taW

;/--n

**TUGAS AKHIR – KI141502**

**DETEKSI API BERBASIS SENSOR VISUAL MENGGUNAKAN METODE *SUPPORT VECTOR MACHINES***

**HAMDI AHMADI MUZAKKIY**

**NRP 5112100091**

**Dosen Pembimbing I**

**Prof. Ir. Handayani Tjandrasa, M.Sc.,Ph.D**

**Dosen Pembimbing II**

**Dr. Chastine Fatichah, S.Kom, M.Kom**

**JURUSAN TEKNIK INFORMATIKA**

**Fakultas Teknologi Informasi**

**Institut Teknologi Sepuluh Nopember**

**Surabaya 2015**

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

**TUGAS AKHIR – KI141502**

**DETEKSI API BERBASIS SENSOR VISUAL MENGGUNAKAN METODE *SUPPORT VECTOR MACHINES***

**HAMDI AHMADI MUZAKKIY**

**NRP 5112100091**

**Dosen Pembimbing I**

**Prof. Ir. Handayani Tjandrasa, M.Sc.,Ph.D**

**Dosen Pembimbing II**

**Dr. Chastine Fatichah, S.Kom, M.Kom**

**JURUSAN TEKNIK INFORMATIKA**

**Fakultas Teknologi Informasi**

**Institut Teknologi Sepuluh Nopember**

**Surabaya 2015**

****

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

**UNDERGRADUATE THESES – KI141502**

**FIRE DETECTION BASED ON VISION SENSOR USING SUPPORT VECTOR MACHINES**

**HAMDI AHMADI MUZAKKIY**

**NRP 5112100091**

**Supervisor I**

**Prof. Ir. Handayani Tjandrasa, M.Sc.,Ph.D**

**Supervisor II**

**Dr. Chastine Fatichah, S.Kom, M.Kom**

**DEPARTMENT OF INFORMATICS**

**FACULTY OF INFORMATION TECHNOLOGY**

**INSTITUT TEKNOLOGI SEPULUH NOPEMBER**

**SURABAYA 2015**

# LEMBAR PENGESAHAN

**DETEKSI API BERBASIS SENSOR VISUAL MENGGUNAKAN METODE SUPPORT VECTOR MACHINES**

**TUGAS AKHIR**

Diajukan Untuk Memenuhi Salah Satu Syarat

Memperoleh Gelar Sarjana Komputer

pada

Bidang Studi Desain dan Terapan Komputasi

Program Studi S-1 Jurusan Teknik Informatika

Fakultas Teknologi Informasi

Institut Teknologi Sepuluh Nopember

Oleh

**HAMDI AHMADI MUZAKKIY**

**NRP : 5112 100 091**

Disetujui oleh Dosen Pembimbing Tugas Akhir:

1. Prof. Ir. Handayani Tjandrasa, M.Sc.,Ph.D .....................

NIP: (Pembimbing 1)

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

NIP: (Pembimbing 2)

**SURABAYA**

**DESEMBER, 2015**

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

**Deteksi Api Berbasis Sensor Visual Menggunakan Metode Support Vector Machines**

|  |  |  |
| --- | --- | --- |
| **Nama Mahasiswa** | **:** | **HAMDI AHMADI MUZAKKIY** |
| **NRP** | **:** | **5112100091** |
| **Jurusan** | **:** | **Teknik Informatika FTIF-ITS** |
| **Dosen Pembimbing 1** | **:** | **Prof. Ir. Handayani Tjandrasa, M.Sc.,Ph.D** |
| **Dosen Pembimbing 2** | **:** | **Dr. Chastine Fatichah, S.Kom, M.Kom** |

# *Abstrak*

*Kebakaran adalah salah satu bencana yang sering terjadi. Hal ini terjadi karena lalainya manusia dalam melakukan aktifitas, misalnya meninggalkan kompor yang sedang menyala. Hal ini berakibat fatal, banyak korban jiwa yang melayang karena masalah ini. Sudah banyaknya alat pendeteksi api menggunakan sensor panas, ion, infrared. Namun penggunaan sistem alarm ini tidak akan bekerja hingga partikel mencapai sensor. Oleh kerena itu diperlukan sistem deteksi api yang dapat mendeteksi kebakaran dengan cepat.*

*Dalam Tugas Akhir ini diimplementasikan perangkat lunak pendeteksi api menggunakan deteksi gerak, deteksi warna menggunakan probabilitas warna, region growing, dan deteksi bentuk menggunakan fitur wavelet dengan metode klasifikasi support vector machines. Hasil dari deteksi bentuk akan digunakan dalam proses penentuan api.*

*Dataset yang digunakan dalam proses uji coba berisi lima puluh tiga video api dengan panjang video enam sampai enambelas detik yang diambil dari berbagai sumber. Performa terbaik yang dihasilkan adalah true positif sebesar xx%, false negatif sebesar xx% dan missing rate sebesar xx%*

***Kata Kunci: Deteksi Gerak, Deteksi Warna, Probabilitas, Wavelet, Support Vector Machines.***

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

**FIRE DETECTION BASED ON VISION SENSOR USING SUPPORT VECTOR MACHINES**

|  |  |  |
| --- | --- | --- |
| **Student’s Name** | **:** | **HAMDI AHMADI MUZAKKIY** |
| **Student’s ID** | **:** | **5112100091** |
| **Department** | **:** | **Teknik Informatika FTIF-ITS** |
| **First Advisor** | **:** | **Prof. Ir. Handayani Tjandrasa, M.Sc.,Ph.D** |
| **Second Advisor** | **:** | **Dr. Chastine Fatichah, S.Kom, M.Kom** |

# *Abstract*

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

# KATA PENGANTAR



Puji syukur penulis kehadirat Tuhan YME karena berkat rahmat dan karunia-NYA penulis dapat menyelesaikan Tugas Akhir yang berjudul

**DETEKSI API BERBASIS SENSOR VISUAL MENGGUNAKAN METODE SUPPORT VECTOR MACHINES**

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. Tuhan YME karena berkat rahmat dan karunianya penulis berhasil menyelesaikan Tugas Akhir dengan baik.
2. Kedua orang tua, dan keluarga penulis, terima kasih atas doa dan bantuan moral dan material selama penulis belajar di Teknik Informatika ITS.
3. Ibu Dr. Eng. Nanik Suciati, S.Kom., M.Kom., selaku ketua jurusan Teknik Informatika ITS
4. Bapak Radityo Anggoro, S.Kom., M.Sc. selaku Koordinator Tugas Akhir di Teknik Informatika ITS.
5. Ibu Prof. Ir. Handayani Tjandrasa, M.Sc.,Ph.D selaku Dosen Pembimbing I Tugas Akhir yang telah memberikan bimbingan dan dukungan selama penulis menyelesaikan Tugas Akhir.
6. Ibu Dr. Chastine Fatichah, S.Kom, M.Kom selaku pembimbing II Tugas Akhir yang telah memberikan banyak waktu untuk berdiskusi dan memberi semangat dan motivasi kepada penulis untuk menyelesaikan Tugas Akhir.
7. Bapak dan Ibu Dosen di Jurusan Teknik Informatika yang telah memberikan ilmu selama penulis kuliah di Teknik Informatika
8. Seluruh Staf dan karyawan Teknik Informatika yang telah memberikan bantuan selama penulis kuliah di Teknik Informatika.
9. Rekan-rekan di laboratorium Pemrograman yang telah bersedia dan betah dengan adanya penulis di lab selama pengerjaan Tugas Akhir.

Penulis Mohon maaf apabila terdapat kekurangan dalam penulisan Tugas Akhir ini. Kritik dan saran penulis harapkan untuk perbaikan dan pembelajaran di kemudian hari. Semoga Tugas Akhir ini dapat memberikan Manfaat yang sebesar besarnya.

Surabaya, Desember 2015

Penulis

# DAFTAR ISI

[LEMBAR PENGESAHAN v](#_Toc438554954)

[*Abstrak* vii](#_Toc438554955)

[*Abstract* ix](#_Toc438554956)

[KATA PENGANTAR xi](#_Toc438554957)

[DAFTAR ISI xiii](#_Toc438554958)

[DAFTAR GAMBAR xv](#_Toc438554959)

[DAFTAR TABEL xvi](#_Toc438554960)

[DAFTAR KODE SUMBER xviii](#_Toc438554961)

[BAB I PENDAHULUAN 1](#_Toc438554962)

[1.1 Latar Belakang 1](#_Toc438554963)

[1.2 Rumusan Masalah 2](#_Toc438554964)

[1.3 Batasan Masalah 2](#_Toc438554965)

[1.4 Tujuan 2](#_Toc438554966)

[1.5 Manfaat 3](#_Toc438554967)

[1.6 Metodologi 3](#_Toc438554968)

[BAB II TINJAUAN PUSTAKA 6](#_Toc438554969)

[2.1 Adaptive Background Mixture Models 6](#_Toc438554970)

[2.2 Gaussian Pyramid 6](#_Toc438554971)

[2.3 Probabilitas Distribusi Gaussian 8](#_Toc438554972)

[2.4 Region Growing 9](#_Toc438554973)

[2.5 Daubachies D4 Wavelet 10](#_Toc438554974)

[2.6 Normalisasi Min-Max 13](#_Toc438554975)

[2.7 Support Vector Machines 14](#_Toc438554976)

[BAB III DESAIN PERANGKAT LUNAK 17](#_Toc438554977)

[3.1 Data 17](#_Toc438554978)

[3.1.1 Data Masukkan 17](#_Toc438554979)

[3.1.2 Data Pembelajaran 18](#_Toc438554980)

[3.1.3 Data Keluaran 18](#_Toc438554981)

[3.2 Desain Sistem Secara Umum 19](#_Toc438554982)

[3.3 Preprocessing 21](#_Toc438554983)

[3.3.1 Reduksi Size Frame 21](#_Toc438554984)

[3.3.2 Deteksi Gerak 22](#_Toc438554985)

[3.3.3 Deteksi Warna Piksel 23](#_Toc438554986)

[3.3.4 Region Growing 24](#_Toc438554987)

[3.3.5 Perhitungan Luasan Region Api 25](#_Toc438554988)

[3.4 Verifikasi 26](#_Toc438554989)

[3.4.1 Ekstraksi Fitur dengan Wavelet 27](#_Toc438554990)

[3.4.2 Klasifikasi 29](#_Toc438554991)

[3.5 Menandai Region Api 29](#_Toc438554992)

[BAB IV IMPLEMENTASI 31](#_Toc438554993)

[4.1 Lingkungan Implementasi 31](#_Toc438554994)

[4.2 Implementasi 31](#_Toc438554995)

[4.2.1 Implementasi Tahap Reduksi Size Frame 31](#_Toc438554996)

[4.2.2 Implementasi Tahap Deteksi Gerak 32](#_Toc438554997)

[4.2.3 Implementasi Tahap Deteksi Warna Piksel 33](#_Toc438554998)

[4.2.4 Implementasi Tahap Region Growing 37](#_Toc438554999)

[4.2.5 Implementasi Tahap Luasan Region Api 39](#_Toc438555000)

[4.2.6 Implementasi Tahap Ekstraksi Fitur dengan Wavelet 40](#_Toc438555001)

[4.2.7 Implementasi Tahap Klasifikasi 41](#_Toc438555002)

[4.2.8 Implementasi Tahap Menandai Region Api 42](#_Toc438555003)

[BAB V UJI COBA DAN EVALUASI 45](#_Toc438555004)

[5.1 Lingkungan Uji Coba 45](#_Toc438555005)

[5.2 Data Uji Coba 45](#_Toc438555006)

[5.3 Skenario Uji Coba 46](#_Toc438555007)

[5.3.1 Skenario Uji Coba 1 46](#_Toc438555008)

[5.3.2 Skenario Uji Coba 2 47](#_Toc438555009)

[5.4 Analisis Hasil Uji Coba 48](#_Toc438555010)

[BAB VI KESIMPULAN DAN SARAN 50](#_Toc438555011)

[6.1 Kesimpulan 50](#_Toc438555012)

[6.2 Saran 50](#_Toc438555013)

[DAFTAR ACUAN 51](#_Toc438555014)

[LAMPIRAN A 53](#_Toc438555015)

[BIODATA PENULIS 61](#_Toc438555016)

# DAFTAR GAMBAR

[Gambar 2.2.1 Ilustrasi Gaussian Pyramid 7](#_Toc438555017)

[Gambar 2.4.1 *Seed* Awal *Region Growing* 10](#_Toc438555018)

[Gambar 2.4.2 Piksel yang diamati dan *Region* 10](#_Toc438555019)

[Gambar 2.5.1 Ilustrasi *Low Pass Filter* 12](#_Toc438555020)

[Gambar 2.5.2 Ilustrasi *Filter* yang dilakukan 12](#_Toc438555021)

[Gambar 2.5.3 Citra Masukkan 13](#_Toc438555022)

[Gambar 2.5.4 Citra Keluaran Hasil Wavelet 13](#_Toc438555023)

[Gambar 2.7.1 Ilustrasi *Support Vector Machines* 15](#_Toc438555024)

[Gambar 3.1.1 Contoh Data Masukkan 18](#_Toc438555025)

[Gambar 3.1.2 Contoh Data Keluaran 19](#_Toc438555026)

[Gambar 3.2.1 Diagram alir rancangan perangkat lunak secara umum 21](#_Toc438555027)

[Gambar 3.3.1 Reduksi *Size Frame* 22](#_Toc438555028)

[Gambar 3.3.2 Contoh Deteksi Gerak 23](#_Toc438555029)

[Gambar 3.3.3 Contoh *Dataset* Gambar Api 24](#_Toc438555030)

[Gambar 3.3.4 *Pseudocode* Deteksi Warna Piksel 24](#_Toc438555031)

[Gambar 3.3.5 *Pseudocode Regon Growing* 25](#_Toc438555032)

[Gambar 3.3.6 Diagram Alir Luasan *Region* Api 26](#_Toc438555033)

[Gambar 3.4.1 Diagram alir Ekstraksi Fitur 28](#_Toc438555034)

[Gambar 3.5.1 *Pseudocode* fungsi menandai *region* 29](#_Toc438555035)

[Gambar 5.2.1 Contoh Video Kejadian 46](#_Toc438555036)

# DAFTAR TABEL

[Tabel 4.1.1 Lingkungan Perancangan Perangkat Lunak 31](#_Toc438555037)

[Tabel 5.3.1 Hasil Uji Coba 1 47](#_Toc438555038)

[Tabel 5.3.2 Hasil Uji Coba 2 48](#_Toc438555039)

[Tabel 6.2.1 Hasil Uji Coba Manggunakan C = 3.5 dan *Threshold*  = 10-7 53](#_Toc438555040)

[Tabel 6.2.2 Hasil Uji Coba Manggunakan C = 3.5 dan *Threshold*  = 10-8 54](#_Toc438555041)

[Tabel 6.2.3 Hasil Uji Coba Manggunakan C = 3.5 dan *Threshold*  = 5x10-9 56](#_Toc438555042)

[Tabel 6.2.4 Hasil Uji Coba Manggunakan C = 1 dan *Threshold*  = 5x10-9 57](#_Toc438555043)

[Tabel 6.2.5 Hasil Uji Coba Manggunakan C = 5 dan *Threshold*  = 5x10-9 59](#_Toc438555044)

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

# DAFTAR KODE SUMBER

[Kode Sumber 4.2.1 Implementasi Tahap Reduksi *Size Frame* 32](#_Toc438555045)

[Kode Sumber 4.2.2 Penggunaan Fungsi pyrDown() 32](#_Toc438555046)

[Kode Sumber 4.2.3 Implementasi Tahap Deteksi Gerak 33](#_Toc438555047)

[Kode Sumber 4.2.4 Implementasi Penyimpanan Piksel 33](#_Toc438555048)

[Kode Sumber 4.2.5 Implementasi Menghitung Nilai Standar Deviasi dan Mean Setiap *Channel* 34](#_Toc438555049)

[Kode Sumber 4.2.6 *Generate list* piksel api 35](#_Toc438555050)

[Kode Sumber 4.2.6 memanggil fungsi getGaussianProbability(). Fungsi getGaussianProbability() digunakan untuk menghitung nilai probabilitas tiap *channel* piksel. Fungsi tersebut mengimplementasikan rumus dari probabilitas distribusi gaussian. Implementasi fungsi getGaussianProbability() dapat dilihat pada Kode Sumber 4.2.7. 35](#_Toc438555051)

[Kode Sumber 4.2.8 Fungsi Menghitung Nilai Probabilitas Distribusi Gaussian 36](#_Toc438555052)

[Kode Sumber 4.2.9 Mendapatkan Threshold 36](#_Toc438555053)

[Kode Sumber 4.2.10 Membaca *List* Piksel Api 36](#_Toc438555054)

[Pada Kode Sumber 4.2.11, *list* piksel api dimasukkan kedalam *array* tiga dimensi. *Array* tiga dimensi tersebut bernilai True atau False, dimana jika indeks dimensi *array* tersebut mewakilkan indeks R,G,B. Setelah didapatkan *array* dengan indeks yang mewakili nilai R,G,B dan hasilnya, akan dilakukan pengecekan terhadap kandidat piksel yang didapatkan pada proses deteksi gerak. Implementasi proses pengecekan kandidat piksel dapat dilihat pada Kode Sumber 4.2.10 36](#_Toc438555055)

[Kode Sumber 4.2.12 Implementasi Tahap Deteksi Warna Piksel 37](#_Toc438555056)

[Kode Sumber 4.2.13 Implementasi Tahap *Region Growing* 38](#_Toc438555057)

[Kode Sumber 4.2.14 Implementasi Tahap *Flood Fill* 38](#_Toc438555058)

[Kode Sumber 4.2.15 Implementasi Tahap *Clock Wise* 39](#_Toc438555059)

[Kode Sumber 4.2.16 Implementasi Tahap Variasi Warna *Region* 40](#_Toc438555060)

[Kode Sumber 4.2.17 Implementasi Memasukkan Nilai Wavelet Kedalam *List* 40](#_Toc438555061)

[Kode Sumber 4.2.18 Pemanggilan Fungsi Wavelet 41](#_Toc438555062)

[Kode Sumber 4.2.19 *Training* Klasifikasi 41](#_Toc438555063)

[Kode Sumber 4.2.20 Implementasi Tahap Klasifikasi 42](#_Toc438555064)

[Kode Sumber 4.2.21 Implementasi Tahap Menandai Region Api 43](#_Toc438555065)

***[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

Banyaknya kamera pengawas yang digunakan pada bangunan-bangunan saat ini tidak terlalu optimal digunakan, hal ini dikarenakan masih banyaknya kamera pengawas masih diawasi oleh operator. Disini penting sekali pengaplikasian pemrosesan gambar yang dapat mempermudah dalam pendeteksian suatu objek. Salah satu deteksi objek yang saat ini penting untuk digunakan adalah deteksi api, banyak dari sistem yang sekarang digunakan dalam mendeteksi api adalah penggunaan sensor panas, ion, atau infrared yang bergantung pada karakteristik tertentu seperti api, asap, suhu, atau radiasi.

Penggunaan deteksi api berdasarkan sensor visual memberikan banyak keuntungan. Pertama, peralatan yang digunakan relatif murah seperti sistem yang berbasis CDC (*Charge Coupled Device*) *cameras*, yang mana sudah banyak dipasang ditempat umum. Kedua, kecepatan untuk mendeteksi lebih cepat karena kamera tidak menunggu asap atau panas menyebar. Ketiga,petugas bisa mengonfirmasi keberadaan api tanpa mengunjungi lokasi kejadian.

Dari masalah yang ada, tujuan dari usulan tugas akhir ini yaitu, membuat sistem pencegahan kebakaran menggunakan rekaman video. Data yang akan digunakan adalah data rekaman video, dalam prosesnya sistem akan memproses gambar per *frame* dengan jumlah frame yang telah ditentukan. Tiap input *frame* akan dilakukan preprocessing, pertama proses pemisahan antara obyek dan latar belakang. Metode yang digunakan dalam tugas akhir ini adalah 1) Reduksi *size* *frame*; 2) Deteksi gerak; 3) Deteksi warna piksel; 4) *Region growing*; 5) Perhitungan luasan *region*; 6) Ekstraksi Fitur; 7) Klasifikasi menggunakan *support vector machines*.

## Rumusan Masalah

Rumusan masalah yang diangkat dalam tugas akhir dapat dipaparkan sebagai berikut.

1. Bagaimana melakukan deteksi warna piksel.
2. Bagaimana melakukan *region growing* pada setiap kandidat piksel.
3. Bagaimana melakukan ekstraksi fitur pada setiap kandidat piksel.
4. Bagaimana melakukan klasifikasi untuk verisikasi piksel api.

## Batasan Masalah

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

1. Implementasi menggunakan bahasa pemrograman Python berbasis dekstop.
2. Jumlah piksel objek api yang dideteksi lebih besar dari 1% dari luas piksel *frame*.
3. Data yang digunakan adalah data video dengan panjang video 6 – 16 detik.
4. Data video memiliki ukuran 240 x 320 piksel dengan *channel* R,G,B.
5. Warna api yang didefinisikan adalah *range* warna kuning hingga merah.
6. Pergerakan dari kamera tidak terlalu besar.

## Tujuan

Tujuan dari pembuatan tugas akhir ini adalah merancang dan membangun perangkat lunak deteksi api menggunakan data video yang dapat melakukan deteksi api pada setiap *frame*.

## Manfaat

Pengerjaan tugas akhir ini dilakukan dengan harapan bisa memberikan kontribusi pada sistem keamanan kebakaran dan mempercepat deteksi kebakaran.

## 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 pembuatan perangkat lunak untuk melakukan deteksi api menggunakan data video.

1. Studi Literatur

Pada tahap ini dilakukan pencarian, pengumpulan, penyaringan, pemahaman, dan pembelajaran literatur yang berhubungan dengan reduksi *frame*, deteksi gerak, *region growing*, distribusi probabilitas gaussian, wavelet, dan *support vector machines*. Literatur yang digunakan meliputi: buku referensi, jurnal, dan dokumentasi internet

1. Implementasi dan pembuatan perangkat lunak

Pada tahap ini dilakukan impelementasi perangkat lunak sesuai dengan rancangan perangkat lunak yang dibuat.

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 parameter *threshold*  pada deteksi warna piksel, dan nilai konstanta C pada 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.

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

# TINJAUAN PUSTAKA

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

## Adaptive Background Mixture Models

*Adaptive background mixture models* adalah metode deteksi gerak. Didefinisikan piksel pada *frame* yang sedang di proses sebagai dengan nilai sebelumnya dari *frame* 1 sampai *frame* t. Model dapat dilihat pada persamaan berikut.

|  |  |
| --- | --- |
|  | (2.1.1) |

Probabilitas piksel didefinisikan sebagai berikut.

|  |  |
| --- | --- |
|  | (2.1.2) |

Dimana adalah nomor gaussian (antara 3 hingga 5), adalah *covariance* yang didapatkan dari variance \* identity matrix, adalah *weight*, adalah *gaussian probability destiny function*. *Gaussian probability destiny function* dapat dilihat pada persamaan berikut.

|  |  |
| --- | --- |
|  | (2.1.3) |

Jika standar deviasi dari *mean*, maka diberi label *matched*, selain itu diberi label *unmatched*. …

## Gaussian Pyramid

Gaussian pyramid digunakan untuk melakukan reduksi resolusi citra. Citra yang tereduksi resolusinya akan berkurang menjadi setengah dari resolusi awal. Hal ini dilakukan untuk mempercepat proses perhitungan.



Gambar 2.2.1 Ilustrasi Gaussian Pyramid

Gaussian pyramid dilakukan dengan dua operasi, yaitu *smoothing* dan *down sampling*. *Smoothing* dilakukan dengan menggunakan filter 5x5. *Smoothing* dilakukan dengan menggunakan persamaan berikut.

|  |  |
| --- | --- |
|  | (2.2.1) |

Dimana adalah filter 5x5. Detail persamaan setiap piksel dilakukan menggunakan persamaan berikut.

|  |  |
| --- | --- |
|  | (2.2.2) |

Setelah mendapatkan citra yang telah di *smoothing*, langkah selanjutnya adalah melakukan *down sampling*. *Down sampling* dilakukan untuk mengubah resolusi citra asli menjadi resolusi yang lebih kecil. *Down sampling* dilakukan dengan persamaan berikut.

|  |  |
| --- | --- |
|  | (2.2.3) |

Untuk melakukan reduksi, dilakukan menggunakan dua proses tersebut, yaitu *smoothing* dan *down sampling*. Perhitungan dapat dilakukan dengan menggabungkan ke dua persamaan tersebut menjadi [1].

|  |  |
| --- | --- |
|  | (2.2.4) |

## Probabilitas Distribusi Gaussian

Probabilitas distribusi gaussian adalah sebuah metode untuk menghitung probabilitas dari suatu data. Probabilitas dilakukan dengan menghitung nilai rata-rata dan standar deviasi dari suatu data. Persamaan umum probabilitas distribusi gausian dapat dilihat pada persamaan berikut [2].

|  |  |
| --- | --- |
|  | (2.3.1) |

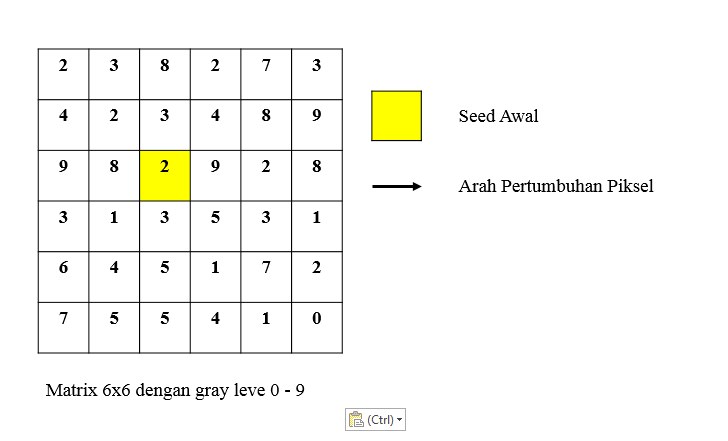
Pada kasus tugas akhir ini, probabilitas gaussian digunakan untuk mendapatkan probabilitas warna piksel. Setiap piksel dihitung nilai probabilitas R,G,B. Setelah dilakukan perhitungan probabilitas R,G,B selanjunya probabilitas tersebut dikalikan sehingga mendapatkan nilai probabilitas piksel. Perhitungan probabilitas setiap piksel dapat dilihat pada persamaan berikut.

|  |  |
| --- | --- |
|  | (2.3.2) |

## Region Growing

*Region Growing* adalah metode untuk melakukan segmentasi citra. Pendekatan dasar adalah degan memulai titik yang sudah diinisialisasi (*seed*) dan dari titik tersebut dilakukan menumbuhkan daerah dengan cara menambahkan tetangga piksel dari *seed* yang mempunyai kesamaan dengan *seed* [3].

Untuk metode ini, dibutuhkan aturan yang mengatur mekanisme tumbuhnya *seed* dan suatu aturan lain yang menguji kehomogenan dari *region* setelah satu tahap tumbuh selesai.pertumbuhan *region* dimulai dari seed awal dengan menambahkan tetangga piksel (menggunakan 8-tetangga) yang serupa untuk menumbuhkan *region.* Ketika suatu pertumbuhan *region* selesai, langkah selanjutnya adalah memilih *seed* baru dan melakukan *region growing* kembali. Proses tersebut dilakukan hingga semua piksel berhasil dikelompokkan dalam beberapa *region*.



Gambar 2.4.1 *Seed* Awal *Region Growing*



Gambar 2.4.2 Piksel yang diamati dan *Region*

## Daubachies D4 Wavelet

Wavelet adalah fungsi matematika yang membagi data menjadi beberapa komponen frekuensi yang berbeda-beda dan menganalisis setiap komponen tersebut dengan menggunakan resolusi yang sesuai dengan skalanya. Transformasi wavelet mendekomposisi signal kedalam *frequency bands* dengan memproyeksikan signal kedalam set fungsi dasar [4]. Pada citra, transformasi wavelet adalah transformasi yang melakukan filter terhadap suatu masukkan. Filter yang digunakan dalam wavelet adalah *high pass filter* dan *low pass filter*. Pada daubachies D4, terdapat dua koefisien yang digunakan untuk melakukan *filter*, yaitu *scaling function coefficient* dan *wavelet function coefficient*. Dimana *scaling function coeficient* adalah koefisien yang digunakan dalam melakukan *low pass filter*, sedangkan *wavelet function coeficient* digunakan dalam melakukan *high pass filter* . Berikut persamaan *scaling function coeficient.*

|  |  |
| --- | --- |
|  | (2.5.1) |

*Wavelet function coeficient* dapat dilihat pada persamaan berikut.

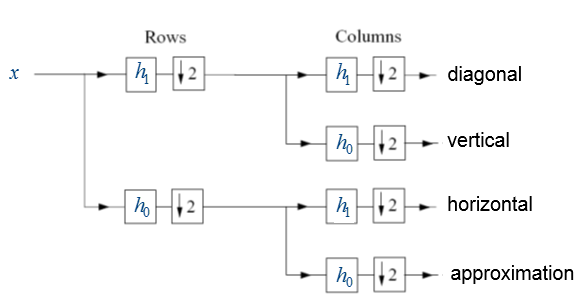
|  |  |
| --- | --- |
|  | (2.5.1) |

Tahapan dari proses *filter* adalah melakukan *filter* terhadap baris citra terlebih dahulu. Dilakukan *high pass filter* dan *low pass filter*, dimana hasil dari proses ini adalah dua citra, yaitu citra *high pass* dan citra *low pass*. Ilustrasi tahap *filter* dapat dilihat pada Gambar 2.5.1

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  | **.** |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

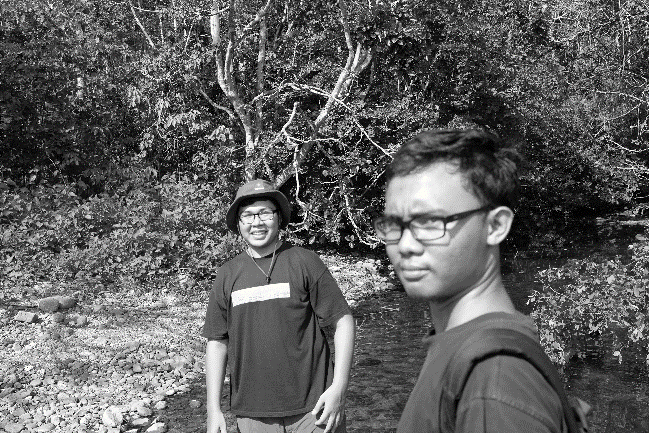
Gambar 2.5.1 Ilustrasi *Low Pass Filter*

Citra yang dilakukan *filter* akan ter-reduksi menjadi setengah. Dilanjutkan dengan melakukan *filter* terhadap kolom citra, dilakukan *high pass filter* dan *low pass filter*. Hasil dari proses ini adalah empat citra dengan satu citra approximasi, dan tiga citra detail. Citra detail yang didapat adalah detail horizontal, vertikal dan diagonal. Gambar 2.5.2 Adalah lustrasi dari proses *filter* yang dilakukan.



Gambar 2.5.2 Ilustrasi *Filter* yang dilakukan

Ilustrasi citra yang dikeluarkan pada proses ini dapat dilihat pada Gambar 2.5.4.



Gambar 2.5.3 Citra Masukkan



Gambar 2.5.4 Citra Keluaran Hasil Wavelet

## Normalisasi Min-Max

Normalisasi adalah sebuah proses untuk mengubah suatu data ke dalam rentang nilai tertentu. Tujuannya adalah untuk menghindari persebaran data yang terlalu jauh sehingga sebuah variabel tidak mendominasi terhadap variabel lain. Salah satu jenis normalisasi adalah normalisasi skala. Pada normalisasi, skala rentang yang umum digunakan yaitu 0 hingga 1. Rumus umum skala adalah sebagai berikut .

|  |  |
| --- | --- |
|  | (2.6.1) |

Variabel adalah nilai yang akan dinormalisasi. Variabel dan adalah nilai minimum dan maximum pada nilai-nilai atribut dimana berada. Variabel dan adalah nilai minimum dan maximum yang diinginkan. Apabila rentang yang digunakan adalah 0 hingga 1 maka rumus diatas menjadi seperti berikut [5].

|  |  |
| --- | --- |
|  | (2.5.2) |

## Support Vector Machines

*Support vector machines* adalah metode klasifikasi yang mengklasifikasikan dua kelas, yaitu kelas +1 dan -1. Pada metode klasifikasi *support vector machines*, akan dibentuk suatu *hyperplane*. *Hyperplane* adalah garis pemisah yang memisahkan dua kelas yang berbeda. Dalam metode *support vector machines* dikenal istilah *margin*. *Margin* adalah jarak antara kelas +1 dengan kelas -1 yang paling dekat, pada *support vector machines* dicari *margin* terpanjang. Ilustrasi *support vector machines* dapat dilihat pada Gambar 2.6.1.



Gambar 2.7.1 Ilustrasi *Support Vector Machines*

Diberikan masukkan berupa data belajar dan masing-masing kelas dianotasikan untuk i = 1,2,3,…,l , dimana l adalah banyaknya data. Fungsi *hyperplane* dibuat dengan persamaan berikut.

|  |  |
| --- | --- |
|  | (2.6.1) |

Dalam mencari nilai w dan b yang optimal, dilakukan dengan persamaan berikut.

|  |  |
| --- | --- |
|  | (2.7.2) |

Persamaan diatas mempunyai variabel C, dimana variabe tersebut adalah konstanta nilai pinalti dari kesalahan klasifikasi. Pencarian nilai w dan b dilakukan dengan batasan yang ditulis menggnakan persamaan berikut.

|  |  |
| --- | --- |
|  | (2.7.1) |

Pada kebanyakan kasus, masukkan dari klasifikasi berdimensi lebih dari dua. Oleh karena itu, digunakan fungsi kernel untuk mengatasi permasalahan tersebut. Salah satu kernel yang biasa dipakai adalah kernel RBF. Persamaan kernel RBF dapat dilihat pada persamaan berikut.

|  |  |
| --- | --- |
|  | (2.6.4) |

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

# DESAIN PERANGKAT LUNAK

Pada bab ini dijelaskan mengenai rancangan sistem perangkat lunak yang akan dibuat. Perancangan yang dijelaskan meliputi data dan proses. Data yang dimaksud adalah data yang akan diolah dalam perangkat lunak baik digunakan sebagai pebelajaran maupun pengujian sehingga tujuan Tugas Akhir ini bisa tercapai. Proses yaitu tahap-tahap yang ada dalam sistem sebagai pengolah data meliputi reduksi *size* *frame*, deteksi gerak, deteksi warna piksel, *region growing*, perhitungan luasan *region* api, ekstraksi fitur dan klasifikasi.

## Data

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

### Data Masukkan

Data masukkan adalah data yang digunakan sebagai masukan dari sistem. Data yang digunakan adalah data video yang memiliki frekuensi minimal 20 Hz, kualitasvideo yang digunakan adalah 240x320 piksel.



Gambar 3.1.1 Contoh Data Masukkan

### Data Pembelajaran

Data pembelajaran digunakan sebagai data belajar klasifikasi. Data yang digunakan adalah data video yang dibagi dalam dua jenis yaitu video berisi objek api dan video berisi objek bukan api.

### Data Keluaran

Data masukkan akan diputar ulang dan diproses setiap *frame* menggunakan metode reduksi *frame*, deteksi gerak, deteksi warna piksel, *region growing*, perhitungan luasan *region* api, ekstraksi fitur, dan klasifikasi. Dari proses tersebut diharapkan keluaran barupa tanda pada tiap frame jika memiliki piksel api. Tanda yang dimaksud adalah tanda berwarna biru yang menghubungkan empat nilai titik koordinat ekstrim piksel api.



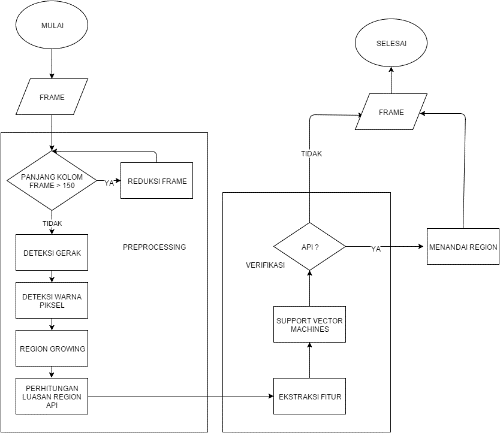
Gambar 3.1.2 Contoh Data Keluaran

## Desain Sistem Secara Umum

Rancangan perangkat lunak deteksi api berbasis sensor visual menggunakan *support vector machines* dimulai dengan membaca masukan berupa file video. Setiap *frame* akan dilakukan *preprocessing* sebelum dilakukan verifikasi menggunakan *support vector machines*. Tahap pertama *preprocessing* adalah mengubah *size frame* yang diproses. Ukuran *frame* diubah hingga panjang kolom *frame* lebih kecil dari 150 piksel. Tahap berikutnya adalah malakukan deteksi gerak *frame*, hasil dari deteksi gerak adalah piksel-piksel bergerak terhadap *frame* sebelumnya. Setelah mendapatkan piksel-piksel bergerak, tahap selanjutnya adalah deteksi warna tiap piksel menggunakan probabilitas distribusi gaussian. Warna api yang didefinisikan pada sistem ini adalah warna yang memiliki *range* dariwarna kuning hingga merah. Hasil keluaran dari metode deteksi warna piksel adalah piksel-piksel yang masuk kedalam kandidat piksel api.

Setelah mendapatkan kandidat piksel api, dilakukan metode *region growing* untuk mendapatkan *region* kandidat piksel api. Hasil dari *region growing* digunakan pada tahap selanjutnya yaitu luasan *region* api. Pada metode luasan *region* , jika luasan *region* melebihi *threshold,* maka kandidat piksel yang masuk pada *region* tersebut merupakan kandidat piksel api selanjutnya.

Setelah melalui tahap *preprocessing*, piksel-piksel yang termasuk kandidat api akan di verifikasi menggunakan metode *support vector machines*. Fitur didapatkan dari nilai konstanta wavelet setiap piksel statis dengan sepuluh *frame* yang berurutan. Hasil akhir yang dikeluarkan adalah adanya penanda pada *frame* yang diproses jika *frame* tersebut mengandung piksel api. Data dibagi menjadi data pembelajaran dan data uji sehingga dapat diperoleh nilai *true positif*, *false positif*, dan *missing rate* dari hasil klasifikasi. Diagram alir desain umum perangkat lunak ditunjukkan pada Gambar 3.2.1 berikut ini.



Gambar 3.2.1 Diagram alir rancangan perangkat lunak secara umum

## Preprocessing

Setiap piksel pada *Frame* yang diproses tidak langsung dilakukan verifikasi untuk menentukan apakah piksel tersebut piksel api atau bukan. Tahap awal yang dilakukan adalah *preprocessing. Preprocessing* bertujuan untuk menghilangkan piksel-piksel yang tidak memiliki karakteristik piksel api. *Preprocessing* dilakukan melalui lima tahap yaitu reduksi *size frame*, deteksi gerak, deteksi warna piksel, *region growing,* perhitungan luasan *region* api.

### Reduksi Size Frame

Tahap reduksi *size frame* adalah tahap dimana *size* dari *frame* direduksi. Kualitas *frame* masukkan direduksi dengan tujuan mempercepat proses pendeteksian api. Gambar 3.3.1 adalah diagram alir dari proses reduksi *size frame*.

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

Gambar 3.3.1 Reduksi *Size Frame*

### Deteksi Gerak

Didalam proses ini dilakukan proses deteksi gerak piksel. Masukkan dari proses ini adalah *frame* yang sedang di proses. Untuk mendapatkan kandidat piksel yang bergerak, dilakukan dengan metode *adaptive background mixture model*. Dari metodetersebut didapatkan piksel-piksel yang bergerak. Ilustrasi deteksi gerak ditunjukkan pada Gambar 3.3.2.



Gambar 3.3.2 Contoh Deteksi Gerak

Setelah mendapatkan piksel yang bergerak, kumpulan piksel tersebut disimpan untuk diproses di tahap berikutnya, yakni deteksi warna piksel.

### Deteksi Warna Piksel

Penentuan warna piksel yang termasuk kandidat piksel api menggunakan metode distribusi probabilitas gaussian. Masukkan dari tahap ini adalah *frame* dengan *channel* R,G,B yang sedang di proses dan *list* piksel bergerak yang didapatkan pada tahap deteksi gerak. Setiap piksel bergerak dihitung perkalian probabilitas R,G,B dan ditentukan apakah piksel tersebut masuk kedalam kandidat piksel api atau tidak.

Sebelum melakukan perhitungan probabilitas pada setiap piksel yang bergerak, dilakukan proses mendapatkan nilai rata-rata dan standar deviasi *channel* R,G,B. Nilai rata-rata dan standar deviasi *channel* R,G,B didapatkan dari data yang diambil dari memproses *dataset* gambar api sebanyak sebelas gambar.



Gambar 3.3.3 Contoh *Dataset* Gambar Api

Perhitungan probabilitas piksel dilakukan dengan mencari probabilitas setiap nilai R,G,B menggunakan probabilitas distribusi gaussian. Selanjutnya nilai probabilitas R,G,B dikalikan untuk mendapatkan nilai probabilitas dari piksel tersebut. Jika nilai probabilitas piksel tersebut melebihi *threshold*, maka piksel tersebut dimasukkan kedalam kandidat piksel api. *Pseudocode* fungsi ini dapat dilihat pada Gambar 3.3.4

|  |  |
| --- | --- |
| 1.  2.  3.  4.  5.  6.  7. | for i=1 to length(CandidatePiksel)  R,G,B = CandidatePiksel[i].RGBvalue  If probability(R)\*probability(G)\* probability(B) > threshold  add CandidatePiksel[i] to firePiksel  else  add CandidatePiksel[i] to nonFirePiksel  return firePiksel |

Gambar 3.3.4 *Pseudocode* Deteksi Warna Piksel

Keluaran dari tahap ini adalah *list* kandidat piksel api yang akan diproses selanjutnya di tahap *region growing* dan luasan *region* api.

### Region Growing

Kandidat piksel api yang didapatkan pada tahap deteksi warna piksel dilakukan *region growing* untuk mendapatkan *region* dari setiap piksel. Masukkan dari tahap ini adalah *list* kandidat piksel api dan *frame* yang diproses. Setiap piksel akan dilakukan *region growing* dengan cara melakukan pengecekan terhadap delapan tetangga piksel tersebut. Untuk menentukan apakah piksel tetangga tersebut termasuk *region* api atau tidak, dilakukan dengan cara menentukan nilai probabilitas warna R,G,B menggunakan distribusi probabilitas gaussian. Jika nilai probabilitas tersebut melebihi *threshold*, maka piksel tersebut dianggap satu *region*. *Pseudocode* fungsi ini dapat dilihat pada Gambar 3.3.5.

|  |  |
| --- | --- |
| 1.  2.  3.  4.  5.  6.  7.  8.  9.  10.  11.  12.  13.  14.  15.  16. | regionCounter = 0  region.size = image.size  clockWise = clockWise()  for i=1 to length(CandidatePiksel)  push CandidatePiksel[i] to stack regionStack  if CandidatePiksel[i].is\_visit == False  increment regionCounter  while regionStack is not empty  pop regionStack and assign to temporary  R,G,B = temporary.RGBvalue  if probability(R)\*probability(G)\*probability(B) > threshold and temporary.is\_visit == False  temporary.is\_visit = True  region[temporary] = regionCounter  for j to clockWise  push temporary.index – j to stack region Stack  return region |

Gambar 3.3.5 *Pseudocode Regon Growing*

Keluaran dari metode *region growing* adalah *region* dengan nilai yang berbeda tiap *region*. Nilai tersebut membedakan antara satu *region* dengan *region* lainnya.

### Perhitungan Luasan Region Api

Perhitungan luasan *region* api adalah metode untuk melakukan *filter* terhadap *region* api. Masukkan dari tahap ini adalah *region* yang telah diperoleh dari metode *region growing*, kandaidat piksel api yang didapatkan dari metode deteksi warna piksel. Pada tahap ini dilakukan proses perhitungan luasansetiap *region*. Jika luasan *region* melebihi *threshold*, maka kandidat piksel api yang ada pada *region* tersebut masuk kedalam kandidat piksel api. Gambar 3.3.6 adalah diagram alir dari proses luasan *region* api.

F:\KULIAH\TA\bukuTA\source\variasi warna region.png

Gambar 3.3.6 Diagram Alir Luasan *Region* Api

Keluaran dari proses ini adalah *list* piksel yang termasuk kedalam kandidat piksel api.

## Verifikasi

Pada sub bab ini akan dijelaskan mengenai proses verifikasi piksel sehingga menghasilkan keluaran yang diharapkan seperti yang sudah dijelaskan sebelumnya. Setelah melalui tahap *preprocessing*, piksel-piksel yang masuk kedalam kandidat piksel api dilakukan tahap verifikasi menggunakan metode klasifikasi *support vector machines*. Sebelum dilakukan klasifikasi, terdapat proses untuk mendapatkan fitur sebagai data masukan klasifikasi. Pencarian fitur dilakukan dengan mengubah gambar spasial kedalam domain wavelet. Setelah mendapatkan fitur yang dicari, dilakukan klasifikasi untuk menentukan apakah piksel tersebut masuk kedalam piksel api atau bukan.

### Ekstraksi Fitur dengan Wavelet

Data masukkan yang digunakan pada ekstraksi fitur adalah data *frame* sebanyak sepuluh *frame* secara berurutan. *Frame* diambil dari *frame* yang sedang diproses berserta sembilan *frame* sebelumnya. Ekstraksi fitur dilakukan dengan mengubah *frame* kedalam domain wavelet. *Frame* yang diubah kedalam domain wavelet ini berupa sepuluh *frame* yang berurutan. Tiap *frame* akan menghasilkan empat sub *frame* baru, yaitu sub *frame* *low-low* (LL), *low-high* (LH), *high-low* (HL), dan *high-high* (HH). Untuk ekstraksi fitur, sub *frame* yang digunakan adalah sub *frame* LL, LH, HH. Wavelet yang digunakan adalah wavelet daubechies D4 . Tiap kandidat piksel api dicari nilai piksel pada sub *frame* wavelet yang telah dilakukan. Setiap piksel mendapatkan tiga nilai untuk setiap *framenya*, yaitu nilai LL, LH, HH. Dari ketiga nilai tersebut akan dihitung menggunakan persamaan :

|  |  |
| --- | --- |
|  | (3.4.1) |

Total nilai yang didapatkan adalah sepuluh buah nilai fitur. Sebelum masuk tahap klasifikasi, nilai fitur yang didapatkan dinormalisasi menggunakan normalisasi min-max. Nilai min-max didapatkan dengan melakukan perhitungan nilai kuadrat sub *frame* LH, HL, HH. Selanjutnya nilai LH, HL, dan HH dikalkulasi menjadi satu, dicari nilai minimum dan maximum dari hasil perhitungan tersebut. Dari proses ini akan dilakukan terhadap seluruh *frame*, dimana nilai min-max yang didapatkan ada sepuluh buah. Normalisasi dilakukan dengan menyesuaikan nilai min-max terhadap *frame* yang dicari nilai wavelet setiap pikselnya. Setelah dilakukan normalisasi, dilakukan pengurutan nilai fitur secara *ascending*. Gambar 3.4.1 adalah diagram alir dari proses ekstraksi fitur.

F:\KULIAH\TA\bukuTA\source\ekstraksi fitur.png

Gambar 3.4.1 Diagram alir Ekstraksi Fitur

Fitur akhir yang didapatkan adalah fitur dengan dimensi sebesar sepuluh dimensi yang sudah terurut. Selanjutnya akan masuk kedalam tahap klasifikasi.

### Klasifikasi

Klasifikasi dilakukan dengan mengolah data fitur yang sudah didapatkan dari proses ekstraksi fitur. Metode klasifikasi yang digunakan adalah *support vector machines* menggunakan kernel RBF. Masukkan dari tahap klasifikasi adalah nilai fitur wavelet sebanyak sepuluh dimensi yang sudah dijelaskan sebelumnya . Hasil akhir dari proses ini adalah piksel-piksel yang masuk kedalam piksel api.

## Menandai Region Api

Penanda *region* api dilakukan untuk menunjukkan bagian yang terdeteksi api. Masukkan dari proses ini adalah piksel-piksel yang lolos tahap verifikasi dan *frame* yang sedang di proses. *Pseudocode* fungsi ini dapat dilihat pada Gambar 3.3.5.

|  |  |
| --- | --- |
| 1.  2.  3.  4.  5.  6. | Min\_x, Max\_x = min(X), max(X)  Min\_y, Max\_y = min(Y), max(Y)  For i = Min\_y to Max\_y:  For j = Min\_x to Max\_x :  Mark(Image[i][j])  Return Image |

Gambar 3.5.1 *Pseudocode* fungsi menandai *region*

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

# IMPLEMENTASI

Pada bab ini akan dibahas mengenai implementasi yang dilakukan berdasarkan rancangan yang telah dijabarkan pada bab sebelumnya. Implementasi kode program dilakukan dengan menggunakan bahasa Python.

## Lingkungan Implementasi

Spesifikasi perangkat keras dan perangkat lunak yang digunakan dalam implementasi ini ditampilkan pada Tabel berikut.

|  |  |
| --- | --- |
| Perangkat | Spesifikasi |
| Perangkat keras | Prosesor: Intel® Core™ i3-2350M CPU @ 2.30GHz 2.30GHz  Memori: 4.00 GB |
| Perangkat lunak | Sistem Operasi: Microsoft Windows 8 64-bit Pro  Perangkat Pengembang: PyCharm  Perangkat Pembantu:  Microsoft Excel 2013 |

Tabel 4.1.1 Lingkungan Perancangan Perangkat Lunak

## Implementasi

Subbab implementasi ini menjelaskan tentang implementasi proses yang sudah dijelaskan pada bab desain perangkat lunak.

### Implementasi Tahap Reduksi Size Frame

Subbab ini membahas implementasi tahap reduksi *size frame*. Pada tahap ini data masukan berupa *frame* dan data keluaran yang dihasilkan pada tahap ini adalah *frame* yang telah tereduksi kualitasnya. Implementasi dilakukan dengan menggunakan fungsi yang sudah disediakan oleh OpenCV yaitu pyrDown dan di tunjukkan oleh Kode Sumber 4.2.1

|  |  |
| --- | --- |
| 1.  2.  3.  4. | while (len(currentFrame)>150):  if (len(currentFrame)<=300):  currentFrame2 = copy.copy(currentFrame)  currentFrame = ImageProcessing.getDownSize(currentFrame) |

Kode Sumber 4.2.1 Implementasi Tahap Reduksi *Size Frame*

|  |  |
| --- | --- |
| 1.  2. | def getDownSize(self,image):  return cv2.pyrDown(image) |

Kode Sumber 4.2.2 Penggunaan Fungsi pyrDown()

Kode Sumber 4.2.1 melakukan reduksi hinggal piksel kolom masukkan lebih kecil dari 150 piksel dengan memanggil fungsi getDownSize(). Variabel currentFrame2 akan digunakan sebagai masukan dari ekstraksi fitur dan *frame* keluaran yang ditampilkan, sedangkan proses lainnya akan menggunakan variabel currentFrame.

### Implementasi Tahap Deteksi Gerak

Subbab ini membahas implementasi tahap deteksi gerak piksel. Masukkan dari tahap ini adalah *frame* yang sedang diproses. Implementasi dilakukan menggunakan fungsi yang sudah disediakan oleh OpenCV yaitu BackgroundSubtractorMOG(). Implementasi ditunjukkan oleh Kode Sumber 4.2.3

|  |  |
| --- | --- |
| 1.  2.  3. | def getMovingForeGround(self, image):  self.BckgrSbsMOG = cv2.BackgroundSubtractorMOG()  return self.BckgrSbsMOG.apply(image,learningRate = 0.00) |

Kode Sumber 4.2.3 Implementasi Tahap Deteksi Gerak

Hasil yang dikembalikan oleh fungsi getMovingForeGround() adalah gambardengan nilai tiap pikselnya antara 0 atau 255 dengan. Dimana nilai 255 adalah nilai piksel yang bergerak terhadap piksel pada *frame* sebelumnya. Hasil dari fungsi getMovingForeGround() tidak secara langsung diproses kedalam tahap selanjutnya. Dari hasil *frame* ini akan diambil indeks piksel yang bergerak, implementasi ditunjukkan oleh Kode Sumber 4.2.4

|  |  |
| --- | --- |
| 1.  2.  3. | def getMovingCandidatePiksel(self, moving\_frame):  listY,listX = np.where( moving\_frame == 255 )  return np.vstack((listY,listX)) |

Kode Sumber 4.2.4 Implementasi Penyimpanan Piksel

Fungsi getMovingCandidatePiksel() melakukan iterasi untuk mendapatkan piksel-piksel yang bernilai 255. Fungsi ini mengembalikan hasil berupa *list* indeks piksel yang bergerak. Hasil keluaran dari keseluruhan proses deteksi gerak adalah *list* yang berisi indeks piksel y dan x.

### Implementasi Tahap Deteksi Warna Piksel

Deteksi warna piksel dilakukan dengan pengecekan warna setiap piksel pada *list* pikselyang dihasilkan melalui proses deteksi gerak. Masukkan dari tahap ini adalah *list* piksel yang telah didapatkan pada tahap deteksi gerak dan *frame* dengan *channel* R,G,B. Sebelum menghitung probabilitas warna piksel, terlebih dahulu dilakukan perhitungan mencari nilai rata-rata dan standar deviasi untuk nilai piksel R,G,B. Sebelas gambar api disimpan dan dilakukan proses perhitungan nilai rata-rata dan nilai standar deviasi untuk setiap *channel*. Pada implementasi sistem, proses menghitung nilai rata-rata dan standar deviasi setiap *channel* dapat dilihat pada Kode Sumber 4.2.5

|  |  |
| --- | --- |
| 1.  2.  3.  4.  5.  6.  7.  8.  9.  10.  11.  12.  13.  14.  15.  16.  17.  18.  19.  20.  21.  22. | def getStdDevAndMean(self,path):  list\_file = File.readFolder(self,path)  R = []  G = []  B = []  for x in list\_file:  image = ImageProcessing.readImage(self,path+'/'+x)  r = np.array(image[:,:,2]).ravel()  g = np.array(image[:,:,1]).ravel()  b = np.array(image[:,:,0]).ravel()  R += (r.tolist())  G += (g.tolist())  B += (b.tolist())  mean = []  mean.append(np.average(B))  mean.append(np.average(G))  mean.append(np.average(R))  standard\_deviasi = []  standard\_deviasi.append(np.std(B))  standard\_deviasi.append(np.std(G))  standard\_deviasi.append(np.std(R))  return standard\_deviasi, mean |

Kode Sumber 4.2.5 Implementasi Menghitung Nilai Standar Deviasi dan Mean Setiap *Channel*

Pada Kode Sumber 4.2.5 setiap gambar *dataset* dibaca dan dilakukan pemisahan setiap *channel*. Selanjutnya, dilakukan perhitungan rata-rata dan perhitungan standar deviasi setiap *channel*. Setelah didapatkan nilai rata-rata dan standar deviasi setiap *channel*, proses penentuan kandidat api menggunakan metode probabilitas warna piksel dilakukan. Setap piksel dihitung nilai probabilitas *channel* R,G,B. Selanjutnya, menghitung nilai probabilitas piksel dengan mengalikan nilai probabilitas R,G,B. Proses perhitungan probabilitas piksel memakan waktu yang cukup lama jika menghitung setiap kemungkinan kandidat piksel api yang lolos pada tahap deteksi gerak. Pada implementasi, nilai probabilitas tiap piksel yang masuk kedalam probabilitas warna api dimasukkan kedalam file. Nilai yang disimpan adalah nilai R,G,B piksel yang termasuk kedalam warna piksel api. Hal ini dilakukan karena proses perhitungan yang lama jika menghitungan probabilitas setiap piksel. Isi dari file adalah *list* kemungkinan piksel-piksel yang masuk dalam probabilitas warna api. Proses *generate* *list* piksel dapat dilihat pada Kode Sumber 4.2.6

|  |  |
| --- | --- |
| 1.  2.  3.  4.  5.  6.  7.  8.  9.  10. | def getListColorPiksel  (self,list\_standard\_deviasi, list\_mean):  res = []  threshold = self.getThreshold()  for B in range(0,256):  for G in range(B,256):  for R in xrange(G,256):  if Data.getGaussianProbability(self,B, list\_standard\_deviasi[0], list\_mean[0])\* Data.getGaussianProbability(self,G, list\_standard\_deviasi[1], list\_mean[1])\* Data.getGaussianProbability(self,R, list\_standard\_deviasi[2], list\_mean[2]) > threshold:  res.append([B,G,R])  return res |

Kode Sumber 4.2.6 *Generate list* piksel api

Kode Sumber 4.2.6 melakukan iterasi pengecekan piksel. Iterasi tidak dilakukan sebanyak kombinasi nilai R,G,B. hal ini dikarenakan nilai *channel* R dari piksel api lebih besar dari nilai *channel* G dan nilai *channel* G lebih besar dari nilai *channel* B [3]. Dalam Kode Sumber 4.2.6, memanggil fungsi getGaussianProbability(). Fungsi getGaussianProbability() digunakan untuk menghitung nilai probabilitas tiap *channel* piksel. Fungsi tersebut mengimplementasikan rumus dari probabilitas distribusi gaussian. Implementasi fungsi getGaussianProbability() dapat dilihat pada Kode Sumber 4.2.7.

|  |  |
| --- | --- |
| 1.  2.  3.  4.  5.  6.  7.  8.  9.  10 | def getGaussianProbability(self, data, standard\_deviasi, mean):  data = float(data)  standard\_deviasi = float(standard\_deviasi)  mean = float(mean)  result = pow((data-mean),2)  div = 2\*pow(standard\_deviasi,2)  exp = np.exp(-result/div)  result = standard\_deviasi\*np.sqrt(2\*np.pi)  result = 1/result  return result\* exp |

Kode Sumber 4.2.8 Fungsi Menghitung Nilai Probabilitas Distribusi Gaussian

|  |  |
| --- | --- |
| 1.  2. | def getThreshold(self):  return 5\*pow(10,-9) |

Kode Sumber 4.2.9 Mendapatkan Threshold

Kode Sumber 4.2.8 melakukan perhitungan probabilitas piksel api setiap *channel*. Nilai *threshold* pada Kode Sumber 4.2.8 didapatkan dari hasil analisa yang telah dilakukan. Ketika program dijalankan, program akan membaca file yang berisi *list* piksel yang sudah di *generate* sebelumnya dan menyimpan nilai piksel tersebut *array*. Proses pembacaan file bisa dilihat pada Kode Sumber 4.2.9

|  |  |
| --- | --- |
| 1.  2.  3.  4.  5.  6.  7. | def getFireArray(self,path):  lists = [[[False for k in xrange(256)] for j in xrange(256)]for i in xrange(256)]  data = open(path,'r')  for x in data:  color = (x.split('\n')[0]).split(' ')  lists[int(color[0])] [int(color[1])][int(color[2])] = True  return lists |

Kode Sumber 4.2.10 Membaca *List* Piksel Api

Pada Kode Sumber 4.2.11, dilakukan peroses pembuatan *array* tiga dimensi, dengan *default*  *False* pada nilai indeks. Pada *line* enam, dilakukan perubahan nilai sesuai indeks yang ada pada *list* dengan mengubah nilai *array* tiga dimensi tersebut menjadi *True*. Nilai indeks dimensi *array* tersebut mewakilkan indeks R,G,B. Setelah didapatkan *array* dengan indeks yang mewakili nilai R,G,B dan hasilnya, akan dilakukan pengecekan terhadap kandidat piksel yang didapatkan pada proses deteksi gerak. Implementasi proses pengecekan kandidat piksel dapat dilihat pada Kode Sumber 4.2.12

|  |  |
| --- | --- |
| 1.  2.  3.  4.  5.  6.  7.  8.  9.  10.  11.  12. | def getColorCandidatePiksel(self, list\_standard\_deviasi, list\_mean):  true\_piksel = []  false\_piksel = []  for x in range(0,len(list\_candidate[0])):  data = image[list\_candidate[0][x]] [list\_candidate[1][x]]  B,G,R = data[0],data[1],data[2]  if color\_dataset[B][G][R] == True:  true\_piksel.append([list\_candidate[0][x], list\_candidate[1][x]])  else : false\_piksel.append([list\_candidate[0][x], list\_candidate[1][x]])  return true\_piksel,false\_piksel |

Kode Sumber 4.2.12 Implementasi Tahap Deteksi Warna Piksel

Hasil yang dikeluarkan pada tahap ini adalah *list* nilai indeks piksel yang masuk kedalam piksel api.

### Implementasi Tahap Region Growing

Sub bab ini membahas implementasi tahap *region growing* yang menggunakan kandidat piksel api pada tahap deteksi warna api sebagai masukkan. *Region growing* dilakukan menggunakan *list* kandidat piksel api sebagai masukan awal dari piksel yang dicari *region*nya. Titik piksel kandidat api akan 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 4.2.13

|  |  |
| --- | --- |
| 1.  2.  3.  4.  5.  6.  7.  8.  9.  10.  11.  12.  13.  14.  15.  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 4.2.13 Implementasi Tahap *Region Growing*

Kode Sumber 4.2.13 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 dimasukkan kedalam *stack* dan dilakukan *growing*. Implementasi *growing* dapat dilihat pada Kode Sumber 4.2.14.

|  |  |
| --- | --- |
| 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 4.2.14 Implementasi *Growing*

Kode Sumber 4.2.14 melakukan pegecekan terhadap *stack* piksel yang akan dicek. Jika stack masih memiliki nilai, maka nilai tersebut akan digunakan sebagai titik 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 dimasukkan kedalam *stack* untuk melakukan pengecekan tetangga selanjutnya. Pada fungsi growing() melakukan pemenggilan fungsi getClockWise(), dimana fungsi ini akan mengembalikan *array* yang akan digunakan untuk iterasi pengecekan delapan tetangga piksel. Implementasi getClockWise() dapat dilihat pada Kode Sumber 4.2.14.

|  |  |
| --- | --- |
| 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 4.2.15 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 Luasan Region Api

Subbab ini membahas implementasi tahap luasan *region* api. Masukkan dari tahap ini adalah kandidat piksel api yang telah didapatkan dari tahap probabilitas warna api, dan *region*. Pada tahap luasan *region* api 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 mendapatkan nilai variasi warna regiondapat dilihat pada Kode Sumber 4.2.16

|  |  |
| --- | --- |
| 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 4.2.16 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 kedala 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* di ubah kedalam domain wavelet dan disimpan kedalam *list*. Implementasi mengubah dan manyimpan *frame* domain wavelet dapat di lihat pada Kode Sumber 4.2.15

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

Kode Sumber 4.2.17 Implementasi Memasukkan Nilai Wavelet Kedalam *List*

Pada Kode Sumber 4.2.20 ,fungsi toWavelet() digunakan untuk mengubah *frame* kedalam domain wavelet. Implementasi mengubah *frame* kedalam domain wavelet dapat dilihat pada Kode Sumber 4.2.16

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

Kode Sumber 4.2.18 Pemanggilan Fungsi Wavelet

### Implementasi Tahap Klasifikasi

Sub bab ini membahas implementasi tahap klasifikasi. Pada tahap ini sistem diberi masukkan kandidat piksel api dan sepuluh buah *frame* domain wavelet, dimana masing-masing *frame* berisi tiga buah sub *frame* seperti penjelasan subbab 3.4.1 . Sebelum melakukan klasifikasi, data training terlebih dahulu diproses sebagai data *training* untuk klasifikasi. Implementasi *training* klasifikasi dapat dilihat pada Kode Sumber 4.2.19

|  |  |
| --- | --- |
| 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 4.2.19 *Training* Klasifikasi

Setiap piksel pada kandidat piksel api dilakukan perhitungan nilai fitur seperti yang sudah dijelaskan pada subbab 3.4.1. Implementasi klasifikasi dapat dilihat pada Kode Sumber 4.2.20

|  |  |
| --- | --- |
| 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 4.2.20 Implementasi Tahap Klasifikasi

Pada Kode Sumber 4.2.20 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 dimasukkan 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. Masukkan 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 4.2.21.

|  |  |
| --- | --- |
| 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 4.2.21 Implementasi Tahap Menandai Region Api

Hasil keluaran dari Kode Sumber 4.2.21 adalah *frame* dengan tanda persegi jika terdapat piksel api pada *frame* yang diproses.

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

# 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 meiputi dua buah jenis video. Video dengan objek api dan video dengan objek bukan api. Jumlah video yang di uji berjumlah lima puluh tiga video dengan jumlah video api sejumlah dua puluh lima dan video bukan api berjumlah dua puluh delapan.

|  |  |  |  |
| --- | --- | --- | --- |
| 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 5.2.1 Contoh Video Kejadian

## Skenario Uji Coba

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

1. Perbandingan hasil akurasi 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 akurasi berdasarkan variasi nilai C (*penalty error term*) pada klasifikkasi dengan kernel tetap yaitu rbf. Nilai C yang akan diuji yaitu 1, 3.5, 5.

### Skenario Uji Coba 1

Sekenario uji coba 1 adalah perhitungan *true positif*, *false positif*, dan *missing rate*. Dimana *false 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 sekenario 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.

|  |  |  |  |
| --- | --- | --- | --- |
| *Threshold* | *True Positif* | *False Positif* | *Missing Rate* |
| 10-7 | 72.64 | 0.36 | 27.01 |
| 10-8 | 88.80 | 0.44 | 10.76 |
| 5x10-9 | 91.69 | 0.66 | 7.65 |
| 10-9 | 94.02 | 3.20 | 2.78 |

Tabel 5.3.1 Hasil Uji Coba 1

Dari hasil uji yang dilakukan, makin kecil nilai *threshold* yang digunakan, hasil dari *true positif* akan semakin besar. Begitu pula 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 di lampiran.

### Skenario Uji Coba 2

Sekenario uji coba 2 dilakukan dengan menghitung nilai *true positif, false positif,* dan *missing rate*. Pada sekenario uji 2 dilakukan uji coba variasi nilai C pada klasifikasi, dimana nilai C adalah nilai *penalty error term*. Nilai C yang diuji yaitu 1, 3.5, 5

|  |  |  |  |
| --- | --- | --- | --- |
| C | *True Positif* | *False Positif* | *Missing Rate* |
| 1 | 90.25 | 0.63 | 9.12 |
| 3.5 | 91.69 | 0.66 | 7.65 |
| 5 | 91.89 | 0.66 | 7.45 |

Tabel 5.3.2 Hasil Uji Coba 2

Hasil uji coba tahap 2 didapatkan nilai *false positif* yang sama pada nilai C 3.5 dengan 5. Namun, nilai C = 5 memiliki nilai *true positif* yang lebih tinggi dari C = 3.5.

## 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 C pada klasifikasi. Uji coba dilakukan dengan membandingkan nilai *true positif* , *false pisitif*, dan *missing rate*.

Dari uji coba yang dilakukan, parameter-parameter tersebut menghasilkan presentase terbaik ketika *threshold* yang digunakan 5 x 10-9 dan C yang digunakan sebesar 5.

***[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. Metode reduksi *size frame* mengurangi proses komputasi sehingga deteksi api bisa lebih cepat dilakukan.
2. Deteksi warna menyaring piksel-piksel yang tidak masuk kedalam *range* warna api dapat menyaring piksel-piksel yang tidak termasuk piksel api dengan baik.
3. Metode perhitungan luasan *region* api dapat menghilangkan *noise* atau objek-objek kecil yang mempengaruhi hasil dari klasifikasi.
4. Hasil terbaik pada uji coba adalah menggunakan nilai *threshold =* 5 x 10-9 dan nilai C = 5. Menghasilkan nilai *true positif* sebesar 91.89, *false positif* sebesar 0.66 dan *missing rate* sebesar 7.45

## Saran

Saran yang diberikan untuk pengembangan perangkat lunak ini adalah :

1. Proses deteksi api dan klasifikasi perlu dikembangkan, tidak melakukan proses verifikasi setiap piksel.

# DAFTAR ACUAN

|  |  |
| --- | --- |
| [1] | E. H. A. PETER J. BURT, "The Laplacian Pyramid as a Compact Image Code," *IEEE ,* Vols. COM-31, pp. 522-540, 1983. |
| [2] | I. S. T. Maria Isabel Ribeiro, "Gaussian Probability Density Functions: Properties and Error Characterization," 2004. |
| [3] | R. E. W. Refael C. Gonzalez, Digital Image Processing third edition, p. 785. |
| [4] | 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 6.2.1 Hasil Uji Coba Manggunakan C = 3.5 dan *Threshold*  = 10-7

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Nama File | True Positif | False Positif | Missing Rate |
| 1 | api\_cars.avi | 0.00 | 0.00 | 100.00 |
| 2 | api\_doll\_house2.avi | 79.70 | 0.00 | 20.30 |
| 3 | api\_dora.avi | 5.94 | 0.00 | 94.06 |
| 4 | api\_hutan\_besar1.avi | 15.46 | 0.00 | 84.54 |
| 5 | api\_kayu\_hutan\_besar1.avi | 56.70 | 0.00 | 43.30 |
| 6 | api\_kayu\_hutan1.avi | 86.60 | 0.00 | 13.40 |
| 7 | api\_kayu1.avi | 58.91 | 0.00 | 41.09 |
| 8 | api\_kayu2.avi | 66.34 | 0.00 | 33.66 |
| 9 | api\_kertas1.avi | 0.00 | 0.00 | 100.00 |
| 10 | api\_kertas2.avi | 2.50 | 0.00 | 97.50 |
| 11 | api\_ladang1.avi | 85.03 | 0.00 | 14.97 |
| 12 | api\_mainan1.avi | 100.00 | 0.00 | 0.00 |
| 13 | api\_mainan2.avi | 42.51 | 0.00 | 57.49 |
| 14 | api\_mobil1.avi | 90.59 | 0.00 | 9.41 |
| 15 | api\_mobil2.avi | 42.08 | 0.00 | 57.92 |
| 16 | api\_rc2.avi | 9.58 | 0.00 | 90.42 |
| 17 | api\_ruang\_tamu1.avi | 0.00 | 0.00 | 100.00 |
| 18 | api\_senter1.avi | 0.00 | 0.00 | 100.00 |
| 19 | api\_senter2.avi | 0.00 | 0.00 | 100.00 |
| 20 | api\_terjun.avi | 46.53 | 0.00 | 53.47 |
| 21 | api\_tol2.avi | 59.90 | 0.00 | 40.10 |
| 22 | api\_truck1.avi | 47.03 | 0.00 | 52.97 |
| 23 | api\_truck2.avi | 57.92 | 0.00 | 42.08 |
| 24 | api\_truck3.avi | 15.35 | 0.00 | 84.65 |
| 25 | non\_api\_anak\_kecil1.avi | 100.00 | 0.00 | 0.00 |
| 26 | non\_api\_anak\_kecil2.avi | 100.00 | 0.00 | 0.00 |
| 27 | non\_api\_anak\_kecil3.avi | 100.00 | 0.00 | 0.00 |
| 28 | non\_api\_bertemu.avi | 100.00 | 0.00 | 0.00 |
| 29 | non\_api\_hamdi2.avi | 81.01 | 18.99 | 0.00 |
| 30 | non\_api\_jaket\_merah.avi | 100.00 | 0.00 | 0.00 |
| 31 | non\_api\_jalan\_malam1.avi | 100.00 | 0.00 | 0.00 |
| 32 | non\_api\_jalan\_malam2.avi | 100.00 | 0.00 | 0.00 |
| 33 | non\_api\_jalan\_raya1.avi | 100.00 | 0.00 | 0.00 |
| 34 | non\_api\_jalan\_raya2.avi | 100.00 | 0.00 | 0.00 |
| 35 | non\_api\_jalan\_raya3.avi | 100.00 | 0.00 | 0.00 |
| 36 | non\_api\_ke\_mobil1.avi | 100.00 | 0.00 | 0.00 |
| 37 | non\_api\_ke\_mobil2.avi | 100.00 | 0.00 | 0.00 |
| 38 | non\_api\_ke\_mobil3.avi | 100.00 | 0.00 | 0.00 |
| 39 | non\_api\_kecelakaan1.avi | 100.00 | 0.00 | 0.00 |
| 40 | non\_api\_kecelakaan2.avi | 100.00 | 0.00 | 0.00 |
| 41 | non\_api\_kecelakaan3.avi | 100.00 | 0.00 | 0.00 |
| 42 | non\_api\_las\_vegas1.avi | 100.00 | 0.00 | 0.00 |
| 43 | non\_api\_parkiran1.avi | 100.00 | 0.00 | 0.00 |
| 44 | non\_api\_parkiran2.avi | 100.00 | 0.00 | 0.00 |
| 45 | non\_api\_parkiran3.avi | 100.00 | 0.00 | 0.00 |
| 46 | non\_api\_pencuri1.avi | 100.00 | 0.00 | 0.00 |
| 47 | non\_api\_pencuri3.avi | 100.00 | 0.00 | 0.00 |
| 48 | non\_api\_tauran1.avi | 100.00 | 0.00 | 0.00 |
| 49 | non\_api\_tauran2.avi | 100.00 | 0.00 | 0.00 |
| 50 | non\_api\_tauran3.avi | 100.00 | 0.00 | 0.00 |
| 51 | non\_api\_televisi1.avi | 100.00 | 0.00 | 0.00 |
| 52 | non\_api\_televisi2.avi | 100.00 | 0.00 | 0.00 |
| 53 | non\_api\_tembak3.avi | 100.00 | 0.00 | 0.00 |

Tabel 6.2.2 Hasil Uji Coba Manggunakan C = 3.5 dan *Threshold*  = 10-8

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Nama File | True Positif | False Positif | Missing Rate |
| 1 | api\_cars.avi | 83.17 | 0.00 | 16.83 |
| 2 | api\_doll\_house2.avi | 100.00 | 0.00 | 0.00 |
| 3 | api\_dora.avi | 57.43 | 0.00 | 42.57 |
| 4 | api\_hutan\_besar1.avi | 47.42 | 0.00 | 52.58 |
| 5 | api\_kayu\_hutan\_besar1.avi | 82.47 | 0.00 | 17.53 |
| 6 | api\_kayu1.avi | 88.61 | 0.00 | 11.39 |
| 7 | api\_kayu2.avi | 97.03 | 0.00 | 2.97 |
| 8 | api\_kertas1.avi | 61.25 | 0.00 | 38.75 |
| 9 | api\_kertas2.avi | 70.63 | 0.00 | 29.38 |
| 10 | api\_ladang1.avi | 100.00 | 0.00 | 0.00 |
| 11 | api\_mainan1.avi | 100.00 | 0.00 | 0.00 |
| 12 | api\_mainan2.avi | 100.00 | 0.00 | 0.00 |
| 13 | api\_mobil1.avi | 97.03 | 0.00 | 2.97 |
| 14 | api\_mobil2.avi | 86.63 | 0.00 | 13.37 |
| 15 | api\_mobil2.avi | 86.63 | 0.00 | 13.37 |
| 16 | api\_rc2.avi | 76.65 | 0.00 | 23.35 |
| 17 | api\_ruang\_tamu1.avi | 22.75 | 0.00 | 77.25 |
| 18 | api\_senter1.avi | 8.42 | 0.00 | 91.58 |
| 19 | api\_senter2.avi | 1.49 | 0.00 | 98.51 |
| 20 | api\_terjun.avi | 75.25 | 0.00 | 24.75 |
| 21 | api\_tol2.avi | 91.09 | 0.00 | 8.91 |
| 22 | api\_truck1.avi | 99.01 | 0.00 | 0.99 |
| 23 | api\_truck2.avi | 97.52 | 0.00 | 2.48 |
| 24 | api\_truck3.avi | 99.50 | 0.00 | 0.50 |
| 25 | non\_api\_anak\_kecil1.avi | 100.00 | 0.00 | 0.00 |
| 26 | non\_api\_anak\_kecil2.avi | 100.00 | 0.00 | 0.00 |
| 27 | non\_api\_anak\_kecil3.avi | 100.00 | 0.00 | 0.00 |
| 28 | non\_api\_bertemu.avi | 100.00 | 0.00 | 0.00 |
| 29 | non\_api\_hamdi2.avi | 76.54 | 23.46 | 0.00 |
| 30 | non\_api\_jaket\_merah.avi | 100.00 | 0.00 | 0.00 |
| 31 | non\_api\_jalan\_malam1.avi | 100.00 | 0.00 | 0.00 |
| 32 | non\_api\_jalan\_malam2.avi | 100.00 | 0.00 | 0.00 |
| 33 | non\_api\_jalan\_raya1.avi | 100.00 | 0.00 | 0.00 |
| 34 | non\_api\_jalan\_raya2.avi | 100.00 | 0.00 | 0.00 |
| 35 | non\_api\_jalan\_raya3.avi | 100.00 | 0.00 | 0.00 |
| 36 | non\_api\_ke\_mobil1.avi | 100.00 | 0.00 | 0.00 |
| 37 | non\_api\_ke\_mobil2.avi | 100.00 | 0.00 | 0.00 |
| 38 | non\_api\_ke\_mobil3.avi | 100.00 | 0.00 | 0.00 |
| 39 | non\_api\_kecelakaan1.avi | 100.00 | 0.00 | 0.00 |
| 40 | non\_api\_kecelakaan2.avi | 100.00 | 0.00 | 0.00 |
| 41 | non\_api\_kecelakaan3.avi | 100.00 | 0.00 | 0.00 |
| 42 | non\_api\_las\_vegas1.avi | 100.00 | 0.00 | 0.00 |
| 43 | non\_api\_parkiran1.avi | 100.00 | 0.00 | 0.00 |
| 44 | non\_api\_parkiran2.avi | 100.00 | 0.00 | 0.00 |
| 45 | non\_api\_parkiran3.avi | 100.00 | 0.00 | 0.00 |
| 46 | non\_api\_pencuri1.avi | 100.00 | 0.00 | 0.00 |
| 47 | non\_api\_pencuri3.avi | 100.00 | 0.00 | 0.00 |
| 48 | non\_api\_tauran1.avi | 100.00 | 0.00 | 0.00 |
| 49 | non\_api\_tauran2.avi | 100.00 | 0.00 | 0.00 |
| 50 | non\_api\_tauran3.avi | 100.00 | 0.00 | 0.00 |
| 51 | non\_api\_televisi1.avi | 100.00 | 0.00 | 0.00 |
| 52 | non\_api\_televisi2.avi | 100.00 | 0.00 | 0.00 |
| 53 | non\_api\_tembak3.avi | 100.00 | 0.00 | 0.00 |

Tabel 6.2.3 Hasil Uji Coba Manggunakan C = 3.5 dan *Threshold*  = 5x10-9

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Nama File | True Positif | False Positif | Missing Rate |
| 1 | api\_cars.avi | 95.05 | 0.00 | 4.95 |
| 2 | api\_doll\_house2.avi | 100.00 | 0.00 | 0.00 |
| 3 | api\_dora.avi | 79.21 | 0.00 | 20.79 |
| 4 | api\_hutan\_besar1.avi | 55.67 | 0.00 | 44.33 |
| 5 | api\_kayu\_hutan\_besar1.avi | 84.54 | 0.00 | 15.46 |
| 6 | api\_kayu\_hutan1.avi | 100.00 | 0.00 | 0.00 |
| 7 | api\_kayu1.avi | 88.61 | 0.00 | 11.39 |
| 8 | api\_kayu2.avi | 97.03 | 0.00 | 2.97 |
| 9 | api\_kertas1.avi | 67.50 | 0.00 | 32.50 |
| 10 | api\_kertas2.avi | 97.50 | 0.00 | 2.50 |
| 11 | api\_ladang1.avi | 100.00 | 0.00 | 0.00 |
| 12 | api\_mainan1.avi | 100.00 | 0.00 | 0.00 |
| 13 | api\_mainan2.avi | 100.00 | 0.00 | 0.00 |
| 14 | api\_mobil1.avi | 97.03 | 0.00 | 2.97 |
| 15 | api\_mobil2.avi | 90.10 | 0.00 | 9.90 |
| 16 | api\_rc2.avi | 89.22 | 0.00 | 10.78 |
| 17 | api\_ruang\_tamu1.avi | 49.10 | 0.00 | 50.90 |
| 18 | api\_senter1.avi | 26.24 | 0.00 | 73.76 |
| 19 | api\_senter2.avi | 6.93 | 0.00 | 93.07 |
| 20 | api\_terjun.avi | 80.20 | 0.00 | 19.80 |
| 21 | api\_tol2.avi | 91.58 | 0.00 | 8.42 |
| 22 | api\_truck1.avi | 100.00 | 0.00 | 0.00 |
| 23 | api\_truck2.avi | 99.01 | 0.00 | 0.99 |
| 24 | api\_truck3.avi | 100.00 | 0.00 | 0.00 |
| 25 | non\_api\_anak\_kecil1.avi | 100.00 | 0.00 | 0.00 |
| 26 | non\_api\_anak\_kecil2.avi | 100.00 | 0.00 | 0.00 |
| 27 | non\_api\_anak\_kecil3.avi | 100.00 | 0.00 | 0.00 |
| 28 | non\_api\_bertemu.avi | 100.00 | 0.00 | 0.00 |
| 29 | non\_api\_hamdi2.avi | 72.63 | 27.37 | 0.00 |
| 30 | non\_api\_jaket\_merah.avi | 100.00 | 0.00 | 0.00 |
| 31 | non\_api\_jalan\_malam1.avi | 100.00 | 0.00 | 0.00 |
| 32 | non\_api\_jalan\_malam2.avi | 100.00 | 0.00 | 0.00 |
| 33 | non\_api\_jalan\_raya1.avi | 100.00 | 0.00 | 0.00 |
| 34 | non\_api\_jalan\_raya2.avi | 100.00 | 0.00 | 0.00 |
| 35 | non\_api\_jalan\_raya3.avi | 100.00 | 0.00 | 0.00 |
| 36 | non\_api\_ke\_mobil1.avi | 100.00 | 0.00 | 0.00 |
| 37 | non\_api\_ke\_mobil2.avi | 100.00 | 0.00 | 0.00 |
| 38 | non\_api\_ke\_mobil3.avi | 100.00 | 0.00 | 0.00 |
| 39 | non\_api\_kecelakaan1.avi | 100.00 | 0.00 | 0.00 |
| 40 | non\_api\_kecelakaan2.avi | 100.00 | 0.00 | 0.00 |
| 41 | non\_api\_kecelakaan3.avi | 100.00 | 0.00 | 0.00 |
| 42 | non\_api\_las\_vegas1.avi | 100.00 | 0.00 | 0.00 |
| 43 | non\_api\_parkiran1.avi | 100.00 | 0.00 | 0.00 |
| 44 | non\_api\_parkiran2.avi | 100.00 | 0.00 | 0.00 |
| 45 | non\_api\_parkiran3.avi | 100.00 | 0.00 | 0.00 |
| 46 | non\_api\_pencuri1.avi | 92.22 | 7.78 | 0.00 |
| 47 | non\_api\_pencuri3.avi | 100.00 | 0.00 | 0.00 |
| 48 | non\_api\_tauran1.avi | 100.00 | 0.00 | 0.00 |
| 49 | non\_api\_tauran2.avi | 100.00 | 0.00 | 0.00 |
| 50 | non\_api\_tauran3.avi | 100.00 | 0.00 | 0.00 |
| 51 | non\_api\_televisi1.avi | 100.00 | 0.00 | 0.00 |
| 52 | non\_api\_televisi2.avi | 100.00 | 0.00 | 0.00 |
| 53 | non\_api\_tembak3.avi | 100.00 | 0.00 | 0.00 |

Tabel 6.2.4 Hasil Uji Coba Manggunakan C = 1 dan *Threshold*  = 5x10-9

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Nama File | True Positif | False Positif | Missing Rate |
| 1 | api\_cars.avi | 92.57 | 0.00 | 7.43 |
| 2 | api\_doll\_house2.avi | 100.00 | 0.00 | 0.00 |
| 3 | api\_dora.avi | 78.71 | 0.00 | 21.29 |
| 4 | api\_hutan\_besar1.avi | 24.74 | 0.00 | 75.26 |
| 5 | api\_kayu\_hutan\_besar1.avi | 80.41 | 0.00 | 19.59 |
| 6 | api\_kayu\_hutan1.avi | 93.81 | 0.00 | 6.19 |
| 7 | api\_kayu1.avi | 84.16 | 0.00 | 15.84 |
| 8 | api\_kayu2.avi | 92.57 | 0.00 | 7.43 |
| 9 | api\_kertas1.avi | 67.50 | 0.00 | 32.50 |
| 10 | api\_kertas2.avi | 96.88 | 0.00 | 3.13 |
| 11 | api\_ladang1.avi | 100.00 | 0.00 | 0.00 |
| 12 | api\_mainan1.avi | 100.00 | 0.00 | 0.00 |
| 13 | api\_mainan2.avi | 100.00 | 0.00 | 0.00 |
| 14 | api\_mobil1.avi | 97.03 | 0.00 | 2.97 |
| 15 | api\_mobil2.avi | 78.22 | 0.00 | 21.78 |
| 16 | api\_rc2.avi | 89.22 | 0.00 | 10.78 |
| 17 | api\_ruang\_tamu1.avi | 40.72 | 0.00 | 59.28 |
| 18 | api\_senter1.avi | 23.76 | 0.00 | 76.24 |
| 19 | api\_senter2.avi | 6.93 | 0.00 | 93.07 |
| 20 | api\_terjun.avi | 80.20 | 0.00 | 19.80 |
| 21 | api\_tol2.avi | 91.58 | 0.00 | 8.42 |
| 22 | api\_truck1.avi | 100.00 | 0.00 | 0.00 |
| 23 | api\_truck2.avi | 98.51 | 0.00 | 1.49 |
| 24 | api\_truck3.avi | 99.01 | 0.00 | 0.99 |
| 25 | non\_api\_anak\_kecil1.avi | 100.00 | 0.00 | 0.00 |
| 26 | non\_api\_anak\_kecil2.avi | 100.00 | 0.00 | 0.00 |
| 27 | non\_api\_anak\_kecil3.avi | 100.00 | 0.00 | 0.00 |
| 28 | non\_api\_bertemu.avi | 100.00 | 0.00 | 0.00 |
| 29 | non\_api\_hamdi2.avi | 74.30 | 25.70 | 0.00 |
| 30 | non\_api\_jaket\_merah.avi | 100.00 | 0.00 | 0.00 |
| 31 | non\_api\_jalan\_malam1.avi | 100.00 | 0.00 | 0.00 |
| 32 | non\_api\_jalan\_malam2.avi | 100.00 | 0.00 | 0.00 |
| 33 | non\_api\_jalan\_raya1.avi | 100.00 | 0.00 | 0.00 |
| 34 | non\_api\_jalan\_raya2.avi | 100.00 | 0.00 | 0.00 |
| 35 | non\_api\_jalan\_raya3.avi | 100.00 | 0.00 | 0.00 |
| 36 | non\_api\_ke\_mobil1.avi | 100.00 | 0.00 | 0.00 |
| 37 | non\_api\_ke\_mobil2.avi | 100.00 | 0.00 | 0.00 |
| 38 | non\_api\_ke\_mobil3.avi | 100.00 | 0.00 | 0.00 |
| 39 | non\_api\_kecelakaan1.avi | 100.00 | 0.00 | 0.00 |
| 40 | non\_api\_kecelakaan2.avi | 100.00 | 0.00 | 0.00 |
| 41 | non\_api\_kecelakaan3.avi | 100.00 | 0.00 | 0.00 |
| 42 | non\_api\_las\_vegas1.avi | 100.00 | 0.00 | 0.00 |
| 43 | non\_api\_parkiran1.avi | 100.00 | 0.00 | 0.00 |
| 44 | non\_api\_parkiran2.avi | 100.00 | 0.00 | 0.00 |
| 45 | non\_api\_parkiran3.avi | 100.00 | 0.00 | 0.00 |
| 46 | non\_api\_pencuri1.avi | 100.00 | 0.00 | 0.00 |
| 47 | non\_api\_pencuri3.avi | 92.22 | 7.78 | 0.00 |
| 48 | non\_api\_tauran1.avi | 100.00 | 0.00 | 0.00 |
| 49 | non\_api\_tauran2.avi | 100.00 | 0.00 | 0.00 |
| 50 | non\_api\_tauran3.avi | 100.00 | 0.00 | 0.00 |
| 51 | non\_api\_televisi1.avi | 100.00 | 0.00 | 0.00 |
| 52 | non\_api\_televisi2.avi | 100.00 | 0.00 | 0.00 |
| 53 | non\_api\_tembak3.avi | 100.00 | 0.00 | 0.00 |

Tabel 6.2.5 Hasil Uji Coba Manggunakan C = 5 dan *Threshold*  = 5x10-9

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Nama File | True Positif | False Positif | Missing Rate |
| 1 | api\_cars.avi | 95.05 | 0.00 | 4.95 |
| 2 | api\_doll\_house2.avi | 100.00 | 0.00 | 0.00 |
| 3 | api\_dora.avi | 79.21 | 0.00 | 20.79 |
| 4 | api\_hutan\_besar1.avi | 59.79 | 0.00 | 40.21 |
| 5 | api\_kayu\_hutan\_besar1.avi | 85.57 | 0.00 | 14.43 |
| 6 | api\_kayu\_hutan1.avi | 100.00 | 0.00 | 0.00 |
| 7 | api\_kayu1.avi | 89.11 | 0.00 | 10.89 |
| 8 | api\_kayu2.avi | 97.52 | 0.00 | 2.48 |
| 9 | api\_kertas1.avi | 67.50 | 0.00 | 32.50 |
| 10 | api\_kertas2.avi | 97.50 | 0.00 | 2.50 |
| 11 | api\_ladang1.avi | 100.00 | 0.00 | 0.00 |
| 12 | api\_mainan1.avi | 100.00 | 0.00 | 0.00 |
| 13 | api\_mainan2.avi | 100.00 | 0.00 | 0.00 |
| 14 | api\_mobil1.avi | 97.03 | 0.00 | 2.97 |
| 15 | api\_mobil2.avi | 90.59 | 0.00 | 9.41 |
| 16 | api\_rc2.avi | 89.22 | 0.00 | 10.78 |
| 17 | api\_ruang\_tamu1.avi | 53.29 | 0.00 | 46.71 |
| 18 | api\_senter1.avi | 26.24 | 0.00 | 73.76 |
| 19 | api\_senter2.avi | 6.93 | 0.00 | 93.07 |
| 20 | api\_terjun.avi | 80.20 | 0.00 | 19.80 |
| 21 | api\_tol2.avi | 91.58 | 0.00 | 8.42 |
| 22 | api\_truck1.avi | 100.00 | 0.00 | 0.00 |
| 23 | api\_truck2.avi | 99.01 | 0.00 | 0.99 |
| 24 | api\_truck3.avi | 100.00 | 0.00 | 0.00 |
| 25 | non\_api\_anak\_kecil1.avi | 100.00 | 0.00 | 0.00 |
| 26 | non\_api\_anak\_kecil2.avi | 100.00 | 0.00 | 0.00 |
| 27 | non\_api\_anak\_kecil3.avi | 100.00 | 0.00 | 0.00 |
| 28 | non\_api\_bertemu.avi | 100.00 | 0.00 | 0.00 |
| 29 | non\_api\_hamdi2.avi | 72.63 | 27.37 | 0.00 |
| 30 | non\_api\_jaket\_merah.avi | 100.00 | 0.00 | 0.00 |
| 31 | non\_api\_jalan\_malam1.avi | 100.00 | 0.00 | 0.00 |
| 32 | non\_api\_jalan\_malam2.avi | 100.00 | 0.00 | 0.00 |
| 33 | non\_api\_jalan\_raya1.avi | 100.00 | 0.00 | 0.00 |
| 34 | non\_api\_jalan\_raya2.avi | 100.00 | 0.00 | 0.00 |
| 35 | non\_api\_jalan\_raya3.avi | 100.00 | 0.00 | 0.00 |
| 36 | non\_api\_ke\_mobil1.avi | 100.00 | 0.00 | 0.00 |
| 37 | non\_api\_ke\_mobil2.avi | 100.00 | 0.00 | 0.00 |
| 38 | non\_api\_ke\_mobil3.avi | 100.00 | 0.00 | 0.00 |
| 39 | non\_api\_kecelakaan1.avi | 100.00 | 0.00 | 0.00 |
| 40 | non\_api\_kecelakaan2.avi | 100.00 | 0.00 | 0.00 |
| 41 | non\_api\_kecelakaan3.avi | 100.00 | 0.00 | 0.00 |
| 42 | non\_api\_las\_vegas1.avi | 100.00 | 0.00 | 0.00 |
| 43 | non\_api\_parkiran1.avi | 100.00 | 0.00 | 0.00 |
| 44 | non\_api\_parkiran2.avi | 100.00 | 0.00 | 0.00 |
| 45 | non\_api\_parkiran3.avi | 100.00 | 0.00 | 0.00 |
| 46 | non\_api\_pencuri1.avi | 92.22 | 7.78 | 0.00 |
| 47 | non\_api\_pencuri3.avi | 100.00 | 0.00 | 0.00 |
| 48 | non\_api\_tauran1.avi | 100.00 | 0.00 | 0.00 |
| 49 | non\_api\_tauran2.avi | 100.00 | 0.00 | 0.00 |
| 50 | non\_api\_tauran3.avi | 100.00 | 0.00 | 0.00 |
| 51 | non\_api\_televisi1.avi | 100.00 | 0.00 | 0.00 |
| 52 | non\_api\_televisi2.avi | 100.00 | 0.00 | 0.00 |
| 53 | non\_api\_tembak3.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.