**(2.Ado.Ok**

**TUGAS AKHIR – KI141502**

**APLIKASI PENGENALAN AKSARA JAWA MELALUI TULISAN TANGAN MENGGUNAKAN METODE K-NEAREST NEIGHBOR BERBASIS OPENCV ANDROID**

HASHFI ALFIAN CIYUDA

NRP 5111 100 104

Dosen Pembimbing

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

Wijayanti Nurul Khotimah, S.Kom., M.Sc.

JURUSAN TEKNIK INFORMATIKA

Fakultas Teknologi Informasi

Institut Teknologi Sepuluh Nopember

Surabaya 2015



**TUGAS AKHIR – KI141502**

**APLIKASI PENGENALAN AKSARA JAWA MELALUI TULISAN TANGAN MENGGUNAKAN METODE K-NEAREST NEIGHBOR BERBASIS OPENCV ANDROID**

HASHFI ALFIAN CIYUDA

NRP 5111 100 104

Dosen Pembimbing

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

Wijayanti Nurul Khotimah, S.Kom., M.Sc.

JURUSAN TEKNIK INFORMATIKA

Fakultas Teknologi Informasi

Institut Teknologi Sepuluh Nopember

Surabaya 2015

[Halaman ini sengaja dikosongkan]



**FINAL PROJECT – KI141502**

**APPLICATION OF HANDWRITING JAVANESE CHARACTER RECOGNITION USING K-NEAREST NEIGHBOR BASED ON OPENCV ANDROID**

HASHFI ALFIAN CIYUDA

NRP 5111 100 088

Advisor

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

Wijayanti Nurul Khotimah, S.Kom., M.Sc.

INFORMATICS DEPARTMENT

Faculty of Information Technology

Institut Teknologi Sepuluh Nopember

Surabaya 2015

[Halaman ini sengaja dikosongkan]

**APLIKASI PENGENALAN AKSARA JAWA MELALUI TULISAN TANGAN MENGGUNAKAN METODE K-NEAREST NEIGHBOR BERBASIS OPENCV ANDROID**

**TUGAS AKHIR**Diajukan Guna Memenuhi Salah Satu Syarat   
Memperoleh Gelar Sarjana Komputer  
pada   
Rumpun Mata Kuliah Komputasi Cerdas dan Visi  
Program Studi S-1 Jurusan Teknik Informatika  
Fakultas Teknologi Informasi  
Institut Teknologi Sepuluh Nopember

Oleh :  
**HASHFI ALFIAN CIYUDA**NRP : 5111 100 104

Disetujui oleh Dosen Pembimbing Tugas Akhir :

|  |  |
| --- | --- |
| Dr. Eng. Chastine Fatichah, S.Kom., M.Kom.  NIP: 197512202001122002 | ........................... (pembimbing 1) |
| Wijayanti Nurul Khotimah, S.Kom., M.Sc.  NIP: 198603122012122004 | ........................... (pembimbing 2) |

**Surabaya  
juni 2015**

[Halaman ini sengaja dikosongkan]

**APLIKASI PENGENALAN AKSARA JAWA MELALUI TULISAN TANGAN MENGGUNAKAN METODE K-NEAREST NEIGHBOR BERBASIS OPENCV ANDROID**

Nama Mahasiswa : HASHFI ALFIAN CIYUDA

NRP : 5111 100 104

Jurusan : Teknik Informatika ITS

Dosen Pembimbing I : Dr. Eng. Chastine Fatichah, S.Kom., M.Kom.

Dosen Pembimbing II : Wijayanti Nurul Khotimah, S.Kom., M.Sc.

**Abstrak**

Optical Character Recognition (OCR) adalah sebuah teknologi pengolahan citra digital yang menghasilkan informasi sebuah pola pada suatu objek. Sebuah informasi pola yang masih belum banyak diketahui adalah tulisan tangan aksara Jawa yang dapat ditemui dalam naskah buku Jawa kuno dimana naskah tersebut memiliki kontribusi berharga untuk pengetahuan.

Tugas akhir ini mengimplementasikan sistem pengenalan tulisan tangan aksara Jawa. Sistem ini diterapkan pada Android yang sudah menggunakan konsep layar sentuh sehingga pengguna dapat menuliskan karakter aksara Jawa kapanpun dimanapun. Terdapat empat tahap dalam sistem ini. Tahap pertama adalah deteksi dan segmentasi tulisan aksara Jawa. Tahap kedua adalah penipisan tulisan aksara Jawa. Tahap ketiga adalah ekstraksi fitur tulisan aksara Jawa, dan tahap terakhir adalah klasifikasi karakter aksara Jawa.

Uji coba pada tugas akhir ini menggunakan citra-citra tulisan tangan aksara Jawa sejumlah 240 data pelatihan untuk mengetahui seberapa akurat metode yang diterapkan. Uji coba ini menggunakan parameter asumsi-asumsi penggunaan fitur dan nilai K pada algoritma klasifikasi K-Nearest Neighbor. Hasil uji coba sistem menunjukkan nilai akurasi sebesar 90.42% berdasarkan parameter-parameter yang digunakan.

***Kata kunci: pengenalan tulisan tangan, aksara Jawa, ekstraksi fitur, klasifikasi K-Nearest Neighbor.***

[Halaman ini sengaja dikosongkan]

**IMPLEMENTATION OF EYEPHONE SYSTEM FOR MENU ACTIVATION ON SMARTPHONE APPLICATION USING OPENCV ANDROID**

Name : HASHFI ALFIAN CIYUDA

NRP : 5111 100 104

Major : Informatics Department – ITS

Supervisor I : Dr. Eng. Chastine Fatichah, S.Kom., M.Kom.

Supervisor II : Wijayanti Nurul Khotimah, S.Kom., M.Sc.

**Abstract**

Optical Character Recognition (OCR) is an image processing technology which produce information of an object’s pattern. A pattern information which is still not widely known is handwriting Javanese character that can be found on manuscript of ancient Java’s book which those manuscript have valueble contribution to knowledge.

This final project implement handwriting Javanese character recognition system. This sistem is implemented on Android device which already use touch screen’s concept so user can write Javanese character anytime and anywhere. There are four steps in this system. First step is handwriting Javanese character detection and segmentation. Second step is handwriting Javanese character thinning. Third step is handwriting Javanese character feature extraction and the last step is the classification of Javanese character.

The testing uses 240 image of handwriting Javanese character on training data to find out how accurate the method used. Testing was done with assumtion the use of features and K value on K-Nearest Neighbor classification algorithm as parameters. The result of this system shows that accuracy value is 90.42% based on the parameters.

**Keywords: handwriting recognition, Javanese character, feature extraction, K-Nearest Neighbor classification.**

[Halaman ini sengaja dikosongkan]

**KATA PENGANTAR**



Segala puji dan syukur ke hadirat Allah SWT yang telah memberikan rahmat dan hidayah-Nya sehingga penulis dapat menyelesaikan tugas akhir yang berjudul **“*Aplikasi Pengenalan Aksara Jawa Melalui Tulisan Tangan Menggunakan Metode K-Nearest Neighbor Berbasis Opencv Android*”**.

Dalam pelaksanaan tugas akhir ini tentu penulis sebagai makhluk sosial tidak dapat menyelesaikannya tanpa bantuan dari pihak lain. Tanpa mengurangi rasa hormat, penulis memberikan penghargaan serta ucapan terima kasih yang sebesar-besarnya kepada:

1. Orang tua serta keluarga besar yang senantiasa memberikan semangat dan doa agar penulis dapat menyelesaikan tugas akhir dengan tepat waktu.
2. Ibu Dr. Eng. Chastine Fatichah, S.Kom., M.Kom. selaku dosen pembimbing tugas akhir pertama yang telah membimbing, memotivasi dan memberikan banyak masukan dalam pengerjaan tugas akhir ini.
3. Ibu Wijayanti Nurul Khotimah, S.Kom., M.Sc. selaku dosen pembimbing tugas akhir kedua yang selalu memberikan koreksi serta masukan-masukan yang dapat dikembangkan dari tugas akhir ini.
4. Bapak dan Ibu dosen Jurusan Teknik Informatika ITS yang telah mengajarkan banyak ilmu berharga kepada penulis.
5. Bapak dan Ibu karyawan Jurusan Teknik Informatika ITS atas berbagai bantuan yang telah diberikan kepada penulis selama masa perkuliahan.
6. Teman-teman mahasiswa bidang minatKomputasi Cerdas dan Visi yang telah menemani perjuangan mencari ilmu selama mengambil mata kuliah RMK KCV.
7. Teman-teman administrator Laboratorium KCV yang selalu setia menemani hari-hari saya dalam mengerjakan tugas akhir.
8. Teman-teman HMTC dan KMI yang telah mengisi hari-hari penulis selain wajib berkuliah namun tetap aktif di organisasi kemahasiswaan.
9. Teman-teman Teknik Informatika ITS angkatan 2011, yang telah memberikan warna-warni kehidupan mahasiswa mulai sejak mahasiswa baru hingga lulus.
10. Sahabat-sahabat yang selalu menemani saat suka dan duka dan membantu dalam pengerjaan tugas akhir, Ahmad Hayam, Muhammad Bagus Andra, Ardian Atminanto, dkk.
11. Teman-teman futsal *dadakan* yang selalu memberikan hiburan di saat penulis penat dengan perkuliahan dan pengerjaan tugas akhir.
12. Pihak-pihak lain yang tidak sempat penulis sebutkan, yang telah membantu kelancaran pengerjaan tugas akhir ini.

Penulis sangat berharap bahwa apa yang dihasilkan dari tugas akhir ini bisa memberikan manfaat bagi semua pihak, khususnya bagi diri penulis sendiri dan seluruh *civitas academica* Teknik Informatika ITS, serta bagi agama, bangsa, dan negara. Tak ada manusia yang sempurna sekalipun penulis berusaha sebaik mungkin dalam menyelesaikan tugas akhir ini. Karena itu, penulis memohon maaf apabila terdapat kesalahan, kekurangan, maupun kelalaian yang telah penulis lakukan. Kritik dan saran yang membangun sangat diharapkan oleh penulis untuk dapat disampaikan untuk perbaikan selanjutnya.

Surabaya, Juni 2015

Hashfi Alfian Ciyuda

# DAFTAR ISI

[Abstrak viii](#_Toc421872281)

[Abstract x](#_Toc421872282)

[KATA PENGANTAR xii](#_Toc421872283)

[DAFTAR ISI xiv](#_Toc421872284)

[DAFTAR GAMBAR xviii](#_Toc421872285)

[DAFTAR TABEL xxiii](#_Toc421872286)

[BAB I PENDAHULUAN 1](#_Toc421872287)

[1.1. Latar Belakang 1](#_Toc421872288)

[1.2. Rumusan Masalah 3](#_Toc421872289)

[1.3. Batasan Masalah 3](#_Toc421872290)

[1.4. Tujuan 3](#_Toc421872291)

[1.5. Metodologi 4](#_Toc421872292)

[1.6. Sistematika Penulisan 5](#_Toc421872293)

[BAB II TINJAUAN PUSTAKA 7](#_Toc421872294)

[2.1. Ekstraksi Fitur 7](#_Toc421872297)

[2.2. Aksara Jawa 7](#_Toc421872298)

[2.3. Klasifikasi K-Nearest Neighbor 8](#_Toc421872299)

[2.4. OpenCV Android 10](#_Toc421872300)

[2.5. Region Growing 10](#_Toc421872301)

[2.6. Algoritma *Thinning* Guo-Hall 11](#_Toc421872302)

[BAB III ANALISIS DAN PERANCANGAN 15](#_Toc421872303)

[3.1. Analisis Implementasi Metode Secara Umum 15](#_Toc421872305)

[3.2. Perancangan Data 17](#_Toc421872306)

[3.2.1. Data Masukan 18](#_Toc421872307)

[3.2.2. Data Proses 18](#_Toc421872308)

[3.2.3. Data Keluaran 20](#_Toc421872309)

[3.3. Perancangan Proses 20](#_Toc421872310)

[3.3.1. Tahap Deteksi dan Segmentasi Tulisan Aksara Jawa 20](#_Toc421872311)

[3.3.2. Tahap Penipisan (Thinning) Tulisan Aksara Jawa 24](#_Toc421872312)

[3.3.3. Tahap Ekstraksi Fitur Tulisan Aksara Jawa 26](#_Toc421872313)

[3.3.4. Tahap Klasifikasi Karakter Aksara Jawa 36](#_Toc421872314)

[3.4. Perancangan Antarmuka Perangkat Lunak 40](#_Toc421872315)

[3.4.1. Halaman Menu 40](#_Toc421872316)

[3.4.2. Halaman Tulis Aksara 40](#_Toc421872317)

[3.4.3. Halaman Foto Aksara 43](#_Toc421872318)

[3.4.4. Halaman Info Aplikasi 44](#_Toc421872319)

[3.4.5. Halaman Klasifikasi 44](#_Toc421872320)

[BAB IV IMPLEMENTASI 49](#_Toc421872321)

[4.1. Lingkungan Implementasi 49](#_Toc421872323)

[4.2. Implementasi Proses 49](#_Toc421872324)

[4.2.1. Implementasi Tahap Deteksi dan Segmentasi Tulisan Aksara Jawa 49](#_Toc421872325)

[4.2.2. Implementasi Tahap Penipisan (*Thinning*) Tulisan Aksara Jawa 54](#_Toc421872326)

[4.2.3. Implementasi Tahap Ekstraksi Fitur Karakter Aksara Jawa 60](#_Toc421872327)

[4.2.4. Implementasi Tahap Klasifikasi Karakter Aksara Jawa 70](#_Toc421872328)

[4.3. Implementasi Antarmuka Pengguna 71](#_Toc421872329)

[4.3.1. Halaman Menu 71](#_Toc421872330)

[4.3.2. Halaman Tulis Aksara 72](#_Toc421872331)

[4.3.3. Halaman Foto Aksara 72](#_Toc421872332)

[4.3.4. Halaman Info Aplikasi 72](#_Toc421872333)

[4.3.5. Halaman Klasifikasi 73](#_Toc421872334)

[BAB V UJI COBA DAN EVALUASI 75](#_Toc421872335)

[5.1. Lingkungan Uji Coba 75](#_Toc421872341)

[5.2. Data Uji Coba 75](#_Toc421872342)

[5.3. Skenario Uji Coba 76](#_Toc421872343)

[5.4. Skenario Pengujian 1: Perbandingan Kombinasi Pemakaian Kondisi Penentuan Fitur 76](#_Toc421872344)

[5.5. Skenario Pengujian 2: Perbandingan Hasil Akurasi Berdasarkan Variasi Nilai K untuk Kombinasi Asumsi Penentuan Fitur dengan Hasil Optimal 84](#_Toc421872345)

[5.6. Skenario Pengujian 3: Uji Coba Responden yang Berbeda untuk Pengenalan Aksara Jawa Melalui Tulisan Tangan pada Layar *Smartphone* 85](#_Toc421872346)

[5.7. Skenario Pengujian 4: Uji Coba Responden yang Berbeda untuk Pengenalan Aksara Jawa Melalui Citra Karakter Aksara Jawa yang Diambil Melalui Kamera Utama 87](#_Toc421872347)

[5.8. Evaluasi 89](#_Toc421872348)

[BAB VI KESIMPULAN DAN SARAN 91](#_Toc421872349)

[6.1. Kesimpulan 91](#_Toc421872351)

[6.2. Saran 92](#_Toc421872352)

[DAFTAR PUSTAKA 93](#_Toc421872353)

[LAMPIRAN 95](#_Toc421872354)

[BIODATA PENULIS 100](#_Toc421872355)

[Halaman ini sengaja dikosongkan]

# DAFTAR GAMBAR

[Gambar 2. 1 Struktur Java dan *native* (C++) pada sistem operasi Android **Error! Bookmark not defined.**](#_Toc409624928)

[Gambar 2. 2 Mekanisme penerjemahan Java API menjadi *native* menggunakan JNI **Error! Bookmark not defined.**](#_Toc409624929)

[Gambar 2. 3 Diagram aktivitas OpenCV Manager dan pustaka OpenCV pada aplikasi Android **Error! Bookmark not defined.**](#_Toc409624930)

[Gambar 2. 4 Beberapa contoh fitur Haar-like **Error! Bookmark not defined.**](#_Toc409624931)

[Gambar 2. 5 Implementasi fitur Haar-like pada proses deteksi wajah **Error! Bookmark not defined.**](#_Toc409624932)

[Gambar 2. 6 *Template Matching* antara citra *template* mata dengan citra wajah **Error! Bookmark not defined.**](#_Toc409624933)

[Gambar 2. 7 Konvolusi pada metode *Template Matching* **Error! Bookmark not defined.**](#_Toc409624934)

[Gambar 2. 8 Contoh penggunaan faktor skala pada dilasi bangun geometri **Error! Bookmark not defined.**](#_Toc409624935)

[Gambar 2. 9 Proporsi wajah berdasarkan tinggi wajah **Error! Bookmark not defined.**](#_Toc409624936)

[Gambar 2. 10 Proporsi wajah berdasarkan lebar wajah **Error! Bookmark not defined.**](#_Toc409624937)

[Gambar 2. 11 *Interpupillary Distance* sebagai jarak antara kedua pupil mata **Error! Bookmark not defined.**](#_Toc409624938)

[Gambar 3. 1 Diagram alir implementasi EyePhone 16](#_Toc409624901)

[Gambar 3. 2 Citra hasil kamera depan *smartphone* dengan wajah pengguna sebagai data masukan 18](#_Toc409624902)

[Gambar 3. 3 Data keluaran berupa deteksi kedipan mata **Error! Bookmark not defined.**](#_Toc409624903)

[Gambar 3. 4 Data keluaran aktivasi menu *smartphone* dengan pelacakan dan deteksi kedipan mata **Error! Bookmark not defined.**](#_Toc409624904)

[Gambar 3. 5 Diagram alir tahap deteksi wajah dan ROI mata 21](#_Toc409624905)

[Gambar 3. 6 Digram alir proses klasifikasi *Haar Cascade Multi Scaling* **Error! Bookmark not defined.**](#_Toc409624906)

[Gambar 3. 7 Proporsi anatomi mata pada wajah **Error! Bookmark not defined.**](#_Toc409624907)

[Gambar 3. 8 ROI mata kanan dan kiri hasil *cropping* **Error! Bookmark not defined.**](#_Toc409624908)

[Gambar 3. 9 Diagram alir tahap kalibrasi *template* mata 24](#_Toc409624909)

[Gambar 3. 10 Contoh *open eye template* (warna hijau) dan *template* iris (warna putih) 26](#_Toc409624910)

[Gambar 3. 11 Diagram alir tahap pelacakan mata **Error! Bookmark not defined.**](#_Toc409624911)

[Gambar 3. 12 Contoh *template matching* dengan menggunakan *template* iris mata **Error! Bookmark not defined.**](#_Toc409624912)

[Gambar 3. 13 Diagram alir proses *template matching* **Error! Bookmark not defined.**](#_Toc409624913)

[Gambar 3. 14 *Pseudocode* mencari titik tengah *interpupillary* **Error! Bookmark not defined.**](#_Toc409624914)

[Gambar 3. 15 Peletakan *window* (kotak berwarna merah) untuk pelacakan mata **Error! Bookmark not defined.**](#_Toc409624915)

[Gambar 3. 16 Contoh transformasi titik tengah *interpupillary* dari *window* ke *screen smartphone* **Error! Bookmark not defined.**](#_Toc409624916)

[Gambar 3. 17 *Pseudocode* transformasi titik dengan *scale factor* **Error! Bookmark not defined.**](#_Toc409624917)

[Gambar 3. 18 Digram alir tahap deteksi kedipan mata 39](#_Toc409624918)

[Gambar 3. 19 Diagram alir perhitungan korelasi dan *thresholding* terhadap *open eye template* **Error! Bookmark not defined.**](#_Toc409624919)

[Gambar 3. 20 Rancangan halaman kalibrasi mata 41](#_Toc409624920)

[Gambar 3. 21 Rancangan halaman menu uji coba 42](#_Toc409624921)

[Gambar 3. 22 Rancangan halaman menu aplikasi **Error! Bookmark not defined.**](#_Toc409624922)

[Gambar 3. 23 Rancangan halaman deteksi kedipan mata 45](#_Toc409624923)

[Gambar 3. 24 Rancangan halaman pelacakan mata 46](#_Toc409624924)

[Gambar 3. 25 Rancangan halaman aplikasi telepon **Error! Bookmark not defined.**](#_Toc409624925)

[Gambar 3. 26 Rancangan halaman aplikasi memasukkan isi SMS **Error! Bookmark not defined.**](#_Toc409624926)

[Gambar 3. 27 Rancangan halaman aplikasi memasukkan nomor telepon penerima SMS **Error! Bookmark not defined.**](#_Toc409624927)

[Gambar 4. 1 *Pseudocode* proses pengubahan *channel* RGB ke *grayscale* 50](#_Toc409624884)

[Gambar 4. 2 *Pseudocode* proses deteksi wajah dengan *Haar Cascade Classifier* 50](#_Toc409624885)

[Gambar 4. 3 *Pseudocode* proses *cropping* ROI mata 52](#_Toc409624886)

[Gambar 4. 4 *Pseudocode* proses deteksi iris mata dengan *Haar Cascade Classifier* 60](#_Toc409624887)

[Gambar 4. 5 *Pseudocode* proses perluasan ROI iris menjadi *open eye template* **Error! Bookmark not defined.**](#_Toc409624888)

[Gambar 4. 6 *Pseudocode* proses pelacakan mata berdasarkan pupil mata **Error! Bookmark not defined.**](#_Toc409624889)

[Gambar 4. 7 *Pseudocode* proses perhitungan titik tengah *interpupillary* **Error! Bookmark not defined.**](#_Toc409624890)

[Gambar 4. 8 *Pseudocode* proses transformasi titik tengah *interpupillary* dari *window* ke *screen* **Error! Bookmark not defined.**](#_Toc409624891)

[Gambar 4. 9 *Pseudocode* proses deteksi kedipan mata dan *thresholding* korelasi **Error! Bookmark not defined.**](#_Toc409624892)

[Gambar 4. 10 Implementasi halaman kalibrasi mata 71](#_Toc409624893)

[Gambar 4. 11 Implementasi halaman menu uji coba 72](#_Toc409624894)

[Gambar 4. 12 Implementasi halaman menu aplikasi **Error! Bookmark not defined.**](#_Toc409624895)

[Gambar 4. 13 Implementasi halaman deteksi kedipan mata 73](#_Toc409624896)

[Gambar 4. 14 Implementasi halaman pelacakan mata 73](#_Toc409624897)

[Gambar 4. 15 Implementasi halaman aplikasi telepon **Error! Bookmark not defined.**](#_Toc409624898)

[Gambar 4. 16 Implementasi halaman aplikasi memasukkan isi SMS **Error! Bookmark not defined.**](#_Toc409624899)

[Gambar 4. 17 Implementasi halaman aplikasi memasukkan nomor telepon penerima SMS **Error! Bookmark not defined.**](#_Toc409624900)

[Gambar 5. 1 Citra keluaran deteksi kedipan iterasi 1 dengan *threshold* 0,6 dan pencahayaan 22 - 26 lux **Error! Bookmark not defined.**](#_Toc409624868)

[Gambar 5. 2 Citra keluaran deteksi kedipan iterasi 2 dengan *threshold* 0,6 dan pencahayaan 22 - 26 lux **Error! Bookmark not defined.**](#_Toc409624869)

[Gambar 5. 3 Citra keluaran deteksi kedipan iterasi 1 dengan *threshold* 0,7 dan pencahayaan 22 - 26 lux **Error! Bookmark not defined.**](#_Toc409624870)

[Gambar 5. 4 Citra keluaran deteksi kedipan iterasi 2 dengan *threshold* 0,7 dan pencahayaan 22 - 26 lux **Error! Bookmark not defined.**](#_Toc409624871)

[Gambar 5. 5 Citra keluaran deteksi kedipan iterasi 1 dengan *threshold* 0,8 dan pencahayaan 22 - 26 lux **Error! Bookmark not defined.**](#_Toc409624872)

[Gambar 5. 6 Citra keluaran deteksi kedipan iterasi 2 dengan *threshold* 0,8 dan pencahayaan 22 - 26 lux **Error! Bookmark not defined.**](#_Toc409624873)

[Gambar 5. 7 Citra keluaran deteksi kedipan iterasi 1 dengan *threshold* 0,6 dan pencahayaan 194 - 229 lux **Error! Bookmark not defined.**](#_Toc409624874)

[Gambar 5. 8 Citra keluaran deteksi kedipan iterasi 2 dengan *threshold* 0,6 dan pencahayaan 194 - 229 lux **Error! Bookmark not defined.**](#_Toc409624875)

[Gambar 5. 9 Citra keluaran deteksi kedipan iterasi 1 dengan *threshold* 0,7 dan pencahayaan 194 - 229 lux **Error! Bookmark not defined.**](#_Toc409624876)

[Gambar 5. 10 Citra keluaran deteksi kedipan iterasi 2 dengan *threshold* 0,7 dan pencahayaan 194 - 229 lux **Error! Bookmark not defined.**](#_Toc409624877)

[Gambar 5. 11 Citra keluaran deteksi kedipan iterasi 1 dengan *threshold* 0,8 dan pencahayaan 194 - 229 lux **Error! Bookmark not defined.**](#_Toc409624878)

[Gambar 5. 12 Citra keluaran deteksi kedipan iterasi 2 dengan *threshold* 0,8 dan pencahayaan 194 - 229 lux **Error! Bookmark not defined.**](#_Toc409624879)

[Gambar 5. 13 Posisi *button* untuk uji coba pelacakan mata **Error! Bookmark not defined.**](#_Toc409624880)

[Gambar 5. 14 Posisi *button* untuk uji coba aktivasi menu **Error! Bookmark not defined.**](#_Toc409624881)

[Gambar 5. 15 Citra keluaran kalibrasi *template* pada iterasi 1 tanpa menggunakan *cropping* ROI mata **Error! Bookmark not defined.**](#_Toc409624882)

[Gambar 5. 16 Citra keluaran kalibrasi *template* pada iterasi 1 dengan menggunakan *cropping* ROI mata **Error! Bookmark not defined.**](#_Toc409624883)

# DAFTAR TABEL

[Tabel 3. 1 Data Proses 19](#_Toc409624961)

[Tabel 3. 2 Spesifikasi atribut antarmuka halaman kalibrasi mata 41](#_Toc409624962)

[Tabel 3. 3 Spesifikasi atribut antarmuka halaman menu uji coba 42](#_Toc409624963)

[Tabel 3. 4 Spesifikasi atribut antarmuka halaman menu aplikasi **Error! Bookmark not defined.**](#_Toc409624964)

[Tabel 3. 5 Spesifikasi atribut antarmuka halaman deteksi kedipan mata 45](#_Toc409624965)

[Tabel 3. 6 Spesifikasi atribut antarmuka halaman pelacakan mata 46](#_Toc409624966)

[Tabel 3. 7 Spesifikasi atribut antarmuka halaman aplikasi telepon **Error! Bookmark not defined.**](#_Toc409624967)

[Tabel 3. 8 Spesifikasi atribut antarmuka halaman aplikasi memasukkan isi SMS **Error! Bookmark not defined.**](#_Toc409624968)

[Tabel 3. 9 Spesifikasi atribut antarmuka halaman aplikasi memasukkan nomor telepon penerima SMS **Error! Bookmark not defined.**](#_Toc409624969)

[Tabel 4. 1 Lingkungan Implementasi Sistem 49](#_Toc409624960)

[Tabel 5. 1 Data masukan uji coba **Error! Bookmark not defined.**](#_Toc409624939)

[Tabel 5. 2 Hasil akurasi deteksi kedipan kombinasi 1 dengan *threshold* 0,6 dan pencahayaan 22 - 26 lux 77](#_Toc409624940)

[Tabel 5. 3 Hasil akurasi deteksi kedipan kombinasi 2 dengan *threshold* 0,7 dan pencahayaan 22 - 26 lux **Error! Bookmark not defined.**](#_Toc409624941)

[Tabel 5. 4 Hasil akurasi deteksi kedipan kombinasi 3 dengan *threshold* 0,8 dan pencahayaan 22 - 26 lux **Error! Bookmark not defined.**](#_Toc409624942)

[Tabel 5. 5 Hasil akurasi deteksi kedipan kombinasi 4 dengan *threshold* 0,6 dan pencahayaan 194 - 229 lux **Error! Bookmark not defined.**](#_Toc409624943)

[Tabel 5. 6 Hasil akurasi deteksi kedipan kombinasi 5 dengan *threshold* 0,7 dan pencahayaan 194 - 229 lux **Error! Bookmark not defined.**](#_Toc409624944)

[Tabel 5. 7 Hasil akurasi deteksi kedipan kombinasi 6 dengan *threshold* 0,8 dan pencahayaan 194 - 229 lux **Error! Bookmark not defined.**](#_Toc409624945)

[Tabel 5. 8 Hasil akurasi pelacakan mata dengan pencahayaan 22 - 26 lux (cahaya lampu) 84](#_Toc409624946)

[Tabel 5. 9 Hasil akurasi pelacakan mata dengan pencahayaan 194 - 229 lux (cahaya matahari) **Error! Bookmark not defined.**](#_Toc409624947)

[Tabel 5. 10 Hasil akurasi aktivasi menu kombinasi 1 dengan threshold 0,6 dan pencahayaan 22 - 26 lux **Error! Bookmark not defined.**](#_Toc409624948)

[Tabel 5. 11 Hasil akurasi aktivasi menu kombinasi 2 dengan threshold 0,7 dan pencahayaan 22 - 26 lux **Error! Bookmark not defined.**](#_Toc409624949)

[Tabel 5. 12 Hasil akurasi aktivasi menu kombinasi 3 dengan threshold 0,8 dan pencahayaan 22 - 26 lux **Error! Bookmark not defined.**](#_Toc409624950)

[Tabel 5. 13 Hasil akurasi aktivasi menu kombinasi 4 dengan *threshold* 0,6 dan pencahayaan 194 - 229 lux **Error! Bookmark not defined.**](#_Toc409624951)

[Tabel 5. 14 Hasil akurasi aktivasi menu kombinasi 5 dengan *threshold* 0,7 dan pencahayaan 194 - 229 lux **Error! Bookmark not defined.**](#_Toc409624952)

[Tabel 5. 15 Hasil akurasi aktivasi menu kombinasi 6 dengan *threshold* 0,8 dan pencahayaan 194 - 229 lux **Error! Bookmark not defined.**](#_Toc409624953)

[Tabel 5. 16 Hasil akurasi kalibrasi *template* tanpa menggunakan *cropping* ROI mata **Error! Bookmark not defined.**](#_Toc409624954)

[Tabel 5. 17 Hasil *frame rate* kalibrasi *template* tanpa menggunakan *cropping* ROI mata **Error! Bookmark not defined.**](#_Toc409624955)

[Tabel 5. 18 Hasil akurasi kalibrasi *template* dengan menggunakan *cropping* ROI mata **Error! Bookmark not defined.**](#_Toc409624956)

[Tabel 5. 19 Hasil *frame rate* kalibrasi *template* dengan menggunakan *cropping* ROI mata **Error! Bookmark not defined.**](#_Toc409624957)

[Tabel 5. 20 Data masukan uji coba aktivasi menu dengan pengguna yang berbeda 85](#_Toc409624958)

[Tabel 5. 21 Hasil akurasi aktivasi menu dengan pengguna yang berbeda pada kondisi terbaik uji coba **Error! Bookmark not defined.**](#_Toc409624959)

# BAB I PENDAHULUAN

Bab ini membahas garis besar penyusunan tugas akhir yang meliputi latar belakang, tujuan pembuatan, rumusan dan batasan permasalahan, metodologi penyusunan tugas akhir, dan sistematika penulisan.

## Latar Belakang

Berkembangnya teknologi memberikan pengaruh besar bagi kesejahteraan hidup manusia. Salah satu teknologi yang memberikan banyak manfaat dan sudah banyak diterapkan dalam masyarakat adalah pengolahan citra digital. Teknologi ini tidak hanya untuk mengolah sebuah citra dengan hasil keluaran citra, namun hasil keluaran pengolahan ini dapat berupa sebuah informasi yang bermanfaat bagi manusia baik dalam bidang kesehatan, pendidikan, dan sebagainya. Salah satu teknologi yang memberikan hasil keluaran berupa informasi dari pengolahan citra adalah *Optical Character Recognition* (OCR) dimana teknologi ini menghasilkan informasi sebuah pola suatu objek dari sebuah citra [1].

Salah satu penerapan teknologi OCR ini digunakan untuk mengenali pola objek pada tulisan tangan. Pengenalan pola objek yang dilakukan adalah pengenalan pola terhadap karakter aksara Jawa dari tulisan tangan. Permasalahan yang terdapat pada tulisan tangan adalah bervariasinya tulisan tangan dalam setiap karakter yang menyebabkan susah untuk dikenali. Teknologi OCR ini berguna untuk memudahkan dalam mengenali tulisan tangan tersebut terutama pada karakter aksara Jawa untuk hal ini.

Tulisan tangan dalam aksara Jawa sering ditemui dalam buku-buku kuno yang menjelaskan kekayaan budaya Jawa. Sayangnya, hanya sedikit orang yang mampu membaca tulisan aksara Jawa tersebut. Karena naskah buku kuno tersebut memiliki kontribusi berharga untuk pengetahuan, sistem pengenalan tulisan tangan aksara Jawa secara otomatis sangat dibutuhkan untuk pembacaan tulisan dalam naskah buku kuno tersebut [2].

Berdasarkan penelitian yang telah dilakukan mengenai pengenalan pola tulisan tangan, digunakan dua proses yaitu ekstraksi fitur dan klasifikasi. Proses ekstraksi fitur bertujuan untuk mendapatkan informasi-informasi penting pada sebuah citra, sehingga dapat membedakan antara citra satu dengan yang lain. Proses klasifikasi menghasilkan prediksi kelas dari data citra yang sudah diolah berdasarkan aturan-aturan atau *rule* dari algoritma klasifikasi yang digunakan [3]. Pada penelitian sebelumnya, pengenalan aksara Jawa menggunakan ekstraksi fitur berupa panjang kaki, banyak lingkaran, banyak kaki, serta banyak huruf untuk membedakan sebuah aksara. Penelitian tersebut menggunakan algoritma klasifikasi *decision tree* dalam mengklasifikasikan karakter aksara Jawa dan menerapkan dalam platform *Windows Phone*.

Dalam tugas akhir ini, penelitian ini akan diterapkan pada *smartphone* Android yang menerapkan layar sentuh dan banyak digunakan oleh masyarakat luas. Oleh karena itu, dibutuhkan sebuah algoritma klasifikasi yang mengolah data pelatihan secara cepat tetapi tetap memiliki tingkat akurasi tinggi. Algoritma klasifikasi K-Nearest Neighbor mampu mengolah data pelatihan lebih cepat karena tidak membutuhkan model tertentu dan juga memiliki tingkat akurasi tinggi. Algoritma ini yang memberikan prediksi kelas sebuah data berdasarkan kelas terbanyak dari K-tetangga dengan jarak terdekat. Sehingga dalam kasus ini, prediksi kelas dari citra hasil ekstraksi fitur ditentukan berdasarkan kelas terbanyak dari K-tetangga dengan jarak terdekat dari data pelatihan. Dari hasil klasifikasi tersebut, diharapkan dapat membedakan dengan cepat input karakter aksara Jawa secara baik sehingga pengenalan karakter aksara Jawa yang di-input-kan menjadi lebih mudah dan cepat.

## Rumusan Masalah

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

1. Bagaimana membuat aplikasi yang mampu melakukan ekstraksi fitur terhadap karakter aksara Jawa dari input tulisan tangan.
2. Bagaimana membuat aplikasi yang mampu mengenali karakter aksara Jawa dari input tulisan tangan.
3. Bagaimana membuat aplikasi *mobile* android untuk mengenali karakter aksara Jawa berbasis OpenCV Android.

## Batasan Masalah

Permasalahan yang dibahas dalam tugas akhir memiliki beberapa batasan antara lain:

1. Aplikasi ini berbasis *platform mobile android* dengan bahasa pemrograman Java menggunakan OpenCV Java API
2. *Smartphone* yang digunakan untuk uji coba adalah merek Lenovo A526 dengan spesifikasi utama yaitu prosesor Quad-Core 1.3 GHz, RAM 1 GB, dan kamera utama 5 MP, serta berjalan pada sistem operasi Android 4.2.2 JellyBean.
3. Karakter aksara Jawa yang mampu dikenali adalah aksara dasar dalam aksara Jawa.
4. Citra masukan dari kamera utama adalah citra dengan *background* putih, tulisan menggunakan spidol, tidak terdapat bayangan di dalamnya, serta pencahayaan yang cukup.

## Tujuan

Tujuan dari pembuatan tugas akhir ini adalah membangun sebuah aplikasi yang dapat mengenali karakter pada aksara Jawa dari tulisan tangan menggunakan OpenCV Android.

## Metodologi

Tahap yang dilakukan untuk menyelesaikan Tugas Akhir ini adalah sebagai berikut:

* + - 1. **Penyusunan proposal tugas akhir**

Proposal Tugas Akhir ini berisi tentang deskripsi pendahuluan dari tugas akhir yang akan dibuat. Pendahuluan ini terdiri atas hal yang menjadi latar belakang diajukannya usulan tugas akhir, rumusan masalah yang diangkat, batasan masalah untuk tugas akhir, tujuan dari pembuatan tugas akhir, dan manfaat dari hasil pembuatan tugas akhir. Selain itu dijabarkan pula tinjauan pustaka yang digunakan sebagai referensi pendukung pembuatan tugas akhir. Sub bab metodologi berisi penjelasan mengenai tahapan penyusunan tugas akhir mulai dari penyusunan proposal hingga penyusunan buku tugas akhir. Terdap at pula sub bab jadwal kegiatan yang menjelaskan jadwal pengerjaan tugas akhir.

* + - 1. **Studi literatur**

Pada tugas akhir ini. referensi yang akan dipelajari adalah sejumlah referensi tentang ekstraksi fitur citra, algoritma klasifikasi K-Nearest Neighbor, dan penggunaan OpenCV Android.

* + - 1. **Implementasi**

Pada tahap ini dilakukan implementasi perangkat lunak dalam platform android dengan bahasa pemrograman Java dan menggunakan *library* OpenCV. Rencana kakas bantu yang digunakan adalah Eclipse JUNO.

* + - 1. **Uji Coba dan Evaluasi**

Pada tahap ini dilakukan uji coba aplikasi dan evaluasi terhadap implementasi metode pada aplikasi. Tahap uji coba difokuskan pada kumpulan kondisi input dari tester serta pengetesan terhadap spesifikasi fungsional program. Tahap evaluasi dilakukan dengan menghitung tingkat akurasi yang didapatkan dari pembacaan kelas pada *dataset* tulisan aksara Jawa menggunakan metode yang telah ditentukan dan menghitung tingkat akurasi yang didapatkan dari implementasi aplikasi dalam mengenali aksara Jawa.

* + - 1. **Penyusunan Buku Tugas Akhir**

Tahap ini merupakan tahap dokumentasi dari tugas akhir. Buku tugas akhir berisi dasar teori, perancangan, implementasi dan hasil uji coba dan evaluasi dari aplikasi yang dibangun.

## Sistematika Penulisan

Buku tugas akhir ini terdiri atas beberapa bab yang tersusun secara sistematis, yaitu sebagai berikut.

1. Bab I. Pendahuluan

Bab pendahuluan berisi penjelasan mengenai latar belakang masalah, rumusan masalah, batasan masalah, tujuan, manfaat dan sistematika penulisan tugas akhir.

1. Bab II. Tinjauan Pustaka

Bab tinjauan pustakan berisi penjelasan mengenai dasar teori yang mendukung pengerjaan tugas akhir.

1. Bab III. Analisis dan Perancangan

Bab analisis dan perancangan berisi penjelasan mengenai analisis kebutuhan, perancangan sistem dan perangkat yang digunakan dalam pengerjaan tugas akhir serta urutan pelaksanaan proses.

1. Bab IV. Implementasi

Bab implementasi berisi pembangunan aplikasi pengenalan aksara jawa melalui tulisan tangan menggunakan metode K-Nearest Neighbor berbasis OpenCV Android sesuai dengan rumusan dan batasan yang sudah dijelaskan pada bagian pendahuluan.

1. Bab V. Uji Coba dan Evaluasi

Bab uji coba dan evaluasi berisi pembahasan mengenai hasil dari uji coba yang dilakukan terhadap aplikasi pengenalan aksara jawa melalui tulisan tangan menggunakan metode K-Nearest Neighbor berbasis OpenCV Android.

1. Bab VI. Kesimpulan dan Saran

Bab kesimpulan dan saran berisi kesimpulan hasil penelitian. Selain itu, bagian ini berisi saran untuk pengerjaan lebih lanjut atau permasalahan yang dialami dalam proses pengerjaan tugas akhir.

# BAB II TINJAUAN PUSTAKA

Bab tinjauan pustaka berisi mengenai penjelasan teori yang berkaitan dengan implementasi perangkat lunak. Penjelasan tersebut bertujuan untuk memberikan gambaran mengenai sistem yang akan dibangun dan berguna sebagai

pendukung dalam pengembangan perangkat lunak.



## Ekstraksi Fitur

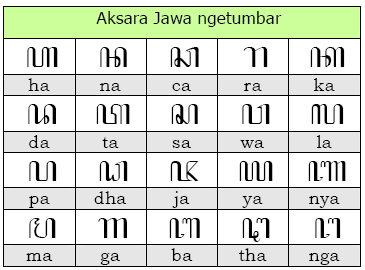
*Feature Extraction* atau ekstraksi fitur merupakan suatu pengambilan ciri / fitur dari suatu bentuk yang nantinya nilai yang didapatkan akan dianalisis untuk proses selanjutnya. *Feature extraction* dilakukan dengan cara menghitung jumlah titik atau piksel yang ditemui dalam setiap pengecekan, dimana pengecekan dilakukan dalam berbagai arah *tracing* pengecekan pada koordinat kartesian dari citra digital yang dianalisis, yaitu vertikal, horizontal, diagonal kanan, dan diagonal kiri [4].

## Aksara Jawa

Aksara Jawa atau lebih dikenal dengan Hanacaraka adalah aksara turunan aksara Brahmi yang digunakan atau pernah digunakan untuk penulisan naskah-naskah berbahasa Jawa, Makasar, Madura, Melayu, Sunda, Bali, dan Sasak.

Pada bentuknya yang asli, aksara Jawa Hanacaraka ditulis menggantung (di bawah garis), seperti aksara Hindi. Namun pada pengajaran modern menuliskannya di atas garis.

Aksara Hanacaraka memiliki 20 huruf dasar, 20 huruf pasangan yang berfungsi menutup bunyi vokal, 8 huruf “utama” (aksara murda, ada yang tidak berpasangan), 8 pasangan huruf utama, lima aksara swara (huruf vokal depan), lima aksara rekan dan lima pasangannya, beberapa sandhangan sebagai pengatur vokal, beberapa huruf khusus, beberapa tanda baca, dan beberapa tanda pengatur tata penulisan (pada) [5]. Gambar 2. 1 merupakan bentuk dari aksara dasar pada aksara Jawa.



Gambar 2. 1 Bentuk Karakter Aksara Dasar Aksara Jawa

## Klasifikasi K-Nearest Neighbor

Klasifikasi K-Nearest Neighbor merupakan sebuah algoritma klasifikasi yang menggunakan jarak antara data percobaan dengan data pelatiihan serta jumlah tetangga terdekat dari data percobaan sebagai tolak ukur klasifikasinya. Algoritma ini sudah digunakan dalam estimasi statistik dan pengenalan pola (*pattern recognition*) pada awal tahun 1970 [6]. Penentuan atribut kelas pada data percobaan ditentukan dengan jumlah *voting* terbanyak dari tetangganya.

Hal-hal terpenting pada klasifikasi ini adalah data pelatihan dengan atribut yang sama jenis datanya dan tidak ada nilai yang kosong, nilai jarak antar data, nilai K untuk mengetahui berapa jumlah tetangga yang digunakan dalam klasifikasi. Metodologi klasifikasi ini secara sederhana dapat dijabarkan sebagai berikut:

1. Hitung jarak antara data percobaan dengan data pelatihan dengan fungsi jarak. Fungsi jarak adalah fungsi jarak Euclidean yang didefinisikan pada persamaan 1 antara data percobaan *p* dan data pelatihan *q* pada indeks ke *i*.

|  |  |
| --- | --- |
|  | (2. 1) |

1. Identifikasi K nearest-neighbor dengan nilai K yang sudah ditentukan. Nilai K yang dipilih harus menghasilkan klasifikasi yang baik (tidak terlalu kecil atau besar). Nilai K yang terlalu kecil sangat sensitif terhadap data *noise*, sedangkan jika terlalu besar menyebabkan ketetanggaannya dapat meliputi data dari kelas lain [7].
2. Gunakan label kelas dari klasifikasi untuk menentukan kelas dari data percobaan dengan *voting*.

Gambar 2. 2 merupakan contoh dari klasifikasi K-Nearest Neighbor dengan nilai K yang berbeda.



Gambar 2. 2 Contoh Klasifikasi K-Nearest Neighbor

Seperti yang ditunjukkan oleh Gambar 2. 2, klasifikasi ini menggunakan parameter data pelatihan dengan jarak terdekat sejumlah K. Penentuan kelas untuk X (sesuai Gambar 2. 2) berdasarkan kelas dari K-tetangga yang digunakan dalam klasifikasi.

Dalam tugas akhir ini, klasifikasi ini digunakan untuk klasifikasi citra tulisan tangan dengan atribut-atribut berupa hasil ekstraksi fitur.

## OpenCV Android

OpenCV (Open Source Computer Vision) adalah sebuah BSD-license open-source library yang mencakup ratusan algoritma visi komputer [8]. OpenCV ini gratis baik untuk penggunaan akademis maupun komersial. Library ini memiliki C++, C, Phyton, dan Java interface yang mendukung Windows, Linux, Mac OS, iOS dan Android. OpenCV dirancang untuk efisiensi komputasi dan dengan fokus yang kuat pada aplikasi real-time. Dituliskan dalam bahasa C/C++ yang optimal, menyebabkan library ini memiliki keuntungan dalam pada multi-core processing. Diaktifkan dengan OpenCL, OpenCV dapat mengambil keuntungan dari akselerasi hardware pada komputasi pokok platform heterogen. Diadopsi di seluruh dunia, OpenCV memiliki lebih dari 47 ribu orang dari komunitas pengguna dan diperkirakan jumlah download melebihi 9 juta [9].

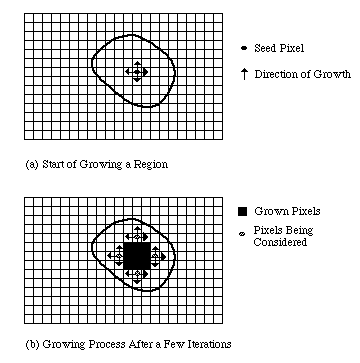
Untuk menggunakan OpenCV pada sebuah aplikasi Android, harus meng-install OpenCV Manager sebelumnya. OpenCV Manager adalah sebuah service Android yang ditargetkan untuk mengelola OpenCV binary library pada device end-user. OpenCV Manager ini mengizinkan untuk berbagi OpenCV dynamic library antar aplikasi dengan perangkat yang sama [10].

## Region Growing

*Region growing* adalah sebuah metode segmentasi citra sederhana dengan memulai dari beberapa piksel (biasa disebut *seed*). *Seed* tersebut merepresentasikan region-region citra yang berbeda dan “menumbuhkannya” sampai memenuhi seluruh citra.

Untuk metode ini, dibutuhkan aturan yang mengatur mekanisme tumbuhnya *seed* dan suatu aturan lain yang menguji kehomogenan dari region setelah satu tahap tumbuh selesai [11]. Pertumbuhan region dimulai dari *seed* awal dengan menambahkan tetangga piksel (menggunakan 4-tetangga) yang serupa, memperluas ukuran region.

Ketika pertumbuhan sebuah region selesai, langkah selanjutya adalah memilih *seed* baru dan melakukan region growing kembali. Proses tersebut dilakukan sampai semua piksel berhasil dikelompokkan dalam beberapa region [12]. Gambar 2. 3 adalah gambar yang menjelaskan pemilihan *seed* awal dan proses dari metode *region growing*.



Gambar 2. 3 Ilustrasi *Region Growing*

## Algoritma *Thinning* Guo-Hall

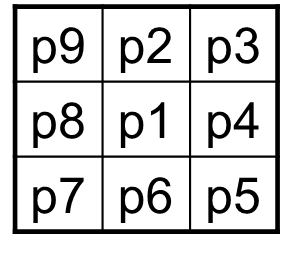
*Thinning* adalah salah satu cara untuk *skeletonizing*. Dalam pengolahan citra, *thinning* biasa dilakukan dengan citra biner. Sebelum melakukan *thinning* pada citra selain biner (misal citra *grayscale*), dilakukan thresholding terlebih dahulu untuk merubahnya ke citra biner. Citra biner yang digunakan adalah citra biner dengan nilai 0 sebagai *background* dan nilai 1 sebagai *foreground*.

Algoritma *thinning* yang dilakukan terhadap citra biner seharusnya memiliki kriteria-kriteria sebagai berikut:

* Skeleton dari citra kira-kira berada di bagian tengah dari citra awal sebelum dilakukan *thinning*.
* Citra hasil dari algoritma *thinning* harus tetap menjaga struktur keterhubungan yang sama dengan citra awal.
* Suatu skeleton seharusnya memiliki bentuk yang hampir mirip dengan citra awal.
* Suatu skeleton seharusnya mengandung jumlah piksel yang seminimal mungkin namun tetap memenuhi kriteria-kriteria sebelumnya.

Algoritma Guo-Hall merupakan salah satu algoritma *thinning* yang cukup populer dan diimplementasikan terhadap proses *thinning* pada MATLAB. Setiap iterasi pada metode ini terdiri dari dua sub-iterasi yang berurutan yang dilakukan terhadap *contour points* dari wilayah citra. *Contour point* adalah setiap piksel dengan nilai 1 dan memiliki setidaknya satu *8-neighbor* dengan nilai 0.

Gambar 2. 4 adalah representasi dari *contour point* p dengan *8-neighbor* yang dimiliki oleh p.



Gambar 2. 4 Representasi *Contour Point* p

Langkah pertama implementasi algoritma Guo-Hall adalah menandai *contour point* p untuk dihapus jika semua kondisi ini dipenuhi:

(a) C(p1) = 1;

Dimana C(p1) = !p2 & (p3 | p4) + !p4 & (p5 | p6) + !p6 &

(p7 | p8) + !p8 & (p1 | p2).

(b) 2 ≤ N(p1) ≤ 3;

Dimana N(p1) adalah nilai minimum dari N1p1 dan N2p2 yang mana nilai N1p1 adalah (p9 | p2) + (p3 | p4) + (p5 | p6) + (p7 | p8) dan nilai N2p1 adalah (p2 | p3) + (p4 | p5) + (p6 | p7) + (p8 | p9).

(c) (p2 | p3 | !p5) & p4 = 0;

Langkah kedua hampir sama dengan langkah pertama, yaitu titik p akan dihapus jika memenuhi kondisi (a) dan (b), sedangkan kondisi (c) diubah menjadi (p6 | p7 | !p9) & p8 = 0 [13].

Langkah pertama dilakukan terhadap semua *border pixel* di citra. Jika salah satu dari ketiga kondisi di atas tidak dipenuhi atau dilanggar maka nilai piksel yang bersangkutan tidak diubah. Sebaliknya jika semua kondisi tersebut dipenuhi maka piksel tersebut ditandai untuk dihapus.

Piksel yang ditandai tidak dihapus sebelum semua *border points* selesai diproses. Setelah langkah 1 selesai dilakukan untuk semua *border points* maka dilakukan penghapusan untuk titik yang telah ditandai (diubah menjadi 0). Setelah itu dilakukan langkah 2 pada data hasil dari langkah 1 dengan cara yang sama dengan langkah 1.

Sehingga dalam satu kali iterasi, urutan algoritmanya sebagai berikut:

1. Menjalankan langkah 1 untuk menandai *border points* yang akan dihapus.
2. Hapus titik-titik yang ditandai dengan menggantinya menjadi angka 0.
3. Menjalankan langkah 2 pada sisa *border points* yang pada langkah 1 belum dihapus lalu yang sesuai dengan semua kondisi yang seharusnya dipenuhi pada langkah 2 kemudian ditandai untuk dihapus.
4. Hapus titik-titik yang ditandai dengan menggantinya menjadi angka 0 [14].

Prosedur ini dilakukan secara iteratif sampai tidak ada lagi titik yang dapat dihapus. Pada saat algoritma ini selesai maka dihasilkan *skeleton* dari citra awal.

[Halaman ini sengaja dikosongkan]

# BAB III ANALISIS DAN PERANCANGAN

Pada Bab 3 ini akan dijelaskan mengenai analisis dan perancangan perangkat lunak untuk mencapai tujuan dari tugas akhir. Perancangan ini meliputi perancangan data, perancangan proses, dan perancangan antar muka, serta juga akan dijelaskan tentang analisis implementasi metode secara umum pada sistem.

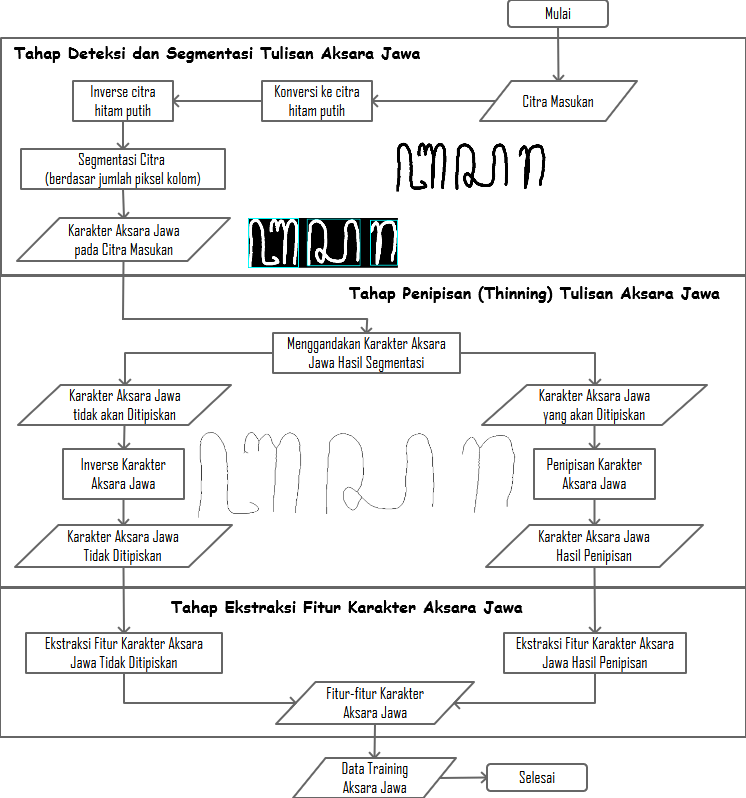


## Analisis Implementasi Metode Secara Umum

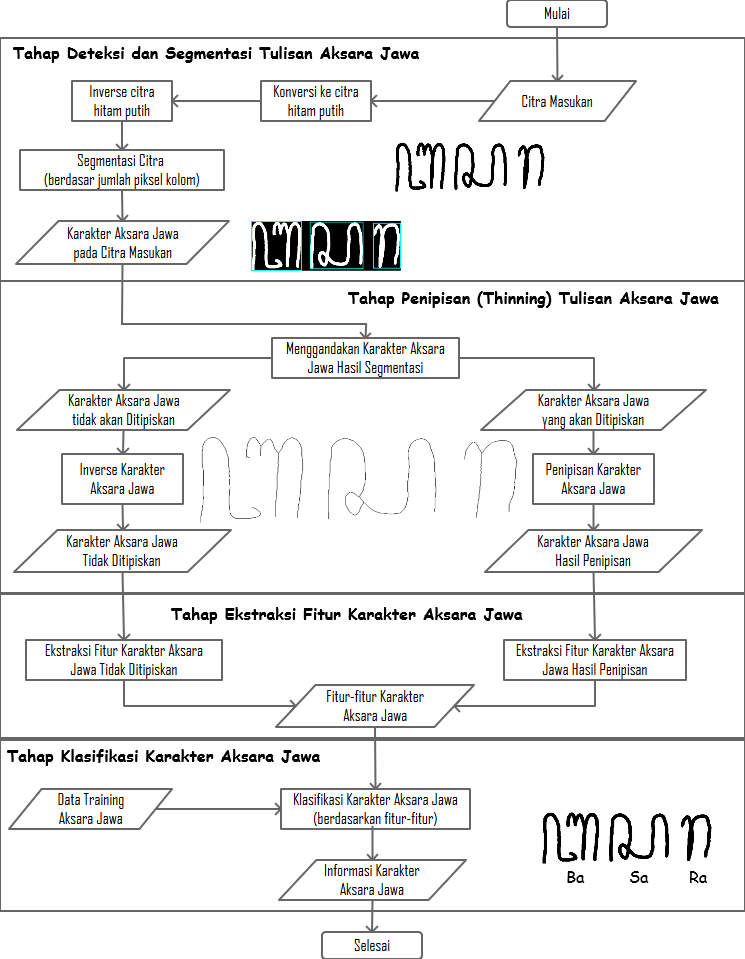
Pada tugas akhir ini akan dibangun sebuah sistem untuk mengenali aksara dasar dalam aksara Jawa dengan bantuan OpenCV Android. Secara garis besar, tugas akhir ini memiliki dua proses *training* yang digunakan untuk mendapatkan data pelatihan dan *testing* yang digunakan dalam implementasi sistem. Proses-proses yang terlibat dalam proses *testing* atau implementasi sistem ini meliputi tahap deteksi atau segmentasi input tulisan aksara Jawa, tahap penipisan (*thinning*) citra input tulisan aksara Jawa, tahap ekstraksi fitur dari tulisan aksara Jawa, dan tahap klasifikasi tulisan aksara Jawa. Sedangkan proses *training* menggunakan tahap-tahap dalam proses *testing* kecuali tahap klasifikasi tulisan aksara Jawa.

Tahap deteksi atau segmentasi merupakan tahap pertama yang dilakukan untuk mengenali citra masukan karakter-karakter dasar aksara Jawa yang akan diolah. Dalam tahap ini, setiap karakter dasar aksara Jawa pada citra masukan akan dibaca agar proses penipisan hanya diterapkan pada setiap karakter sehingga menjadi lebih cepat dan mudah dikenali. Proses selanjutnya adalah proses penipisan (*thinning*) yang diimplementasikan pada setiap karakter dasar aksara Jawa yang berhasil dibaca sebelumnya. Proses ini menghasilkan *skeleton* dari setiap karakter aksara Jawa yang mana *skeleton* tersebut digunakan untuk ekstraksi fitur pada karakter aksara Jawa.

Tahap utama dari implementasi metode ini adalah tahap ekstraksi fitur dari tulisan aksara Jawa. Fitur-fitur yang didapatkan berasal dari citra hasil penipisan dan citra asli hitam putih sebelum proses penipisan. Tahap terakhir adalah tahap klasifikasi tulisan aksara Jawa dimana fitur yang digunakan klasifikasi adalah fitur yang dihasilkan proses sebelumnya. Hasil klasifikasi adalah informasi tentang setiap karakter aksara Jawa pada citra masukan. Diagram alir kesuluruhan dari proses *training* atau pelatihan ditunjukkan oleh Gambar 3. 1. Sedangkan diagram alir dari keseluruhan proses *testing* atau implementasi sistem ditunjukkan oleh Gambar 3. 1.



Gambar 3. 1 Diagram Alir Proses *Training* atau Pelatihan



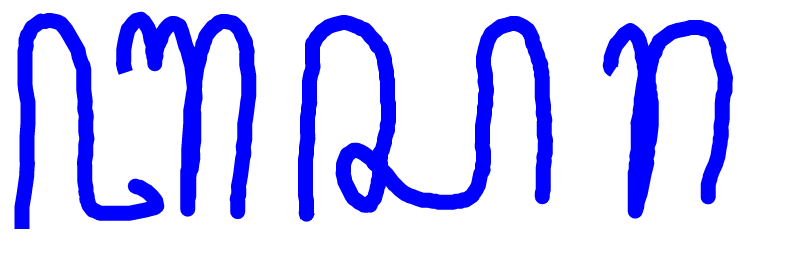
Gambar 3. 2 Diagram Alir Proses Implementasi Sistem

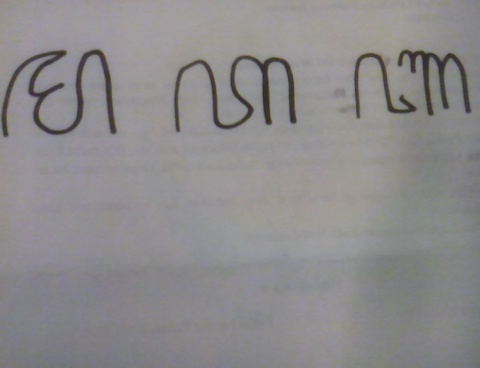
## Perancangan Data

Pada subbab ini akan dibahas mengenai perancangan data yang merupakan bagian penting karena data sebagai objek yang akan diolah oleh perangkat lunak dalam tugas akhir ini dan menghasilkan sebuah informasi. Data yang akan digunakan pada sistem ini adalah data masukan (*input*), data proses, dan data keluaran (*output*) yang memberikan hasil pengolahan sistem ini untuk pengguna.

### Data Masukan

Data masukan merupakan data awal yang akan diproses oleh sistem untuk mengenali aksara Jawa dari citra tulisan tangan. Data masukan tersebut berupa citra yang diambil menggunakan *fingerpaint* (tulisan tangan yang digambarkan pada layar Android) dari pengguna yang disimpan dalam citra RGB dengan ukuran 806 x 360 piksel atau citra hasil kamera utama *smartphone* beresolusi 5 MP (*megapixel*) dengan ukuran 800 x 480 piksel dan memiliki *image* *channel* berupa RGB. Contoh citra masukan yang digunakan sebagai data masukan ditunjukkan oleh Gambar 3. 3.





Gambar 3. 3 Citra hasil *fingerpaint* (atas) dan kamera utama *smartphone* (bawah) sebagai data masukan

### Data Proses

Data proses adalah data yang digunakan selama proses berjalannya sistem yang merupakan hasil pengolahan dari data masukan untuk diproses kembali menjadi data keluaran di tahap selanjutnya. Data proses yang digunakan di dalam proses ini ditunjukkan oleh Tabel 3. 1.

Tabel 3. 1 Data Proses

| No. | Nama Data | Keterangan |
| --- | --- | --- |
| 1. | imgMat | Citra masukan awal seperti pada Gambar 3. dengan ukuran 806 x 322 piksel atau 480 x 368 piksel dan *image channel* RGB yang dipetakan dalam sebuah matriks oleh OpenCV |
| 2. | binaryMat | Citra masukan awal seperti pada Gambar 3. dengan ukuran 806 x 322 piksel atau 480 x 368 piksel dan *image channel* hitam putih yang dipetakan dalam sebuah matriks oleh OpenCV |
| 3 | inverseBinary | Citra hitam putih hasil *inverse* dari binaryMat yang dipetakan dalam sebuah matriks oleh OpenCV |
| 4 | aksaraList | *List* berisikan kumpulan citra karakter aksara Jawa dari inverseBinary setelah segmentasi |
| 5 | thinMat | Citra hasil *thinning* setiap citra karakter aksara Jawa pada aksaraList yang dipetakan dalam sebuah matriks oleh OpenCV |
| 6 | bwKarakter | Citra *inverse* hitam putih setiap karakter pada aksaraList yang dipetakan dalam sebuah matriks oleh OpenCV |
| 7 | regionArea | Citra hasil *region growing* dengan *seed* awal adalah piksel posisi (0,0) yang dipetakan dalam sebuah matriks oleh OpenCV |
| 8 | regionKarakter | Citra hasil *region growing* dengan *seed* awal adalah sebuah piksel bernilai 0 yang dipetakan dalam sebuah matriks oleh OpenCV |

### Data Keluaran

Data keluaran dari sistem ini berupa fitur-fitur yang akan digunakan untuk proses klasifikasi karakter aksara Jawa dan hasil dari proses klasifikasi yang berupa informasi tentang karakter aksara Jawa itu sendiri. Data keluaran yang berupa fitur tidak akan diinformasikan kepada pengguna *smartphone*, sebaliknya hasil klasifikasi akan diinformasikan untuk mengetahui apakah informasi yang diinginkan pengguna sama dengan informasi yang diberikan oleh sistem.

## Perancangan Proses

Pada subbab ini akan dibahas mengenai perancangan proses yang dilakukan untuk memberikan gambaran secara rinci pada setiap alur implementasi metode dalam sistem pengenalan aksara Jawa melalui tulisan tangan pada Android ini. Alur tersebut nantinya akan digunakan dalam tahap implementasi.

### Tahap Deteksi dan Segmentasi Tulisan Aksara Jawa

Di dalam tahap deteksi dan segmentasi tulisan aksara Jawa ini terdapat beberapa proses antara lain mengubah citra masukan menjadi citra hitam putih, meng-*inverse* citra hitam putih hasil konversi citra masukan sehingga piksel karakter aksara Jawa berwarna putih dan *background* berwarna hitam, segmentasi citra hasil *inverse* berdasarkan jumlah piksel setiap kolom untuk mendapatkan citra setiap karakter aksara Jawa yang ada dalam citra masukan. Diagram alir mengenai tahap ini dapat dilihat pada Gambar 3. 4.



Gambar 3. 4 Diagram alir tahap deteksi dan segmentasi tulisan aksara Jawa

#### Konversi ke CitraHitam Putih

Konversi citra masukan ke citra hitam putihadalah proses yang pertama di dalam tahap deteksi dan segmentasi tulisan aksara Jawa. Proses ini dilakukan dengan maksud untuk memudahkan sistem dalam mendapatkan fitur karena *channel* hitam putih memiliki kedalaman citra sebesar 1 dan hanya memiliki 2 nilai yaitu 0 dan 1. Hasil yang didapatkan dari perubahan citra ini adalah citra bernilai 0 untuk karakter aksara Jawa dan bernilai 1 untuk *background*.

#### Melakukan *Inverse* Citra Hitam Putih

Proses ini merupakan proses yang sangat penting karena membuat proses segmentasi lebih mudah dimana karakter aksara Jawa disegmentasi berdasarkan jumlah piksel setiap kolom. Hasil proses ini adalah citra hitam putih dengan piksel bernilai 1 untuk karakter aksara Jawa dan piksel bernilai 0 untuk *background*. Selain proses segmentasi, citra ini juga digunakan untuk proses penipisan (*thinning*) dan proses ekstraksi fitur.

#### Segmentasi Citra Berdasarkan Jumlah Piksel Kolom

Seperti penjelasan pada diagram alir tentang tahap deteksi dan segmentasi, segmentasi karakter aksara Jawa ini berdasarkan jumlah piksel secara vertikal atau jumlah piksel per kolom. Oleh karena itu, citra hasil *inverse* ini sangat mendukung metode segmentasi ini. Berikut ini adalah langkah-langkah proses segmentasi:

* Menjumlahkan piksel per kolom dari citra hasil *inverse*

Dari hasil penjumlahan tersebut, jika hasilnya bernilai lebih dari nol maka piksel pada kolom tersebut merupakan bagian dari sebuah karakter aksara Jawa pada citra masukan.

* Mendapatkan indeks kolom pada citra yang merepresentasikan karakter aksara Jawa

Indeks kolom yang akan diambil adalah indeks kolom yang jumlah piksel kolomnya lebih dari nol tetapi indeks-indeks tersebut tidak hanya merepresentasikan satu karakter aksara Jawa. Untuk mengetahui jumlah karakter yang ada dan indeks yang merepresentasikannya, ada beberapa hal yang harus diperhatikan diantaranya:

* + - * Pencarian indeks diurutkan dari kolom pertama sampai terakhir.
      * Dalam pencarian indeks, karakter aksara Jawa satu dengan lainnya dipisahkan oleh indeks dengan jumlah piksel kolom bernilai nol.
      * Indeks karakter aksara Jawa pertama dimulai dari indeks dengan jumlah piksel kolom bernilai tidak nol pertama dan diakhiri sebelum indeks dengan jumlah piksel kolom bernilai nol.
      * Indeks karakter aksara Jawa selanjutnya dimulai dari indeks dengan jumlah piksel kolom tidak bernilai nol terhitung setelah indeks terakhir karakter sebelumnya dan diakhiri sebelum indeks dengan jumlah piksel kolom bernilai nol.
* Menyimpan indeks kolom pertama dan terakhir dari setiap karakter aksara Jawa yang tersegmentasi

Indeks pertama dan terakhir dari setiap karakter aksara Jawa yang tersegmentasi digunakan untuk menunjukkan daerah setiap karakter askara Jawa berdasarkan kolom.

* Menghitung jumlah piksel setiap baris dari setiap daerah karakter aksara Jawa berdasarkan kolom

Dari hasil penjumlahan tersebut, jika hasilnya bernilai lebih dari nol maka piksel pada baris tersebut merupakan bagian dari karakter aksara Jawa dalam daerah tersebut.

* Mendapatkan indeks baris pada setiap daerah karakter aksara Jawa berdasarkan kolom yang merupakan bagian dari karakter aksara Jawa pada daerah bersangkutan

Indeks baris yang akan diambil adalah indeks baris yang jumlah piksel barisnya lebih dari nol.

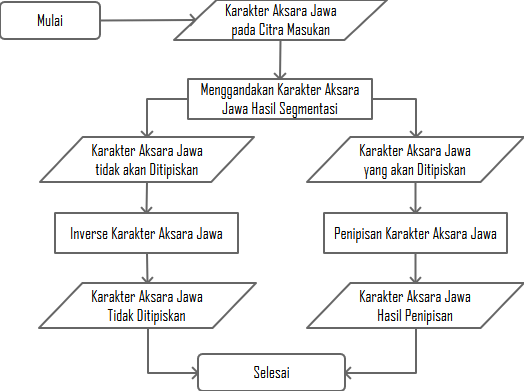
* Mendapatkan daerah setiap karakter aksara Jawa

Setelah mendapatkan indeks baris dari masing-masing daerah karakter aksara Jawa berdasarkan kolom, langkah selanjutnya adalah mencari indeks pertama dan terakhirnya untuk mendapatkan daerah aksara Jawa berdasarkan baris. Langkah terakhir adalah mendapatkan daerah setiap karakter aksara Jawa yang dimulai indeks pertama kolom dan diakhiri indeks terakhir kolom untuk kolom daerah karakter aksara Jawa dan cara yang sama pada baris karakter aksara Jawa untuk baris daerah karakter aksara Jawa.

Setelah mendapatkan semua daerah karakter aksara Jawa, indeks-indeks penting dari setiap daerah (indeks awal dan akhir baris dan kolom) disimpan dalam sebuah *list* untuk proses selanjutnya. Sebelum menuju proses selanjutnya, setiap citra karakter ditambah 2 kolom dengan piksel bernilai nol di kiri dan kanan citra dan ditambah 2 baris dengan piksel bernilai nol di atas dan bawah citra. Hal tersebut dimaksudkan untuk proses ekstraksi fitur yang dilakukan terhadap citra yang tidak ditipiskan pada proses penipisan (*thinning*).

### Tahap Penipisan (Thinning) Tulisan Aksara Jawa

Dalam tahap penipisan (*thinning*), terdapat proses penggandaan karakter aksara Jawa hasil proses sebelumnya karena pada proses ekstraksi fitur membutuhkan citra yang tidak ditipiskan. Citra bukan hasil penipisan yang digunakan dalam ekstraksi fitur adalah hasil *inverse* dari citra hasil segmentasi sehingga citra yang diperoleh adalah citra dengan piksel bernilai 0 untuk karakter aksara Jawa dan piksel bernilai 1 untuk *background*. Proses *thinning* ini menggunakan algoritma *thinning* Guo-Hall yang juga diimplementasikan pada MATLAB untuk citra hitam putih. Diagram alir dari tahap penipisan (*thinning*) tulisan aksara Jawa ini dapat dilihat pada Gambar 3. 5.



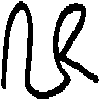
Gambar 3. 5 Diagram alir tahap penipisan (*thinning*) tulisan aksara Jawa

#### Penggandaan Karakter Aksara Jawa Hasil Segmentasi

Proses ini dilakukan karena terdapat fitur yang membutuhkan citra karakter aksara Jawa tanpa melalui proses penipisan. Seperti yang digambarkan pada Gambar 3. 5., salah satu citra akan dilanjutkan ke proses penipisan sedangkan citra yang lain tidak.

#### *Inverse* Karakter Aksara Jawa

Proses ini dilakukan terhadap citra hasil penggandaan yang tidak akan ditipiskan. Proses ini dilakukan karena citra yang digunakan dalam ekstraksi fitur adalah citra yang memiliki piksel bernilai 0 untuk karakter aksara Jawa dan piksel bernilai 1 untuk *background*. Contoh sebuah citra karakter aksara Jawa yang tidak akan ditipiskan ketika sebelum dan sesudah proses *inverse* dapat dilihat pada Gambar 3. 6.

Gambar 3. 6 Citra hasil segmentasi (kiri) dan citra hasil *inverse* (kanan)

#### Penipisan Karakter Aksara Jawa

Proses penipisan ini menggunakan algoritma penipisan Guo-Hall seperti yang diterapkan pada MATLAB untuk penipisan citra biner atau hitam putih. Proses penipisan citra ini ditunjukkan oleh persamaan (3. 1):

|  |  |
| --- | --- |
|  | (3. 1) |

Dimana adalah citra biner karakter aksara Jawa dengan ketentuan piksel bernilai 1 untuk karakter aksara Jawa dan piksel bernilai 0 untuk background. Dari hasil proses penipisan, daerah citra yang tidak memiliki piksel bernilai 1 baik kolom maupun baris dihilangkan. Hal tersebut dilakukan untuk memperkecil daerah karakter aksara Jawa yang membuat proses ekstraksi fitur menjadi lebih cepat. Sehingga hasil akhir dari proses ini adalah daerah citra hasil penipisan kolom dimulai dari indeks terkiri yang memiliki piksel bernilai 1 (putih) dikurangi satu indeks (jika indeks terkiri tidak sama dengan nol) dan diakhiri indeks terkanan yang memiliki piksel bernilai 1 (putih) ditambah satu indeks (jika indeks terkanan tidak sama dengan jumlah kolom dikurangi satu). Sedangkan daerah baris dimulai dari indeks teratas yang memiliki piksel bernilai 1 (putih) dikurangi satu indeks (jika indeks teratas tidak sama dengan nol) dan diakhiri indeks terbawah yang memiliki piksel bernilai 1 (putih) ditambah satu indeks (jika indeks terbawah tidak sama dengan jumlah baris dikurangi satu).

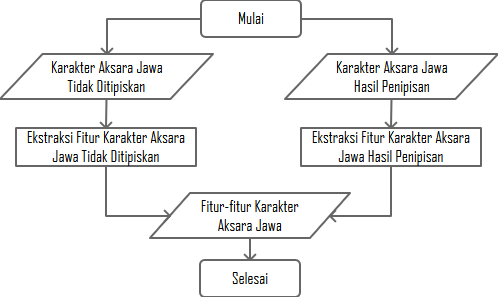
Contoh sebuah citra karakter aksara Jawa ketika sebelum dan sesudah proses penipisan (thinning) dapat dilihat pada Gambar 3. 7.

Gambar 3. 7 Citra hasil segmentasi (kiri) dan citra hasil penipisan (kanan)

### Tahap Ekstraksi Fitur Tulisan Aksara Jawa

Pada tahap ini terdapat dua proses yaitu proses ekstraksi fitur terhadap citra hasil proses penipisan dan ekstraksi fitur terhadap citra yang tidak ditipiskan. Diagram alir mengenai tahap ekstraksi fitur ini dapat dilihat pada **Error! Reference source not found.**8.



Gambar 3. 8 Diagram alir tahap ekstraksi fitur aksara Jawa

#### Ekstraksi Fitur Aksara Jawa Citra Tidak Ditipiskan

Terdapat beberapa fitur yang akan diekstraksi dari citra ini diantaranya jumlah area tertutup seluruh citra, jumlah area tertutup kiri, jumlah area tertutup kanan, jumlah area tertutup atas, jumlah area tertutup bawah dan jumlah karakter pada sebuah karakter aksara Jawa. Proses ekstraksi fitur ini menggunakan algoritma *region growing* dengan *seed* awal yang berbeda untuk setiap fitur. Proses *region growing* ini ditunjukkan dengan persamaan (3. 2):

|  |  |
| --- | --- |
|  | (3. 2) |

Dimana adalah citra biner karakter aksara Jawa dengan ketentuan piksel bernilai 0 untuk karakter aksara Jawa dan piksel bernilai 1 untuk *background*. Algoritma ini memiliki koordinat baris dan koordinat kolom sebagai *seed* awal yang akan menghasilkan sebuah daerah dimana piksel-pikselnya memiliki karakter sama dengan *seed* awal. Hasil dari *region growing* ini adalah citra biner dimana piksel bernilai 1 merupakan piksel *seed* awal dan piksel-piksel yang sama karakternya dengan *seed* awal dan sebaliknya piksel bernilai 0 adalah piksel yang berbeda karakternya dengan *seed* awal.

Untuk ekstraksi fitur jumlah area tertutup, koordinat *seed* awal yang digunakan adalah (0,0) yang bernilai 1 dan tidak dikelilingi oleh piksel-piksel bernilai nol. Sedangkan koordinat *seed* awal untuk ekstraksi fitur jumlah karakter adalah koordinat salah satu piksel bernilai nol. Hasil *region growing* untuk fitur area tertutup tidak hanya digunakan untuk fitur area tertutup seluruh citra tetapi digunakan juga untuk fitur area tertutup kiri, kanan, atas, dan bawah. Deskripsi tentang fitur yang diekstraksi dijelaskan dalam Tabel 3. 2.

Tabel 3. 2 Fitur-fitur Citra Tidak Ditipiskan

| No. | Nama Fitur | Penjelasan |
| --- | --- | --- |
| 1. | Area | Nilai fitur sama dengan 1 jika terdapat piksel bernilai 1 pada pengurangan citra awal dengan citra hasil *region growing* (koordinat *seed* awal (0,0)). Jika tidak, nilai fitur sama dengan nol. |
| 2. | AreaKiri | Nilai fitur sama dengan 1 jika terdapat piksel bernilai 1 pada pengurangan citra awal kiri dengan citra hasil *region growing* kiri (koordinat *seed* awal (0,0)). Jika tidak, nilai fitur sama dengan nol. |
| 3 | AreaKanan | Nilai fitur sama dengan 1 jika terdapat piksel bernilai 1 pada pengurangan citra awal kanan dengan citra hasil *region growing* kanan (koordinat *seed* awal (0,0)). Jika tidak, nilai fitur sama dengan nol. |
| 4 | AreaAtas | Nilai fitur sama dengan 1 jika terdapat piksel bernilai 1 pada pengurangan citra awal atas dengan citra hasil *region growing* atas (koordinat *seed* awal (0,0)). Jika tidak, nilai fitur sama dengan nol. |
| 5 | AreaBawah | Nilai fitur sama dengan 1 jika terdapat piksel bernilai 1 pada pengurangan citra awal bawah dengan citra hasil *region growing* bawah (koordinat *seed* awal (0,0)). Jika tidak, nilai fitur sama dengan nol. |
| 6 | Karakter | Nilai fitur sama dengan 2 jika terdapat piksel bernilai 1 pada pengurangan *inverse* citra awal dengan citra hasil *region growing* bawah (koordinat *seed* awal adalah salah satu koordinat piksel bernilai 0 pada citra awal). Jika tidak, nilai fitur sama dengan satu. |

Pembagian daerah citra baik daerah atas, bawah, kiri, maupun kanan memiliki aturan-aturan tersendiri. Bagian atas citra adalah 40 persen daerah teratas dan bagian bawah citra adalah 40 persen daerah terbawah. Pemilihan angka 40 persen ini merupakan hasil yang optimal yang didapatkan setelah uji coba terhadap angka 35, 40, 45, dan 50. Hal itu karena masih terdapat kesalahan hasil pada beberapa data aksara Ja saat percobaan angka 45 dan 50, sedangkan hasil fitur yang diperoleh dari angka 40 sama dengan fitur *groundtruth* dan untuk percobaan angka 35 memiliki hasil yang sama dengan percobaan angka 40.

Sedangkan pembagian daerah kiri dan kanan berdasarkan titik tengah dari *range* terpanjang jumlah piksel kolom lebih dari nol pada daerah 80 persen atau 30 persen terbawah citra. Pemilihan daerah 80 persen terbawah dikarenakan terdapat data karakter Na dan Ka yang memiliki *range* terpanjang jumlah piksel kolom lebih dari nol pada daerah 31-100 persen teratas citra. Sedangkan pemilihan angka 30 persen terbawah merupakan alternatif daerah jika *range* yang didapatkan pada 80 persen terbawah adalah daerah kesuluruhan karakter aksara Jawa. Selain itu, pemilihan daerah 30 persen terbawah merupakan daerah yang tidak memiliki piksel-piksel pembentuk area tertutup pada karakter Ja. Kondisi ini diterapkan terhadap karakter Ja atau karakter aksara Jawa yang tertulis sedikit miring.

Pencarian titik tengah pada *range* terpanjang jumlah piksel kolom lebih dari nol memiliki langkah-langkah sebagai berikut:

* Menjumlah piksel setiap kolom pada daerah 80 persen terbawah
* Mendapatkan indeks yang memiliki jumlah piksel kolom lebih dari nol
* Menghitung selisih antar indeks yang memiliki jumlah piksel kolom lebih dari nol dan menyimpannya dalam sebuah *list*.
* Mendapatkan indeks selisih yang lebih dari satu dan disimpan dalam sebuah *list*
* Jika indeks yang didapatkan hanya satu, ulangi langkah mulai langkah pertama dengan daerah 30 persen terbawah
* Mencari nilai terbesar dari selisih-selisih tersebut dan mendapatkan indeksnya
* Mendapatkan indeks kolom awal *range* terpanjang jumlah piksel kolom lebih dari nol dengan persamaan (3. 3):

(3. 3)

Dimana adalah indeks awal *range* terpanjang jumlah piksel kolom lebih dari nol yang nilainya didapatkan dari nilai *list* selisih indeks jumlah piksel kolom lebih dari nol yang bernilai lebih dari satu pada indeks selisih terbesar . Selanjutnya, nilai tersebut ditambah satu.

* Mendapatkan indeks kolom akhir *range* terpanjang jumlah piksel kolom lebih dari nol dengan persamaan (3. 4):

(3. 4)

Dimana adalah indeks akhir *range* terpanjang jumlah piksel kolom lebih dari nol yang nilainya didapatkan dari nilai *list* selisih indeks jumlah piksel kolom lebih dari nol yang bernilai lebih dari satu pada indeks selisih terbesar ditambah satu.

* Mendapatkan nilai tengah *range* terpanjang jumlah piksel kolom lebih dari nol dengan persamaan (3. 5):

(3. 5)

Dimana adalah indeks tengah yang dapat membagi karakter menjadi kiri dan kanan.

Setelah mendapatkan titik tengah, bagian kiri karakter aksara Jawa dimulai dari indeks kolom pertama sampai indeks tengah dan bagian kanan dimulai dari indeks tengah ditambah satu sampai indeks kolom terakhir.

#### Ekstraksi Fitur Aksara Jawa Citra Hasil Penipisan

Pada proses ini, ekstraksi fitur menggunakan citra hasil penipisan. Terdapat delapan fitur yang akan diekstraksi pada proses ini antara lain kaki, kaki kiri, kaki kanan, kaki vertikal, rata-rata daerah atas, jumlah daerah bawah, jarak antara baris terakhir dan piksel tidak nol terbawah, dan jumlah piksel kolom terbesar. Penjelasan fitur-fitur kaki yang akan diekstraksi dalam proses ini (kaki, kaki kiri, kaki kanan, dan kaki vertikal) terdapat dalam Tabel 3. 3.

Tabel 3. 3 Fitur-fitur Citra Hasil Penipisan Bagian 1

| No. | Nama Fitur | Penjelasan |
| --- | --- | --- |
| 1. | Kaki | Jumlah garis vertikal pada daerah 35-45 persen teratas seluruh citra hasil *thinning* dengan syarat jumlah piksel setiap kaki lebih dari sama dengan jumlah baris daerah. Setiap kaki dipisahkan oleh kolom dengan jumlah piksel kolom nol. |
| 2. | Kaki Kiri | Jumlah garis vertikal pada daerah 35-45 persen teratas bagian kiri citra hasil *thinning* dengan syarat jumlah piksel setiap kaki lebih dari sama dengan jumlah baris daerah. Setiap kaki dipisahkan oleh kolom dengan jumlah piksel kolom nol. |
| 3 | Kaki Kanan | Jumlah garis vertikal pada daerah 35-45 persen teratas bagian kanan citra hasil *thinning* dengan syarat jumlah piksel setiap kaki lebih dari sama dengan jumlah baris daerah. Setiap kaki dipisahkan oleh kolom dengan jumlah piksel kolom nol. |
| 4 | Kaki Vertikal | Jumlah garis horizontal pada daerah 35-45 persen dari kiri seluruh citra hasil *thinning*. Setiap kaki dipisahkan oleh baris dengan jumlah piksel baris nol. |

Pemilihan daerah 35-45 persen teratas untuk ekstraksi fitur kaki, kaki kiri, dan kaki kanan karena daerah ini merupakan daerah yang menghasilkan hasil yang optimal untuk semua data yang ada. Selain itu, pemilihan daerah itu karena pemilihan daerah yang lebih bawah seperti daerah 45-55 persen teratas, 55-65 persen teratas, dan 65-75 persen teratas menyebabkan kesalahan dalam penghitungan fitur terutama untuk karakter yang memiliki fitur area bawah sama dengan satu seperti karakter Na, Da, Ka, Sa, dan Ca. Pemilihan daerah 35-45 persen teratas juga disebabkan oleh kesalahan dalam ekstraksi fitur untuk daerah yang lebih atas seperti 15-25 persen teratas dan 25-35 persen teratas terutama pada karakter Nya, Ba, Tha, dan Nga.

Fitur kaki vertikal diekstraksi dengan tujuan untuk mempermudah membaca karakter Ma. Pemilihan daerah 35-45 persen dari bagian kiri citra juga menghasilkan hasil yang optimal untuk fitur ini. Hal tersebut dikarenakan pada daerah yang lebih kanan seperti 45-55 persen dari kiri dan seterusnya sebagian besar merupakan daerah bagian kanan dan nilai fiturnya sama dengan karakter-karakter lain. Selain itu, pada daerah yang lebih kiri seperti 25-35 persen dan seterusnya menghasilkan nilai fitur sama dengan karakter-karakter lain juga.

Hal-hal yang harus diperhatikan dalam penentuan jumlah kaki baik kaki seluruh citra, kaki kiri, kaki kanan, maupun kaki vertikal adalah sebagai berikut:

* Penghitungan calon kaki dimulai dari kolom terkiri, sedangkan kaki vertikal dimulai dari baris teratas pada citra dengan daerah yang telah ditentukan.
* Setiap calon kaki dipisahkan oleh kolom dengan jumlah piksel kolom nol, sedangkan untuk fitur kaki vertikal dipisahkan oleh baris dengan jumlah piksel baris nol.
* Nilai dari setiap calon kaki adalah hasil penjumlahan setiap jumlah piksel kolom selama jumlah piksel kolom tidak bernilai nol. Sedangkan untuk kaki vertikal menggunakan jumlah piksel baris.
* Jumlah calon kaki bertambah ketika penghitungannya mendapatkan kolom dengan jumlah piksel kolom tidak bernilai nol dengan syarat jumlah piksel kolom sebelumnya bernilai nol. Sedangkan untuk jumlah kaki vertikal menggunakan jumlah piksel baris.
* Dari hasil penghitungan nilai setiap calon kaki, jumlah kaki ditentukan berdasarkan banyaknya calon kaki yang memenuhi syarat. Sedangkan untuk kaki vertikal, semua calon kaki vertikal adalah banyaknya kaki vertikal tanpa ada syarat.

Pembagian daerah kiri dan kanan menggunakan *range* terpanjang jumlah piksel kolom lebih dari nol dengan daerah yang sama dengan pembagian fitur area kiri dan kanan. Cara pembagiannya juga sama dengan pembagian fitur area kiri dan kanan.

Empat fitur lainnya yaitu rata-rata daerah atas, jumlah daerah bawah, jarak antara baris terakhir dan piksel tidak nol terbawah, dan jumlah piksel kolom terbesar menggunakan sebuah daerah pengamatan atau *region of interest* (ROI). ROI yang digunakan adalah daerah yang dimulai pada kuartil pertama *range* terpanjang jumlah piksel kolom lebih dari nol sampai kuartil ketiganya. Penjelasan empat fitur ini terdapat dalam Tabel 3. 4.

Tabel 3. 4 Fitur-fitur Citra Hasil Penipisan Bagian 2

| No. | Nama Fitur | Penjelasan |
| --- | --- | --- |
| 1. | Rata-rata Daerah Atas | Rata-rata piksel daerah atas ROI yaitu 20 persen daerah teratas dengan *range* dimulai dari indeks tengah sampai indeks terakhirnya. Fitur ini digunakan untuk pembacaan karakter Sa dan Ca. |
| 2. | Jumlah Daerah Bawah | Jumlah piksel daerah bawah ROI yaitu 80 persen daerah terbawah. Fitur ini untuk membedakan karakter dengan 5 kaki dan jumlah karakter sama dengan satu. |
| 3 | Jarak antara baris terakhir dan piksel tidak nol terbawah | Jarak antara indeks baris terbawah tidak nol pada kolom kuartil pertama dengan baris terakhir. Tujuan ekstraksi fitur ini untuk memudahkan dalam membaca karakter Na dan Da. |
| 4 | Jumlah piksel kolom terbesar | Jumlah piksel kolom terbesar pada daerah yang digunakan untuk pembagian daerah kiri dan kanan dengan *range* sama dengan ROI. Tujuannya untuk memudahkan dalam membaca karakter Dha. |

Penentuan nilai kuartil pertama dan kuartil ketiga menggunakan persamaan (3. 6) dan (3. 7):

(3. 6)

(3. 7)

Dimana adalah indeks awal ROI dan merupakan indeks akhir ROI yang melibatkan indeks awal (lihat persamaan (3. 3)), indeks akhir (lihat persamaan (3. 4)), dan indeks tengah (lihat persamaan (3. 6)) pada daerah yang digunakan untuk pembagian daerah kiri dan kanan.

Daerah pengamatan ini merupakan daerah piksel-piksel penghubung antara bagian kiri dan kanan. Pemilihan *range* pada daerah pengamatan dikarenakan jika indeks awal ROI adalah indeks awal daerah yang digunakan untuk pembagian daerah kiri dan kanan, terdapat beberapa karakter memiliki indeks awal daerahnya termasuk bagian kiri sehingga tidak sesuai dengan daerah yang akan diamati. Begitu juga kondisi indeks akhir ROI terhadap indeks akhir daerah yang digunakan untuk pembagian daerah kiri dan kanan.

Penentuan daerah untuk ekstraksi empat fitur terakhir dipengaruhi oleh beberapa faktor. Berikut ini penjelasan faktor-faktor yang mempengaruhi untuk setiap fitur:

* Rata-rata Daerah Atas

Daerah 20 persen teratas pada ROI dengan *range* antara indeks tengah sampai indeks terakhirya memberikan hasil yang optimal untuk fitur ini. Pemilihan daerah ini dikarenakan daerah yang lebih atas (daerah 10 persen daerah teratas) memberikan kesalahan nilai pada karakter-karakter yang seharusnya. Selain itu, pemilihan *range* dari indeks tengah sampai akhir karena piksel-piksel yang dihitung merupakan piksel-piksel bagian kanan.

* Jumlah Daerah Bawah

Fitur ini menggunakan daerah 80 persen terbawah daerah ROI sebagai daerah yang menghasilkan nilai fitur optimal. Pemilihan daerah ini karena daerah yang lebih bawah seperti 30 persen daerah terbawah belum bisa membedakan karakter Ha dan Ta.

* Jarak antara Baris Terakhir dan Piksel Tidak Nol Terbawah

Fitur ini memberikan nilai optimal dengan menggunakan jarak piksel tidak nol terbawah pada kolom indeks pertama daerah ROI dengan baris terakhir. Sebelum menggunakan metode ini, fitur menggunakan jarak baris terakhir dengan baris yang memiliki jumlah piksel baris terbesar dalam daerah ROI tetapi hasilnya masih salah dalam beberapa data karakter Da. Pemilihan kolom pertama ini karena kolom pertama memiliki piksel tidak nol terbawah daripada kolom lain sehingga lebih mudah membedakan karakter Na dan Da.

* Jumlah Piksel Kolom Terbesar

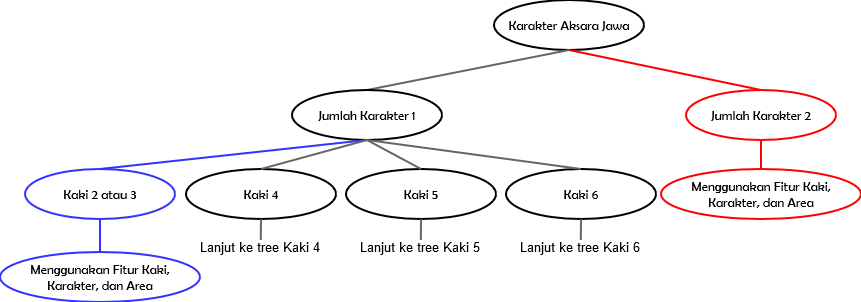
Pengambilan daerah untuk fitur ini adalah daerah yang digunakan untuk pembagian daerah kiri dan kanan pada *range* ROI karena daerah tersebut hanya menghitung jumlah piksel kolom daerah bawah tanpa melibatkan piksel-piksel daerah atas (20 persen daerah teratas) dimana daerah atas untuk karakter Wa dan Dha hampir sama. Sehingga pengambilan daerah ini memudahkan sistem dalam membaca karakter Wa dan Dha.

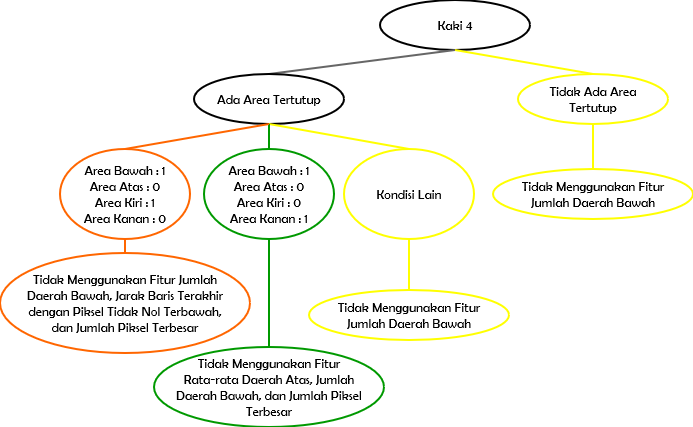
### Tahap Klasifikasi Karakter Aksara Jawa

Proses terakhir adalah klasifikasi karakter aksara Jawa pada citra masukan dengan fitur-fitur dari proses ekstraksi fitur. Fitur-fitur yang digunakan berbeda untuk setiap kelompok karakter bergantung pada kondisi fitur lain. Fitur yang tidak digunakan diubah nilainya menjadi nol. Berikut ini asumsi-asumsi penggunaan fitur yang dipakai untuk klasifikasi karakter aksara Jawa:

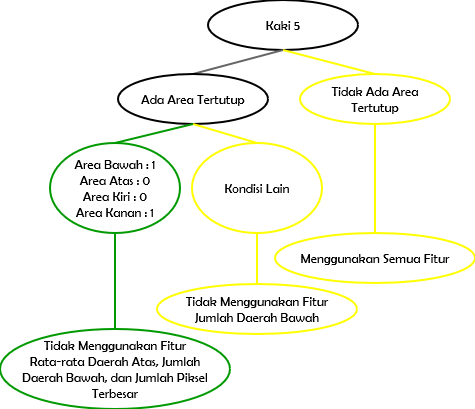
1. Jika jumlah karakter dua, maka fitur yang **digunakan** adalah kaki, karakter, dan area.
2. Jika jumlah karakter satu dan kaki berjumlah 2,3, atau 6 tanpa ada area tertutup, maka fitur yang **digunakan** adalah kaki, karakter, dan area.
3. Jika jumlah karakter satu dan jumlah area satu dengan ketentuan nilai area atas nol, area bawah satu, area kiri satu, dan area kanan nol, maka fitur yang **tidak** **digunakan** adalah jumlah daerah bawah, jarak baris terakhir dan piksel tidak nol terbawah, dan jumlah piksel terbesar.
4. Jika jumlah karakter satu dan jumlah area satu dengan ketentuan nilai area atas nol, area bawah satu, area kiri nol, dan area kanan satu, maka fitur yang **tidak** **digunakan** adalah rata-rata daerah atas, jumlah daerah bawah, dan jumlah piksel terbesar.
5. Jika jumlah karakter satu, jumlah kaki sama dengan lima, tanpa ada area tertutup maka semua fitur **digunakan**. Sedangkan karakter lainnya **tidak** **menggunakan** jumlah daerah bawah.

**Error! Reference source not found.**. 9 merupakan *decision tree* yang menjelaskan asumsi-asumsi penggunaan fitur yang digunakan dalam proses klasifikasi secara umum. *Decision tree* tersebut memiliki beberapa cabang dimana cabang-cabang tersebut menjelaskan asumsi-asumsi penggunaan fitur. Cabang warna merah merupakan asumsi 1, cabang warna biru merupakan asumsi 2, cabang warna jingga merupakan asumsi 3, cabang warna hijau merupakan asumsi 4, dan cabang warna kuning merupakan asumsi 5.

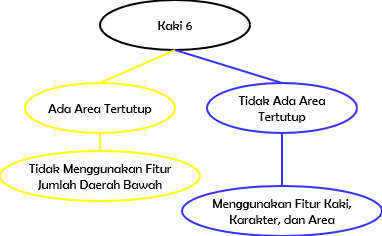
**(a) *decision tree* secara umum**



**(b) *decision tree* kaki 4**



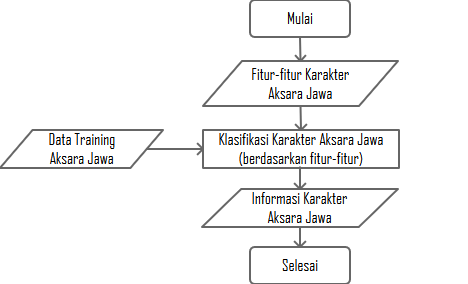
**(c) *decision tree* kaki 5**



**(d) *decision tree* kaki 6**

Gambar 3. 9 *Decision tree* asumsi penggunaan fitur

Data pelatihan yang digunakan berjumlah 240 data dengan 12 data setiap karakter aksara Jawa. Data pelatihan berupa citra biner dengan ukuran 100 x 100 piksel. Diagram alir mengenai tahap ini dapat dilihat pada Gambar 3. 10.



Gambar 3. 10 Diagram alir tahap klasifikasi

#### Klasifikasi Karakter Aksara Jawa Berdasarkan Fitur

Proses klasifikasi ini menggunakan algoritma klasifikasi *K-Nearest Neighbor* dengan nilai K sama dengan 4(empat). Pemilihan nilai K ini berdasarkan uji coba yang dilakukan **(penjelasan lanjut di BAB V)** terhadap nilai K dengan rentang nilai K mulai dari 1 sampai 10. Hasil akhir dari proses ini adalah informasi tentang setiap karakter aksara Jawa pada citra masukan.

## Perancangan Antarmuka Perangkat Lunak

Pada subbab ini akan dibahas mengenai perancangan antarmuka perangkat lunak yang bertujuan untuk dapat mempermudah interaksi antara perangkat lunak dengan pengguna. Sistem ini memiliki beberapa halaman antara lain halaman menu, halaman tulis aksara, halaman foto aksara, halaman info aplikasi, dan halaman klasifikasi.

### Halaman Menu

Halaman ini merupakan halaman yang akan muncul saat sistem dijalankan. Terdapat beberapa menu pada halaman ini antara lain menu menulis aksara, menu foto aksara, menu info aplikasi, dan pilihan keluar. Selain itu, terdapat juga gambar karakter aksara Jawa dasar agar pengguna mengetahui bagaimana bentuk karakter aksara Jawa. Untuk rancangan halaman menu uji coba dapat dilihat pada Gambar 3. 11. Kemudian untuk spesifikasi atribut antarmuka halaman menu ini ditunjukkan oleh Tabel 3. 5.

### Halaman Tulis Aksara

Halaman ini adalah halaman yang digunakan pengguna untuk menuliskan karakter aksara Jawa pada layar *smartphone*. Hasil tulisan tersebut akan disimpan sebagai matriks citra dan akan diproses lebih lanjut pada halaman klasifikasi. Pada halaman ini, terdapat dua pilihan yaitu pilihan untuk menghapus tulisan secara keseluruhan dan pilihan untuk menyimpan tulisan dan melanjutkan ke halaman klasifikasi. Untuk rancangan halaman tulis aksara dapat dilihat pada Gambar 3. 12. Kemudian untuk spesifikasi atribut antarmuka halaman tulis aksara ini ditunjukkan oleh Tabel 3. 6.



Gambar 3. 11 Rancangan halaman menu

Tabel 3. 5 Spesifikasi atribut antarmuka halaman menu

| No | Nama Atribut Antarmuka | Jenis Atribut | Kegunaan | Jenis Masukan / Keluaran |
| --- | --- | --- | --- | --- |
| 1 | karakter\_aksara | Image View | Menampilkan bentuk karakter aksara Jawa | *Image* |
| 2 | button\_tulisAksara | Button | Pilihan aksi untuk masuk ke halaman tulis aksara | *Event Clicked* |
| 3 | button\_fotoAksara | Button | Pilihan aksi untuk masuk ke halaman foto aksara | *Event Clicked* |
| 4 | button\_info | Button | Pilihan aksi untuk masuk ke halaman info aplikasi | *Event Clicked* |
| 5 | button\_keluar | Button | Pilihan aksi untuk keluar dari aplikasi | *Event Clicked* |



Gambar 3. 12 Rancangan halaman tulis aksara

Tabel 3. 6 Spesifikasi atribut antarmuka halaman tulis aksara

| No | Nama Atribut Antarmuka | Jenis Atribut | Kegunaan | Jenis Masukan / Keluaran |
| --- | --- | --- | --- | --- |
| 1 | *fingerpaint*\_  aksaraJawa | Layout | Tempat menulis karakter aksara Jawa | *Image Matrix* |
| 2 | button\_Hapus | Button | Pilihan aksi untuk menghapus semua tulisan | *Event Clicked* |
| 3 | button\_Simpan | Button | Pilihan aksi untuk menyimpan gambar dan lanjut ke halaman klasifikasi | *Event Clicked* |

### Halaman Foto Aksara

Halaman ini adalah halaman yang digunakan pengguna untuk memasukkan citra karakter aksara Jawa menggunakan kamera utama pada *smartphone*. Berdasarkan penjelasan pada batasan masalah, objek yang diambil gambarnya adalah citra karakter aksara Jawa dengan ketentuan *background* putih, tulisan aksara Jawa hitam menggunakan spidol, serta hampir tidak ada bayangan saat mengambil gambar. Langkah yang dilakukan untuk mengambil gambar adalah dengan menyentuh layar *smartphone*.

Setelah pengguna berhasil mengambil gambar, gambar tersebut akan disimpan dan sistem akan melanjutkan ke halaman klasifikasi untuk proses selanjutnya. Untuk rancangan halaman foto aksara dapat dilihat pada Gambar 3. 13. Kemudian untuk spesifikasi atribut antarmuka halaman foto aksara ini ditunjukkan oleh Tabel 3. 7.



Gambar 3. 13 Rancangan halaman foto aksara

Tabel 3. 7 Spesifikasi atribut antarmuka halaman foto aksara

| No | Nama Atribut Antarmuka | Jenis Atribut | Kegunaan | Jenis Masukan / Keluaran |
| --- | --- | --- | --- | --- |
| 1 | fotoAksara\_  activity | Java Camera View | Menampilkan data masukan citra aksara Jawa | *Image Matrix* |

### Halaman Info Aplikasi

Halaman ini memiliki fungsi untuk memberikan info atau cara menggunakan halaman-halaman pada aplikasi ini. Pada halaman ini, terdapat tiga info penggunaan halaman yaitu halaman tulis aksara, halaman foto aksara, dan halaman klasifikasi. Pada info halaman tulis aksara dan foto aksara, terdapat pilihan untuk melanjutkan ke info selanjutnya. Sedangkan pada info halaman aplikasi, terdapat pilihan untuk kembali ke halaman menu. Rancangan antarmuka halaman info aplikasi dapat dilihat pada Gambar 3. 14. Selain itu, spesifikasi atribut pada halaman ini ditunjukkan oleh Tabel 3. 8.

### Halaman Klasifikasi

Halaman ini adalah halaman utama dari aplikasi ini. Halaman ini menampilkan info hasil klasifikasi karakter aksara Jawa pada citra masukan. Pada halaman ini, terdapat pilihan segmentasi yang berfungsi untuk menampilkan hasil segmentasi citra masukan dan pilihan klasifikasi yang berfungsi untuk menampilkan hasil klasifikasi dan citra biner dari citra masukan. Rancangan antarmuka dari halaman klasifikasi ini dapat dilihat pada Gambar 3. 15. Sedangkan untuk spesifikasi atribut antarmuka halaman ini ditunjukkan oleh Tabel 3. 9.



Gambar 3. 14 Rancangan halaman info aplikasi

Tabel 3. 8 Spesifikasi atribut antarmuka halaman info aplikasi

| No | Nama Atribut Antarmuka | Jenis Atribut | Kegunaan | Jenis Masukan / Keluaran |
| --- | --- | --- | --- | --- |
| 1 | btn\_nextIcon | Button | Pilihan untuk lanjut info berikutnya | *Event Clicked* |
| 2 | btn\_homeIcon | Button | Pilihan untuk kembali ke halaman menu | *Event Clicked* |



Gambar 3. 15 Rancangan halaman klasifikasi

Tabel 3. 9 Spesifikasi atribut antarmuka halaman klasifikasi

| No | Nama Atribut Antarmuka | Jenis Atribut | Kegunaan | Jenis Masukan / Keluaran |
| --- | --- | --- | --- | --- |
| 1 | btn\_segmentasi | Button | Pilihan untuk menampilkan hasil segmentasi | *Event Clicked* |
| 2 | btn\_klasifikasi | Button | Pilihan untuk menampilkan hasil klasifikasi | *Event Clicked* |
| 3 | Image\_hasilProses | Image View | Menampilkan citra hasil proses (citra masukan, segmentasi, dan klasifikasi) | *Image Matrix* |
| 4 | info\_aksara | Text View | Menampilkan info karakter aksara Jawa pada citra masukan | *String* |
| 5 | btn\_backIcon | Button | Pilihan aksi untuk kembali ke halaman menu | *Event Clicked* |

*[Halaman ini sengaja dikosongkan]*

# BAB IV IMPLEMENTASI

Bab ini membahas implementasi dari perancangan sistem sesuai dengan perancangan yang telah dibuat. Bahasa pemrograman yang digunakan untuk implementasi sistem adalah bahasa pemrograman Java dengan pustaka OpenCV Android.



## Lingkungan Implementasi

Lingkungan implementasi sistem yang digunakan untuk mengembangkan tugas akhir memiliki spesifikasi perangkat keras dan perangkat lunak seperti yang ditampilkan pada Tabel 4. 1.

Tabel 4. 1 Lingkungan Implementasi Sistem

|  |  |
| --- | --- |
| Perangkat | Spesifikasi |
| Perangkat keras | Prosesor: Intel® Core™ i3-2370M CPU @ 2.40GHz, ~2.4GHz  Memori: 4096 MB |
| Perangkat lunak | Sistem Operasi: Microsoft Windows 8.1 Pro 64-bit  Perangkat Pengembang: IDE Eclipse Juno 4.2, MATLAB R2014a  Perangkat Pembantu: Notepad++, Microsoft Excel 2013, Microsoft Word 2013, Adobe Photoshop CS6, Evolus Pencil 2.0.3, Weka 3.6.9 |

## Implementasi Proses

Implementasi proses dilakukan berdasarkan perancangan proses yang sudah dijelaskan pada bab analisis dan perancangan.

### Implementasi Tahap Deteksi dan Segmentasi Tulisan Aksara Jawa

Subbab ini membahas implementasi tahap deteksi dan segmentasi tulisan aksara Jawa. Implementasi tahap ini dilakukan dengan pengubahan citra ke *channel* hitam putih, melakukan *inverse* citra hitam putih*,* dan segmentasi citra hasil *inverse*. Hasil dari tahap ini adalah citra hasil segmentasi dengan piksel putih untuk karakter aksara Jawa dan piksel hitam untuk *background* sehingga citra ini lebih memudahkan tahap berikutnya. Tahap deteksi dan segmentasi tulisan aksara Jawa terdiri dari beberapa proses seperti yang telah dijelaskan pada subbab 3.3.1. Berikut ini adalah implementasi proses pengubahan citra masukan ke *channel* hitam putih pada sistemyang ditunjukkan oleh Gambar 4. 1.

|  |
| --- |
| //convert to grayscale  Mat grayscaleMat **=** **new** Mat**(**imgMat**.**size**(),**CvType**.**CV\_8U**);**  Imgproc**.**cvtColor**(**imgMat**,** grayscaleMat**,** Imgproc**.**COLOR\_RGB2GRAY**);**  //convert to binary  Mat binaryMat **=** **new** Mat**(**grayscaleMat**.**size**(),**grayscaleMat**.**type**());**  Imgproc**.**threshold**(**grayscaleMat**,** binaryMat**,** 100**,** 255**,** Imgproc**.**THRESH\_BINARY**);** |

Gambar 4. 1 Kode implementasi proses pengubahan citra ke *channel* hitam putih

Setelah diubah menjadi citra biner, langkah selanjutnya adalah melakukan *inverse* terhadap citra biner tersebut untuk mempermudah proses segmentasi. Berikut ini adalah implementasi proses *inverse* citra biner pada sistemyang ditunjukkan oleh Gambar 4. 12.

|  |
| --- |
| Mat inverseBinary **=** **new** Mat**(**t**.**size**(),**t**.**type**());**  Core**.**bitwise\_not**(**t**,** inverseBinary**);**  //merubah piksel menjadi 0 dan 1  Mat binary10 **=** **new** Mat**(**inverseBinary**.**size**(),**CvType**.**CV\_32F**);**  inverseBinary**.**convertTo**(**binary10**,** CvType**.**CV\_32F**,** 1.**/**255**);** |

Gambar 4. 2 Kode implementasi proses *inverse* citra biner

Langkah terakhir dari proses ini adalah segmentasi citra hasil *inverse*. Untuk mempercepat proses pada sistem baik segmentasi maupun proses lainnya, citra *inverse* tersebut dipetakan pada *array boolean* dua dimensi. Implementasi pada sistem untuk langkah ini ditunjukkan pada Gambar 4. 13.

|  |
| --- |
| //perubahan ke bentuk logical  boolean**[][]** img1 **=** **new** boolean**[**binary10**.**rows**()][**binary10**.**cols**()];**  int row **=** binary10**.**rows**();**  int col **=** binary10**.**cols**();**  **for** **(**int i **=** 0**;** i **<=** row**-**1**;** i**++)** **{**  **for** **(**int j **=** 0**;** j **<=** col**-**1**;** j**++)** **{**  **if** **(**binary10**.**get**(**i**,** j**)[**0**]** **==** 0.0**)** img1**[**i**][**j**]** **=** **false;**  **else** img1**[**i**][**j**]** **=** **true;**  **}**  **}** |

Gambar 4. 3 Kode implementasi proses pengubahan citra ke *array boolean* dua dimensi

Seperti penjelasan subbab 3.3.1.3, proses segmentasi ini terdiri dari beberapa langkah. Langkah pertama dan kedua adalah menghitung jumlah piksel kolom setiap kolom pada citra dan mendapatkan indeks kolom yang merepresentasikan karakter aksara Jawa. Implementasi pada sistem untuk proses penghitungan piksel kolom setiap kolom pada citra ditunjukkan oleh Gambar 4. 14. Sedangkan implementasi proses mendapatkan indeks kolom karakter aksara Jawa ditunjukkan oleh Gambar 4. 15.

|  |
| --- |
| int jmlPixelColom **[]** **=** **new** int **[**binary10**.**cols**()];**  **for(**int i**=**0**;**i**<**binary10**.**cols**();**i**++){**  int jml **=** 0**;**  **for(**int j**=**0**;**j**<**binary10**.**rows**();**j**++){**  jml**+=(**img1**[**j**][**i**]?**1**:**0**);**  **}**  jmlPixelColom**[**i**]=**jml**;**  **)** |

**Gambar 4. 4 Kode implementasi proses penghitungan piksel kolom setiap kolom**

|  |
| --- |
| ArrayList**<**ArrayList**<**Integer**>>**segment **=** **new** ArrayList**<**ArrayList**<**Integer**>>();**  ArrayList**<**Integer**>**pixelPos **=** **new** ArrayList**<**Integer**>();**  **for(**int i**=**0**;**i**<**binary10**.**cols**();**i**++){**  **if** **(**jmlPixelColom**[**i**]** **!=** 0**){**  pixelPos**.**add**(**i**);**  **if(**i **==** binary10**.**cols**()-**1**){**  ArrayList**<**Integer**>**posCopy **=** **new** ArrayList**<**Integer**>();**  posCopy **=** **(**ArrayList**<**Integer**>)** pixelPos**.**clone**();**  segment**.**add**(**posCopy**);**  pixelPos**.**clear**();**  **}**  **}**  **else{**  **if(!**pixelPos**.**isEmpty**()){**  ArrayList**<**Integer**>**posCopy **=** **new** ArrayList**<**Integer**>();**  posCopy **=** **(**ArrayList**<**Integer**>)** pixelPos**.**clone**();**  segment**.**add**(**posCopy**);**  pixelPos**.**clear**();**  **}**  **}**  **}** |

**Gambar 4. 5 Kode implementasi proses pengambilan indeks kolom karakter aksara Jawa**

Setelah mendapatkan indeks kolomnya, langkah berikutnya yaitu menghitung jumlah piksel baris setiap baris dengan *range* indeks pertama sampai indeks terakhir kolom setiap karakter aksara Jawa yang dilanjutkan dengan mendapatkan indeks baris yang merepresentasikan karakter aksara Jawa. Implementasi pada sistem untuk langkah-langkah ini ditunjukkan oleh Gambar 4. 16.

|  |
| --- |
| ArrayList**<**ArrayList**<**Integer**>>**segmentBaris **=** **new** ArrayList**<**ArrayList**<**Integer**>>();**  ArrayList**<**Integer**>**pixelPosBaris **=** **new** ArrayList**<**Integer**>();**  **for(**int i**=**0**;**i**<**segment**.**size**();**i**++){**  **for(**int j**=**0**;**j**<**binary10**.**rows**();**j**++){**  int jmlBaris **=** 0**;**  **for(**int k**=**segment**.**get**(**i**).**get**(**0**);**k**<=**segment**.**get**(**i**).**get**(**segment**.**get**(**i**).**size**()-**1**);**k**++){**  jmlBaris**+=(**img1**[**j**][**k**]?**1**:**0**);**  **}**  **if** **(**jmlBaris **!=** 0**){**  pixelPosBaris**.**add**(**j**);**  **if(**j **==** binary10**.**rows**()-**1**){**  ArrayList**<**Integer**>**posCopy **=** **new** ArrayList**<**Integer**>();**  posCopy **=** **(**ArrayList**<**Integer**>)** pixelPosBaris**.**clone**();**  segmentBaris**.**add**(**posCopy**);**  pixelPosBaris**.**clear**();**  **}**  **}**  **else{**  **if(!**pixelPosBaris**.**isEmpty**()){**  ArrayList**<**Integer**>**posCopy **=** **new** ArrayList**<**Integer**>();**  posCopy **=** **(**ArrayList**<**Integer**>)** pixelPosBaris**.**clone**();**  segmentBaris**.**add**(**posCopy**);**  pixelPosBaris**.**clear**();**  **}**  **}**  **}**  **}** |

**Gambar 4. 6 Kode implementasi proses penghitungan jumlah piksel baris pada *range* karakter aksara Jawa dan pengambilan indeks baris karakter aksara Jawa**

Setelah mendapatkan indeks barisnya, hasil yang diperoleh adalah daerah setiap karakter aksara Jawa. Langkah terakhir adalah menambahkan 2 baris di bagian atas dan bawah serta 2 kolom di sebelah kiri dan kanan daerah karakter aksara Jawa. Hasilnya disimpan dalam sebuah *list* untuk diproses pada tahap berikutnya. Implementasi pada sistem untuk langkah-langkah ini ditunjukkan oleh Gambar 4. 17.

|  |
| --- |
| for(int i =0;i<segment.size();i++){ |
| Mat huruf = binary10.submat(segmentBaris.get(i).get(0), segmentBaris.get(i).get(segmentBaris.get(i).size()-1), segment.get(i).get(0), segment.get(i).get(segment.get(i).size()-1)); |
|  |
| //insert into Mat list |
| int baris = huruf.rows(); |
| int kolom = huruf.cols(); |
| int baris1 = baris + 4, kolom1 = kolom + 4; |
| Mat finals = new Mat(baris1,kolom1,binary10.type()); |
| boolean[][] hurufFinal = new boolean[baris1][kolom1]; |
| boolean[][] huruf1 = new boolean[baris][kolom]; |
|  |
| //perubahan ke nilai boolean |
| for (int j = 0; j <=baris1-1 ; j++) { |
| for (int k = 0; k <= kolom1-1; k++) { |
| hurufFinal[j][k]=false;  **}**  **}**  **for** **(**int j **=** 0**;** j **<=** huruf**.**rows**()-**1**;** j**++)** **{**  **for** **(**int k **=** 0**;** k **<=** huruf**.**cols**()-**1**;** k**++)** **{**  **if** **(**huruf**.**get**(**j**,** k**)[**0**]** **==** 0.0**)**  huruf1**[**j**][**k**]** **=** **false;**  **else** huruf1**[**j**][**k**]** **=** **true;**  **}**  **}**  **for** **(**int j **=** 0**;** j **<=**baris**-**1 **;** j**++)** **{**  **for** **(**int k **=** 0**;** k **<=** kolom**-**1**;** k**++)** **{**  hurufFinal**[**j**+**2**][**k**+**2**]=**huruf1**[**j**][**k**];**  **}**  **}**  **for** **(**int j **=** 0**;** j **<=**baris1**-**1 **;** j**++)** **{**  **for** **(**int k **=** 0**;** k **<=** kolom1**-**1**;** k**++)** **{**  **if(**hurufFinal**[**j**][**k**]==false)**  finals**.**put**(**j**,** k**,** 0.0**);**  **else**  finals**.**put**(**j**,** k**,** 1.0**);**  **}**  **}**  aksaraList**.**add**(**finals**);**  **}** |

**Gambar 4. 7 Kode implementasi proses pengambilan daerah karakter aksara Jawa dan penyimpanan hasil segmentasi ke dalam sebuah *list***

### Implementasi Tahap Penipisan (*Thinning*) Tulisan Aksara Jawa

Subbab ini membahas implementasi tahap penipisan (*thinning*) tulisan aksara Jawa. Implementasi pada tahap ini terdiri dari penggandaan citra karakter aksara Jawa dan proses penipisan citra karakter aksara Jawa. Hasil dari proses ini adalah citra hasil proses penggandaan dan citra hasil penipisan (*thinning*). Berikut ini implementasi dari proses penggandaan citra karakter aksara Jawa yang ditunjukkan oleh Gambar 4. 18.

|  |
| --- |
| Mat bwKarakter = im.clone(); |

**Gambar 4. 8 Kode implementasi proses penggandaan citra karakter aksara Jawa**

Setelah mendapatkan hasil penggandaan citra, salah satu citra akan ditipiskan dengan algoritma Guo-Hall seperti yang dijelaskan pada subbab 3.3.2.2. Sebelum memulai proses penipisan, langkah yang harus dilakukan adalah mengubah citra ke *array boolean* dua dimensi agar mempercepat kerja sistem. Implementasi langkah ini ditunjukkan pada Gambar 4. 19.

|  |
| --- |
| boolean**[][]** img **=** **new** boolean**[**im**.**rows**()][**im**.**cols**()];**  ArrayList**<**ArrayList**<**Integer**>>**index **=** **new** ArrayList**<**ArrayList**<**Integer**>>();**  int row **=** im**.**rows**();**  int col **=** im**.**cols**();**  **for** **(**int i **=** 0**;** i **<=** im**.**rows**()-**1**;** i**++)** **{**  **for** **(**int j **=** 0**;** j **<=** im**.**cols**()-**1**;** j**++)** **{**  **if** **(**im**.**get**(**i**,** j**)[**0**]** **==** 0.0**)** img**[**i**][**j**]** **=** **false;**  **else** img**[**i**][**j**]** **=** **true;**  **}**  **}** |

**Gambar 4. 9 Kode implementasi proses pengubahan citra ke *array boolean* dua dimensi proses *thinning***

Seperti penjelasan pada subbab 2.5, algoritma penipisan ini mempunyai dua sub iterasi pada setiap iterasinya yang diulangi sampai tidak ada piksel yang perlu dihapus. Implementasi algoritma *thinning* untuk sub iterasi pertama ditunjukkan pada Gambar 4. 10. Sedangkan sub iterasi keduanya ditunjukkan pada Gambar 4. 11.

|  |
| --- |
| **while** **(true)** **{**  boolean stop **=** **true;**  **for** **(**int i **=** 1**;** i **<=** row**-**2**;** i**++)** **{**  **for** **(**int j **=** 1**;** j **<=** col**-**2**;** j**++)** **{**  **if** **(!**img**[**i**][**j**])** **continue;**  boolean**[]** pixelTetangga **=** **new** boolean**[**8**];**  pixelTetangga**[**0**]** **=** img**[**i**][**j**+**1**];**  pixelTetangga**[**1**]** **=** img**[**i**+**1**][**j**+**1**];**  pixelTetangga**[**2**]** **=** img**[**i**+**1**][**j**];**  pixelTetangga**[**3**]** **=** img**[**i**+**1**][**j**-**1**];**  pixelTetangga**[**4**]** **=** img**[**i**][**j**-**1**];**  pixelTetangga**[**5**]** **=** img**[**i**-**1**][**j**-**1**];**  pixelTetangga**[**6**]** **=** img**[**i**-**1**][**j**];**  pixelTetangga**[**7**]** **=** img**[**i**-**1**][**j**+**1**];**  int A **=** 0**,** B1 **=** 0**,** B2 **=** 0**;**  **for** **(**int k **=** 1**;** k **<=**4**;** k**++)** **{**  int b **=** 0**;**  **if(!**pixelTetangga**[(**2**\***k**-**1**-**1**)%**8**]** **&&(**pixelTetangga**[(**2**\***k**-**1**)%**8**]** **||** pixelTetangga**[(**2**\***k**+**1**-**1**)%**8**]))**  b **=** 1**;**  A **+=** b**;**  B1**+=((**pixelTetangga**[(**2**\***k**-**1**-**1**)%**8**]** **|** pixelTetangga**[(**2**\***k**-**1**)%**8**])** **?** 1 **:** 0**);**  B2**+=((**pixelTetangga**[(**2**\***k**-**1**)%**8**]** **|** pixelTetangga**[(**2**\***k**+**1**-**1**)%**8**])** **?** 1 **:** 0**);**  **}**  int m1 **=** **((**pixelTetangga**[**1**]** **||** pixelTetangga**[**2**]** **||** **!**pixelTetangga**[**7**])** **&&** pixelTetangga**[**0**])** **?** 1 **:** 0**;**  **if** **(**A **==** 1 **&&** **(**Math**.**min**(**B1**,** B2**)** **==** 2 **||** Math**.**min**(**B1**,** B2**)** **==** 3**)** **&&** m1 **==** 0**)** **{**  ArrayList**<**Integer**>**index1 **=** **new** ArrayList**<**Integer**>();**  index1**.**add**(**i**);** index1**.**add**(**j**);**  index**.**add**(**index1**);**  stop **=** **false;**  **}**  **}**  **}**  **for(**int i**=**0**;**i**<**index**.**size**();**i**++){** img**[**index**.**get**(**i**).**get**(**0**)][**index**.**get**(**i**).**get**(**1**)]=false;**  **}**  index**.**clear**();**  //kode subiterasi kedua pada Gambar 4.10.  **if** **(**stop**)** **break;**  **}** |

**Gambar 4. 10 Kode implementasi algoritma *thinning* subiterasi pertama**

|  |
| --- |
| **for** **(**int i **=** 1**;** i **<=** row**-**2**;** i**++)** **{**  **for** **(**int j **=** 1**;** j **<=** col**-**2**;** j**++)** **{**  **if** **(!**img**[**i**][**j**])** **continue;**  boolean**[]** pixelTetangga **=** **new** boolean**[**8**];**  pixelTetangga**[**0**]** **=** img**[**i**][**j**+**1**];**  pixelTetangga**[**1**]** **=** img**[**i**+**1**][**j**+**1**];**  pixelTetangga**[**2**]** **=** img**[**i**+**1**][**j**];**  pixelTetangga**[**3**]** **=** img**[**i**+**1**][**j**-**1**];**  pixelTetangga**[**4**]** **=** img**[**i**][**j**-**1**];**  pixelTetangga**[**5**]** **=** img**[**i**-**1**][**j**-**1**];**  pixelTetangga**[**6**]** **=** img**[**i**-**1**][**j**];**  pixelTetangga**[**7**]** **=** img**[**i**-**1**][**j**+**1**];**  int A **=** 0**,** B1 **=** 0**,** B2 **=** 0**;**  **for** **(**int k **=** 1**;** k **<=**4**;** k**++)** **{**  int b **=** 0**;**  **if(!**pixelTetangga**[(**2**\***k**-**1**-**1**)%**8**]** **&&** **(**pixelTetangga**[(**2**\***k**-**1**)%**8**]** **||** pixelTetangga**[(**2**\***k**+**1**-**1**)%**8**]))**  b **=** 1**;**  A **+=** b**;**  B1**+=((**pixelTetangga**[(**2**\***k**-**1**-**1**)%**8**]** **|** pixelTetangga**[(**2**\***k**-**1**)%**8**])** **?** 1 **:** 0**);**  B2**+=((**pixelTetangga**[(**2**\***k**-**1**)%**8**]** **|** pixelTetangga**[(**2**\***k**+**1**-**1**)%**8**])** **?** 1 **:** 0**);**  **}**  int m1 **=** **((**pixelTetangga**[**5**]** **||** pixelTetangga**[**6**]** **||** **!**pixelTetangga**[**3**])** **&&** pixelTetangga**[**4**])** **?** 1 **:** 0**;**    **if** **(**A **==** 1 **&&** **(**Math**.**min**(**B1**,** B2**)** **==** 2 **||** Math**.**min**(**B1**,** B2**)** **==** 3**)** **&&** m1 **==** 0**)** **{**  ArrayList**<**Integer**>**index1 **=** **new** ArrayList**<**Integer**>();**  index1**.**add**(**i**);** index1**.**add**(**j**);**  index**.**add**(**index1**);**  stop **=** **false;**  **}**  **}**  **}**  **for(**int i**=**0**;**i**<**index**.**size**();**i**++){**  img**[**index**.**get**(**i**).**get**(**0**)][**index**.**get**(**i**).**get**(**1**)]=false;** **}**  index**.**clear**();** |

**Gambar 4. 11 Kode implementasi algoritma *thinning* subiterasi kedua**

Setelah mendapatkan hasil *thinning*, citra dalam bentuk *array boolean* dua dimensi diubah kembali ke dalam bentuk matriks citra. Implementasi langkah ini ditunjukkan pada Gambar 4. 12.

|  |
| --- |
| **for** **(**int i **=** 0**;** i **<=** row**-**1**;** i**++)** **{**  **for** **(**int j **=** 0**;** j **<=** col**-**1**;** j**++)** **{**  **if** **(**img**[**i**][**j**]){**  // im isi putih  im**.**put**(**i**,** j**,** 1.0**);**  **}**  **else{**  // im isi hitam  im**.**put**(**i**,** j**,** 0.0**);**  **}**  **}**  **}**  im**.**convertTo**(**im**,** CvType**.**CV\_8U**,** 255**);** |

**Gambar 4. 12 Kode implementasi proses pengubahan *array boolean* dua dimensi ke matriks citra**

Citra hasil *thinning* akan diperkecil dengan membuang baris dan kolom terluar yang memiliki jumlah piksel baris atau kolom bernilai nol. Implementasi proses tersebut ditunjukkan oleh Gambar 4. 13.

|  |
| --- |
| protected void thinSegmen**(**Mat im**){**  im**.**convertTo**(**im**,** CvType**.**CV\_32F**,** 1.**/**255**);**  int minX **=** 9999999**,** maxX **=** **-**1**,** minY **=** 9999999**,** maxY **=** **-**1**,** iter **=** 0**;**  **for(**int i**=**0**;**i**<=**im**.**rows**()-**1**;**i**++){**  **for(**int j**=**0**;**j**<=**im**.**cols**()-**1**;**j**++){**  **if(**im**.**get**(**i**,** j**)[**0**]==**0.0**){**  minX **=** Math**.**min**(**i**,** minX**);**  maxX **=** Math**.**max**(**i**,** maxX**);**  minY **=** Math**.**min**(**j**,** minY**);**  maxY **=** Math**.**max**(**j**,** maxY**);**  iter **=** iter **+** 1**;**  **}**  **}**  **}**  **if(**iter **==** 0**){**  minX **=** 0**;** minY **=** 0**;** maxX **=** im**.**rows**()-**1**;** maxY **=** im**.**cols**()-**1**;**  **}**  **if(**minX**!=**0 **&&** minY**!=**0 **&&** maxX**!=**im**.**rows**()-**1 **&&** maxY**!=**im**.**cols**()-**1**)**  im **=** im**.**submat**(**minX**-**1**,** maxX**+**1**,** minY**-**1**,** maxY**+**1**);**  **else** **if(**minX**!=**0 **&&** minY**!=**0 **&&** maxX**!=**im**.**rows**()-**1 **&&** maxY**==**im**.**cols**()-**1**)**  im **=** im**.**submat**(**minX**-**1**,** maxX**+**1**,** minY**-**1**,** maxY**);**  **else** **if(**minX**!=**0 **&&** minY**!=**0 **&&** maxX**==**im**.**rows**()-**1 **&&** maxY**!=**im**.**cols**()-**1**)**  im **=** im**.**submat**(**minX**-**1**,** maxX**,** minY**-**1**,** maxY**+**1**);**  **else** **if(**minX**!=**0 **&&** minY**!=**0 **&&** maxX**==**im**.**rows**()-**1 **&&** maxY**==**im**.**cols**()-**1**)**  im **=** im**.**submat**(**minX**-**1**,** maxX**,** minY**-**1**,** maxY**);**  **else** **if(**minX**!=**0 **&&** minY**==**0 **&&** maxX**!=**im**.**rows**()-**1 **&&** maxY**!=**im**.**cols**()-**1**)**  im **=** im**.**submat**(**minX**-**1**,** maxX**+**1**,** minY**,** maxY**+**1**);**  **else** **if(**minX**!=**0 **&&** minY**==**0 **&&** maxX**!=**im**.**rows**()-**1 **&&** maxY**==**im**.**cols**()-**1**)**  im **=** im**.**submat**(**minX**-**1**,** maxX**+**1**,** minY**,** maxY**);**  **else** **if(**minX**!=**0 **&&** minY**==**0 **&&** maxX**==**im**.**rows**()-**1 **&&** maxY**!=**im**.**cols**()-**1**)**  im **=** im**.**submat**(**minX**-**1**,** maxX**,** minY**,** maxY**+**1**);**  **else** **if(**minX**!=**0 **&&** minY**==**0 **&&** maxX**==**im**.**rows**()-**1 **&&** maxY**==**im**.**cols**()-**1**)**  im **=** im**.**submat**(**minX**-**1**,** maxX**,** minY**,** maxY**);**  **else** **if(**minX**==**0 **&&** minY**!=**0 **&&** maxX**!=**im**.**rows**()-**1 **&&** maxY**!=**im**.**cols**()-**1**)**  im **=** im**.**submat**(**minX**,** maxX**+**1**,** minY**-**1**,** maxY**+**1**);**  **else** **if(**minX**==**0 **&&** minY**!=**0 **&&** maxX**!=**im**.**rows**()-**1 **&&** maxY**==**im**.**cols**()-**1**)**  im **=** im**.**submat**(**minX**,** maxX**+**1**,** minY**-**1**,** maxY**);**  **else** **if(**minX**==**0 **&&** minY**!=**0 **&&** maxX**==**im**.**rows**()-**1 **&&** maxY**!=**im**.**cols**()-**1**)**  im **=** im**.**submat**(**minX**,** maxX**,** minY**-**1**,** maxY**+**1**);**  **else** **if(**minX**==**0 **&&** minY**!=**0 **&&** maxX**==**im**.**rows**()-**1 **&&** maxY**==**im**.**cols**()-**1**)**  im **=** im**.**submat**(**minX**,** maxX**,** minY**-**1**,** maxY**);**  **else** **if(**minX**==**0 **&&** minY**==**0 **&&** maxX**!=**im**.**rows**()-**1 **&&** maxY**!=**im**.**cols**()-**1**)**  im **=** im**.**submat**(**minX**,** maxX**+**1**,** minY**,** maxY**+**1**);**  **else** **if(**minX**==**0 **&&** minY**==**0 **&&** maxX**!=**im**.**rows**()-**1 **&&** maxY**==**im**.**cols**()-**1**)**  im **=** im**.**submat**(**minX**,** maxX**+**1**,** minY**,** maxY**);**  **else** **if(**minX**==**0 **&&** minY**==**0 **&&** maxX**==**im**.**rows**()-**1 **&&** maxY**!=**im**.**cols**()-**1**)**  im **=** im**.**submat**(**minX**,** maxX**,** minY**,** maxY**+**1**);**  **else** **if(**minX**==**0 **&&** minY**==**0 **&&** maxX**==**im**.**rows**()-**1 **&&** maxY**==**im**.**cols**()-**1**)**  im **=** im**.**submat**(**minX**,** maxX**,** minY**,** maxY**);**  **}** |

**Gambar 4. 13 Kode implementasi proses pengecilan citra hasil *thinning***

### Implementasi Tahap Ekstraksi Fitur Karakter Aksara Jawa

Subbab ini membahas implementasi tahap ekstraksi fitur karakter aksara Jawa. Dalam tahap ini, terdapat dua proses yaitu ekstraksi fitur untuk citra tidak *thinning* dan citra *thinning*. Seperti penjelasan subbab 3.3.3.1, proses ekstraksi fitur citra tidak *thinning* menggunakan algoritma *region growing* yang didahului proses perubahan citra ke *array boolean* dua dimensi untuk mempercepat proses pada sistem. Implementasi proses tersebut ditunjukkan pada Gambar 4. 14.

|  |
| --- |
| boolean[][] img = new boolean[im.rows()][im.cols()]; |
| int row = im.rows();  int col = im.cols(); |
| for (int i = 0; i <= im.rows()-1; i++) { |
| for (int j = 0; j <= im.cols()-1; j++) { |
| if (im.get(i, j)[0] == 0.0) img[i][j] = false; |
| else img[i][j] = true; |
| hasil[i][j] = 0; |
| } |
| } |

Gambar 4. 14 Kode implementasi proses pengubahan citra ke *array boolean* dua dimensi proses *region growing*

Langkah pertama algoritma *region growing* adalah menandai *seed* awal sebagai bagian dari daerah hasil *region growing* yang sudah dikunjungi saat pencarian daerah hasil *region growing*. Implementasi langkah pertama ini ditunjukkan oleh Gambar 4. 15.

|  |
| --- |
| Mat region **=** Mat**.**zeros**(**im**.**size**(),** im**.**type**());** //Output  int**[][]** hasil **=** **new** int**[**im**.**rows**()][**im**.**cols**()];**  ArrayList**<**ArrayList**<**Integer**>>**neg\_list **=** **new** ArrayList**<**ArrayList**<**Integer**>>();**  int reg\_mean **=** **(**img**[**x**][**y**]?**1**:**0**);**  ArrayList**<**Integer**>** data\_list **=** **new** ArrayList**<**Integer**>();**  data\_list**.**add**(**x**);** data\_list**.**add**(**y**);**  neg\_list**.**add**(**data\_list**);**  hasil**[**x**][**y**]=**2**;** |

Gambar 4. 15 Kode implementasi proses langkah pertama *region growing*

Langkah kedua algoritma *region growing* adalah mencari daerah hasil *region growing* dimulai dari tetangga-tetangga *seed* awal sampai semua piksel pada citra terkunjungi. Implementasi langkah pertama ini ditunjukkan oleh Gambar 4. 16.

|  |
| --- |
| **while(!**neg\_list**.**isEmpty**()){**  neg\_list**.**remove**(**0**);**  **for(**int j**=**1**;**j**<=**4**;**j**++){**  int xn**,**yn**;**  **if(**j**==**1**){**  xn **=** x **-** 1**;** yn **=** y**;**  **}**  **else** **if(**j**==**2**){**  xn **=** x **+** 1**;** yn **=** y**;**  **}**  **else** **if(**j**==**3**){**  xn **=** x**;** yn **=** y **-** 1**;**  **}**  **else{**  xn **=** x**;** yn **=** y **+** 1**;**  **}**  boolean ins **=** **(**xn**>=**0 **&&** yn**>=**0 **&&** xn**<=**im**.**rows**()-**1 **&&** yn**<=**im**.**cols**()-**1**);**  **if(**ins**&&**hasil**[**xn**][**yn**]==**0**){**  int dist **=** Math**.**abs**((**img**[**xn**][**yn**]?**1**:**0**)-**reg\_mean**);**  **if(**dist**==**0**){**  ArrayList**<**Integer**>**data **=** **new** ArrayList**<**Integer**>();**  data**.**add**(**xn**);** data**.**add**(**yn**);**  neg\_list**.**add**(**data**);**  hasil**[**xn**][**yn**]=**2**;**  **}**  **else{**  hasil**[**xn**][**yn**]=**1**;**  **}**  **}**  **}**  **try** **{**  x **=** neg\_list**.**get**(**0**).**get**(**0**);**  y **=** neg\_list**.**get**(**0**).**get**(**1**);**  **}** **catch** **(**IndexOutOfBoundsException e**)** **{**  // TODO: handle exception  **break;**  **}**  **}** |

**Gambar 4. 16 Kode implementasi proses langkah kedua *region growing***

Langkah terakhir algoritma *region growing* adalah mengembalikan hasil pencarian *region growing* ke dalam citra biner dengan ketentuan piksel putih sebagai piksel hasil *region growing* dan piksel hitam sebagai piksel **bukan** hasil *region growing*. Implementasi langkah terakhir ini ditunjukkan oleh Gambar 4. 17.

|  |
| --- |
| **for(**int i**=**0**;**i**<=**row**-**1**;**i**++){**  **for(**int j**=**0**;**j**<=**col**-**1**;**j**++){**  **if(**hasil**[**i**][**j**]>**1**)**  region**.**put**(**i**,** j**,** 1.0**);**  **else**  region**.**put**(**i**,** j**,** 0.0**);**  **}**  **}**  region**.**convertTo**(**region**,** CvType**.**CV\_8U**,** 255**);** |

**Gambar 4. 17 Kode implementasi proses langkah terakhir *region growing***

Berdasarkan penjelasan subbab 3.3.3.1, terdapat fitur area kiri dan area kanan yang daerahnya dibagi berdasarkan titik tengah *range* terpanjang jumlah piksel kolom lebih dari nol. Berikut ini adalah implementasi proses pencarian indeks awal dan akhir *range* terpanjang jumlah piksel kolom lebih dari nol yang ditunjukkan oleh Gambar 4. 18. Sedangkan implementasi penghitungan titik tengah didapat dari hasil implementasi pada Gambar 4. 18 yang ditunjukkan oleh Gambar 4. 19.

|  |
| --- |
| protected int**[]** hitungPikselTengahThinning**(**Mat im**){**  im**.**convertTo**(**im**,** CvType**.**CV\_32F**,** 1.**/**255**);**  int startRow **=** **(**int**)**Math**.**round**(**0.2**\***im**.**rows**());**  int startRow1 **=** **(**int**)**Math**.**round**(**0.7**\***im**.**rows**());**  Mat thinKaki **=** im**.**rowRange**(**startRow**,** im**.**rows**()-**1**);**  ArrayList**<**Integer**>**sumthinKaki **=** **new** ArrayList**<**Integer**>();**  ArrayList**<**Integer**>**sumNotNull **=** **new** ArrayList**<**Integer**>();**  ArrayList**<**Integer**>**indeksNotNull **=** **new** ArrayList**<**Integer**>();**  ArrayList**<**Integer**>**indexNotOne **=** **new** ArrayList**<**Integer**>();**  //proses mendapatkan range-range kolom dengan jumlah piksel kolom tidak nol  **for(**int i**=**0**;**i**<=**thinKaki**.**cols**()-**1**;**i**++){**  int jumlah **=** 0**;**  **for(**int j**=**0**;**j**<=**thinKaki**.**rows**()-**1**;**j**++){**  jumlah**+=(**int**)**thinKaki**.**get**(**j**,** i**)[**0**];**  **}**  sumthinKaki**.**add**(**jumlah**);**  **}**  **for(**int i**=**0**;**i**<**sumthinKaki**.**size**();**i**++){**  **if(**sumthinKaki**.**get**(**i**)!=**0**)**  sumNotNull**.**add**(**i**);**  **}**  **for(**int j**=**0**;**j**<**sumNotNull**.**size**()-**1**;**j**++){**  indeksNotNull**.**add**(**sumNotNull**.**get**(**j**+**1**)-**sumNotNull**.**get**(**j**));**  **}**  **for(**int k**=**0**;**k**<**indeksNotNull**.**size**();**k**++){**  **if(**indeksNotNull**.**get**(**k**)!=**1**)**  indexNotOne**.**add**(**k**);**  **}**  **if(**indexNotOne**.**size**()==**1**){**  thinKaki**.**release**();**  sumthinKaki**.**clear**();**  sumNotNull**.**clear**();**  indeksNotNull**.**clear**();**  indexNotOne**.**clear**();**  //mengulang proses sebelumnya dengan daerah antara startRow1 sampai baris terakhir  **}**  ArrayList**<**Integer**>**selisihPiksel **=** **new** ArrayList**<**Integer**>();**  **for(**int j**=**0**;**j**<**indexNotOne**.**size**()-**1**;**j**++){**  selisihPiksel**.**add**(**indexNotOne**.**get**(**j**+**1**)-**indexNotOne**.**get**(**j**));**  **}**  int nilaiMaks **=** 0**,** indeksMaks **=** 0**;**  **for(**int k**=**0**;**k**<**selisihPiksel**.**size**();**k**++){**  nilaiMaks **=** Math**.**max**(**nilaiMaks**,** selisihPiksel**.**get**(**k**));**  **}**  **for(**int j**=**0**;**j**<**selisihPiksel**.**size**();**j**++){**  **if(**selisihPiksel**.**get**(**j**)==**nilaiMaks**){**  indeksMaks **=** indeksMaks **+** j**;**  **break;**  **}**  **}**  int**[]** index **=** **new** int**[**3**];**  **try** **{**  index**[**0**]** **=** sumNotNull**.**get**(**indexNotOne**.**get**(**indeksMaks**)+**1**);**  index**[**1**]** **=** sumNotNull**.**get**(**indexNotOne**.**get**(**indeksMaks**+**1**));**  **}** **catch** **(**IndexOutOfBoundsException e**)** **{**  index**[**0**]** **=** 0**;**  index**[**1**]** **=** im**.**cols**()-**1**;**  **}**  //pencarian fitur jumlah piksel kolom terbesar (lebih lanjut pada Gambar 4.27)  **return** index**;**  **}** |

**Gambar 4. 18 Kode implementasi proses pencarian indeks awal dan akhir pada *range* terpanjang**

|  |
| --- |
| int colTengah **=** **(**int**)**Math**.**round**(((**index**[**1**]-**index**[**0**])/**2**)+**index**[**0**]);** |

**Gambar 4. 19 Kode implementasi proses penghitungan titik tengah pada *range* terpanjang**

Implementasi penghitungan fitur area ditunjukkan oleh Gambar 4. 120 Sedangkan implementasi penghitungan fitur karakter ditunjukkan oleh Gambar 4. 1.

|  |
| --- |
| protected int hitungArea**(**Mat reg**,** Mat src**){**  int jumlahArea **=** 0**;**  Mat diffArea **=** **new** Mat**();**  Core**.**absdiff**(**src**,** reg**,** diffArea**);**  **if(**Core**.**countNonZero**(**diffArea**)==**0**)**  jumlahArea **=** jumlahArea **+** 0**;**  **else**  jumlahArea **=** jumlahArea **+** 1**;**  **return** jumlahArea**;**  **}** |

**Gambar 4. 20 Kode implementasi proses penghitungan fitur area**

|  |
| --- |
| protected int hitungKarakter**(**Mat reg**,** Mat src**){**  int jumlahKarakter **=** 0**;**  Core**.**bitwise\_not**(**src**,** src**);**  Mat diff **=** **new** Mat**();**  Core**.**absdiff**(**src**,** reg**,** diff**);**  **if(**Core**.**countNonZero**(**diff**)==**0**)**  jumlahKarakter **=** jumlahKarakter **+** 1**;**  **else**  jumlahKarakter **=** jumlahKarakter **+** 2**;**  Core**.**bitwise\_not**(**src**,** src**);**  **return** jumlahKarakter**;**  **}** |

**Gambar 4. 21 Kode implementasi proses penghitungan fitur karakter**

Implementasi proses penghitungan fitur area kiri, area kanan, area atas, dan area bawah menggunakan implementasi pada Gambar 4. 1 dengan daerah yang berbeda-beda sesuai dengan penjelasan subbab 3.3.3.1.

Setelah mendapatkan fitur-fitur dari citra tidak *thinning*, langkah selanjutnya mendapatkan fitur-fitur dari citra hasil *thinning*. Fitur pertama yang dihitung adalah fitur kaki (kaki seluruh citra, kaki kanan, kaki kiri, dan kaki vertikal). Implementasi penghitungan fitur kaki seluruh citra ditunjukkan oleh Gambar 4. 1. Sedangkan implementasi penghitungan kaki vertikal ditunjukkan oleh Gambar 4. 1.

|  |
| --- |
| protected int hitungKaki**(**Mat im**){**  Core**.**bitwise\_not**(**im**,** im**);**  im**.**convertTo**(**im**,** CvType**.**CV\_32F**,** 1.**/**255**);**  int rowKakiStart **=** **(**int**)** Math**.**round**(**0.35**\***im**.**rows**())-**1**;**  int rowKakiEnd **=** **(**int**)** Math**.**round**(**0.45**\***im**.**rows**())-**1**,** flag **=** 0**,** jmlKakiKolom **=** 0**;**  Mat kakiPenuh **=** im**.**rowRange**(**rowKakiStart**,** rowKakiEnd**);**  //implementasi perubahan citra ke *array boolean* dua dimensi sama dengan Gambar 4.14  ArrayList**<**Integer**>**jumlahPerKolom **=** **new** ArrayList**<**Integer**>();**  ArrayList**<**Integer**>**jumlahPerKaki **=** **new** ArrayList**<**Integer**>();**  **for(**int i**=**0**;**i**<=**col**-**1**;**i**++){**  int jumlah **=** 0**;**  **for(**int j**=**0**;**j**<=**row**-**1**;**j**++){**  jumlah**+=(**img**[**j**][**i**]?**1**:**0**);**  **}**  jumlahPerKolom**.**add**(**jumlah**);**  **}**  **for(**int i**=**0**;**i**<**jumlahPerKolom**.**size**();**i**++){**  **if(**jumlahPerKolom**.**get**(**i**)!=**0**){**  flag**++;**  jmlKakiKolom**+=**jumlahPerKolom**.**get**(**i**);**  **if(**i**==**jumlahPerKolom**.**size**()-**1**&&**jmlKakiKolom**>=**kakiPenuh**.**rows**())**  jumlahPerKaki**.**add**(**jmlKakiKolom**);**  **}**  **else{**  **if(**flag**>**0**&&**jmlKakiKolom**>=**kakiPenuh**.**rows**()){**  jumlahPerKaki**.**add**(**jmlKakiKolom**);**  **}**  jmlKakiKolom **=** 0**;**  flag **=** 0**;**  **}**  **}**  **return** jumlahPerKaki**.**size**();**  **}** |

**Gambar 4. 22 Kode implementasi proses penghitungan fitur kaki**

|  |
| --- |
| protected int hitungKakiVertikal**(**Mat im**){**  Core**.**bitwise\_not**(**im**,** im**);**  im**.**convertTo**(**im**,** CvType**.**CV\_32F**,** 1.**/**255**);**  int colKakiStart **=** **(**int**)** Math**.**round**(**0.35**\***im**.**cols**())-**1**;**  int colKakiEnd **=** **(**int**)** Math**.**round**(**0.45**\***im**.**cols**())-**1**,** flag **=** 0**,** jmlKakiBaris **=** 0**;**  Mat kakiPenuh **=** im**.**colRange**(**colKakiStart**,** colKakiEnd**);**  //implementasi perubahan citra ke *array boolean* dua dimensi sama dengan Gambar 4.14  ArrayList**<**Integer**>**jumlahPerBaris **=** **new** ArrayList**<**Integer**>();**  ArrayList**<**Integer**>**jumlahPerKaki **=** **new** ArrayList**<**Integer**>();**  **for(**int i**=**0**;**i**<=**row**-**1**;**i**++){**  int jumlah **=** 0**;**  **for(**int j**=**0**;**j**<=**col**-**1**;**j**++){**  jumlah**+=(**img**[**i**][**j**]?**1**:**0**);**  **}**  jumlahPerBaris**.**add**(**jumlah**);**  **}**  **for(**int i**=**0**;**i**<**jumlahPerBaris**.**size**();**i**++){**  **if(**jumlahPerBaris**.**get**(**i**)!=**0**){**  flag**++;**  jmlKakiBaris**+=**jumlahPerBaris**.**get**(**i**);**  **if(**i**==**jumlahPerBaris**.**size**()-**1**)**  jumlahPerKaki**.**add**(**jmlKakiBaris**);**  **}**  **else{**  **if(**flag**>**0**){**  jumlahPerKaki**.**add**(**jmlKakiBaris**);**  **}**  jmlKakiBaris **=** 0**;**  flag **=** 0**;**  **}**  **}**  **return** jumlahPerKaki**.**size**();**  **}** |

**Gambar 4. 23 Kode implementasi proses penghitungan fitur kaki vertikal**

Implementasi proses penghitungan fitur kaki kiri dan kaki kanan menggunakan implementasi pada Gambar 4. 1 dengan daerah yang berbeda-beda sesuai dengan penjelasan subbab 3.3.3.2.

Fitur-fitur lain yang harus dihitung dengan menggunakan ROI seperti penjelasan subbab 3.3.3.2 adalah rata-rata daerah atas, jumlah daerah bawah, jarak antara baris terakhir dan piksel tidak nol terbawah, dan jumlah piksel kolom terbesar. Implementasi penentuan indeks awal dan akhir ROI menggunakan indeks awal, tengah, dan akhir *range* terpanjang jumlah piksel kolom lebih dari nol seperti yang ditunjukkan oleh Gambar 4. 1. Implementasi penghitungan fitur rata-rata daerah atas ditunjukkan pada Gambar 4. 1 Implementasi penghitungan fitur jumlah daerah bawah ditunjukkan pada Gambar 4. 1.

|  |
| --- |
| double median **=** **((**index**[**1**]-**index**[**0**])/**2**)+**index**[**0**];**  double per4an **=** **((**index**[**1**]-**index**[**0**])+**1**)/**4**;**  int pikselPer4Awal **=** **(**int**)**Math**.**round**(**median**-**per4an**);**  int pikselPer4Akhir **=** **(**int**)**Math**.**round**(**median**+**per4an**);** |

**Gambar 4. 24 Kode implementasi proses indeks awal dan akhir ROI**

|  |
| --- |
| protected double hitungJumlahCeil**(**Mat im**,** int pikselTengah**,** int pikselAkhir**){**  Core**.**bitwise\_not**(**im**,** im**);**  im**.**convertTo**(**im**,** CvType**.**CV\_32F**,** 1.**/**255**);**  int endRow **=** **(**int**)**Math**.**round**(**0.2**\***im**.**rows**());**  double sumRataCeil **=** 0.0**;**  Mat ceil **=** im**.**submat**(**0**,** endRow**-**1**,** pikselTengah**,** pikselAkhir**-**1**);**  //implementasi perubahan citra ke *array boolean* dua dimensi sama dengan Gambar 4.14 ArrayList**<**Double**>**rataCeil **=** **new** ArrayList**<**Double**>();**  **for(**int i**=**0**;**i**<=**row**-**1**;**i**++){**  double jumlahRata **=** 0.0**;**  **for(**int j**=**0**;**j**<=**col**-**1**;**j**++){**  jumlahRata **=** jumlahRata **+** **(**img**[**i**][**j**]?**1**:**0**);**  **}**  rataCeil**.**add**(**jumlahRata**/**ceil**.**cols**());**  **}**  **for(**int k**=**0**;**k**<**rataCeil**.**size**();**k**++){**  sumRataCeil **=** sumRataCeil **+** rataCeil**.**get**(**k**);**    **return** sumRataCeil**;**  **}** |

**Gambar 4. 25 Kode implementasi proses penghitungan fitur rata-rata daerah atas**

|  |
| --- |
| protected int**[]** hitungFloor **(**Mat im**,** int pikselPer4Awal**,** int pikselPer4Akhir**){**  Core**.**bitwise\_not**(**im**,** im**);**  im**.**convertTo**(**im**,** CvType**.**CV\_32F**,** 1.**/**255**);**  int startRow **=** **(**int**)**Math**.**round**(**0.2**\***im**.**rows**());**  Mat floor **=** im**.**submat**(**startRow**,** im**.**rows**()-**1**,** pikselPer4Awal**,** pikselPer4Akhir**);**  Mat floor2 **=** im**.**col**(**pikselPer4Awal**);**  //implementasi perubahan citra ke array boolean dua dimensi sama dengan Gambar 4.14  ArrayList**<**Integer**>**floorSum **=** **new** ArrayList**<**Integer**>();**  ArrayList**<**Integer**>**pikselPertama **=** **new** ArrayList**<**Integer**>();**  **for(**int i **=** 0**;** i **<=** floor2**.**rows**()-**1**;** i**++)**  **if(**floor2**.**get**(**i**,** 0**)[**0**]** **==** 1.0**)**  pikselPertama**.**add**(**i**);**  **for(**int i**=**0**;**i**<=**row**-**1**;**i**++){**  int jumlahRata **=** 0**;**  **for(**int j**=**0**;**j**<=**col**-**1**;**j**++){**  jumlahRata **=** jumlahRata **+** **(**img**[**i**][**j**]?**1**:**0**);**  **}**  floorSum**.**add**(**jumlahRata**);**  **}**  int maxJumlah **=** 0**;**  **for(**int k**=**0**;**k**<**floorSum**.**size**();**k**++)**  maxJumlah **=** Math**.**max**(**maxJumlah**,** floorSum**.**get**(**k**));**  ArrayList**<**Integer**>**indexMax **=** **new** ArrayList**<**Integer**>();**  **for(**int k**=**0**;**k**<**floorSum**.**size**();**k**++){**  **if(**floorSum**.**get**(**k**)==**maxJumlah**)**  indexMax**.**add**(**k**);**  **}**  int indeksFloorMax **=** indexMax**.**get**(**indexMax**.**size**()-**1**);**  int floorSums **=** 0**;**  **for(**int i**=**0**;**i**<=**indeksFloorMax**;**i**++)**  floorSums **=** floorSums **+** floorSum**.**get**(**i**);**  int**[]** floors **=** **new** int**[**2**];**  floors**[**0**]** **=** floorSums**;**  //penghitungan fitur jarak antara baris terakhir dan piksel tidak nol terbawah  floor**.**convertTo**(**floor**,** CvType**.**CV\_8U**,** 255**);**  **return** floors**;**  **}** |

**Gambar 4. 26 Kode implementasi proses penghitungan fitur jumlah daerah bawah**

Berdasarkan Gambar 4. 1, implementasi penghitungan fitur jarak antara baris terakhir dan piksel tidak nol terbawah terdapat dalam fungsi yang sama dengan proses penghitungan fitur jumlah daerah bawah. Berikut ini adalah implementasi dari penghitungan fitur jarak antara baris terakhir dan piksel tidak nol terbawah yang ditunjukkan oleh Gambar 4. 1

|  |
| --- |
| int floorJarak **=** floor2**.**rows**()-**pikselPertama**.**get**(**pikselPertama**.**size**()-**1**)-**1**;**  floors**[**1**]** **=** **(**int**)**Math**.**round**(**floorJarak**\***100**/**im**.**rows**());** |

**Gambar 4. 27 Kode implementasi proses penghitungan fitur jarak antara baris terakhir dan piksel tidak nol terbawah**

Implementasi penghitungan fitur jumlah piksel kolom terbesar terdapat dalam implementasi penghitungan indeks awal dan akhir *range* terpanjang jumlah piksel kolom lebih dari nol pada Gambar 4. 18 Berikut ini adalah implementasi penghitungan jumlah piksel kolom terbesar yang ditunjukkan Gambar 4. 1

|  |
| --- |
| double colTengah **=** **((**index**[**1**]-**index**[**0**])/**2**)+**index**[**0**];**  double per4an **=** **((**index**[**1**]-**index**[**0**])+**1**)/**4**;**  int pikselPer4Awal **=** **(**int**)**Math**.**round**(**colTengah**-**per4an**);**  int pikselPer4Akhir **=** **(**int**)**Math**.**round**(**colTengah**+**per4an**);**  index**[**2**]** **=** 0**;**  **for(**int i**=**pikselPer4Awal**;**i**<=**pikselPer4Akhir**;**i**++)**  index**[**2**]** **=** Math**.**max**(**index**[**2**],** sumthinKaki**.**get**(**i**));** |

**Gambar 4. 28 Kode implementasi proses penghitungan fitur jumlah piksel kolom terbesar**

### Implementasi Tahap Klasifikasi Karakter Aksara Jawa

Subbab ini membahas implementasi tahap klasifikasi karakter aksara Jawa dengan hasil ekstraksi fitur sebagai data masukan. Seperti yang dijelaskan pada subbab 3.3.4.1, klasifikasi menggunakan algoritma K-Nearest Neighbor dengan nilai k sama dengan empat. Pada implementasinya, fitur-fitur yang akan diproses diubah dalam bentuk *string* dimana setiap nilai fitur dipisahkan oleh tanda baca koma (,). Berikut ini adalah implementasi pada sistem untuk proses klasifikasi karakter aksara Jawa yang ditunjukkan oleh Gambar 4. 29.

|  |
| --- |
| public String Classify**(**String feature**)** **throws** Exception  **{**  String str\_kelas **=** **null;**  Instance row **=** **new** Instance**(**15**);**  row**.**setDataset**(**Data**);**  List**<**String**>** featureList **=** Arrays**.**asList**(**feature**.**split**(**","**));**  **for(**int i **=**0**;**i**<**featureList**.**size**();**i**++)**  **{** row**.**setValue**(**i**,**Double**.**valueOf**(**featureList**.**get**(**i**)));**  **}**  row**.**setValue**(**14**,** 0.0**);**  Log**.**w**(**"info"**,**"row telah berhasil dibuat"**);**  Log**.**w**(**"row val"**,**row**.**toString**());**  double kelas **=** 0.0**;**  **try{**  kelas **=** knn**.**classifyInstance**(**row**);**  str\_kelas **=** Data**.**classAttribute**().**value**((**int**)**kelas**);**  **}**  **catch(**Exception ex**)**  **{**  Log**.**w**(**"error klasifikasi:" **,** ex**.**getMessage**());**  **}**  Log**.**w**(**"info"**,**"klasifikasi selesai"**);**  **return** str\_kelas**;**  **}** |

**Gambar 4. 29 Kode implementasi proses klasifikasi karakter aksara Jawa**

## Implementasi Antarmuka Pengguna

Implementasi tampilan antarmuka pengguna pada perangkat lunak berbasis *mobile* dan berjalan pada sistem operasi Android dilakukan dengan menggunakan berkas XML. Berikut ini dijelaskan mengenai implementasi tampilan antarmuka pengguna yang terdapat pada perangkat lunak.

### Halaman Menu

Halaman ini merupakan halaman pertama yang muncul saat sistem dijalankan. Rancangan mengenai halaman ini dapat dilihat pada subbab 3.4.1 dan implementasinya bisa dilihat pada Gambar 4. 30. Pada gambar terlihat bahwa halaman memiliki ***icon* pensil** untuk menuju halaman tulis aksara, ***icon* kamera** untuk menuju halaman foto aksara, ***icon* informasi** untuk menuju halaman info aplikasi, dan ***icon* pintu** untuk keluar dari sistem. Selain itu, terdapat gambar yang berisi informasi karakter aksara Jawa.



Gambar 4. 30 Implementasi halaman kalibrasi mata

### Halaman Tulis Aksara

Halaman ini adalah halaman yang digunakan oleh pengguna memasukkan karakter aksara Jawa dengan menulisnya di layar *smartphone*. Rancangan mengenai halaman ini dapat dilihat pada subbab 3.4.2 dan implementasinya bisa dilihat pada Gambar 4. 31. Terdapat tombol Hapus untuk menghapus semua tulisan dan tombol Simpan untuk menyimpan tulisan dan melanjutkan ke halaman klasifikasi.



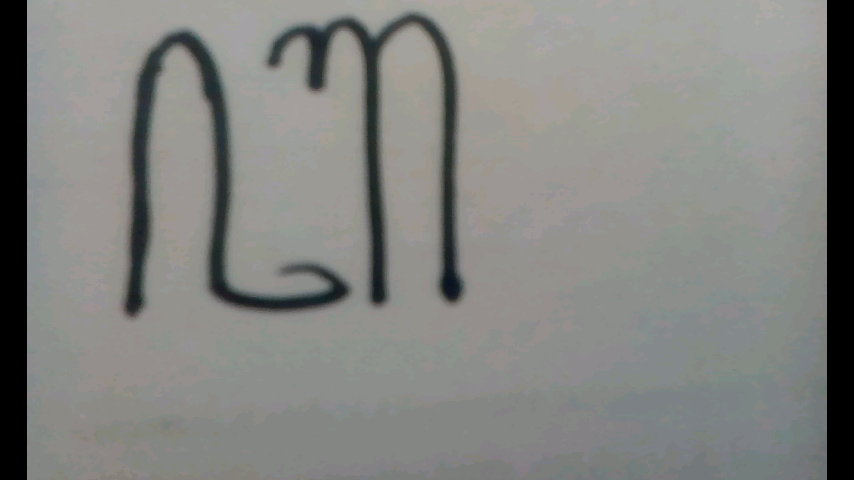
Gambar 4. 31 Implementasi halaman tulis aksara

### Halaman Foto Aksara

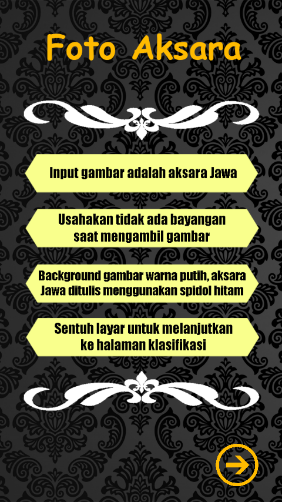
Halaman ini adalah halaman yang digunakan oleh pengguna untuk memasukkan citra tulisan karakter aksara Jawa melalui kamera utama *smartphone* dengan batasan-batasan yang sudah dijelasan dalam batasan masalah. Rancangan mengenai halaman ini dapat dilihat pada subbab 3.4.4 dan implementasinya bisa dilihat pada Gambar 4. 32. Pengguna cukup menyentuh layar *smartphone* untuk melanjutkan ke halaman klasifikasi dan menyimpan gambar.

### Halaman Info Aplikasi

Halaman ini merupakan halaman informasi halaman-halaman yang terdapat dalam sistem. Info halaman yang ada adalah info halaman tulis aksara, foto aksara, dan halaman klasifikasi. Rancangan mengenai halaman ini dapat dilihat pada subbab 3.4.5 dan implementasinya bisa dilihat pada Gambar 4. 33.



Gambar 4. 32 Implementasi halaman foto aksara

Gambar 4. 33 Implementasi halaman info aplikasi

### Halaman Klasifikasi

Halaman ini merupakan halaman yang digunakan oleh pengguna untuk mendapatkan informasi karakter aksara Jawa dari citra masukan. Pada halaman ini, terdapat tombol segmentasi untuk menampilkan hasil segmentasi, tombol klasifikasi untuk menampilkan hasil klasifikasi setiap karakter pada citra masukan, dan tombol *icon* panah ke kiri untuk kembali ke halaman menu. Rancangan mengenai halaman ini dapat dilihat pada subbab **Error! Reference source not found.** dan implementasinya bisa dilihat pada Gambar 4. 34.



Gambar 4. 34 Implementasi halaman klasifikasi

# BAB V UJI COBA DAN EVALUASI

Bab ini membahas uji coba dan evaluasi terhadap perangkat lunak yang telah dikembangkan dari implementasi sistem pengenalan aksara Jawa melalui tulisan tangan berbasis OpenCV Android.



## Lingkungan Uji Coba

Lingkungan uji coba yang digunakan dalam pembuatan tugas akhir ini meliputi perangkat lunak dan perangkat keras yang digunakan untuk melakukan uji coba implementasi sistem pengenalan aksara Jawa melalui tulisan tangan berbasis OpenCV Android. Lingkungan uji coba merupakan perangkat bergerak (*mobile device*) berupa ponsel pintar (*smartphone*) dengan spesifikasi sebagai berikut:

* Merek : Lenovo A526
* Processor : Quad-core 1.3 GHz Cortex-A7
* RAM : 1 GB
* Sistem Operasi : Android 4.2.2 JellyBean
* Kamera Utama : 5 MP

## Data Uji Coba

Uji coba yang dilakukan adalah uji coba terhadap fitur-fitur yang digunakan untuk klasifikasi dan uji coba hasil akurasi berdasarkan pengguna menggunakan dua citra masukan berbeda. Data uji coba untuk menguji coba fitur-fitur yang digunakan klasifikasi adalah data pelatihan yang dibagi menggunakan 10-*cross validation*. Data pelatihan yang digunakan adalah citra biner karakter aksara Jawa berukuran 100 x 100 piksel. Data pelatihan berjumlah 240 data dengan ketentuan setiap karakter aksara Jawa memiliki 12 data. Sedangkan data uji coba untuk menguji coba hasil akurasi berdasarkan pengguna adalah data masukan seperti yang dijelaskan pada subbab 3.2.1.

## Skenario Uji Coba

Subbab ini akan menjelaskan skenario uji coba yang telah dilakukan. Terdapat beberapa skenario uji coba yang telah dilakukan, antara lain:

* + 1. Perbandingan hasil akurasi berdasarkan asumsi penentuan fitur seperti yang dijelaskan pada subbab 3.3.4.
    2. Perbandingan hasil akurasi berdasarkan variasi nilai K untuk kombinasi asumsi penentuan fitur dengan hasil optimal
    3. Uji coba hasil akurasi berdasarkan responden yang berbeda pada proses pengenalan aksara Jawa melalui tulisan tangan pada layar *smartphone*
    4. Uji coba hasil akurasi berdasarkan responden yang berbeda pada proses pengenalan aksara Jawa melalui citra karakter aksara Jawa yang diambil menggunakan kamera utama *smartphone*

## Skenario Pengujian 1: Perbandingan Kombinasi Pemakaian Kondisi Penentuan Fitur

Nilai pembanding yang digunakan pada uji coba ini adalah nilai akurasi terbaik setiap kombinasi asumsi penentuan fitur dari variasi nilai K antara 1 sampai 10. Akurasi dihitung dengan ketentuan sebagai berikut:

1. Uji coba benar: sesuai dengan *groundtruth*
2. Uji coba salah: tidak sesuai dengan *groundtruth*

Kombinasi asumsi penentuan fitur yang digunakan dalam uji coba ini antara lain:

1. Tidak menggunakan semua asumsi
2. Menggunakan asumsi pertama
3. Menggunakan asumsi pertama dan kedua
4. Menggunakan asumsi pertama, kedua, dan ketiga
5. Menggunakan asumsi pertama, kedua, ketiga, dan keempat
6. Menggunakan semua asumsi

Hasil klasifikasi setiap karakter menggunakan kombinasi 1 asumsi penentuan fitur yang ditunjukkan oleh Tabel 5. 1 yang menghasilkan akurasi terbaik untuk variasi nilai K antara 1 sampai 10 sebesar 86.67%. Dalam menggunakan kombinasi ini, terdapat masalah dalam penentuan titik tengah pembagi daerah kiri dan kanan (kasus beberapa data karakter Ra). Permasalahan pada penentuan titik tengah adalah tidak ditemukannya *range* terpanjang jumlah piksel kolom lebih dari nol sehingga tidak bisa menentukan titik tengah.

Solusi untuk permasalahan tersebut adalah menentukan kolom pertama dan terakhir sebagai indeks awal dan akhir *range* terpanjang jumlah piksel kolom lebih dari nol. Hasil akurasi untuk setiap nilai K pada kombinasi 1 dapat dilihat pada Lampiran A. 3.

Tabel 5. 1 Hasil klasifikasi setiap karakter dengan kombinasi asumsi penentuan fitur 1

| No | Nama Karakter | Klasifikasi Benar | Klasifikasi Salah |
| --- | --- | --- | --- |
| 1 | Ha | 11 | 1 |
| 2 | Na | 10 | 2 |
| 3 | Ca | 9 | 3 |
| 4 | Ra | 12 | 0 |
| 5 | Ka | 11 | 1 |
| 6 | Da | 10 | 2 |
| 7 | Ta | 10 | 2 |
| 8 | Sa | 6 | 6 |
| 9 | Wa | 11 | 1 |
| 10 | La | 12 | 0 |
| 11 | Pa | 10 | 2 |
| 12 | Dha | 8 | 4 |
| 13 | Ja | 12 | 0 |
| 14 | Ya | 12 | 0 |
| 15 | Nya | 11 | 1 |
| 16 | Ma | 9 | 3 |
| 17 | Ga | 9 | 3 |
| 18 | Ba | 12 | 0 |
| 19 | Tha | 11 | 1 |
| 20 | Nga | 12 | 0 |
| TOTAL | | **208** | **32** |
| AKURASI | | **86.67%** | |

Uji coba berikutnya menggunakan kombinasi 2 asumsi penentuan fitur yaitu menambahkan asumsi pertama saat melakukan klasifikasi. Hasil klasifikasi setiap karakter untuk uji coba ini ditunjukkan oleh Tabel 5. 2 yang menghasilkan akurasi terbaik untuk variasi nilai K antara 1 sampai 10 sebesar 87.08%. Uji coba ini masih menggunakan solusi penentuan titik tengah sama seperti kombinasi pertama. Hasil akurasi untuk setiap nilai K pada kombinasi 2 dapat dilihat pada Lampiran A. 3.

Tabel 5. 2 Hasil klasifikasi setiap karakter dengan kombinasi asumsi penentuan fitur 2

| No | Nama Karakter | Klasifikasi Benar | Klasifikasi Salah |
| --- | --- | --- | --- |
| 1 | Ha | 10 | 2 |
| 2 | Na | 10 | 2 |
| 3 | Ca | 10 | 2 |
| 4 | Ra | 12 | 0 |
| 5 | Ka | 11 | 1 |
| 6 | Da | 9 | 3 |
| 7 | Ta | 10 | 2 |
| 8 | Sa | 8 | 4 |
| 9 | Wa | 11 | 1 |
| 10 | La | 12 | 0 |
| 11 | Pa | 10 | 2 |
| 12 | Dha | 9 | 3 |
| 13 | Ja | 12 | 0 |
| 14 | Ya | 11 | 1 |
| 15 | Nya | 11 | 1 |
| 16 | Ma | 9 | 3 |
| 17 | Ga | 9 | 3 |
| 18 | Ba | 12 | 0 |
| 19 | Tha | 11 | 1 |
| 20 | Nga | 12 | 0 |
| TOTAL | | **209** | **31** |
| AKURASI | | **87.08%** | |

Uji coba berikutnya menggunakan kombinasi 3 asumsi penentuan fitur yaitu menambahkan asumsi pertama dan kedua saat melakukan klasifikasi. Hasil klasifikasi setiap karakter untuk uji coba ini ditunjukkan oleh Tabel 5. 3 yang menghasilkan akurasi terbaik untuk variasi nilai K antara 1 sampai 10 sebesar 88.75%. Karena kondisi kedua digunakan, metode penentuan titik tengah kembali seperti semula. Hasil akurasi untuk setiap nilai K pada kombinasi 3 dapat dilihat pada Lampiran A. 3.

Tabel 5. 3 Hasil klasifikasi setiap karakter dengan kombinasi asumsi penentuan fitur 3

| No | Nama Karakter | Klasifikasi Benar | Klasifikasi Salah |
| --- | --- | --- | --- |
| 1 | Ha | 10 | 2 |
| 2 | Na | 11 | 1 |
| 3 | Ca | 9 | 3 |
| 4 | Ra | 12 | 0 |
| 5 | Ka | 11 | 1 |
| 6 | Da | 11 | 1 |
| 7 | Ta | 10 | 2 |
| 8 | Sa | 8 | 4 |
| 9 | Wa | 9 | 3 |
| 10 | La | 12 | 0 |
| 11 | Pa | 9 | 3 |
| 12 | Dha | 11 | 1 |
| 13 | Ja | 12 | 0 |
| 14 | Ya | 12 | 0 |
| 15 | Nya | 11 | 1 |
| 16 | Ma | 9 | 3 |
| 17 | Ga | 11 | 1 |
| 18 | Ba | 12 | 0 |
| 19 | Tha | 11 | 1 |
| 20 | Nga | 12 | 0 |
| TOTAL | | **213** | **27** |
| AKURASI | | **88.75%** | |

Pada uji coba kombinasi 4 asumsi penentuan fitur yaitu menambahkan asumsi pertama, kedua, dan ketiga saat melakukan klasifikasi. Hasil klasifikasi setiap karakter untuk uji coba ini ditunjukkan oleh Tabel 5. 4 yang menghasilkan akurasi terbaik untuk variasi nilai K antara 1 sampai 10 sebesar 90%. Metode penentuan titik tengah tidak berubah. Hasil akurasi untuk setiap nilai K pada kombinasi 4 dapat dilihat pada Lampiran A. 3.

Tabel 5. 4 Hasil klasifikasi setiap karakter dengan kombinasi asumsi penentuan fitur 4

| No | Nama Karakter | Klasifikasi Benar | Klasifikasi Salah |
| --- | --- | --- | --- |
| 1 | Ha | 10 | 2 |
| 2 | Na | 10 | 2 |
| 3 | Ca | 10 | 2 |
| 4 | Ra | 12 | 0 |
| 5 | Ka | 11 | 1 |
| 6 | Da | 11 | 1 |
| 7 | Ta | 10 | 2 |
| 8 | Sa | 10 | 2 |
| 9 | Wa | 10 | 2 |
| 10 | La | 12 | 0 |
| 11 | Pa | 9 | 3 |
| 12 | Dha | 11 | 1 |
| 13 | Ja | 12 | 0 |
| 14 | Ya | 12 | 0 |
| 15 | Nya | 11 | 1 |
| 16 | Ma | 9 | 3 |
| 17 | Ga | 11 | 1 |
| 18 | Ba | 12 | 0 |
| 19 | Tha | 11 | 1 |
| 20 | Nga | 12 | 0 |
| TOTAL | | **216** | **24** |
| AKURASI | | **90%** | |

Uji coba berikutnya adalah kombinasi 5 asumsi penentuan fitur yaitu menambahkan asumsi pertama, kedua, ketiga, dan keempat saat melakukan klasifikasi. Hasil klasifikasi setiap karakter untuk uji coba ini ditunjukkan oleh Tabel 5. 5 yang menghasilkan akurasi terbaik untuk variasi nilai K antara 1 sampai 10 sebesar 90.42%. Metode penentuan titik tengah tidak berubah. Hasil akurasi untuk setiap nilai K pada kombinasi 5 dapat dilihat pada Lampiran A. 3.

Tabel 5. 5 Hasil klasifikasi setiap karakter dengan kombinasi asumsi penentuan fitur 5

| No | Nama Karakter | Klasifikasi Benar | Klasifikasi Salah |
| --- | --- | --- | --- |
| 1 | Ha | 9 | 3 |
| 2 | Na | 11 | 1 |
| 3 | Ca | 10 | 2 |
| 4 | Ra | 12 | 0 |
| 5 | Ka | 11 | 1 |
| 6 | Da | 10 | 2 |
| 7 | Ta | 10 | 2 |
| 8 | Sa | 11 | 1 |
| 9 | Wa | 11 | 1 |
| 10 | La | 12 | 0 |
| 11 | Pa | 10 | 2 |
| 12 | Dha | 10 | 2 |
| 13 | Ja | 12 | 0 |
| 14 | Ya | 12 | 0 |
| 15 | Nya | 11 | 1 |
| 16 | Ma | 9 | 3 |
| 17 | Ga | 11 | 1 |
| 18 | Ba | 12 | 0 |
| 19 | Tha | 11 | 1 |
| 20 | Nga | 12 | 0 |
| TOTAL | | **217** | **23** |
| AKURASI | | **90.42%** | |

Uji coba terakhir adalah kombinasi 6 asumsi penentuan fitur yaitu menggunakan semua asumsi saat melakukan klasifikasi. Hasil klasifikasi setiap karakter untuk uji coba ini ditunjukkan oleh Tabel 5. 6 yang menghasilkan akurasi terbaik untuk variasi nilai K antara 1 sampai 10 sebesar 90.42%. Metode penentuan titik tengah tidak berubah..

Tabel 5. 6 Hasil klasifikasi setiap karakter dengan kombinasi kondisi penentuan fitur 6

| No | Nama Karakter | Klasifikasi Benar | Klasifikasi Salah |
| --- | --- | --- | --- |
| 1 | Ha | 9 | 3 |
| 2 | Na | 11 | 1 |
| 3 | Ca | 10 | 2 |
| 4 | Ra | 12 | 0 |
| 5 | Ka | 11 | 1 |
| 6 | Da | 10 | 2 |
| 7 | Ta | 10 | 2 |
| 8 | Sa | 11 | 1 |
| 9 | Wa | 11 | 1 |
| 10 | La | 12 | 0 |
| 11 | Pa | 10 | 2 |
| 12 | Dha | 10 | 2 |
| 13 | Ja | 12 | 0 |
| 14 | Ya | 12 | 0 |
| 15 | Nya | 11 | 1 |
| 16 | Ma | 9 | 3 |
| 17 | Ga | 11 | 1 |
| 18 | Ba | 12 | 0 |
| 19 | Tha | 11 | 1 |
| 20 | Nga | 12 | 0 |
| TOTAL | | **217** | **23** |
| AKURASI | | **90.42%** | |

Dari uji coba kombinasi kondisi penentuan fitur, terdapat dua kombinasi yang menghasilkan akurasi paling tinggi yaitu kombinasi 5 dan 6 senilai 90.42%. Setelah uji coba penerapan metode dalam sistem, kombinasi kondisi penentuan fitur yang optimal adalah kombinasi 6 karena kombinasi 5 terkadang menghasilkan hasil klasifikasi yang salah pada karakter Dha dan Ta. Hasil perbandingan akurasi setiap kombinasi asumsi dapat dilihat pada Tabel 5. 7.

Tabel 5. 7 Perbandingan Hasil Akurasi Kombinasi Asumsi Penentuan Fitur

|  |  |
| --- | --- |
| **Kombinasi Asumsi** | **Akurasi (%)** |
| 1 | 86.67 |
| 2 | 87.08 |
| 3 | 89.75 |
| 4 | 90 |
| 5 | 90.42 |
| 6 | 90.42 |

## Skenario Pengujian 2: Perbandingan Hasil Akurasi Berdasarkan Variasi Nilai K untuk Kombinasi Asumsi Penentuan Fitur dengan Hasil Optimal

Skenario uji coba ini bertujuan mencari nilai K dengan rentang nilai 1 sampai 10 yang menghasilkan akurasi terbaik dari kombinasi asumsi penggunaan fitur dengan hasil optimal yaitu kombinasi asumsi 6 yang menggunakan semua asumsi. Ketentuan penghitungan akurasi sama dengan skenario uji coba sebelumnya. Hasil uji coba nilai K dari rentang 1 sampai 10 untuk kombinasi 6 asumsi penggunaan fitur ditunjukkan oleh Tabel 5. 8.

Tabel 5. 8 Hasil akurasi setiap nilai K pada kombinasi 6

|  |  |
| --- | --- |
| **Iterasi** | **Akurasi (%)** |
| 1 | 87.92 |
| 2 | 87.92 |
| 3 | 89.58 |
| 4 | 90.42 |
| 5 | 90 |
| 6 | 89.58 |
| 7 | 89.58 |
| 8 | 90 |
| 9 | 88.75 |
| 10 | 85.42 |

Dari perbandingan hasil akurasi pada setiap nilai K, nilai K yang terbaik dan optimal adalah nilai K sama dengan 4 (empat). Selain memiliki akurasi tertinggi, pemilihan nilai K ini dikarenakan akurasi nilai K setelahnya berada di bawah akurasi nilai K sama dengan 4.

## Skenario Pengujian 3: Uji Coba Responden yang Berbeda untuk Pengenalan Aksara Jawa Melalui Tulisan Tangan pada Layar *Smartphone*

Pada skenario pengujian ini dilakukan pengujian terhadap tiga responden berbeda yang dipilih secara acak untuk proses pengenalan aksara Jawa. Pada skenario uji coba ini, responden menulis 8 jenis tulisan aksara Jawa yang berbeda sebanyak 5 kali untuk setiap tulisan. Tulisan tersebut terdiri dari dua tulisan dengan 1 karakter aksara Jawa, dua tulisan dengan 2 karakter aksara Jawa, dua tulisan dengan 3 karakter aksara Jawa, dan dua tulisan dengan 4 karakter aksara Jawa. Contoh tulisan responden 1, 2, dan 3 dapat dilihat secara berurutan pada Gambar 4. 1, Gambar 4. 2, dan Gambar 4. 3.

Hasil rata-rata akurasi uji coba ini dari setiap responden dapat dilihat pada Tabel 5. 9. Detail hasil akurasi setiap iterasi untuk responden 1, 2, dan 3 pada uji coba ini dapat dilihat secara berurutan pada Lampiran A. 3, Lampiran A. 3, dan Lampiran A. 3.

Tabel 5. 9 Hasil uji coba pengenalan aksara Jawa dengan responden yang berbeda melalui tulisan tangan pada layar *smartphone*

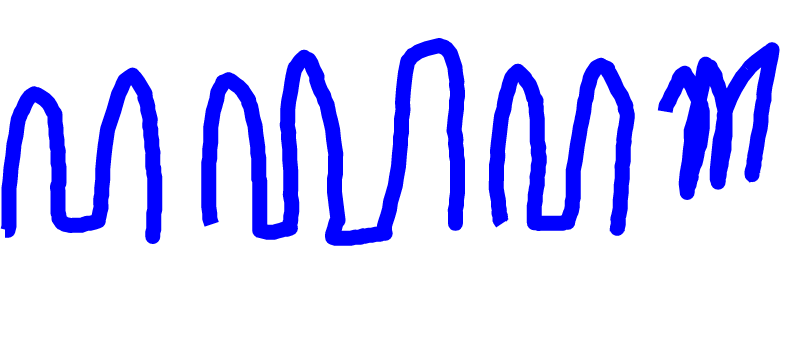
|  |  |  |
| --- | --- | --- |
| **No.** | **Nama Responden** | **Rata-rata Akurasi (%)** |
| 1 | Karsono Puguh Nindyo Cipto | 95.42 |
| 2 | Addien Haniefardy | 91.25 |
| 3 | Ignatius Abraham Susanto | 88.75 |
| **Rata-rata** | | **91.81** |



Gambar 5. 1 Contoh tulisan tangan pada layar *smartphone* responden pertama



Gambar 5. 2 Contoh tulisan tangan pada layar *smartphone* responden kedua



Gambar 5. 3 Contoh tulisan tangan pada layar *smartphone* responden ketiga

Dari hasil akurasi yang didapatkan dapat dilihat bahwa nilai rata-rata akurasi implementasi sistem pengenalan aksara Jawa menggunakan tulisan dari layar *smartphone* mencapai 91.81% dengan kasus tiga responden yang berbeda.

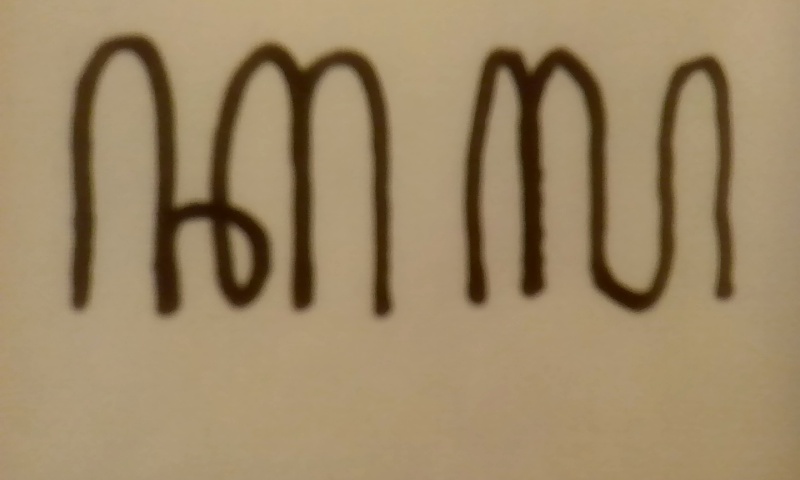
## Skenario Pengujian 4: Uji Coba Responden yang Berbeda untuk Pengenalan Aksara Jawa Melalui Citra Karakter Aksara Jawa yang Diambil Melalui Kamera Utama

Skenario pengujian ini menggunakan konsep yang sama dengan skenario pengujian 3 tetapi dengan citra karakter aksara Jawa yang berasal dari citra hasil pengambilan gambar melalui kamera utama *smartphone*. Pada skenario pengujian ini, responden menulis tulisan-tulisan yang sama dengan skenario pengujian sebelumnya pada kertas putih menggunakan spidol hitam sebanyak 5 kali untuk setiap tulisan. Hasil penulisan pada kertas tersebut diambil gambarnya menggunakan kamera utama pada *smartphone*. Contoh hasil pengambilan gambar tulisan tangan responden 1, 2, dan 3 melalui kamera utama *smartphone* dapat dilihat secara berurutan pada Gambar 4. 4, Gambar 4. 5, dan Gambar 4. 6.

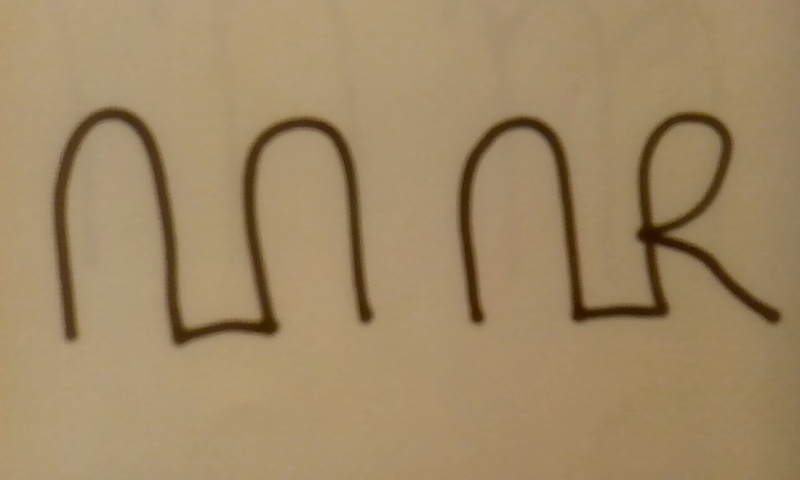
Hasil rata-rata akurasi uji coba ini dari setiap responden dapat dilihat pada Tabel 5. 10. Detail hasil akurasi setiap iterasi untuk responden 1, 2, dan 3 pada uji coba ini dapat dilihat secara berurutan pada Lampiran A. 3, Lampiran A. 3, dan Lampiran A. 3.

Tabel 5. 10 Hasil uji coba pengenalan aksara Jawa dengan responden yang berbeda melalui citra aksara Jawa dari hasil pengambilan gambar melalui kamera utama *smartphone*

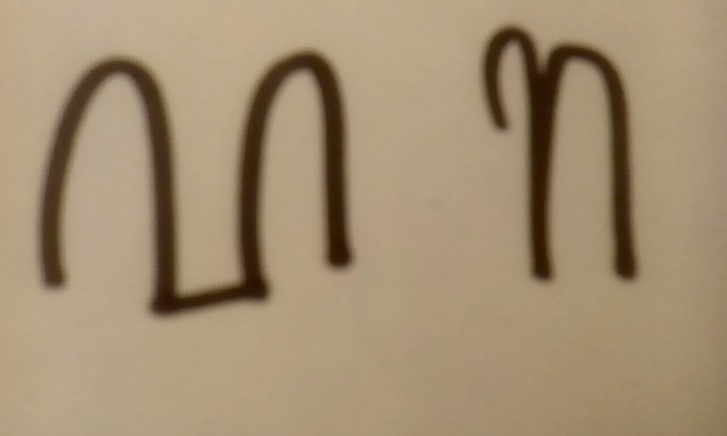
|  |  |  |
| --- | --- | --- |
| **No.** | **Nama Responden** | **Rata-rata Akurasi (%)** |
| 1 | Karsono Puguh Nindyo Cipto | 94.17 |
| 2 | Addien Haniefardy | 97.08 |
| 3 | Ignatius Abraham Susanto | 94.79 |
| **Rata-rata** | | **95.35** |



Gambar 5. 4 Contoh citra karakter aksara Jawa hasil pengambilan gambar melalui kamera utama *smartphone* responden pertama



Gambar 5. 5 Contoh citra karakter aksara Jawa hasil pengambilan gambar melalui kamera utama *smartphone* responden kedua



Gambar 5. 6 Contoh citra karakter aksara Jawa hasil pengambilan gambar melalui kamera utama *smartphone* responden ketiga

Dari hasil akurasi yang didapatkan dapat dilihat bahwa nilai rata-rata akurasi implementasi sistem pengenalan aksara Jawa menggunakan citra karakter aksara Jawa hasil pengambilan gambar melalui kamera utama *smartphone* mencapai 95.35% dengan kasus tiga responden yang berbeda.

## Evaluasi

Dari skenario uji coba yang telah dilakukan, beberapa parameter yang telah digunakan dalam uji coba memberikan pengaruh terhadap hasil akurasi kinerja implementasi sistem pengenalan aksara Jawa ini. Parameter-parameter yang telah digunakan antara lain kombinasi asumsi penentuan fitur untuk klasifikasi dan nilai K pada algoritma klasifikasi K-Nearest Neighbor untuk mendapatkan hasil optimal. Uji coba terhadap parameter-parameter tersebut menghasilkan nilai rata-rata akurasi yang optimal pada kombinasi 6 asumsi penentuan fitur yaitu menggunakan semua asumsi dengan nilai K sama dengan 4 sebesar 90.42%. Walaupun terdapat dua kombinasi yang menghasilkan akurasi tertinggi, tetapi penerapan pada sistem memberikan hasil optimal ketika menggunakan kombinasi 6.

Selain itu, parameter dari kondisi terbaik hasil uji coba pengenalan karakter aksara Jawa juga dilakukan terhadap tiga responden berbeda dan didapatkan hasil bahwa implementasi sistem pengenalan aksara Jawa berbasis OpenCV Android mampu memberikan nilai rata-rata akurasi hingga 91.81% untuk citra karakter aksara Jawa yang berasal dari tulisan tangan pada layar *smartphone* dan 95.35% untuk citra karakter aksara Jawa yang berasal dari hasil pengambilan gambar menggunakan kamera utama *smartphone*. Hasil akurasi pengenalan aksara Jawa dengan citra hasil pengambilan gambar melalui kamera utama *smartphone* pada tiga responden berbeda menunjukkan hasil lebih baik daripada citra hasil tulisan tangan responden pada layar *smartphone* karena responden belum terbiasa menulis aksara Jawa pada layar *smartphone*. Selain itu, menulis aksara Jawa pada kertas lebih mudah dibandingkan pada layar *smartphone*. Variasi nilai akurasi responden pada pengenalan aksara Jawa dari citra hasil tulisan tangan pada layar *smartphone* dipengaruhi oleh keterbiasaan responden dalam menulis aksara Jawa pada layar *smartphone*. Jika responden sudah terbiasa, maka kesalahan pengenalan aksara Jawa dapat dikurangi. Sedangkan variasi nilai akurasi responden pada pengenalan aksara Jawa dari citra hasil pengambilan gambar menggunakan kamera utama *smartphone* dipengaruhi oleh jarak kamera dengan objek.

# BAB VI KESIMPULAN DAN SARAN

Bab ini berisi tentang kesimpulan yang diperoleh selama pengerjaan tugas akhir ini. Selain itu, juga terdapat beberapa saran terhadap tugas akhir ini yang diharapkan bisa membuat tugas akhir ini menjadi lebih baik lagi.



## Kesimpulan

Kesimpulan yang diperoleh berdasarkan uji coba dan evaluasi yang telah dilakukan pada tugas akhir antara lain:

1. Berdasarkan hasil uji coba terhadap kombinasi asumsi penggunaan fitur pada proses klasifikasi karakter aksara Jawa, kombinasi asumsi penggunaan fitur yang memberikan hasil klasifikasi optimal adalah kombinasi 6 yaitu menggunakan semua asumsi yang dijelaskan pada subbab 3.3.4 dengan nilai K sama dengan 4 yang memberikan akurasi sebesar 90.42%.
2. Tidak semua fitur berpengaruh besar terhadap karakter aksara Jawa. Hal itu dibuktikan dengan penggunaan fitur klasifikasi dengan kombinasi asumsi penentuan fitur yang memberikan akurasi lebih tinggi daripada penggunaan fitur klasifikasi yang tidak menggunakan asumsi.
3. Uji coba implementasi sistem pengenalan aksara Jawa dengan responden berbeda masih memberikan hasil yang baik, yakni dengan nilai rata-rata akurasi sebesar 91.81% untuk citra hasil tulisan tangan pada layar *smartphone* dan 95.35% untuk citra hasil pengambilan gambar melalui kamera utama *smartphone*.
4. Selain batasan masalah yang diberikan ketika responden mengambil citra masukan dari kamera utama, jarak antara kamera dan objek juga mempengaruhi hasil klasifikasi.

## Saran

Beberapa saran terkait tugas akhir ini yang diharapkan bisa membuat tugas akhir ini menjadi lebih baik antara lain:

1. Perlu dilakukan optimasi pada sistem agar sistem lebih cepat melakukan klasifikasi terhadap karakter aksara Jawa sehingga pengguna lebih cepat memperoleh informasinya.
2. Perlu penambahan data pelatihan dengan data karakter lebih bervariasi agar pengenalan karakter aksara Jawa lebih tepat terhadap variasi bentuk tulisan karakter aksara Jawa.
3. Sistem ini bisa dikembangkan dalam beberapa *platform* seperti web, *desktop*, dan lainnya agar pengguna bisa mempelajari karakter aksara Jawa pada berbagai macam *gadget*. Selain pembacaan aksara dasar Jawa, sistem ini bisa dikembangkan dengan membaca *sandhangan* aksara Jawa agar pengguna lebih mengenal aksara Jawa.

# DAFTAR PUSTAKA

|  |  |
| --- | --- |
| [1] | E. Juharwidyningsih, C. Fatichah and W. N. Khotimah, "Pengenalan Karakter Tulisan Tangan Angka dan Operator Matematika Berdasarkan Zernike Moments Menggunakan Support Vector Machine," *JURNAL TEKNIK POMITS,* vol. 2, 2013. |
| [2] | P. N. W. Anastasia Rita Widiarti, "Javanese Character Recognition Using Hidden Markov Model," *World Academy of Science, Engineering and Technology,* vol. 3, 2009. |
| [3] | O. Trier, A. K. Jain and T. Taxt, "Feature extraction methods for character recognition-A survey," *Pattern Recognition,* vol. 29, no. 4, pp. 641-662, 1996. |
| [4] | A. P. Putra, "Fiture Extraction," DSI, 31 March 2013. [Online]. Available: http://alfian-p-p-fst10.web.unair.ac.id/artikel\_detail-76028-INTELLIGENT%20SYSTEM-fiture%20extraction.html. [Accessed 1 January 2015]. |
| [5] | M. B. Ulum, "TULISAN AKSARA JAWA LENGKAP (BELAJAR AKSARA JAWA LENGKAP)," August 2012. [Online]. Available: http://dududth.blogspot.com/2012/08/belajar-aksara-jawa-yang-terlupakan.html#comment-form. [Accessed 9 Juny 2015]. |
| [6] | S. Sayad, "K Nearest Neighbors - Classification," 2010. [Online]. Available: http://www.saedsayad.com/k\_nearest\_neighbors.htm. [Accessed 22 December 2014]. |
| [7] | P.-N. Tan, M. Steinbach and V. Kumar, "Data Mining Classification: Alternative Techniques," in *Introduction to Data Mining*, Addison-Wesley, 2005. |
| [8] | o. d. team, "OpenCV 2.4.9.0 documentation » OpenCV API Reference » Introduction," OpenCV Dev Team, 21 April 2014. [Online]. Available: http://docs.opencv.org/modules/core/doc/intro.html. [Accessed 22 December 2014]. |
| [9] | O. D. Team, "OPENCV (OPEN SOURCE COMPUTER VISION)," OpenCV Developers Team, 2014. [Online]. Available: http://opencv.org/. [Accessed 22 December 2014]. |
| [10] | o. d. team, "OpenCV 2.4.9.0 documentation » OpenCV4Android Reference » Android OpenCV Manager » Introduction," OpenCV Dev Team, 21 April 2014. [Online]. Available: http://docs.opencv.org/platforms/android/service/doc/Intro.html. [Accessed 22 December 2014]. |
| [11] | D. F. Arnia, "Pengolahan Citra Digital," March 2011. [Online]. Available: https://ceritakampus.files.wordpress.com/2011/03/kuliah6-regiongrowing\_watershed.pptx.. [Accessed 21 May 2015]. |
| [12] | D. Marshall, "Region Growing," Cardiff University, [Online]. Available: http://www.cs.cf.ac.uk/Dave/Vision\_lecture/node35.html. [Accessed 19 May 2015]. |
| [13] | Nash, "Implementation of Guo-Hall thinning algorithm," 12 January 2013. [Online]. Available: http://opencv-code.com/quick-tips/implementation-of-guo-hall-thinning-algorithm/. [Accessed 21 May 2015]. |
| [14] | A. d. Nataniel, "Tugas 2 Thinning Algorithm," 2003. [Online]. Available: http://staf.cs.ui.ac.id/WebKuliah/citra/2003/Thinning/Kelompok01.ppt. [Accessed 21 May 2015]. |

# LAMPIRAN

Lampiran A. 1 Hasil akurasi setiap nilai K pada kombinasi 1

|  |  |
| --- | --- |
| **Iterasi** | **Akurasi (%)** |
| 1 | 82.5 |
| 2 | 82.08 |
| 3 | 86.25 |
| 4 | 86.25 |
| 5 | 86.25 |
| 6 | 86.67 |
| 7 | 85.83 |
| 8 | 80.83 |
| 9 | 79.17 |
| 10 | 78.33 |

Lampiran A. 2 Hasil akurasi setiap nilai K pada kombinasi 2

|  |  |
| --- | --- |
| **Iterasi** | **Akurasi (%)** |
| 1 | 83.75 |
| 2 | 83.33 |
| 3 | 86.25 |
| 4 | 87.08 |
| 5 | 84.58 |
| 6 | 85.42 |
| 7 | 85 |
| 8 | 80.83 |
| 9 | 80.42 |
| 10 | 77.08 |

Lampiran A. 3 Hasil akurasi setiap nilai K pada kombinasi 3

|  |  |
| --- | --- |
| **Iterasi** | **Akurasi (%)** |
| 1 | 86.67 |
| 2 | 85.42 |
| 3 | 87.92 |
| 4 | 87.92 |
| 5 | 88.75 |
| 6 | 88.33 |
| 7 | 88.75 |
| 8 | 87.5 |
| 9 | 87.5 |
| 10 | 85.42 |

Lampiran A. 4 Hasil akurasi setiap nilai K pada kombinasi 4

|  |  |
| --- | --- |
| **Iterasi** | **Akurasi (%)** |
| 1 | 85 |
| 2 | 84.17 |
| 3 | 87.92 |
| 4 | 89.17 |
| 5 | 89.17 |
| 6 | 89.17 |
| 7 | 90 |
| 8 | 88.33 |
| 9 | 88.33 |
| 10 | 85.42 |

Lampiran A. 5 Hasil akurasi setiap nilai K pada kombinasi 5

|  |  |
| --- | --- |
| **Iterasi** | **Akurasi (%)** |
| 1 | 87.92 |
| 2 | 86.67 |
| 3 | 89.17 |
| 4 | 89.58 |
| 5 | 90.42 |
| 6 | 89.17 |
| 7 | 88.33 |
| 8 | 90 |
| 9 | 89.17 |
| 10 | 85.42 |

Lampiran A. 6 Detail akurasi per iterasi responden 1 pada skenario uji coba ketiga

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No.** | **Tulisan Aksara Jawa** | **Akurasi Per Iterasi (%)** | | | | | **Rata-rata Akurasi (%)** |
| **1** | **2** | **3** | **4** | **5** |
| 1 | Sa | 100 | 100 | 100 | 100 | 100 | 100 |
| 2 | Tha | 100 | 100 | 100 | 100 | 100 | 100 |
| 3 | Da Ra | 100 | 100 | 100 | 100 | 100 | 100 |
| 4 | Ka La | 100 | 100 | 100 | 100 | 50 | 90 |
| 5 | Pa Da Na | 100 | 100 | 100 | 100 | 100 | 100 |
| 6 | Ga Dha La | 100 | 100 | 66.67 | 100 | 100 | 93.33 |
| 7 | Pa La Sa Nga | 100 | 100 | 50 | 75 | 100 | 85 |
| 8 | Nga Ca Wa La | 100 | 75 | 100 | 100 | 100 | 95 |

Lampiran A. 7 Detail akurasi per iterasi responden 2 pada skenario uji coba ketiga

| **No.** | **Tulisan Aksara Jawa** | **Akurasi Per Iterasi (%)** | | | | | **Rata-rata Akurasi (%)** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | **2** | **3** | **4** | **5** |
| 1 | Ha | 100 | 100 | 100 | 100 | 0 | 80 |
| 2 | Sa | 100 | 100 | 100 | 100 | 100 | 100 |
| 3 | Pa Ja | 100 | 50 | 100 | 100 | 100 | 90 |
| 4 | Ra Dha | 100 | 100 | 100 | 100 | 100 | 100 |
| 5 | Pa Nga Ha | 100 | 66.67 | 66.67 | 100 | 100 | 86.67 |
| 6 | Ra Sa Wa | 66.67 | 100 | 100 | 100 | 100 | 93.33 |
| 7 | Pa Sa Da Pa | 100 | 75 | 75 | 100 | 100 | 90 |
| 8 | Ha La Ra Ga | 100 | 100 | 75 | 75 | 100 | 90 |

Lampiran A. 8 Detail akurasi per iterasi responden 3 pada skenario uji coba ketiga

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No.** | **Tulisan Aksara Jawa** | **Akurasi Per Iterasi (%)** | | | | | **Rata-rata Akurasi (%)** |
| **1** | **2** | **3** | **4** | **5** |
| 1 | Sa | 100 | 100 | 100 | 0 | 0 | 60 |
| 2 | Ga | 100 | 0 | 100 | 100 | 100 | 80 |
| 3 | Pa Ra | 100 | 100 | 100 | 100 | 100 | 100 |
| 4 | Pa Ya | 100 | 50 | 100 | 100 | 100 | 90 |
| 5 | Ma Pa Da | 100 | 100 | 66.67 | 100 | 66.67 | 86.67 |
| 6 | Ya Ya Pa | 100 | 66.67 | 100 | 100 | 100 | 93.33 |
| 7 | Pa Ya Pa Ga | 100 | 100 | 100 | 100 | 100 | 100 |
| 8 | Pa Ya Ra Pa | 100 | 100 | 100 | 100 | 100 | 100 |

Lampiran A. 9 Detail akurasi per iterasi responden 1 pada skenario uji coba keempat

| **No.** | **Tulisan Aksara Jawa** | **Akurasi Per Iterasi (%)** | | | | | **Rata-rata Akurasi (%)** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **1** | **2** | **3** | **4** | **5** |
| 1 | Sa | 100 | 100 | 100 | 100 | 100 | 100 |
| 2 | Tha | 100 | 100 | 100 | 100 | 0 | 80 |
| 3 | Da Ra | 100 | 100 | 100 | 100 | 100 | 100 |
| 4 | Ka La | 100 | 100 | 100 | 100 | 100 | 100 |
| 5 | Pa Da Na | 100 | 100 | 100 | 100 | 100 | 100 |
| 6 | Ga Dha La | 100 | 100 | 100 | 100 | 66.67 | 93.33 |
| 7 | Pa La Sa Nga | 75 | 100 | 75 | 75 | 100 | 85 |
| 8 | Nga Ca Wa La | 75 | 100 | 100 | 100 | 100 | 95 |

Lampiran A. 10 Detail akurasi per iterasi responden 2 pada skenario uji coba keempat

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No.** | **Tulisan Aksara Jawa** | **Akurasi Per Iterasi (%)** | | | | | **Rata-rata Akurasi (%)** |
| **1** | **2** | **3** | **4** | **5** |
| 1 | Ha | 100 | 100 | 100 | 100 | 100 | 100 |
| 2 | Sa | 100 | 100 | 100 | 100 | 100 | 100 |
| 3 | Pa Ja | 100 | 100 | 100 | 100 | 100 | 100 |
| 4 | Ra Dha | 100 | 100 | 100 | 100 | 100 | 100 |
| 5 | Pa Nga Ha | 100 | 66.67 | 66.67 | 100 | 100 | 86.67 |
| 6 | Ra Sa Wa | 100 | 100 | 100 | 100 | 100 | 100 |
| 7 | Pa Sa Da Pa | 50 | 100 | 100 | 100 | 100 | 90 |
| 8 | Ha La Ra Ga | 100 | 100 | 100 | 100 | 100 | 100 |

Lampiran A. 11 Detail akurasi per iterasi responden 3 pada skenario uji coba keempat

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No.** | **Tulisan Aksara Jawa** | **Akurasi Per Iterasi (%)** | | | | | **Rata-rata Akurasi (%)** |
| **1** | **2** | **3** | **4** | **5** |
| 1 | Sa | 100 | 100 | 100 | 100 | 100 | 100 |
| 2 | Ga | 100 | 100 | 100 | 100 | 0 | 80 |
| 3 | Pa Ra | 100 | 100 | 50 | 100 | 100 | 90 |
| 4 | Pa Ya | 100 | 100 | 100 | 100 | 100 | 100 |
| 5 | Ma Pa Da | 100 | 100 | 100 | 100 | 100 | 100 |
| 6 | Ya Ya Pa | 100 | 66.67 | 100 | 100 | 100 | 93.33 |
| 7 | Pa Ya Pa Ga | 100 | 100 | 100 | 100 | 100 | 100 |
| 8 | Pa Ya Ra Pa | 75 | 100 | 100 | 100 | 100 | 95 |

# BIODATA PENULIS

****Penulis lahir di Situbondo, 21 September 1993. Penulis telah menempuh pendidikan dasar di SDN 1 Gudang Asembagus Situbondo, kemudian untuk pendidikan menengah pertama di SMPN 2 Jember dan di jenjang menengah atas di SMAN 1 Jember. Sejak kecil, penulis memiliki ketertarikan yang besar pada matematika dan logika sehingga penulis memutuskan untuk mengambil pendidikan sarjana S1 di Jurusan Teknik Informatika, Fakultas Teknologi Informasi, Institut Teknologi Sepuluh Nopember Surabaya. Selama kuliah, penulis pernah mengemban amanah sebagai asisten dosen pada mata kuliah Statistika Komputasional, Kecerdasan Komputasional, dan Matematika Informatika. Selain itu, penulis juga aktif berorganisasi menjadi Bendahara KMI ITS 2013/2014, staf Himpunan Mahasiswa Teknik Computer-Informatika (HMTC) ITS 2013/2014, dan staf BEM FTIF ITS 2013/2014.

Penulis dalam menyelesaikan pendidikan S1 mengambil rumpun mata kuliah (RMK) Komputasi Cerdas dan Visi serta memiliki ketertarikan di bidang Manajemen Basis Data, Pemrograman *Mobile*, *Data Mining*, serta Pengolahan Citra Digital. Untuk komunikasi, penulis dapat dihubungi melalui surel: [hashfi.51.104@gmail.com](mailto:hashfi.51.104@gmail.com).