api-parkiran.avi | 100.00 | 0.00 | 0.00 |
| 57 | non\_api-parkiran2.avi | 100.00 | 0.00 | 0.00 |
| 58 | non\_api-parkiran3.avi | 100.00 | 0.00 | 0.00 |
| 59 | non\_api-pencuri.avi | 100.00 | 0.00 | 0.00 |
| 60 | non\_api-pencuri2.avi | 100.00 | 0.00 | 0.00 |
| 61 | non\_api-penembakan.avi | 100.00 | 0.00 | 0.00 |
| 62 | non\_api-serbet.avi | 72.38 | 27.62 | 0.00 |
| 63 | non\_api-tas.avi | 100.00 | 0.00 | 0.00 |
| 64 | non\_api-tas2.avi | 100.00 | 0.00 | 0.00 |
| 65 | non\_api-tas3.avi | 100.00 | 0.00 | 0.00 |
| 66 | non\_api-televisi.avi | 100.00 | 0.00 | 0.00 |
| 67 | non\_api-televisi2.avi | 100.00 | 0.00 | 0.00 |

Tabel A.20 Hasil Uji Coba Menggunakan Parameter *size frame* = 240 x 320, *Threshold*  = 5x10-9, = 5 dan kernel RBF

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Nama File | True Positif | False Positif | Missing Rate |
| 1 | api-bakar\_sampah.avi | 71.88 | 0.00 | 28.13 |
| 2 | api-boneka\_dora.avi | 48.04 | 0.00 | 51.96 |
| 3 | api-kayu.avi | 100.00 | 0.00 | 0.00 |
| 4 | api-kayu2.avi | 100.00 | 0.00 | 0.00 |
| 5 | api-kebakaran\_barang.avi | 100.00 | 0.00 | 0.00 |
| 6 | api-kebakaran\_barang2.avi | 100.00 | 0.00 | 0.00 |
| 7 | api-kebakaran\_hutan.avi | 100.00 | 0.00 | 0.00 |
| 8 | api-kebakaran\_hutan2.avi | 100.00 | 0.00 | 0.00 |
| 9 | api-kebakaran\_hutan3.avi | 100.00 | 0.00 | 0.00 |
| 10 | api-kebakaran\_hutan4.avi | 100.00 | 0.00 | 0.00 |
| 11 | api-kebakaran\_ladang.avi | 100.00 | 0.00 | 0.00 |
| 12 | api-kebakaran\_mobil.avi | 100.00 | 0.00 | 0.00 |
| 13 | api-kebakaran\_mobil2.avi | 100.00 | 0.00 | 0.00 |
| 14 | api-kebakaran\_mobil3.avi | 100.00 | 0.00 | 0.00 |
| 15 | api-kebakaran\_mobil4.avi | 100.00 | 0.00 | 0.00 |
| 16 | api-kebakaran\_mobil5.avi | 100.00 | 0.00 | 0.00 |
| 17 | api-kebakaran\_mobil6.avi | 100.00 | 0.00 | 0.00 |
| 18 | api-kebakaran\_rumah.avi | 95.10 | 0.00 | 4.90 |
| 19 | api-kebakaran\_rumah2.avi | 100.00 | 0.00 | 0.00 |
| 20 | api-kebakaran\_rumah3.avi | 100.00 | 0.00 | 0.00 |
| 21 | api-kebakaran\_rumah4.avi | 76.47 | 0.00 | 23.53 |
| 22 | api-kebakaran\_tol.avi | 78.92 | 0.00 | 21.08 |
| 23 | api-kebakaran-truck.avi | 97.55 | 0.00 | 2.45 |
| 24 | api-kebakaran-truck2.avi | 100.00 | 0.00 | 0.00 |
| 25 | api-kebakaran-truck3.avi | 74.51 | 0.00 | 25.49 |
| 26 | api-kertas.avi | 24.69 | 0.00 | 75.31 |
| 27 | api-kertas2.avi | 48.15 | 0.00 | 51.85 |
| 28 | api-miniatur\_mainan.avi | 61.54 | 0.00 | 38.46 |
| 29 | api-mobil\_mainan.avi | 96.08 | 0.00 | 3.92 |
| 30 | api-orang\_terjun.avi | 61.27 | 0.00 | 38.73 |
| 31 | api-pesawat\_mainan.avi | 100.00 | 0.00 | 0.00 |
| 32 | api-pesawat\_mainan2.avi | 100.00 | 0.00 | 0.00 |
| 33 | api-ruang\_tamu.avi | 46.75 | 0.00 | 53.25 |
| 34 | api-rumah\_mainan.avi | 100.00 | 0.00 | 0.00 |
| 35 | non\_api-anak\_kecil.avi | 100.00 | 0.00 | 0.00 |
| 36 | non\_api-anak\_kecil2.avi | 100.00 | 0.00 | 0.00 |
| 37 | non\_api-anak\_kecil3.avi | 100.00 | 0.00 | 0.00 |
| 38 | non\_api-bertemu.avi | 100.00 | 0.00 | 0.00 |
| 39 | non\_api-jaket\_merah.avi | 100.00 | 0.00 | 0.00 |
| 40 | non\_api-jalan\_malam.avi | 100.00 | 0.00 | 0.00 |
| 41 | non\_api-jalan\_malam2.avi | 100.00 | 0.00 | 0.00 |
| 42 | non\_api-jalan\_raya.avi | 100.00 | 0.00 | 0.00 |
| 43 | non\_api-jalan\_raya2.avi | 100.00 | 0.00 | 0.00 |
| 44 | non\_api-jalan\_raya3.avi | 100.00 | 0.00 | 0.00 |
| 45 | non\_api-kantor.avi | 100.00 | 0.00 | 0.00 |
| 46 | non\_api-kecelakaan.avi | 100.00 | 0.00 | 0.00 |
| 47 | non\_api-kecelakaan2.avi | 100.00 | 0.00 | 0.00 |
| 48 | non\_api-kecelakaan3.avi | 100.00 | 0.00 | 0.00 |
| 49 | non\_api-kerusuhan.avi | 100.00 | 0.00 | 0.00 |
| 50 | non\_api-kerusuhan2.avi | 100.00 | 0.00 | 0.00 |
| 51 | non\_api-kerusuhan3.avi | 100.00 | 0.00 | 0.00 |
| 52 | non\_api-las\_vegas.avi | 81.48 | 18.52 | 0.00 |
| 53 | non\_api-manuju\_mobil.avi | 100.00 | 0.00 | 0.00 |
| 54 | non\_api-manuju\_mobil2.avi | 100.00 | 0.00 | 0.00 |
| 55 | non\_api-manuju\_mobil3.avi | 100.00 | 0.00 | 0.00 |
| 56 | non\_api-parkiran.avi | 100.00 | 0.00 | 0.00 |
| 57 | non\_api-parkiran2.avi | 100.00 | 0.00 | 0.00 |
| 58 | non\_api-parkiran3.avi | 100.00 | 0.00 | 0.00 |
| 59 | non\_api-pencuri.avi | 100.00 | 0.00 | 0.00 |
| 60 | non\_api-pencuri2.avi | 98.82 | 1.18 | 0.00 |
| 61 | non\_api-penembakan.avi | 100.00 | 0.00 | 0.00 |
| 62 | non\_api-serbet.avi | 66.85 | 33.15 | 0.00 |
| 63 | non\_api-tas.avi | 100.00 | 0.00 | 0.00 |
| 64 | non\_api-tas2.avi | 100.00 | 0.00 | 0.00 |
| 65 | non\_api-tas3.avi | 100.00 | 0.00 | 0.00 |
| 66 | non\_api-televisi.avi | 100.00 | 0.00 | 0.00 |
| 67 | non\_api-televisi2.avi | 100.00 | 0.00 | 0.00 |

Tabel A.21 Hasil Uji Coba Menggunakan Parameter *size frame* = 60 x 80, *Threshold*  = 5x10-9, = 5 dan kernel RBF

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Nama File | True Positif | False Positif | Missing Rate |
| 1 | api-bakar\_sampah.avi | 100.00 | 0.00 | 0.00 |
| 2 | api-boneka\_dora.avi | 91.67 | 0.00 | 8.33 |
| 3 | api-kayu.avi | 100.00 | 0.00 | 0.00 |
| 4 | api-kayu2.avi | 100.00 | 0.00 | 0.00 |
| 5 | api-kebakaran\_barang.avi | 100.00 | 0.00 | 0.00 |
| 6 | api-kebakaran\_barang2.avi | 100.00 | 0.00 | 0.00 |
| 7 | api-kebakaran\_hutan.avi | 100.00 | 0.00 | 0.00 |
| 8 | api-kebakaran\_hutan2.avi | 100.00 | 0.00 | 0.00 |
| 9 | api-kebakaran\_hutan3.avi | 100.00 | 0.00 | 0.00 |
| 10 | api-kebakaran\_hutan4.avi | 100.00 | 0.00 | 0.00 |
| 11 | api-kebakaran\_ladang.avi | 100.00 | 0.00 | 0.00 |
| 12 | api-kebakaran\_mobil.avi | 100.00 | 0.00 | 0.00 |
| 13 | api-kebakaran\_mobil2.avi | 100.00 | 0.00 | 0.00 |
| 14 | api-kebakaran\_mobil3.avi | 98.53 | 0.00 | 1.47 |
| 15 | api-kebakaran\_mobil4.avi | 100.00 | 0.00 | 0.00 |
| 16 | api-kebakaran\_mobil5.avi | 100.00 | 0.00 | 0.00 |
| 17 | api-kebakaran\_mobil6.avi | 100.00 | 0.00 | 0.00 |
| 18 | api-kebakaran\_rumah.avi | 99.51 | 0.00 | 0.49 |
| 19 | api-kebakaran\_rumah2.avi | 100.00 | 0.00 | 0.00 |
| 20 | api-kebakaran\_rumah3.avi | 100.00 | 0.00 | 0.00 |
| 21 | api-kebakaran\_rumah4.avi | 99.51 | 0.00 | 0.49 |
| 22 | api-kebakaran\_tol.avi | 100.00 | 0.00 | 0.00 |
| 23 | api-kebakaran-truck.avi | 100.00 | 0.00 | 0.00 |
| 24 | api-kebakaran-truck2.avi | 100.00 | 0.00 | 0.00 |
| 25 | api-kebakaran-truck3.avi | 100.00 | 0.00 | 0.00 |
| 26 | api-kertas.avi | 98.77 | 0.00 | 1.23 |
| 27 | api-kertas2.avi | 100.00 | 0.00 | 0.00 |
| 28 | api-miniatur\_mainan.avi | 98.82 | 0.00 | 1.18 |
| 29 | api-mobil\_mainan.avi | 100.00 | 0.00 | 0.00 |
| 30 | api-orang\_terjun.avi | 91.18 | 0.00 | 8.82 |
| 31 | api-pesawat\_mainan.avi | 100.00 | 0.00 | 0.00 |
| 32 | api-pesawat\_mainan2.avi | 100.00 | 0.00 | 0.00 |
| 33 | api-ruang\_tamu.avi | 97.04 | 0.00 | 2.96 |
| 34 | api-rumah\_mainan.avi | 100.00 | 0.00 | 0.00 |
| 35 | non\_api-anak\_kecil.avi | 100.00 | 0.00 | 0.00 |
| 36 | non\_api-anak\_kecil2.avi | 100.00 | 0.00 | 0.00 |
| 37 | non\_api-anak\_kecil3.avi | 100.00 | 0.00 | 0.00 |
| 38 | non\_api-bertemu.avi | 100.00 | 0.00 | 0.00 |
| 39 | non\_api-jaket\_merah.avi | 100.00 | 0.00 | 0.00 |
| 40 | non\_api-jalan\_malam.avi | 100.00 | 0.00 | 0.00 |
| 41 | non\_api-jalan\_malam2.avi | 100.00 | 0.00 | 0.00 |
| 42 | non\_api-jalan\_raya.avi | 100.00 | 0.00 | 0.00 |
| 43 | non\_api-jalan\_raya2.avi | 100.00 | 0.00 | 0.00 |
| 44 | non\_api-jalan\_raya3.avi | 100.00 | 0.00 | 0.00 |
| 45 | non\_api-kantor.avi | 100.00 | 0.00 | 0.00 |
| 46 | non\_api-kecelakaan.avi | 100.00 | 0.00 | 0.00 |
| 47 | non\_api-kecelakaan2.avi | 100.00 | 0.00 | 0.00 |
| 48 | non\_api-kecelakaan3.avi | 100.00 | 0.00 | 0.00 |
| 49 | non\_api-kerusuhan.avi | 100.00 | 0.00 | 0.00 |
| 50 | non\_api-kerusuhan2.avi | 100.00 | 0.00 | 0.00 |
| 51 | non\_api-kerusuhan3.avi | 100.00 | 0.00 | 0.00 |
| 52 | non\_api-las\_vegas.avi | 0.00 | 100.00 | 0.00 |
| 53 | non\_api-manuju\_mobil.avi | 100.00 | 0.00 | 0.00 |
| 54 | non\_api-manuju\_mobil2.avi | 100.00 | 0.00 | 0.00 |
| 55 | non\_api-manuju\_mobil3.avi | 100.00 | 0.00 | 0.00 |
| 56 | non\_api-parkiran.avi | 100.00 | 0.00 | 0.00 |
| 57 | non\_api-parkiran2.avi | 100.00 | 0.00 | 0.00 |
| 58 | non\_api-parkiran3.avi | 100.00 | 0.00 | 0.00 |
| 59 | non\_api-pencuri.avi | 31.36 | 68.64 | 0.00 |
| 60 | non\_api-pencuri2.avi | 98.22 | 1.78 | 0.00 |
| 61 | non\_api-penembakan.avi | 89.22 | 10.78 | 0.00 |
| 62 | non\_api-serbet.avi | 65.19 | 34.81 | 0.00 |
| 63 | non\_api-tas.avi | 100.00 | 0.00 | 0.00 |
| 64 | non\_api-tas2.avi | 100.00 | 0.00 | 0.00 |
| 65 | non\_api-tas3.avi | 100.00 | 0.00 | 0.00 |
| 66 | non\_api-televisi.avi | 100.00 | 0.00 | 0.00 |
| 67 | non\_api-televisi2.avi | 100.00 | 0.00 | 0.00 |

Tabel A.22 Hasil Uji Coba Tanpa Menggunakan *region growing* dan perhitungan luasan *region*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Nama File | True Positif | False Positif | Missing Rate |
| 1 | api-bakar\_sampah.avi | 100.00 | 0.00 | 0.00 |
| 2 | api-boneka\_dora.avi | 100.00 | 0.00 | 0.00 |
| 3 | api-kayu.avi | 100.00 | 0.00 | 0.00 |
| 4 | api-kayu2.avi | 100.00 | 0.00 | 0.00 |
| 5 | api-kebakaran\_barang.avi | 100.00 | 0.00 | 0.00 |
| 6 | api-kebakaran\_barang2.avi | 100.00 | 0.00 | 0.00 |
| 7 | api-kebakaran\_hutan.avi | 100.00 | 0.00 | 0.00 |
| 8 | api-kebakaran\_hutan2.avi | 100.00 | 0.00 | 0.00 |
| 9 | api-kebakaran\_hutan3.avi | 100.00 | 0.00 | 0.00 |
| 10 | api-kebakaran\_hutan4.avi | 100.00 | 0.00 | 0.00 |
| 11 | api-kebakaran\_ladang.avi | 100.00 | 0.00 | 0.00 |
| 12 | api-kebakaran\_mobil.avi | 100.00 | 0.00 | 0.00 |
| 13 | api-kebakaran\_mobil2.avi | 100.00 | 0.00 | 0.00 |
| 14 | api-kebakaran\_mobil3.avi | 100.00 | 0.00 | 0.00 |
| 15 | api-kebakaran\_mobil4.avi | 100.00 | 0.00 | 0.00 |
| 16 | api-kebakaran\_mobil5.avi | 100.00 | 0.00 | 0.00 |
| 17 | api-kebakaran\_mobil6.avi | 100.00 | 0.00 | 0.00 |
| 18 | api-kebakaran\_rumah.avi | 100.00 | 0.00 | 0.00 |
| 19 | api-kebakaran\_rumah2.avi | 100.00 | 0.00 | 0.00 |
| 20 | api-kebakaran\_rumah3.avi | 100.00 | 0.00 | 0.00 |
| 21 | api-kebakaran\_rumah4.avi | 100.00 | 0.00 | 0.00 |
| 22 | api-kebakaran\_tol.avi | 100.00 | 0.00 | 0.00 |
| 23 | api-kebakaran-truck.avi | 100.00 | 0.00 | 0.00 |
| 24 | api-kebakaran-truck2.avi | 100.00 | 0.00 | 0.00 |
| 25 | api-kebakaran-truck3.avi | 100.00 | 0.00 | 0.00 |
| 26 | api-kertas.avi | 100.00 | 0.00 | 0.00 |
| 27 | api-kertas2.avi | 100.00 | 0.00 | 0.00 |
| 28 | api-miniatur\_mainan.avi | 100.00 | 0.00 | 0.00 |
| 29 | api-mobil\_mainan.avi | 100.00 | 0.00 | 0.00 |
| 30 | api-orang\_terjun.avi | 100.00 | 0.00 | 0.00 |
| 31 | api-pesawat\_mainan.avi | 100.00 | 0.00 | 0.00 |
| 32 | api-pesawat\_mainan2.avi | 100.00 | 0.00 | 0.00 |
| 33 | api-ruang\_tamu.avi | 100.00 | 0.00 | 0.00 |
| 34 | api-rumah\_mainan.avi | 100.00 | 0.00 | 0.00 |
| 35 | non\_api-anak\_kecil.avi | 100.00 | 0.00 | 0.00 |
| 36 | non\_api-anak\_kecil2.avi | 100.00 | 0.00 | 0.00 |
| 37 | non\_api-anak\_kecil3.avi | 100.00 | 0.00 | 0.00 |
| 38 | non\_api-bertemu.avi | 100.00 | 0.00 | 0.00 |
| 39 | non\_api-jaket\_merah.avi | 81.99 | 18.01 | 0.00 |
| 40 | non\_api-jalan\_malam.avi | 84.67 | 15.33 | 0.00 |
| 41 | non\_api-jalan\_malam2.avi | 95.86 | 4.14 | 0.00 |
| 42 | non\_api-jalan\_raya.avi | 47.93 | 52.07 | 0.00 |
| 43 | non\_api-jalan\_raya2.avi | 31.36 | 68.64 | 0.00 |
| 44 | non\_api-jalan\_raya3.avi | 24.26 | 75.74 | 0.00 |
| 45 | non\_api-kantor.avi | 100.00 | 0.00 | 0.00 |
| 46 | non\_api-kecelakaan.avi | 100.00 | 0.00 | 0.00 |
| 47 | non\_api-kecelakaan2.avi | 100.00 | 0.00 | 0.00 |
| 48 | non\_api-kecelakaan3.avi | 100.00 | 0.00 | 0.00 |
| 49 | non\_api-kerusuhan.avi | 28.40 | 71.60 | 0.00 |
| 50 | non\_api-kerusuhan2.avi | 58.58 | 41.42 | 0.00 |
| 51 | non\_api-kerusuhan3.avi | 55.03 | 44.97 | 0.00 |
| 52 | non\_api-las\_vegas.avi | 0.00 | 100.00 | 0.00 |
| 53 | non\_api-manuju\_mobil.avi | 100.00 | 0.00 | 0.00 |
| 54 | non\_api-manuju\_mobil2.avi | 100.00 | 0.00 | 0.00 |
| 55 | non\_api-manuju\_mobil3.avi | 100.00 | 0.00 | 0.00 |
| 56 | non\_api-parkiran.avi | 100.00 | 0.00 | 0.00 |
| 57 | non\_api-parkiran2.avi | 100.00 | 0.00 | 0.00 |
| 58 | non\_api-parkiran3.avi | 100.00 | 0.00 | 0.00 |
| 59 | non\_api-pencuri.avi | 5.92 | 94.08 | 0.00 |
| 60 | non\_api-pencuri2.avi | 74.56 | 25.44 | 0.00 |
| 61 | non\_api-penembakan.avi | 0.00 | 100.00 | 0.00 |
| 62 | non\_api-serbet.avi | 51.38 | 48.62 | 0.00 |
| 63 | non\_api-tas.avi | 97.89 | 2.11 | 0.00 |
| 64 | non\_api-tas2.avi | 100.00 | 0.00 | 0.00 |
| 65 | non\_api-tas3.avi | 100.00 | 0.00 | 0.00 |
| 66 | non\_api-televisi.avi | 100.00 | 0.00 | 0.00 |
| 67 | non\_api-televisi2.avi | 100.00 | 0.00 | 0.00 |

# BIODATA PENULIS



Hamdi Ahmadi Muzakkiy atau biasa dipanggil Hamdi dilahirkan di Jakarta pada tanggal 15 April 1994 dan dibesarkan di Jakarta. Penulis adalah anak pertama dari tiga bersaudara.

Penulis menempuh pendidikan di SD kasih Ananda (1999-2006), SMP N 84 Jakarta (2006-2009), dan SMA N 75 Jakarta (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.

Selama menempuh kuliah penulis aktif sebagai anggota Himpunan Mahasiswa Teknik Computer (HMTC) ITS. Penulis juga aktif dalam kegiatan kepanitiaan Schematics sebagai staff hubungan masyarakat (Humas) Schematics 2013 dan Wakil Ketua National Programming Contest (NPC) 2014. Selain itu penulis juga aktif menjadi administrator Lab pemrograman(LP) Teknik Informatika ITS. Penulis pernah menjadi asisten dosen dan praktikum untuk mata kuliah Pemrograman Terstruktur (2013), Algoritma dan Struktur Data (2013) , Basis Data (2014) dan Dasar Pemrograman (2015)

Penulis dapat dihubungi melalui alamat *email* hamdiahmadi1504@gmail.com.