PENGENALAN TULISAN TANGAN LATIN (*CURSIVE HANDWRITING*) DENGAN ALGORITMA *FUZZY ART NEURAL NETWORK*

Skripsi

Sebagai salah satu syarat untuk memperoleh gelar
Sarjana Komputer dalam bidang Ilmu Komputer

Oleh :
RIEZQI RIZAL SALASA
0610963053-96

PROGRAM STUDI ILMU KOMPUTER
JURUSAN MATEMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS BRAWIJAYA
MALANG

2011


UNIVERSITAS BRAWIJAYA

LEMBAR PENGESAHAN SKRIPSI

Pengenalan Tulisan Tangan Latin (*Cursive Handwriting*) dengan Algoritma *Fuzzy ART Neural Network*

Oleh :

RIEZQI RIZAL SALASA
0610963053-96

Setelah dipertahankan di depan Majelis Penguji
Pada tanggal 23 Mei2011

dan dinyatakan memenuhi syarat untuk memperoleh gelar Sarjana Komputer dalam bidang Ilmu Komputer

Pembimbing I,	Pembimbing II,
Dewi Yanti L.,S.Kom,M.Kom	**Drs.Marji,MT**
NIP. 198111162005012004	**NIP.196708011992031001**

Mengetahui,
Ketua Jurusan Matematika
Fakultas MIPA Universitas Brawijaya

Dr. Abdul Rouf Alghofari, M.Sc
NIP.196709071992031001

LEMBAR PERNYATAAN

Saya yang bertanda tangan di bawah ini :

Nama : Riezqi Rizal Salasa
NIM : 0610963053-96
Jurusan : Matematika
Program Studi : Ilmu Komputer
Penulis skripsi berjudul : Pengenalan Tulisan Tangan Latin (*Cursive Handwriting*) dengan Algoritma *Fuzzy ART Neural Network*

Dengan ini menyatakan bahwa :

1. **Isi dari Skripsi yang saya buat adalah benar-benar karya sendiri dan tidak menjiplak karya orang lain, selain nama-nama yang termaktub di isi dan tertulis di daftar pustaka dalam Skripsi ini.**
2. **Apabila dikemudian hari ternyata Skripsi yang saya tulis terbukti hasil jiplakan, maka saya akan bersedia menanggung segala resiko yang akan saya terima.**

Demikian pernyataan ini dibuat dengan segala kesadaran.

Malang, 23Mei 2011
Yang menyatakan,

Riezqi Rizal Salasa
NIM. 0610963053-96

ABSTRAK

Pengenalan tulisan tangan yaitu proses untuk mengenali tulisan yang ditulis menggunakan alat tulis biasa kedalam bentuk digital teks. Penelitian tentang pengenalan tulisan tangan telah dilakukan selama bertahun-tahun. Hasil dari penelitian telah banyak diaplikasikan dalam pemenuhan kebutuhan manusia. Sistem pengenalan tulisan tangan secara garis besar terbagi atas 2 kategori, yaitu *online recognition* dan *offline recognition*. Pada *online recognition*, tulisan dihasilkan oleh suatu perangkat elektronik seperti pada layar PDA (touch screen), sedangkan *off-line system* menggunakan citra dari tulisan tangan yang diambil melalui kamera atau *scanner*.Pada skripsi ini akan dibahas pengenalan tulisan tangan latin menggunakan *offline recognition*, dimana citra tulisan diambil melalui *scanner*. Metode yang digunakan adalah pendekatan analitis dengan segmentasi secara eksplisit, dimana proses pengenalan tulisan tangan latin dilakukan dengan cara memecah tulisan tangan kedalam *segment*, kemudian mencoba melakukan pengenalan terhadap *segment* tersebut. Metode ini mempunyai keunggulan bahwa dalam tahap pengenalan terhadap *segment* dapat menggunakan teknik pengenalan karakter (*Optical Character Recognition*) yang sudah ada. Pada proses awal(*pre-processing*) dilakukan pemrosesan citra seperti *binersasi*dan *character segmentation*. Kemudian dilakukan ekstraksi fitur dengan metode *zoning feature extraction* untuk selanjutnya dimasukkan ke dalam tahap pengenalan. *Fuzzy ART Neural Network* dipilih karena memiliki keunggulan dalam hal *increament learning*. Hasil yang didapatkan dalam penelitian ini adalah tingkat akurasi segementasi sebesar 64,54% dan untuk tingkat akurasi pengenalan menggunakan algoritma *Fuzzy ART Neural Network* sebesar 75,17%.

Kata kunci : Pengenalan Tulisan Tangan Latin,*binerisasi*, *projection profile histogram, zoning feature extraction, Fuzzy ART Neural Network.*

ABSTRACT

Handwriting recognition is a process to identify writing text that are written manually using usual writing tools into digital text. Research on handwriting recognition has been done for years and the results from this research has been applied in the fulfillment of human needs. Handwriting recognition system is broadly divided into 2 categories, that is online recognition and off-line recognition. In online recognition, the writing text produced by an electronic device like a PDA screen (touch screen), while writing text of the off-line system produced by captured handwriting images through a camera or scanner. This thesis discuss about cursive handwriting recognition using the off-line recognition, where the image of writing text is taken through the scanner. The method used in this thesis isan analyticalapproachwithexplicitsegmentation, that thecursivehandwriting recognitionprocessis done bybreaking handwriting text into segments, then try to recognize them. The advantage from this method is the use of optical character recognition in segment recognizing phase which is already exist. In the early process called pre-processing, image processing such as binerisasi and character segmentation is done in each writing text. Later, feature extraction process using zoning feature extraction method is applied and end into recognition phase. FuzzyARTneuralnetworkwas choseninterms ofincrement learning. Resultsobtainedfrom this studyisthe level ofsegmentationaccuracyof64.54% andthelevel ofrecognition accuracyusing FuzzyARTNeuralNetwork algorithm of75.17%.

Keywords : *Cursive handwriting recognition*, *binarization*, *projection profile histogram, zoning feature extraction, Fuzzy ART Neural Network.*

KATA PENGANTAR

Puji syukur kehadirat Allah SWT, hanya dengan rahmat dan karunia yang telah diberikan kepada penulis, sehingga dapat menyelesaikan skripsi yang berjudul "*Pengenalan Tulisan Tangan Latin (Cursive Handwriting) dengan Algoritma Fuzzy ART Neural Network*".

Skripsi ini merupakan salah satu syarat untuk memenuhi persyaratan akademis untuk menyelesaikan studi di program Sarjana Ilmu Komputer Universitas Brawijaya.

Pada kesempatan ini penulis mengucapkan banyak terima kasih atas segala bantuan dan dedikasi moral maupun material dalam rangka penyusunan skripsi ini.

1. Dewi Yanti Liliana,S.Kom.,M.Kom dan Drs. Mardji, MT., selaku dosen pembimbing yang telah membimbing dengan bijaksana dan sabar dalam membimbing dengan baik penyusunan skripsi ini.
2. Dr. Abdul Rouf Alghofari, M.Sc., selaku Ketua Jurusan Matematika Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Brawijaya.
3. Drs. Mardji, MT., selaku Ketua Program Studi Ilmu Komputer, Jurusan Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Brawijaya.
4. Reza Andria Siregar, ST., selaku dosen pembimbing akademik.
5. Segenap Bapak dan Ibu dosen yang telah mendidik dan mengajarkan ilmunya kepada penulis selama menempuh pendidikan di Program Studi Ilmu Komputer Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Brawijaya.
6. Segenap staf dan karyawan Jurusan Matematika Universitas Brawijaya yang telah membantu penyusunan skripsi ini.
7. Ayah, Ibu, dan saudara-saudara, serta keluarga besarku yang tersayang, terima kasih atas dukungan dan doanya.
8. Rakanita Nawangrasi, I Gede Adi, Siwie, Tari, Menyun, Lhya, Fafan, teman-teman Prodi Ilmu Komputerserta teman-teman lain yang selalu memberi dukungan dan doanya.
9. Seluruh pihak yang tidak dapat disebut secara langsung yang telah memberikan bantuan demi terselesaikannya skripsi ini.

Penulis menyadari bahwa masih banyak kekurangan dalam penulisan laporan ini yang disebabkan oleh keterbatasan kemampuan dan pengalaman. Oleh karena itu, Penulis sangat menghargai saran dan kritik yang sifatnya membangun demi perbaikan penulisan dan mutu isi penelitian ini.

Penulis berharap semoga laporan ini dapat memberikan manfaat kepada pembaca dan bisa diambil manfaatnya untuk pengembangan selanjutnya.

Malang, 23 Mei 2011

Penulis

DAFTAR ISI

Halaman

HALAMAN JUDUL i
HALAMAN PENGESAHAN iii
LEMBAR PERNYATAAN v
ABSTRAK vii
ABSTRACT ix
KATA PENGANTAR xi
DAFTAR ISI xiii
DAFTAR GAMBAR xvii
DAFTAR TABEL xix
DAFTAR *SOURCECODE* xxi
DAFTAR LAMPIRAN xxiii

BAB I PENDAHULUAN 1
1.1 Latar Belakang 1
1.2 Rumusan Masalah 2
1.3 Tujuan Penelitian 2
1.4 Batasan Masalah 3
1.5 Manfaat Penelitian 3
1.6 Metodologi Pemecahan Masalah 3
1.7 Sistematika Penulis 4

BAB II TINJAUAN PUSTAKA 5
2.1 Pengolahan Citra Digital 5
2.1.1 Pengertian Citra 5
2.1.2 Pengertian Pengolahan Citra 6
2.1.3 Pengenalan Tulisan Tangan 6
2.2 Pre-Processing 7
2.2.1 Binerisasi 7
2.2.1.1 *Grayscale* 8
2.2.1.2 *Thresholding* (pengambangan) 8
2.2.2 Segmentasi 9
2.2.2.1 *Projection Profile-Based Histogram* 9
2.3 Processing 11
2.3.1 *Feature Extraction* 11

2.4 Jaringan Syaraf Tiruan 11
2.4.1 Pengertian Jaringan Syaraf Tiruan 11
2.4.2 Arsitektur Jaringan Syaraf Tiruan 12
2.4.3 Metode Pembelajaran Jaringan Syaraf Tiruan 13
2.5 *Adaptive Resonance Theory* (ART) 14
2.5.1 *Match-Based Learning* 14
2.5.2 Cara Kerja ART 15
2.5.3 ART-1 16
2.5.4 Fuzzy ART 17
2.5.5 Algoritma Fuzzy ART 18
2.6 Tingkat Akurasi PengenalanTulisan Tangan Latin 20

BAB III METODOLOGI DAN PERANCANGAN 21
3.1 Analisis Sistem 22
3.1.1 Deskripsi Sistem 22
3.1.2 Deskripsi Data 23
3.1.3 Analisis Kebutuhan Sistem 23
3.2 Perancangan Perangkat Lunak 24
3.2.1 Perancangan Proses 24
3.2.1.1 *Pre-processing 1* 25
a. Binerisasi 26
3.2.1.2 *Pre-processing 2* 28
a. *Character Segmentation* 28
3.2.1.3 *Processing* 30
a. *Scaling* 30
b. *Zoning Feature Extraction* 30
3.2.1.4 Proses pengenalan karakter 34
3.3Perancangan *Interface* 37
3.4 Perhitungan Manual 38
3.4.1 Perhitungan Fuzzy ART 39
3.5Perancangan Uji Coba 57

BAB IV IMPLEMENTASI DAN PEMBAHASAN 59
4.1 Lingkungan Implementasi 59
4.1.1 Lingkungan Perangkat Keras 59
4.1.2 Lingkungan Perangkat Lunak 59
4.2 Implementasi *Interface* 60
4.3 Implementasi Program 63

4.3.1 Implementasi Tahap *Preprocessing* 63
4.3.1.1 Binerisasi 64
4.3.1.2 *Character Segmentation* 64
4.3.2 Implementasi Tahp Pengenalan (*Processing*) 66
4.4 Skenario Pengujian 72
4.4.1 Data Pengujian 72
4.4.2 Lingkungan Pengujian 72
4.4.3 Hasil Pengujian 73
4.4.3.1 Hasil Pengujian *Preprocessing* 73
4.4.3.1 Hasil Pengujian Pengenalan dengan Fuzzy ART 73
4.5 Analisa Hasil 75
4.5.1 Analisa Hasil *Preprocessing* 75
4.5.1 Analisa Hasil Pengenalan Karakter 76

BAB V KESIMPULAN DAN SARAN 79
5.1 Kesimpulan 79
5.2 Saran 79

DAFTAR PUSTAKA 81
LAMPIRAN 85

DAFTAR GAMBAR

Halaman

Gambar 2.1 Representasi citra digital..... 5
Gambar 2.2 Contoh citra *grayscale* dan citra biner..... 9
Gambar 2.3 Implementasi histogram dalam segmentasi 10
Gambar 2.4 Jaringan Syaraf Tiruan..... 12
Gambar 2.5 Susunan layer-layer pada JST 13
Gambar 2.6 Perceptron pada ART 15
Gambar 2.7 Model jaringan ART-1 16
Gambar 3.1 Alur penelitian 22
Gambar 3.2 Flowchart proses umum dalam sistem..... 24
Gambar 3.3 Flowchart urutan proses dalam sistem..... 25
Gambar 3.4 Flowchart urutan proses *pre-processing 1*..... 26
Gambar 3.5 Ilustrasi proses binerisasi 26
Gambar 3.6 Flowchart proses binerisasi..... 27
Gambar 3.7 Flowchart urutan proses *pre-processing 2*..... 28
Gambar 3.8 Flowchart proses *character segmentation*..... 29
Gambar 3.9 Ilustrasi proses *scaling*..... 30
Gambar 3.10 Ilustrasi proses *zoning*..... 31
Gambar 3.11 Flowchart proses *feature extraction* 31
Gambar 3.12Representasi vektor 34
Gambar 3.13 Flowchart proses pelatihan Fuzzy ART..... 36
Gambar 3.14 Rancangan *interface* tahapan pengujian 37
Gambar 3.15 Rancangan *interface* Fuzzy ART 38
Gambar 3.16 Klasifikasi pola dengan Fuzzy ART 39
Gambar 4.1 *Interface* pengenalan..... 60
Gambar 4.2 *Interface* pelatihan 62
Gambar 4.3Contoh Kesalahan Segmentasi Karakter 75
Gambar 4.4Contoh Kesalahan Segmentasi Karakter 76
Gambar 4.5 Grafik pengaruh nilai parameter Fuzzy ART 76
Gambar 4.6 Contoh Kesalahan Pengenalan Karakter 77


UNIVERSITAS BRAWIJAYA

DAFTAR TABEL

Halaman

Tabel 3.1 Daftar karakter 23
Tabel 3.2 Ilustrasi pemisahan piksel dari citra karakterberukuran 32x20 32
Tabel 3.3 Proses pencarian nilai rata-rata pada tiap area 33
Tabel 3.4Inisialisasi bobot awal 40
Tabel 3.5 Input vector 'A' 40
Tabel 3.6 Nilai $I \wedge w_1^{(awal)}$ 41
Tabel 3.7 Nilai $I \wedge w_1^{(awal)}$ 42
Tabel 3.8 Nilai $w_1^{(baru)}$ 42
Tabel 3.9Input vector 'B' 43
Tabel 3.10 Nilai $I \wedge w_1^{(baru)}$ 43
Tabel 3.11 Nilai $I \wedge w_2^{(awal)}$ 44
Tabel 3.12 Nilai $I \wedge w_2^{(awal)}$ 45
Tabel 3.13 Nilai $w_2^{(baru)}$ 45
Tabel 3.14Input vector 'C' 46
Tabel 3.15 Nilai $I \wedge w_1^{(baru)}$ 46
Tabel 3.16 Nilai $I \wedge w_2^{(baru)}$ 47
Tabel 3.17 Nilai $I \wedge w_3^{(awal)}$ 47
Tabel 3.18 Nilai $I \wedge w_3^{(awal)}$ 48
Tabel 3.19 Nilai $w_3^{(baru)}$ 49
Tabel 3.20Input vector 'D' 49
Tabel 3.21 Nilai $I \wedge w_1^{(baru)}$ 50
Tabel 3.22 Nilai $I \wedge w_2^{(baru)}$ 50
Tabel 3.23 Nilai $I \wedge w_3^{(baru)}$ 51
Tabel 3.24 Nilai $I \wedge w_4^{(awal)}$ 52
Tabel 3.25 Nilai $I \wedge w_4^{(awal)}$ 53
Tabel 3.26 Nilai $w_4^{(baru)}$ 53
Tabel 3.27 Input vector *testing* 54
Tabel 3.28 Nilai $I \wedge w_1^{(baru)}$ 54
Tabel 3.29 Nilai $I \wedge w_2^{(baru)}$ 55
Tabel 3.30 Nilai $I \wedge w_3^{(baru)}$ 55
Tabel 3.31 Nilai $I \wedge w_4^{(baru)}$ 56
Tabel 3.32 Hasil Uji Keberhasilan Segmentasi 57
Tabel 3.33Hasil Uji Parameter Fuzzy ART 57

Tabel 3.34 Hasil Uji Pengenalan Karakter 58
Tabel 4.1Hasil Uji Segmentasi Tulisan Tangan Latin..... 73
Tabel 4.2 Hasil Uji Parameter Fuzzy ART 74
Tabel 4.3 Hasil Uji Pengenalan Karakter 75

DAFTAR *SOURCECODE*

Halaman

Sourcecode 4.1 Struktur data *preprocessing* 63
Sourcecode 4.2 Proses konversi *binerisasi*..... 64
Sourcecode 4.3 Segmentasi Karakter 66
Sourcecode 4.4 Struktur data *processing* 67
Sourcecode 4.5 Konversi blok 68
Sourcecode 4.6 Konversi vector input 68
Sourcecode 4.7 Fungsi-fungsi perhitungan Fuzzy ART 71
Sourcecode 4.8 Pengenalan Fuzzy ART 72


UNIVERSITAS BRAWIJAYA

DAFTAR LAMPIRAN

	Halaman
Lampiran 1 Karakter Data Latih	85
Lampiran 2 Data Uji Segmentasi Karakter	92
Lampiran 3 Data Uji Pengenalan Karakter	10

UNIVERSITAS BRAWIJAYA

BAB I
PENDAHULUAN

1.1 Latar Belakang

Tulisan tangan latin (*Cursive*) adalah gaya tulisan tangan yang digunakan untuk menulis secara cepat (Jean, 1997). Kata "*cursive*" berasal dari bahasa Latin yaitu *cursivus*, yang berarti mengalir. Huruf pada tulisan latin tersambung pada setiap katanya. Tulisan latin sering digunakan dalam kegiatan sehari-hari, seperti mencatat, pengisian formulir dan surat-menyurat. Untuk menulis sering menggunakan pena dan kertas dari pada langsung memakai papan ketik (*keybord*). Setelah catatan tersebut selesai ditulis, maka baru kemudian diketik. Demikian juga untuk mereproduksi naskah-naskah lama dalam tulisan tangan latin sehingga dibutuhkan waktu dan tenaga khusus untuk mengetik ulang. Untuk mengurangi waktu dan tenaga pengetikan tersebut maka diperlukan suatu perangkat lunak pengenal tulisan tangan latin (Sediyono, 2000).

Sistem pengenalan tulisan tangan terbagi atas 2 kategori, yaitu *online system* dan *offline system* (Gader, 1997). Pada *online system*, sistem mengenali secara langsung pada saat pengguna menulis. Dengan cara ini dapat diketahui informasi-informasi secara dinamik dari tulisan tersebut, yaitu: kayuhan (*stroke*), urutan kayuhan dan kecepatan penulisan. Namun metode ini membutuhkan peralatan khusus berupa *digitizer* atau *electronic tablet* yang harganya relatif lebih mahal. Sebaliknya, pada *offline system* tidak memerlukan perangkat khusus melainkan proses pengenalan tulisan tangan menggunakan citra dari tulisan tangan yang diambil melalui kamera atau *scanner*(Sediyono, 2000). Pada penelitian ini digunakan pengenalan tulisan tangan dengan *offline system.* Hasil dari penelitian telah banyak diaplikasikan untuk memenuhi kebutuhan manusia. Beberapa aplikasi pengenalan tulisan tangan antara lain sistem pemilahan surat otomatis, validasi tanda tangan, pengolahan cek di bank dan aplikasi lain. Karena manfaat dari pengenalan tulisan tangan inilah kebutuhan akan proses pengenalan tulisan tangan secara akurat semakin diperlukan, dengan menerapkan berbagai metode yang ada.

Metode yang sering digunakan dalam pengenalan tulisan tangan adalah penerapan dari Jaringan Syaraf Tiruan (*Neural Network*) (Tay & Khalid, 2002). Jaringan Syaraf Tiruan bekerja meniru prinsip jaringan syaraf biologis yang memiliki kemampuan seperti otak manusia untuk mempelajari sesuatu dari contoh-contoh yang ada, maka *Neural Network* biasanya digunakan untuk memecahkan masalah-masalah yang sifatnya pengenalan (*recognition*). Salah satu metode Jaringan Syaraf Tiruan yang memiliki kemampuan pembelajaran sangat baik adalah Jaringan Syaraf Tiruan Fuzzy ART (*Adaptive Resonance Theory*). Fuzzy ART ini merupakan pengembangan dari *Adaptive Resonance Theory* (ART) dengan mengintegrasikan *fuzzy logic* ke sistem tersebut, yang mempunyai kelebihan dalam hal *increment learning*, yaitu jaringan dapat belajar data masukan yang baru, tanpa melupakan data yang lama (Carpenter, 1991).

Pengenalan tulisan tangan pernah dilakukan penelitian sebelumnya oleh Leila dan Mohammed dengan melakukan pengenalan tulisan Arab dengan Fuzzy ART. Hasil yang didapat menunjukkan bahwa tingkat pengenalan sebesar 78% dan Fuzzy ART memiliki kecepatan dan stabilitas jaringan yang baik.

Dalam skripsi ini akan digunakan metode pengenalan tulisan tangan latin dengan Fuzzy ART. Berdasarkan latar belakang yang telah dipaparkan, maka judul yang diambil dalam skripsi ini adalah **"Pengenalan Tulisan Tangan Latin (*Cursive Handwriting*) dengan menggunakan *Fuzzy ART Neural Network*"**.

1.2 Rumusan Masalah

Berdasarkan uraian pada latar belakang, maka dalam skripsi ini dapat dirumuskan permasalahan sebagai berikut:

1. Bagaimana merancang segmentasi karakter pada citra tulisan tangan latin (*cursive handwriting*).
2. Bagaimana merancang arsitektur Fuzzy ART *Neural Network* yang digunakan dalam pengenalan huruf tulisan tangan latin.
3. Bagaimana tingkat akurasi yang dihasilkan oleh sistem untuk mengenali tulisan tangan latin.

1.3 Tujuan Penelitian

Beberapa tujuan yang ingin dicapai dalam skripsi ini adalah:

1. Merancang segmentasi karakter pada citra tulisan tangan latin (*cursive handwriting*).
2. Merancang arsitektur Fuzzy ART *Neural Network* yang digunakan dalam pengenalan huruf tulisan tangan latin.
3. Mengetahui tingkat akurasi yang dihasilkan oleh sistem untuk mengenali tulisan tangan latin.

1.4 Batasan Masalah

Batasan masalah dalam tugas akhir ini adalah perencanaan dan pembuatan perangkat lunak dengan klasifikasi :

1. Pengenalan tulisan tangan latin dilakukan dengan *offline system.*
2. Citra yang akan digunakan dalam skripsi ini berupa citra dalam format *Bitmap* (*.bmp).
3. Citra data uji berupa 1 kata tulisan tangan latin.
4. Pengenalan tulisan dilakukan dengan memecah citra kedalam segment dan dilakukan pengenalan per karakter.
5. Sistem pengenalan karakter dari tulisan latin (*cursive script*) hanya dapat mengenali 1 jenis variasi tulisan.
6. Karakter yang akan dikenali berupa huruf kecil (a-z).
7. Tulisan tangan yang dikenali berupa tulisan tangan latin (bersambung), tegak dan tidak ditulis miring (*slant*).

1.5 Manfaat Penelitian

Manfaat dari hasil skripsi ini dapat digunakan untuk mengenali tulisan tangan latin menggunakan algoritma Fuzzy ART sehingga hasil pengenalan tulisan dapat diproses menjadi informasi yang diperlukan.

1.6 Metodologi Pemecahan Masalah

Untuk mencapai tujuan yang dirumuskan sebelumnya, maka metodologi yang digunakan dalam penulisan tugas akhir ini adalah:

1. Studi Literatur
   Mempelajari teori yang berhubungan dengan pengolahan citra dan metodeFuzzy ART dari berbagai sumber.
2. Pendefinisian dan analisis masalah.
   Mendefinisikan dan menganalisis masalah untuk mencari solusi yang tepat.

3. Perancangan dan implementasi sistem
   Membuat perancangan perangkat lunak dan mengimplementasikan hasilnya untuk membuat perangkat lunak tersebut.
4. Uji coba dan analisis hasil implementasi
   Menguji coba perangkat lunak tersebut dan menganalisa hasil dari implementasi tersebut apakah sudah sesuai dengan tujuan yang dirumuskan sebelumnya, untuk kemudian dievalusi dan disempurnakan.

1.7 Sistematika Penulisan

Sistematika penulisan tugas akhir ini dibagi menjadi lima bab dengan masing-masing bab diuraikan sebagai berikut:

1. BAB I PENDAHULUAN
   Berisi latar belakang masalah, perumusan masalah, batasan masalah, tujuan penulisan, manfaat penulisan, metodologi pemecahan masalah, dan sistematika penulisan.
2. BAB II TINJAUAN PUSTAKA
   Bab ini membahas mengenai teori-teori penunjang yang membahas konsep dasar dari citra, beberapa macam representasinya, serta beberapa teori Jaringan Syaraf Tiruan, pengolahan citra dan metode pelatihan Fuzzy ART.
3. BAB III METODE DAN PERANCANGAN
   Bab ini membahas mengenai penggunaan teori-teori dalam perancangan perangkat lunak ini. Serta membahas bagaimana perangkat lunak ini direncanakan, dibuat dan diimplementasikan.
4. BAB IV HASIL DAN PEMBAHASAN
   Pada bab ini akan dilakukan implementasi sistem, serta analisa hasil yang dikeluarkan oleh perangkat lunak, untuk selanjutnya dilakukan analisa hasil untuk mengevaluasi pengenalan tulisan tangan latin setelah mengalami proses pengenalan Fuzzy ART.
5. BAB V PENUTUP
   Bab ini berisi kesimpulan dan saran dari proses pembuatan sampai proses pengimplementasian perangkat lunak serta untuk pengembangan perangkat lunak lebih lanjut.

BAB II
TINJAUAN PUSTAKA

Skripsi ini menggunakan beberapa tinjauan pustaka yang menunjang dalam sisi teori yang akan digunakan dalam bab pembahasan. Tinjauan pustaka yang digunakan meliputi pengolahan citra digital (pengertian citra digital, pengertian pengolahan citra dan pengenalan tulisan tangan), *pre-processing* (binerisasi dan segmentasi), *processing* (*feature extraction*), jaringan syaraf tiruan, algoritma *Adaptive Resonance Theory* dan Fuzzy ART (*Adaptive Resonance Theory*).

2.1 Pengolahan Citra Digital

2.1.1 Pengertian Citra

Secara harfiah, citra (*image*) adalah gambar pada bidang dua dimensi (Koerich, 2003). Citra dapat dikelompokkan menjadi citra tampak dan citra tidak tampak. Contoh dari citra tampak adalah foto, gambar, lukisan dsb. Sedangkan citra tidak tampak misalnya citra digital (data gambar dalam file), citra distribusi panas di kulit manusia, dimana untuk dapat dilihat oleh mata manusia maka harus ditampilkan dalam monitor atau dicetak diatas kertas. Diantara citra tersebut, hanya citra digital yang dapat diolah menggunakan komputer. Apabila ingin melakukan perubahan terhadap citra analog maka citra tersebut dapat didigitalkan dengan cara dipindai (*scan*) menggunakan *scanner*. (Achmad & Firdausy, 2005)

Citra digital merupakan suatu fungsi intensitas cahaya dua dimensi $f(x,y)$, dimana x dan y menunjukkan koordinat spasial. Nilai f pada setiap titik (x,y)menunjukkan tingkat (nilai) warna dari citra pada titik tersebut (Gonzales & Woods, 1993). Citra digital dapat dibayangkan sebagai suatu matriks dimana baris dan kolomnya merepresentasikan suatu titik di dalam citra, dan nilai elemen matriks tersebut menunjukkan nilai warna di titik tersebut. Contoh jika ada suatu citra berukuran 256x256 piksel dengan intetitas beragam pada tiap pikselnya, direpresentasikan secara numerik dengan matriks terdiri dari 256 baris dan 256 kolom. Nilai pikseltersebutberkisar antara 0 sampai 255, dimana 0 menunjukkan intensitas paling gelap, sedangkan 255 paling terang. Pada Gambar 2.1 ditunjukkan representasi citra digital.

$$f(x,y) = \begin{bmatrix} f(1,1) & f(1,2) & \cdots & f(1,b) \\ f(2,1) & f(2,2) & \vdots & f(2,b) \\ \vdots & \cdots & \cdots & \vdots \\ f(a,1) f(a,2) & \cdots & f(a,b) \end{bmatrix}$$

Gambar 2.1 Representasi citra digital

2.1.2 Pengertian Pengolahan Citra

Pengolahan citra merupakan proses pengolahan dan analisis citra yang banyak melibatkan persepsi visual. Proses ini mempunyai ciri data masukan dan informasi keluaran yang berbentuk citra. Istilah pengolah citra digital secara umum didefinisikan sebagai pemrosesan citra dua dimensi dengan komputer. Dalam definisi yang lebih luas, pengolahan citra digital juga mencakup semua data dua dimensi (Jain, 1995). Pengolahan citra digital dapat dikelompokkan dalam dua jenis kegiatan :

1. Memperbaiki kualitas suatu citra, sehingga dapat lebih mudah diintepretasikan oleh mata manusia.
2. Mengolah informasi yang terdapat pada suatu citra untuk keperluan pengenalan obyek secara otomatis.

Bidang aplikasi kedua sangat erat hubungannya dengan pengenalan pola (*pattern recognition*) yang umumnya bertujuan mengenali suatu obyek dengan cara mengekstrak informasi penting yang terdapat pada suatu citra. Bila pengenalan pola dihubungkan dengan pengolahan citra, diharapkan akan terbentuk suatu sistem yang dapat memproses citra masukan sehingga citra tersebut dapat dikenali polanya. Proses ini disebut pengenalan citra atau *image recognition*. Pengolahan citra dan pengenalan pola menjadi bagian dari proses pengenalan citra. Kedua aplikasi ini akan saling melengkapi untuk mendapatkan ciri khas dari suatu citra yang hendak dikenali (Kausari, 2009).

2.1.3 Pengenalan Tulisan Tangan

Pengenalan tulisan tangan adalah proses untuk mengenali tulisan yang ditulis menggunakan alat tulis biasa kedalam betuk digital teks. Penelitian dalam bidang pengenalan tulisan tangan sudah cukup banyak dilakukan, dan mempunyai banyak metode dan strategi (Lecoinet & Baret, 1994). Salah satu faktor penentu adalah mengenai jumlah karakter dan penulis yang mampu dikenali. Pada sistem *online*, umumnya pengenalan dibatasi untuk satu orang saja (*mono scriptor*), yaitu si pengguna, sedang pada sistem *offline*

umumnya bersifat *multi scriptor,* dimana tulisan yang di-*scanning* berasal dari lebih dari satu orang. Hal ini menunjukkan betapa dibutuhkan sistem yang mampu mengenali berbagai gaya penulisan dari masing-masing orang.

Menurut jenis tulisan yang dikenali, sistem dapat dibagi menjadi Pengenalan Karakter (*Optical Character Recognition - OCR*) dan Pengenalan Tulisan Tangan Latin (*Cursive Handwriting*) (Lecoinet & Baret, 1994). Pada pengenalan tulisan tangan bersambung dikenal dengan metode holistik dan analitis, metode holistik mengenali tulisan per-kata, sedangkan metode analitis mencoba membagi kata menjadi bagian yang lebih kecil, kemudian baru masing-masing bagian tersebut dikenali. Karena metode ini melakukan pengenalan dengan membagi kata ke bagian yang lebih kecil, maka metode ini disebut juga metode segmentasi.

Pada metode segmentasi, dapat dilakukan dengan mengenali bersamaan dengan proses segmentasi (*implicit*), dan melakukan pengenalan secara terpisah (*eksplisit*). Metode implicit mempunyai keuntungan dalam hal memanfaatkan informasi hubungan antara karakter yang satu dengan karakter didekatnya, sedang metode eksplisit mempunyai keuntungan bahwa untuk pengenalan dapat menggunakan teknik *OCR* yang sudah ada.

Metode yang digunakan dalam penelitian ini adalah pendekatan analitis dengan segmentasi secara *eksplisit*, dimana proses pengenalan tulisan dilakukan dengan cara memecah citra kedalam segment, kemudian mencoba melakukan pengenalan karakter terhadap segment tersebut menggunakan algoritma Fuzzy ART.

2.2 *Pre-Processing*

2.2.1 Binerisasi

Binerisasi yaitu proses pengolahan citra berwarna menjadi citra biner. Citra biner adalah citra yang hanya mempunyai dua nilai derajat keabuan yaitu hitam dan putih, dimana 0 menyatakan warna latar belakang (*background*) dan 1 menyatakan warna obyek/tinta (*foreground*) atau dalam bentuk angka 0 untuk warna hitam dan angka 255 untuk warna putih (Munir, 2004). Proses binerisasi meliputi operasi *grayscale* pada citra berwarna, kemudian dilanjutkan dengan operasi *threshold* (pengambangan).

2.2.1.1 *Grayscale*

Proses *grayscale* adalahmengubah citra berwarna menjadi hitam putih dengan mengubah warna setiap komponen RGB (*Red, Gren, Blue*) citra menjadi sama (Arica, 2001). Untuk mengubah RGB menjadi citra *grayscale* dapat digunakan Persamaan 2.1 atau Persamaan 2.2.

$$Grayscale = 0{,}299R + 0{,}587G + 0{,}114B \quad (2.1)$$

Atau

$$Grayscale = \frac{R + G + B}{3} \quad (2.2)$$

Perhitungan nilai *grayscale* yang sebenarnya adalah dengan menggunakan Persamaan 2.1. Namun persamaan yang umum digunakan adalah Persamaan 2.2 sehingga pada penelitian ini digunakan persamaan tersebut. Karena lebih mudah untuk digunakan dan diingat (Budhi, 2006).

2.2.1.2 *Thresholding* (pengambangan)

Menurut Diaz dkk (2008), memilih bentuk-bentuk dalam sebuah citra sangat berguna dalam pengukuran atau pemahaman citra. Secara tradisional, pengambangan didefinisikan sebagai proses pendefinisian jangkauan nilai-nilai gelap-terang pada citra yang sebenarnya. Memilih piksel-piksel dalam jangkauan ini sebagai latar depan (*foreground*) dan menolak sisanya sebagai latar belakang (*background*). Dengan demikian, citra terbagi atas dua bagian, yaitu bagian hitam dan bagian putih. Dalam hal ini tidak ada kesepakatan untuk menetapkan warna hitam atau putih untuk obyek yang diamati.

Thresholding mengubah citra warna menjadi citra biner, dimana ditentukan sebuah nilai level *threshold* kemudianpiksel yang memiliki nilai di bawah level *threshold* diset menjadi nilai warna hitam (1 pada nilai biner) dan nilai di atas level *threshold* diset menjadi nilai warna putih (0 pada nilai biner). Proses *threshold* digunakan untuk mengekstrak *foreground* (tinta) dari *background* (kertas) dan menjadikan citra menjadi biner (Arica, 2001). Fungsi pengambangan secara global dinyatakan pada Persamaan 2.3.

$$f_B(i,j) \begin{cases} 1, & f_g(i,j) \leq T \\ 0, & lainnya \end{cases} \quad (2.3)$$

Keterangan :
$f_B(i,j)$: citra biner
$f_g(i,j)$: citra *grayscale*
T : nilai ambang yang dispesifikasikan (*threshold*)

Dengan operasi pengambangan, obyek dibuat berwana gelap (1 atau hitam) sedangkan latar belakang berwarna terang (0 atau putih). Pada Gambar 2.2 ditunjukkan contoh citra *grayscale* dengan 0-255 kombinasi warna dan citra biner dengan 2 kombinasi warna.

Gambar 2.2 Contoh citra *grayscale* dan citra biner

2.2.2 Segmentasi

Segmentasi citra merupakan suatu proses yang membagi citra ke dalam beberapa bagian, yaitu bagian yang diperlukan dan bagian yang tidak diperlukan oleh sistem (Munir, 2004). Segmentasi karakter perlu dilakukan sebelum proses pengenalan karakter. Untuk memudahkan pemisahan karakter, biasanya citra yang akan disegmentasi sudah berupa citra biner (hitam-putih).

2.2.2.1 *Projection Profile-Based Histogram*

Dalam sistem ini, metode segmentasi yang digunakan adalah metode *Projection Profile-Based Histogram* dengan menjadikan huruf atau objek ke dalam bentuk garis-garis histogram vertikal dan horizontal (Rodrigues, 2000). Metode *Projection Profile-Based Histogram* ini berdasarkan pada penggunaan profil histogram yang telah tercipta dan diproporsikan oleh suatu citra. Profil dari proyeksi merupakan sebuah struktur data yang digunakan untuk menyimpan sejumlah piksel hitam yang merupakan objek ketika suatu citra diproyeksikan melalui sumbu x maupun sumbu y. Dengan cara menghitung jumlah piksel hitam pada suatu citra secara mendatar

(horizontal) dan juga secara menurun (vertikal). Metode ini dapat digunakan dalam segmentasi karakter pada tulisan latin. Berikut ini ditunjukkan Persamaan 2.4 yang digunakan untuk melakukan proyeksi histogram dengan mengambil jumlah piksel yang terkait (Rodrigues, 2000).

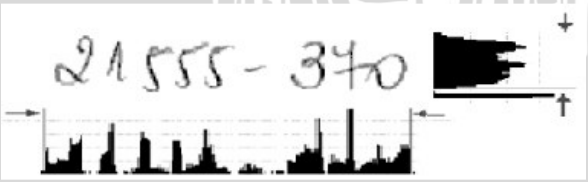
$$X,Y \rightarrow M\,(x,y)$$

$$X_n = \sum_{i=0}^{h} Yi, n \in [0, v]$$

$$Y_n = \sum_{i=0}^{v} Xi, n \in [0, h] \qquad (2.4)$$

Keterangan :
X : sumbu horizontal
Y : sumbu vertikal
h : ketinggian citra untuk x atau lebar bagi y
v : mempresentasikan ukuran dari citra

Menurut Khalifa (2006), pada citra tulisan tangan segmentasi dilakukan dengan menghitung jumlah piksel hitam secara horizontal dan vertikal. Citra yang mengandung karakter akan memiliki lebih banyak piksel hitam daripada latar citra (*background*). Jika jumlah piksel hitam tidak memenuhi nilai yang diinginkan, maka bagian tersebut akan dihilangkan. Bagian yang tidak dihilangkan akan dipisahkan menjadi citra baru hasil dari citra segmentasi. Pada Gambar 2.3ditunjukkan proyeksi jumlah perhitungan piksel hitam secara horizontal dan vertikal pada citra tulisan tangan.



Gambar 2.3 Implementasi histogram dalam segmentasi

2.3 *Processing*

2.3.1 *Feature Extraction*

Pada pengenalan citra, ekstraksi dapat dilakukan untukmempermudah proses pengenalan. Citra yang akan diekstraksi merupakan citra hitam putih dan akan dibagi menjadi beberapa bagian yang diinginkan. Tiap-tiap bagian tersebut akan diekstraksi menjadi nilai tertentu yang dapat mewakilinya. (Lim, 2003).

Pada pengenalan karakter tulisan tangan latin, citra karakter yang telah disegmentasi akan diekstraksi menjadi vektor yang bernilai antara 0 sampai dengan 1. Metode *feature extraction* yang digunakan adalah *zoning feature extraction* yaitu dengan cara mengambil nilai rata-rata nilai suatu citra. Misalkan ukuran asli suatu citra karakter 32x32 dan hasil akhir ukuran citra berdasarkan *zoning* berukuran 8x8. Maka hal itu berarti bahwa nilai 1 piksel pada citra mewakili 4x4 nilai piksel pada citra aslinya. Persamaan 2.5 merupakan persamaan yang digunakan untuk *zoning feature extraction*.

$$V = \frac{1}{16}\sum_{x=1}^{4}\sum_{y=1}^{4} f(x,y) \qquad (2.5)$$

Keterangan :

x : sumbu x pada citra
y : sumbu y pada citra
f(x,y) : reprsentasi citra
V : *zoning feature ectraction*

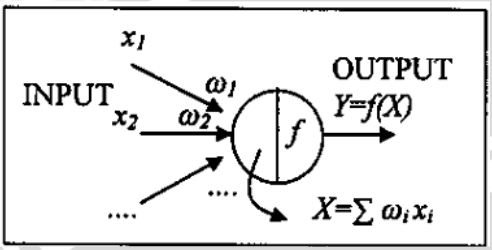
2.4 Jaringan Syaraf Tiruan

2.4.1 Pengertian Jaringan Syaraf Tiruan

Jaringan Syaraf Tiruan (JST) dalam bahasa Inggris dikenal dengan *Artificial Neural Network* (ANN), dimana kata *neural* berasal dari kata *neuron* yang berarti syaraf. JST atau ANN merupakan suatu sistem pemrosesan informasi yang memiliki karakteristik-karakteristik menyerupai jaringan syaraf biologi dalam otak (Fausett, 1994). JST dapat digambarkan sebagai model matematis dan komputasi untuk fungsi aprokmasi nonlinear, klasifikasi data, kluster dan regresi non parametrik atau sebagai sebuah simulasi dari koleksi model syaraf biologi. (Kristanto, 2004).

Cara berpikir otak manusia diterapkan pada JST melalui pengaturan bobot dan koneksi antar neuron yang akan menentukan output. Cara kerja JST adalah menggunakan sekelompok elemen

proses atau *node-node* yang hampir sama dengan otak manusia. *Node-node* ini berhubungan dalam sebuah jaringan yang dapat mengidentifikasi pola yang ditunjukkan dalam kumpulan data.



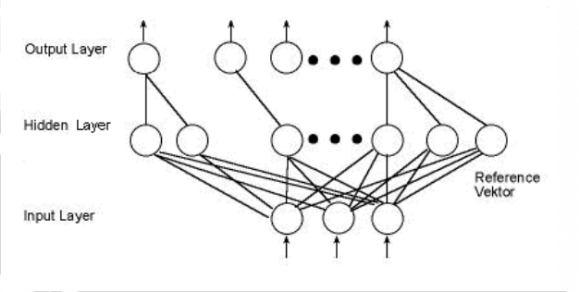
Gambar 2.4 Jaringan Syaraf Tiruan

Gambar 2.5 merupakan proses umum dari Jaringan Syaraf Tiruan. Suatu sinyal input x_i dikalikan dengan bobot w_i, sehingga menjadi $x_i.w_i$, kemudian dijumlahkan semua sinyal yang telah dikalikan dengan bobotnya tersebut ($\sum$). Hasil penjumlahan dinyatakan dalam variable *X*. Dalam unit output variable *X* dimasukkan dalam suatu fungsi *f* tertentu untuk menghasilkan output akhir. Fungsi *f* yang digunakan merupakan fungsi linear atau fungsi-fungsi lain yang lebih kompleks.

2.4.2 Arsitektur Jaringan Syaraf Tiruan

Sebuah Jaringan Syaraf Tiruan umumnya terdiri dari tiga jenis *layer* (lapisan) yaitu *input layer* (lapis masukkan), *hidden layer* (lapis tengah yang tidak terlihat)dan *output layer* (lapis keluaran) (Fausett, 1994). *Input layer* berfungsi untuk menerima informasi dari lingkungan luar, sedangkan *output layer* berfungsi untuk mengkomunikasikan hasil yang diperoleh JST dengan lingkungan luarnya. Diantara *input layer* dan *output layer* terdapat satu atau lebih *hidden layer* yang saling berkomunikasi satu sama lain maupun dengan *output layer*. Disebut *hidden layer* karena memang lapisan ini letaknya tersembunyi dan tidak dapat dilihat oleh pengguna. Pengguna tidak akan mengetahui apa saja yang menjadi masukkan maupun data apa saja yang dihasilkan dari *hidden layer* ini. Namun umumnya jumlah *hidden layer* antara nol sampai dengan tiga lapisan. Hal ini dikarenakan Jaringan Syaraf Tiruan dengan tiga *hidden layer* sudah mencukupi untuk menyelesaikan berbagai permasalahan yang dihadapi (Ibrahim, 2003).

Secara umum ketiga *layer* dalam JST dapat digambarkan seperti pada Gambar 2.5.



Gambar2.5 Susunan layer-layer pada JST

2.4.3 Metode Pembelajaran Jaringan Syaraf Tiruan

Proses pembelajaran Jaringan Syaraf Tiruan diilhami dari pakar yang bernama Kosko. Kosko mengklasifikasikan proses pembelajaran pada Jaringan Syaraf Tiruan dapat dibedakan menjadi dua bagian, yaitu :

1. *Supervised* (terbimbing)

   Proses pembelajaran ini adalah proses yang memasukkan target keluaran dalam data untuk proses pembelajarannya. Pada proses pembelajaran, suatu pola input akan diberikan ke satu neuron pada lapisan masukan. Pola ini akan dirambatkan di sepanjang jaringan syaraf hingga sampai ke neuron pada lapisan keluaran. Lapisan keluaran ini akan membangkitkan pola keluaran yang nantinya akan dicocokan dengan pola targetnya. Apabila terjadi perbedaan pola keluaran dengan pola target, maka dapat dikatakan sebagai *error*. Apabila nilai *error* ini masih cukup besar, maka perlu dilakukan lebih banyak pembelajaran lagi. Ada beberapa metode pembelajaran terbimbing, diantaranya adalah *Single Perceptron, Multi Perceptron, Learning Vector Quantization* (LVQ) dan *Back Propagation.*

2. *Unsupervised* (tak terbimbing)

   Pada proses pembelajaran tak terawasi ini tidak diperlukan target keluaran (*output)*. Pada metode ini tidak dapat ditentukan hasil yang seperti apakah yang diharapkan dalam proses pembelajaran. Selama proses pembelajaran, nilai bobot diproses dalam suatu *range* tertentu tergantung pada nilai *input* yang diberikan. Tujuan pembelajaran ini adalah mengelompokkan

unit-unit yang hampir sama dalam suatu area tertentu. Contoh metode untuk model ini yaitu SOM-Kohonen (*Self Organizing Map*) dan ART (*Adaptive Resonance Theory*).

2.5 Adaptive Resonance Theory (ART)

Adaptive Resonance Theory (ART) yang dikembangkan oleh Carpenter dan Grossberg (1988), mempunyai berbagai keunggulan dibandingkan jenis-jenis yang lain. ART adalah salah satu network dengan *unsupervised learning*. ART juga dikembangkan pula untuk network dengan *supervised learning* seperti pada ARTMAP.

Jaringan Syaraf Tiruan pada umumnya tidak dapat mempelajari pola masukan baru setelah selesai pelatihan, jika ada pola baru maka jaringan harus dilatih terhadap pola yang lama dan yang baru. Hal ini mengilhami Carpenter dan Grossberg membuat suatu jaringan yang dapat memecahkan masalah tersebut, jaringan ini disebut *Adaptive Resonance Theory* (ART). Grossberg merumuskan permasalahan diatas sebagai *plasticity stability dilemma*, yang dapat diatasi oleh ART dengan cara menambahkan mekanisme umpan balik antara *input layer* dan *competitive layer.*

2.5.1 Match-Based Learning

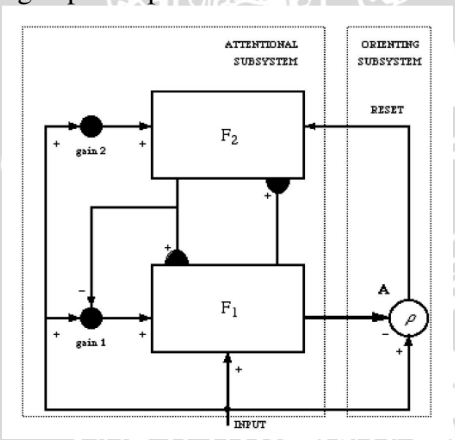
ART memiliki kelebihan dalam hal *increment learning*, yaitu jaringan dapat belajar terhadap data masukan yang baru, tanpa melupakan data yang lama(Carpenter, 1991). Kemampuan ART dalam *increment learning* disebabkan karena ART menggunakan teknik *match-based learning* dalam proses pembelajaran. Pada *match-based learning*, saat pembelajaran proses penyesuaian bobot terjadi pada kondisi resonansi, dimana vector masukan dianggap mendekati vector pewakil atau vector masukan dianggap sama sekali baru sehingga dibuat kategori baru. Pada proses penyesuaian bobot, hanya bobot yang berkaitan dengan neuron pemenang pada *competitive layer* (F2) saja yang di update, bobot yang lain tetap, hal ini menyebabkan ART dapat bersifat adaptif dalam mempelajari vector masukan yang baru tanpa melupakan ingatan sebelumnya. Proses pembelajaran dihentikan jika semua vector masukan telah berhasil dimasukan kedalam kategori yang benar, dimana kedekatan vector masukan dengan vector pewakil diatur oleh *vigilance parameter*.

2.5.2 Cara Kerja ART

Sistem ART terdiri dari 2 subsistem, yaitu *attentional subsystem* dan *orienting subsystem*. *Attentional subsystem* terdiri dari F_1 dan F_2 - *layer, gain control* 1 dan 2. Kedua *layer* ini saling terhubung dengan *weight* koneksi. *Orienting subsystem* yang terdiri dari *reset layer* diperlukan untuk menstabilkan proses dan pembelajaran.

Kegunaan dari F_1 *layer* adalah untuk membandingkan pola *input* dengan pola yang diharapkan pada F_2 *layer*. Jika pola tidak cukup dekat (*closely matched*) maka *orienting subsystem* akan mengirimkan sinyal *reset* ke F_2*layer*. F_1*layer* dikenal juga sebagai *input layer* atau *comparison layer*.

Reset adalah sebuah sinyal yang dikirim kepada F_2*layer*, yang menonaktifkan node F_2 yang aktif selama periode presentasi *input* tersebut. Sinyal *reset* ini dikarenakan tidak cukup miripnya node pada F_2*layer* dengan pola input tersebut.



Gambar 2.6 Perceptron pada ART

Seperti yang dapat dilihat pada Gambar 2.6, F_1 menerima *input* dari 3 sumber yang mungkin. Ketiga sumber tersebut adalah *bottom-up input* ke F_1, *top-down input* dari F_2 dan sinyal *gain control*.

F_2*layer* adalah *competitive layer* (*winner take-all*) yang mencari dan memberitahukan *winning node* (node yang memiliki nilai maksimum pada F_2 *layer*) kepada F_1. F_2 *layer* disebut juga *output* atau *recognitionlayer*. Parameter *vigilance*, ρ, adalah untuk menetukan kapan sinyal *reset* harus dilakukan dengan menentukan tingkat kemiripan antara sebuah *cluster* (pola yang sudah tersimpan) dengan pola *input* yang masuk.

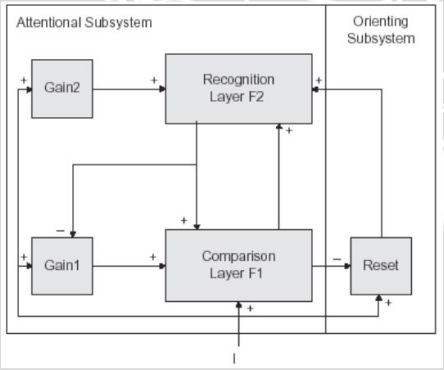
Istilah resonance pada ART (*Adaptive resonance Theory*) mengacu pada keadaan resonansi pada jaringan dimana sebuah node pada F_2*layer* cukup mirip dengan pola input yang ada sehingga *orienting subsystem* tidak mengirimkan sinyal reset ke F_2*layer*. Pada keadaan ini, pola input yang sedang diproses pada F_1 menyebabkan node F_2 yang telah ada terpilih, yang kemudian akan mengirim node tersebut ke F_1, yang sekali lagi cocok/mirip dengan pola input tersebut. Jaringan belajar hanya pada keadaan resonansi. ART mampu untuk membangun clustering yang stabil dari urutan pola input.

Beberapa kelebihan dari ART adalah sebagai berikut :

1. Dapat mengatasi plasticity-stability dilemma
2. Proses pembelajaran yang real-time
3. Proses pencarian untuk best match yang cepat
4. Dapat belajar dengan cepat

2.5.3 ART-1

ART-1 adalah anggota keluarga ART yang pertama yang dirancang untuk mengatasi masalah dalam mengenali pola *input biner*. ART-1 termasuk jaringan *unsupervised*. Jaringan ART-1 secara otomatis mengatur dan menstabilkan pengenalan kode, tidak tergantung dari banyaknya dan kerumitan pola *input biner*.



Gambar 2.7 Model jaringan ART-1

ART-1, menurut Carpenter dan Grossberg, 1988, menawarkan suatu solusi adanya dilemma pada *neural network* yaitu dengan *stability-plasticity*. Dengan mempunyai *stability-plasticity* ini, ART-1 dapat terus melakukan proses pembelajaran dengan masih mempertahankan fleksibilitasnya (*plastic*) atau adaptif untuk

menanggapi masukan yang masih relevan dengan apa yang sudah dikenalnya, tetapi tetap stabil terhadap masukkan yang tidak relevan.

ART-1 terdiri dari 5 modul fungsional: dua lapis neuron (lapis perbandingan dan lapis pengenalan), gain1 dan gain2, dan reset untuk mengendalikan antara untuk pembelajaran atau untuk klasifikasi (Wasserman, 1989). Jumlah neuron untuk lapis pembandingan, atau lapis *input*, sama dengan jumlah elemen dari vector data *input*. Setiap neuron dari lapis ini semuanya dihubungkan ke setiap neuron di lapis pengenalan atau lapis *output*. Penghubungan ini dikenal dengan nama hubungan *bottom-up* dan *top-down*. Jumlah neuron di lapis *output* yang diperlukan adalah sebanyak kelas atau kelompok yang diharapkan akan terjadi. *Output* dari lapis *output* ini juga kemudian diumpanbalikkan ke lapis *input*.

Pada saat data *input* dimasukkan, vector data *input* pertama membentuk kelas pertama. Neuron pertama dilapis *output* disesuaikan untuk mengidentifikasi masukan pertama tersebut. Matrik pembobot (*weight matric*) pada kedua lapisan disesuaikan dengan data masukan untuk kelas pertama tadi yang dikaitkan dengan neuron lapis *output* pertama. Ketika vector *input* berikutnya masuk, ini dibandingkan dengan kelas pertama tadi. Jika *input*-nya mempunyai derajat kemiripan yang diinginkan (lebih dari atau sama dengan *vigilance parameter,* ρ) maka *input* kedua tadi dimasukkan ke kelas pertama dan matrik pembobot disesuaikan lagi. Jika *input*-nya tidak mirip ($<\rho$), maka *input* tersebut menjadi/membentuk kelas kedua, yang dikaitkan dengan neuron lapis *output* kedua. Proses ini diulang untuk semua data *input*, dan jumlah kelas akan bertambah.

2.5.4 Fuzzy ART

Fuzzy ART merupakan pengembangan dari ART-1 dengan mengintegrasikan *Fuzzy Logic* ke sistem tersebut, dimana operasi himpunan pada ART-1 diganti menjadi operasi *fuzzy*. Fuzzy ART mampu menerima semua jenis data masukan baik yang berupa biner maupun data kontinyu (antara 0 dan 1).

Fuzzy ART biasa digunakan untuk melaksanakan proses pengenalan serta pengkategorian pola dan sekaligus memetakan setiap kategori pola dengan hasil keluaran yang sesuai. Selain itu Fuzzy ART mampu untuk diterapkan baik secara *online* maupun *offline*. Dalam pengoperasian secara *online* jaringan harus memproses data tepat ketika data tersebut diterima tanpa harus menyimpan atau menggunakan data tersebut. Sedangkan dalam

pengopersiannya secara *offline,* data tersimpan dan secara berulang ditampilkan atau digunakan oleh jaringan.

Dalam Fuzzy ART terdapat dua subsystem utama yakni *attentional subsystem* dan *orienting subsystem*. *Attentional subsystem* digunakan untuk mengenali dan menklasifikasikan pola sebelumnya yang telah dipelajari, sementara *orienting subsystem* diaktifkan jika ART menemui pola baru. Dalam kondisi seperti ini *orienting subsystem* akan menutup setiap usaha yang bertujuan untuk meneruskan pencocokan dengan pola yang telah disimpan dan membuat kategori baru untuk pola masukan tersebut. Parameter penting yang terdapat dalam Fuzzy ART, yakni *vigilance parameter* (ρ). *Vigilance parameter* digunakan untuk mengatur kedekatan atau tingkat kemiripan suatu pola vector masukan dengan vector pewakil.

2.5.5 Algoritma Fuzzy ART

Jenis masukan untuk algoritma Fuzzy ART dapat berupa data biner maupun data kontinyu (antara 0 dan 1). Apabila data yang digunakan dalam bentuk data kontinyu, maka data masukan diubah dalam bentuk *complement code,* dan hal ini sangat penting untuk keberhasilan suatu Fuzzy ART. Pada Persamaan 2.6 ditunjukkan proses perubahan data masukan ke dalam bentuk *complement* code. Untuk Fuzzy ART jika data masukannya adalah $I = (a_1,\ldots,a_M)$, maka

$$I = (a, a^c) = (a_1,\ldots,a_M,a_1^c,\ldots a_M^c),$$

dimana

$$a_i \in [0,1] \text{dan } a_i^c = 1 - a_i \; ; 1 \leq i \leq M \quad (2.6)$$

Vector pengaktif F_1 ditandai dengan $x = (x_1,\ldots, x_{2M})$ dan vector pengaktif F_2 dengan kategori sebanyak Nditandaidengan $y = (y_1\ldots, y_N)$. Vector bobot dari F_2 ditandai dengan $w_j = (x_{j1},\ldots,x_{jM})$. Inisialisasi vector bobot awal, w_j diset bernilai 1, seperti yang ditunjukkan pada Persamaan 2.7.

$$w_{j1}(0) = \cdots = w_{j2M}(0) = 1, \quad (2.7)$$

Keterangan :

j = masing-masing kategori *node* ($j = 1,\ldots,N$)

w_{ji} = vektor bobot ($w_j \equiv x_{j1},..., x_{jM}$)

Pergerakan Fuzzy ART sangat ditentukan oleh *choice parameter* $\alpha > 0$; *learning parameter* $\beta \in [0,1]$ dan *vigilance parameter* $\rho \in [0,1]$; $0 < \rho < 1$. Pengkategorian vektor-vektor dalam Fuzzy ART didasarkan pada dua kriteria yakni *choice* dan *match*.

Fungsi *choice* didefinisikan pada Persamaan 2.8.

$$T_j(I) = \frac{|I \wedge w_j|}{\alpha + |w_j|} \qquad (2.8)$$

Keterangan:

α = konstanta dengan nilai kecil (mendekati 0)

w_j = vector bobot yang berhubungan dengan kategori *j*.

^ = operator *fuzzy* AND yang memenuhi persamaan $(p \wedge q)_i \equiv \min(p_i, q_i)$

$|.|$ = didefinisikan sebagai norm, misalkan $|x| = \sum_{i=1}^{n} x_i$

Nilai-nilai T_j di F_2 akan diproses, sehingga hanya satu nilai T_j yang terpilih. Pemilihan kategori mengikuti persyaratan pada Persamaan 2.9.

$$T_j = \max(T_j); \quad j = 1,....N \qquad (2.9)$$

Jika terdapat lebih dari satu T_j yang maksimal, maka yang dipilih adalah T_j dengan indeks yang terkecil.

Setelah mendapatkan nilai T_j pemenang maka langkah berikutnya adalah mencari nilai dari fungsi *match*. Untuk vektor masukan *x* dan kategori *j*, maka funsi *match* didefinisikan pada Persamaan 2.10.

$$S_j(I) = \frac{|I \wedge w_j|}{|I|} \qquad (2.10)$$

Fungsi *match* digunakan untuk membandingkan $T_j$pemenang dengan pola *input* untuk mengetahui bahwa pola *input x* bagian dari kelas tertentu. Jika $S_j \geq \rho$, dengan nilai *vigilance* yang telah ditentukan maka terjadi resonansi. Jika tidak, maka terjadi *mismatch reset.*

Proses pencarian kategori untuk *J* dilakukan dengan memenuhi kedua kriteria diatas. Apabila keseluruhan proses telah berhasil

dilaksanakan, maka vektor bobot (w_j) diatur sehingga memenuhi Persamaan 2.11.

$$w_j^{(new)} = \beta \left(I \wedge w_j^{(old)} \right) + (1 - \beta) w_j^{(old)} \quad (2.11)$$

2.6 Tingkat Akurasi Pengenalan Tulisan Tangan

Bagian penting dalam pengenalan adalah menghitung tingkat akurasi pengenalan tersebut. Dalam penelitian ini tingkat akurasi digunakan dalam menghitung beberapa akurasi, yaitu akurasi segmentasi karakter, dan akurasi pengenalan karakter (Lim, 2003).

Persamaan yang digunakan untuk menghitung tingkat akurasi dan tingkat kesalahan dapat dilihat seperti pada Persamaan 2.12.

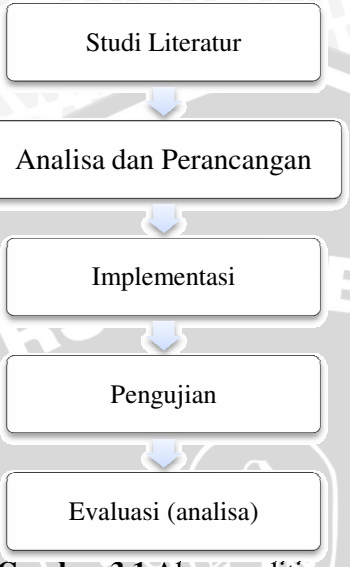
$$Tingkat\ akurasi = \frac{jumlah\ pengenalan\ yang\ akurat}{jumlah\ total\ pengenalan} \times 100\% \quad (2.12)$$

BAB III
METODOLOGI DAN PERANCANGAN

Pada bab ini dibahas tentang metodologi dan perancangan yang digunakan dan seluruh tahapan yang digunakan dalam penelitian ini. Tahapan-tahapan yang harus dilakukan dalam penelitian ini dapat diidentifikasikan sebagai berikut :

1. Mencari dan mempelajari literatur-literatur yang berkaitan dengan permasalahan pengenalan tulisan tangan latin menggunakan metode Fuzzy ART (*Adaptive Resonance Theory*) yang berasal dari buku dan sumber lain dari internet.
2. Mempersiapkan data citra tulisan tangan latin yaitu berupa data *training* dan data *testing*. *Pre-processing* data citra tulisan tangan latin agar siap diolah. Citra tulisan tangan yang digunakan berbentuk citra *grayscale* dan berukuran 350 x 150 piksel.
3. Mengimplementasikan metode pengenalan tulisan tangan latin dengan metode Fuzzy ART (*Adaptive Resonance Theory*) menggunakan data training yang telah disiapkan sebagai proses pembelajaran yang akan digunakan dalam pengenalan ke sebuah sistem perangkat lunak.
4. Melakukan uji coba terhadap hasil pengenalan yang dilakukan oleh sistem. Hasil pengenalan merupakan karakter dari tulisan yang diuji.
5. Mengevaluasi dan menganalisis hasil pengenalan yang dihasilkan oleh sistem berupa tingkat akurasi dari proses segmentasi sampai dengan pengenalan tulisan tangan.

Tahapan-tahapan yang telah dijelaskan dapat digambarkan dengan sebuah diagram alur yang ditunjukkan pada Gambar 3.1.



Gambar 3.1 Alur penelitian

3.1 Analisis Sistem

Pada subbab analisis sistem akan dibahas berbagai hal yang diperlukan dalam proses pengenalan tulisan tangan latin dengan menggunakan metode Fuzzy ART.

3.1.1 Deskripsi Sistem

Pada sistem ini akan diimplementasikan mengenai proses yang digunakan untuk mengenali citra tulisan tangan latin seseorang dengan metode Fuzzy ART. Sistem yang dibangun ini diharapkan mampu mengenali karakter yang terdapat dalam citra. Dalam proses pengenalan tulisan tangan latin terbagi menjadi dua tahap, tahap *pre-processing* dan tahap *processing*.

Dalam tahap *pre-processing,* akan dilakukan proses pengolahan citra untuk mempermudah proses pengenalan karakter. Proses ini melalui beberapa tahap, antara lain mengubah citra masukan menjadi citra biner dan melakukan segmentasi karakter. Hasil dari proses ini adalah berupa citra masing-masing karakter.

Tahap *processing* merupakan proses lanjutan dari tahap *pre-processing*. Citra karakter hasil tahap *pre-processing* akan diubah dalam bentuk vektor melalui tahap *scaling* dan *zoningfeature extraction* sebelum dikenali menggunakan metode Fuzzy ART.

3.1.2 Deskripsi Data

Data *input* terhadap sistem terdiri dari 2 macam. Data pertama adalah citra data latih (*training set*) dan data kedua adalah citra data uji (*testing set*). Citra yang terdapat dalam data latih merupakan kumpulan citra alphabet (huruf kecil) yang akan dilatih oleh sistem.

Pada penelitian ini, citra tulisan tangan yang menjadi data latih (*training set*) berasal dari 5 variasi tulisan tangan latin dari orang yang sama. Citra tulisan tangan ini terdiri atas 26 jenis huruf. Karakter lain selain huruf tersebut tidak termasuk dalam data latih. Pada tahap pelatihan dilakukan pembelajaran per karakter dan hasilnya disimpan dalam sebuah file teks dengan berformat *.txt. File eksternal ini terdiri dari file sampel dan file bobot yang masing-masing berisi karakter serta representasi vektor dari karakter tersebut.

Pada tahap pengenalan, citra yang digunakan sebagai data uji (*testing set*) merupakan citra tulisan tangan latin (bersambung), tegak, tidak ditulis miring (*slant*) dan dalam bahasa Indonesia. Data uji yang digunakan berjumlah 110 kata yang bersal dari tulisan orang yang sama. Tabel 3.1 merupakan daftar karakter yang akan dikenali dalam penelitian ini.

Tabel 3.1 Daftar karakter

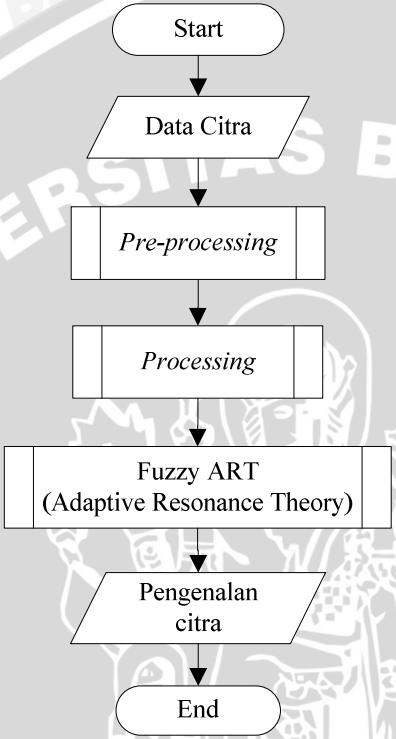
Keterangan	Daftar Karakter	Jumlah
Huruf Kecil	a, b, c,.....z	26 jenis

3.1.3 Analisis Kebutuhan Sistem

Analisis kebutuhan ini didasarkan pada pembangunan sistem pengenalan tulisan tangan latin dengan Fuzzy ART yang baik serta memudahkan *user* dalam mengakses sistem tersebut nantinya. Kebutuhan perangkat lunak meliputi hal-hal sebagai berikut :

1. Sistem harus dapat mengolah citra hingga didapatkan representasi vektor yang mewakili karakter tertentu dalam citra tersebut.
2. Sistem harus dapat mengenali tulisan tangan latin pada citra yang telah melalui tahap pertama dengan baik sesuai dengan algoritma yang telah ditentukan.
3. *Interface* sistem yang dibangun harus bersifat *user friendly* atau dapat dioperasikan dengan mudah oleh *user*.

Flowchart proses umum yang terdapat pada sistem dapat dilihat pada Gambar 3.2. Proses dalam sistem ini terdiri atas proses-proses yang saling berkaitan antara proses yang satu dengan yang lain.



Gambar 3.2 Flowchart proses umum dalam sistem

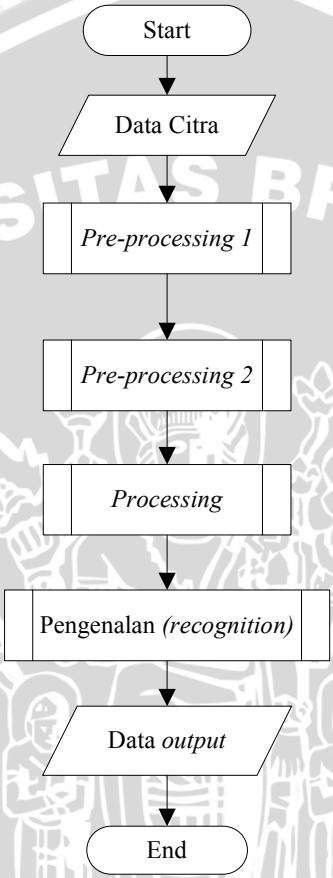
3.2Perancangan Perangkat Lunak

Berdasarkan analisa yang telah dilakukan, maka tahap selanjutnya adalah merancang pembangunan perangkat lunak. Pada subbab ini akan dibahas proses-proses yang terjadi dalam perangkat lunak, dimana proses-proses tersebut memiliki keterkaitan penting antara satu dengan yang lainnya. Selain itu akan dibahas juga mengenai perancangan *interface* sistem dan perancangan uji coba.

3.2.1 Perancangan Proses

Secara garis besar, proses dalam perancangan sistem terdiri dari 5 bagian. Proses tersebut adalah *pre-processing 1, pre-processing 2* (segmentasi), *processing* dan *recognition* (pengenalan). Tahap *pre-*

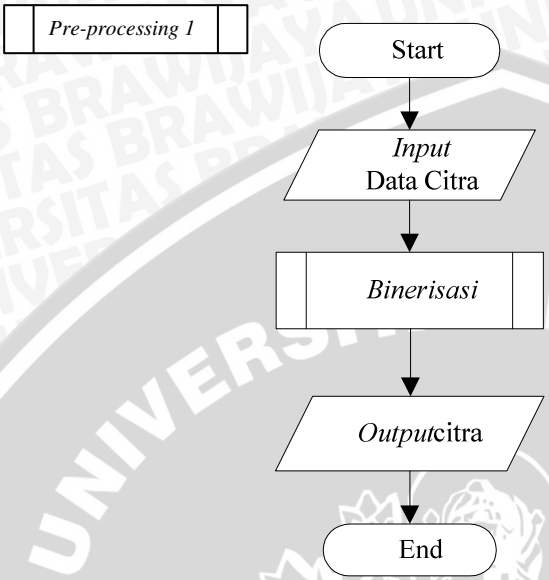
processing 2 hanya akan diterapkan pada citra data uji. Urutan dari proses-proses tersebut ditunjukkan oleh flowchart pada Gambar 3.3.



Gambar 3.3 Flowchart urutan proses dalam sistem

3.2.1.1 *Pre-Processing* 1

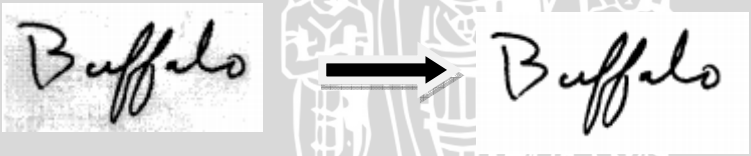
Tahap ini bertujuan untuk mempersiapkan dan memodifikasi citra untuk meningkatkan kualitas dari citra dengan cara menerapkan proses binerisasi. Citra yang menjadi data *input,* pertama kali akan melalui tahapan ini sebelum masuk ke tahap selanjutnya. Urutan proses yang terdapat dalam tahap *pre-processing 1* seperti ditunjukkan pada Gambar 3.4.



Gambar 3.4 Flowchart urutan proses *pre-processing 1*

a. Binerisasi

Tahapan awal dalam penelitian pengenalan tulisan tangan latin adalah pengubahan citra masukan menjadi citra biner. Hal ini karena citra biner akan digunakan pada tahap selanjutnya. Pada Gambar 3.5 ditunjukkan ilustrasi proses binerisasi citra hasil *scanning*.

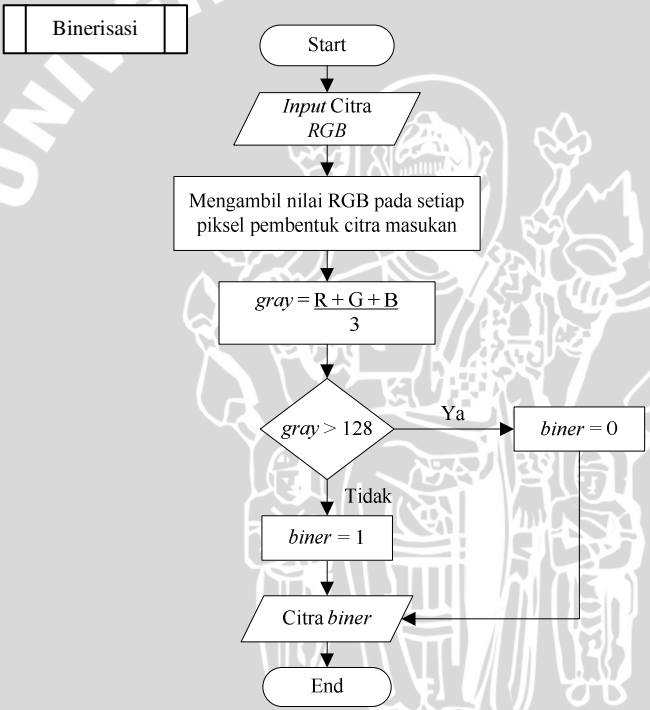


Gambar 3.5 Ilustrasi proses binerisasi

Hasil dari proses binerisasi adalah setiap piksel di dalam citra dipetakan ke dua nilai, 1 atau 0. Dengan operasi pengambangan (*thresholding*), obyek dibuat berwarna gelap (1 atau hitam) sedangkan latar belakang berwarna terang (0 atau putih). Dalam penelitian ini, nilai ambang (*threshold*) yang digunakan yaitu 128. Berikut merupakan langkah-langkah binerisasi :

1. Masukkan citra berwarna atau *grayscale*.

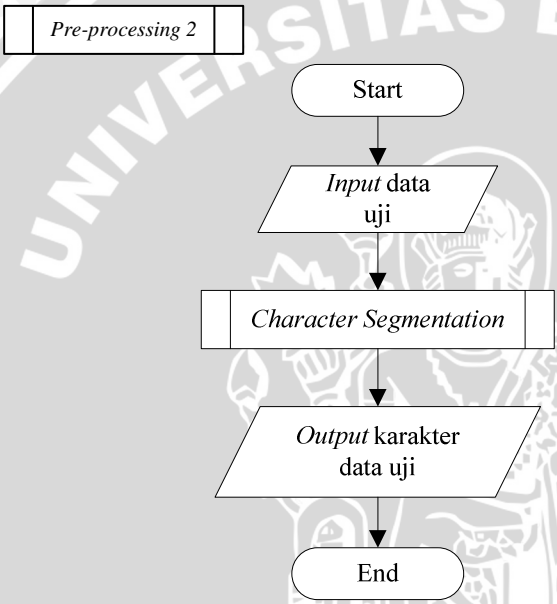
2. Mangambil nilai RGB (*Red, Green, Blue*) pada setiap piksel yang membentuk citra.
3. Mencari nilai *grayscale* dengan menerapkan Persamaan 2.2.
4. Memeriksa apakah nilai *grayscale* lebih dari 128 atau sebaliknya.
5. Jika nilai *grayscale* lebih dari 128 maka piksel tersebut diubah menjadi citra biner bernilai 0 (putih), begitu juga sebaliknya, jika nilai *grayscale* kurang dai 128 maka piksel tersebut diubah menjadi citra biner bernilai 1 (hitam). Proses binerisasi citra RGB ditunjukkan oleh Gambar 3.6.



Gambar 3.6 Flowchart proses binerisasi

3.2.1.2 *Pre-Processing* 2

Pada penelitian ini, proses segementasi adalah memisahkan huruf per huruf yang akan dikenali dari sebuah citra hasil scan kata. Berdasarkan pada perancangan subbab 3.2.1, disebutkan jika data masukan merupakan data uji, maka akan dilakukan tahap *pre-processing 2* terhadap citra. Pada tahap ini terdapat proses *character segmentation* yang digunakan untuk mendapatkan citra per karakter. Proses *pre-processing 2* ditunjukkan pada Gambar 3.7.



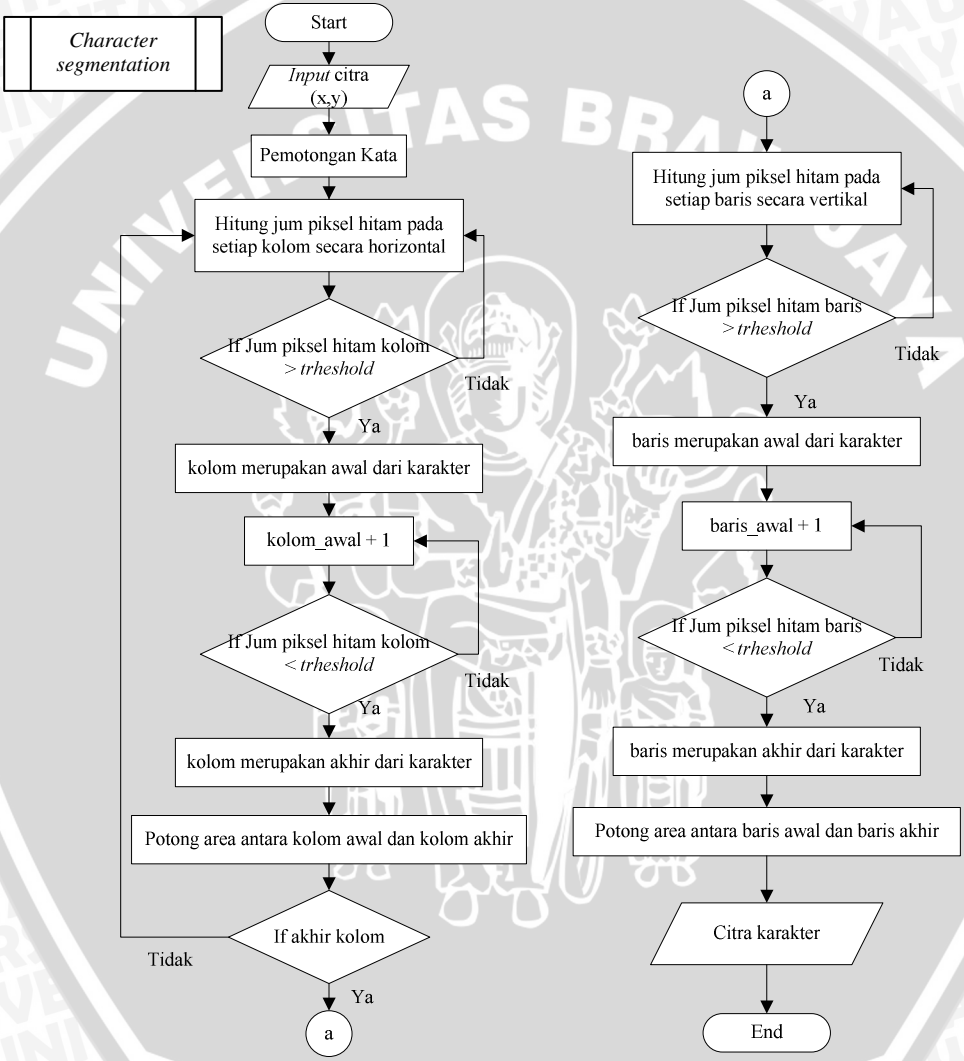
Gambar 3.7 Flowchart urutan proses dalam *pre-processing 2*

a. *Character Segmentation*

Character segmentation merupakan proses yang digunakan untuk mencari karakter pada suatu citra data uji. Proses segmentasi karakter bertujuan untuk mendapatkan citra dari setiap karakter yang ada pada tulisan tangan latin yang nanti akan diproses dalam proses pengenalan.

Metode yang digunakan dalam proses ini adalah dengan menghitung jumlah piksel hitam pada citra tulisan tangan latin. Sebelumnya citra tulisan tangan latin telah diubah menjadi citra biner. Citra biner tulisan tangan latin akan diproyeksikan secara horizontal dan vertikal. Jika jumlah piksel hitam pada baris atau kolom tersebut kurang dari nilai ambang (*threshold*), maka baris atau

kolom itu dianggap bukan sebagai bagian dari karakter. Sebaliknya bagian yang nilainya di atas nilai ambang (*threshold*) akan diambil sebagai citra karakter. Proses *character segmentation* citra tulisan tangan latin ditunjukkan oleh Gambar 3.8.



Gambar 3.8 Flowchart proses *character segmentation*

3.2.1.3 *Processing*

Sebelum tahap pengenalan, citra hasil *pre-processing* 1 dan 2 akan melalui tahap *processing*. Pada tahap ini, semua hasil karakter yang diperoleh diubah ukurannya menjadi 32x32. Kemudian untuk mendapatkan nilai unik dari suatu citra, maka dilakukan *feature extraction.* Proses ini dilakukan terhadap seluruh karakter dalam data latih dan data uji.

a. ***Scaling***

Setelah didapat batasan pada tiap karakter, maka karakter tersebut dilanjutkan dengan proses *scaling*. Pada tahap ini, semua hasil karakter yang diperoleh diubah ukurannya menjadi 32x32 untuk menyamakan kategori masukan bagi FuzzyART.

Dalam proses perubahan dari ukuran asli citra menjadi ukuran 32x32 dilakukan dengan perbandingan rasio antara tinggi dengan lebarnya.

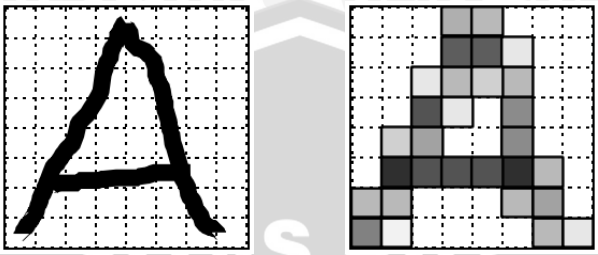
Untuk langkah selanjutnya proses *scaling* citra dilakukan dengan menggunakan rasio yang telah didapat. Teknik *scalling* citra yang dipakai dengan menggunakan fungsi dari Objek Image pada Delphi. Berikut ini pada Gambar 3.9 ditunjukkan contoh karakter citra asli dengan karakter hasil *scaling*.

Ukuran : 40 x 35 Ukuran : 32 x 32

Gambar 3.9 Ilustrasi proses *scaling*

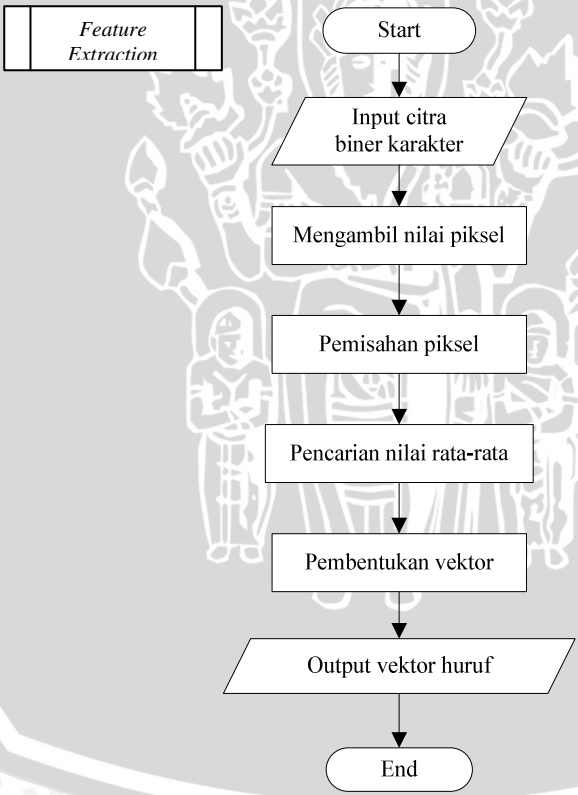
b. ***Zoning Feature Extraction***

Feature extraction merupakan proses untuk mendapatkan nilai-nilai unik dari suatu citra yang telah melakukan *pre-processing* 1dan 2. Nilai-nilai unik tersebut direpresentasikan untuk dijadikan sebagai model masukan pada Fuzzy ART. Pola data masukan yang diterima meliputi nilai yang mempunyai rentangan nilai antara 0 sampai 1. *Feature extraction* yang digunakan dalam penelitian ini adalah *zoning feature extraction*. Pada Gambar 3.10 ditunjukkan ilustrasi proses *zoning feature extraction*.



Gambar 3.10 Ilustrasi proses *zoning*

Nilai akhir yang diperoleh akan terdiri dari 64 nilai yang bernilai antara 0 sampai 1. Nilai-nilai tersebut nantinya akan dijadikan sebagai masukan pada Fuzzy ART. Urutan proses *zoning* dijelaskan pada Gambar 3.11.



Gambar 3.11 Flowchart proses *Feature Extraction*

Dalam proses ini, terdapat 4 sub proses yang akan dilakukan, yaitu :

1. Mengambil nilai piksel dari citra

   Citra masukan pada sub proses ini adalah citra biner karakter berukuran 32x32. Jika piksel pada citra berwarna putih, maka nilai piksel tersebut adalah 0. Sebaliknya jika piksel pada citra berwarna hitam, maka nilai piksel adalah 1.

2. Pemisahan piksel

   Pada sub proses ini, piksel-piksel yang telah di dapat dari citra karakter akan dipisah berdasarkan ukuran yang telah ditetapkan pada *zoning feature extraction*, yaitu sebesar 8x8 area. Jika citra berukuran 32x32, maka setiap area akan mewakili 4x4 nilai piksel pada citra aslinya.Ilustrasi dari *zoning feature extraction* dapat ditunjukkan pada Tabel 3.2 dengan menggunakan contoh ilustrasi karakter huruf 'a' dengan ukuran citra 32x20.

Tabel 3.2 Ilustrasi pemisahan piksel dari citra karakter 32x20

Data piksel dari citra karakter	**Pemisahan piksel**
00000000000111111111111100000000	0000 0000 0001 1111 1111 1111 0000 0000
00000000010000000000000010000000	0000 0000 0010 0000 0000 0000 1000 0000
00000000100000000000000010000000	0000 0000 0100 0000 0000 0000 1000 0000
00000001000000000000000010000000	0000 0000 1000 0000 0000 0000 1000 0000
00000001000000000000000010000000	
00000001000000000000000010000000	0000 0001 0000 0000 0000 0000 1000 0000
00000010000000000000000010000000	0000 0001 0000 0000 0000 0000 1000 0000
00000100000000000000000010000000	0000 0010 0000 0000 0000 0000 1000 0000
00001000000000000000000010000000	0000 0100 0000 0000 0000 0000 1000 0000
00010000000000000000000010000000	
00100000000000000000000010000000	0000 1000 0000 0000 0000 0000 1000 0000
01100000000000000000000010000000	0001 0000 0000 0000 0000 0000 1000 0000
10100000000000000000000010000000	0010 0000 0000 0000 0000 0000 1000 0000
00010000000000000000000010000000	0110 0000 0000 0000 0000 0000 1000 0000
00010000000000000000000010000000	
00001000000000000000000010000000	1010 0000 0000 0000 0000 0000 1000 0000
00001000000000000000000010000000	0001 0000 0000 0000 0000 0000 1000 0000
00000100000000000000000010000000	0001 0000 0000 0000 0000 0000 1000 0000
00000010000000000000000011000011	0000 1000 0000 0000 0000 0000 1000 0000
00000001111111111111111110011000	
	0000 1000 0000 0000 0000 0000 1000 0000
	0000 0100 0000 0000 0000 0000 1000 0000
	0000 0010 0000 0000 0000 0000 1100 0011
	0000 0001 1111 1111 1111 1111 1001 1000

3. Pencarian nilai rata-rata

Proses ini bertujuan untuk mencari nilai rata-rata pada tiap-tiap area yang telah didapatkan dari sub proses sebelumnya. Misalkan untuk contoh sebelumnya, hasil dari pemetaan pada area untuk citra berukuran 32x20 ditunjukkan pada Tabel 3.3.

Tabel 3.3 Proses pencarian nilai rata-rata pada tiap area

Pemisahan piksel								**Nilai rata-rata pada tiap area**							
0000	0000	0001	1111	1111	1111	0000	0000								
0000	0000	0010	0000	0000	0000	1000	0000	0	0	0.25	0.25	0.25	0.25	0.19	0
0000	0000	0100	0000	0000	0000	1000	0000								
0000	0000	1000	0000	0000	0000	1000	0000	0	0.25	0	0	0	0	0.25	0
1	2	3	4	5	6	7	8								
0000	0001	0000	0000	0000	0000	1000	0000	0.25	0.06	0	0	0	0	0.25	0
0000	0001	0000	0000	0000	0000	1000	0000								
0000	0010	0000	0000	0000	0000	1000	0000	0.25	0.06	0	0	0	0	0.25	0
0000	0100	0000	0000	0000	0000	1000	0000								
9	10	11	12	13	14	15	16	0	0.25	0.25	0.25	0.25	0.42	0.19	0.19
0000	1000	0000	0000	0000	0000	1000	0000								
0001	0000	0000	0000	0000	0000	1000	0000								
0010	0000	0000	0000	0000	0000	1000	0000								
0110	0000	0000	0000	0000	0000	1000	0000								
17	18	19	20	21	22	23	24								
1010	0000	0000	0000	0000	0000	1000	0000								
0001	0000	0000	0000	0000	0000	1000	0000								
0001	0000	0000	0000	0000	0000	1000	0000								
0000	1000	0000	0000	0000	0000	1000	0000								
25	26	27	28	29	30	31	32								
0000	1000	0000	0000	0000	0000	1000	0000								
0000	0100	0000	0000	0000	0000	1000	0000								
0000	0010	0000	0000	0000	0000	1100	0011								
0000	0001	1111	1111	1111	1111	1001	1000								
33	34	35	37	38	39	40	42								

Setelahdiketahui hasil pemetaan tiap area, maka dilakukan perhitungan nilai rata-rata tiap area dengan cara menjumlahkan piksel bernilai 1 pada tiap area dan membaginya dengan 16. Sehingga dari contoh tersebut akan terdapat 42 nilai yang nantinya dijadikan sebagai masukan pada Fuzzy ART.

4. Pembentukan vector

Hasil dari sub proses akan diolah untuk pembentukan vector. Nilai yang didapat dari perhitungan rata-rata proses

sebelumnya akan dijadikan deret vector masukan Fuzzy ART. Misalkan proses ini diterapkan terhadap hasil proses sebelumnya, maka representasi vector sepanjang 42 seperti ditunjukkan pada Gambar 3.12.

0 0 0.25 0.25 0.25 0.25 0.19 0 0 0.25 0 0 0 0 0.25 0 0.25 0.06 0 0 0 0
0.25 0 0.25 0.06 0 0 0 0 0.25 0 0 0.25 0.25 0.25 0.25 0.42 0.19 0.19

Gambar 3.12 Representasi vektor

Nilai vector ini akan disimpan dalam sebuah record yang berisi huruf dan data. Huruf merupakan posisi kelas dari citra dan data adalah vector huruf sepanjang 64.

3.2.1.4 Proses Pengenalan Karakter

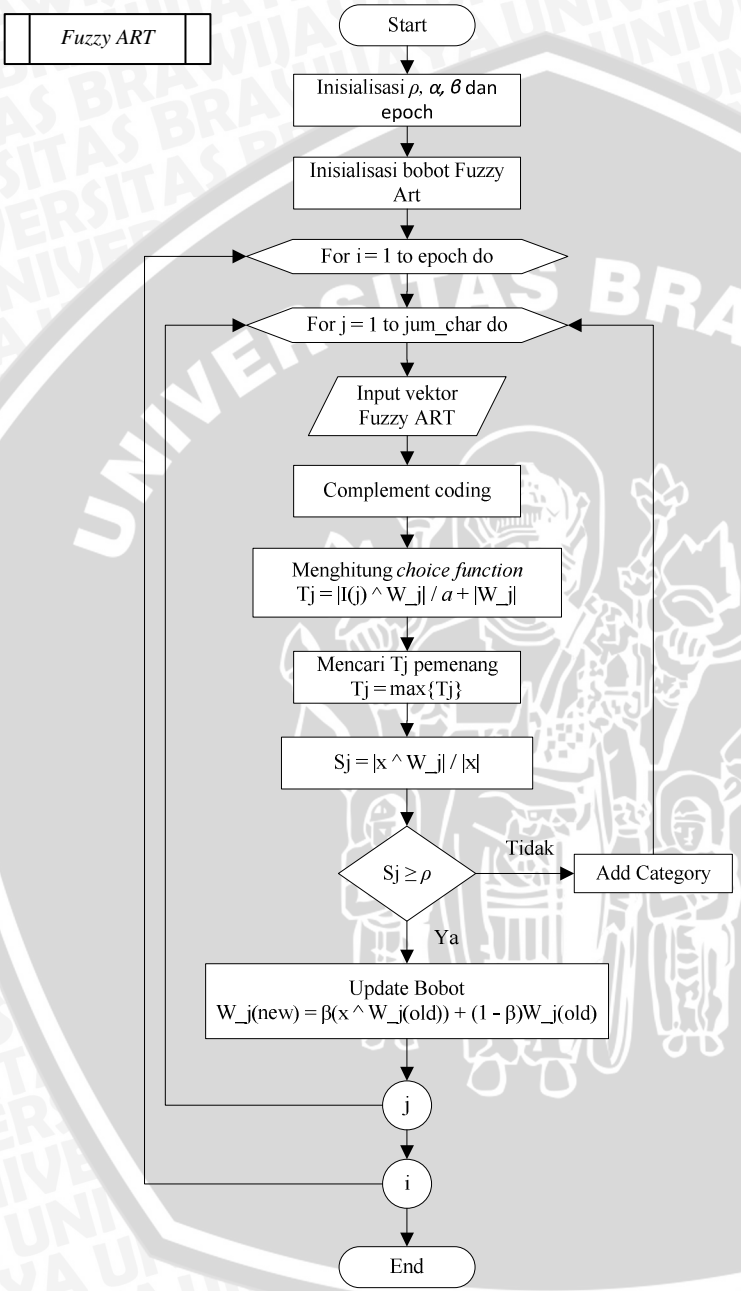
Dalam tahap ini dilakukan pembelajaran terhadap sejumlah *data set*. Vektor input yang didapatkan dari proses *feature extraction* akan digunakan sebagai *input* dalam pengenalan menggunakan algoritma Fuzzy ART. Langkah-langkah dalam tahap pembelajaran sebagai berikut :

1. Menentukan nilai *learning rate* (β), *choice parameter* (α) dan nilai *vigilance parameter* (ρ) dari masukan *user.*
2. Inisialisasi bobot awal. Inisialisai bobot awal dilakukan pada *weight* (w_j) seperti yang ditunjukkan pada Persamaan 2.7
3. Memasukkan vektor input x ke lapis input. Vektor x terdiri dari elemen data riil dari 0 sampai 1.
4. Menambahkan data input x dengan bentuk *complement code*.
5. Menghitung nilai T_j untuk semua *neuron* di lapis *output* yang aktif dengan fungsi *choice* pada Persamaan 2.8.
6. Memilih salah satu node yang memiliki nilai T_j terbesar sebagai nilai yang terbaik dari Persamaan 2.9. Apabila T_j memiliki nilai sama, maka dipilih node yang berindeks kecil.
7. Membandingkan dengan nilai *vigilance* untuk mengetahui bahwa pola *input* x bagian dari kelas tertentu dengan Persamaan 2.10.

Jika $S_j \geq \rho$, maka terjadi resonansi dan dilanjutkan ke langkah 9. Jika tidak, kelas ini dibatalkan dan dilanjutkan ke langkah 8.

8. Menghapus kelas yang paling cocok tadi, karena vektor *x* bukan milik kelas *j*. *Output* dari node *j* yang terpilih pada langkah 5 untuk sementara dihapus dan tidak diikutkan berkompetisi selanjutnya. Lanjutkan pada langkah 3.
9. Memperbarui bobot kelas yang paling baik dengan Persamaan 2.11.
10. Semua node yang dihapus pada langkah 7 diaktifkan untuk berkompetisi lagi dan kembali pada langkah 3.

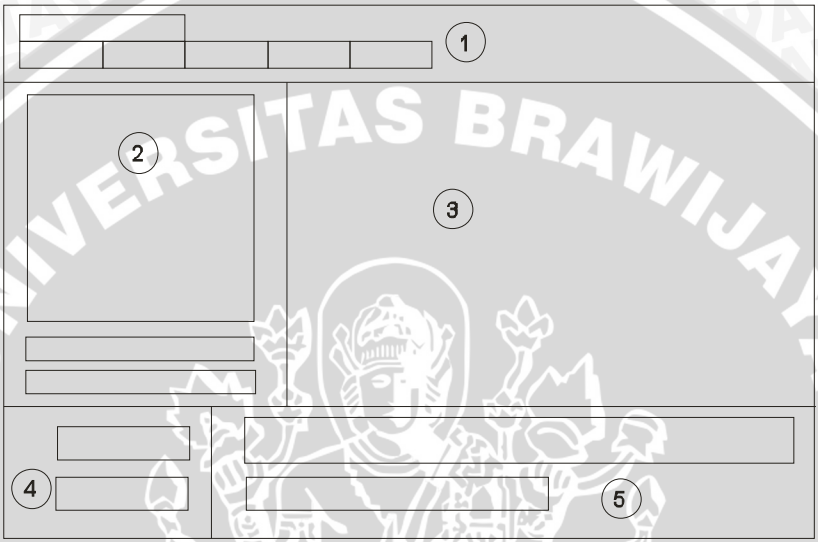
Pada Gambar 3.13 ditunjukkan urutan proses dari pengenalan karakter menggunakan algoritma Fuzzy ART.



Gambar 3.13 Flowchart proses pelatihan dengan Fuzzy ART

3.3 Perancangan *Interface*

Interface yang akan dibangun secara garis besar dibagi menjadi 2 bagian utama. Yaitu *interface* pada tahap pembelajaran dan pengujian serta *interface* Fuzzy ART. *Interface* yang dimaksud diantaranya ditunjukkan pada Gambar 3.14 dan Gambar 3.15.



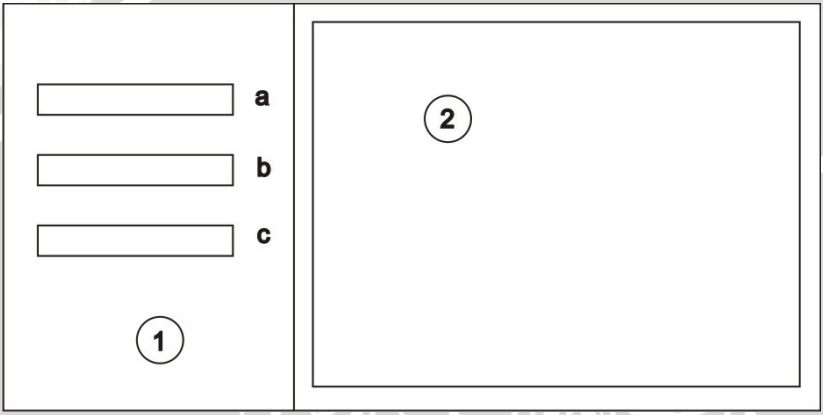
Gambar 3.14 Rancangan *interface* tahapan pengujian

Adapun penjelasan dari bagian-bagian *interface* tahapan pengujian adalah :

1. Judul dan menu utama
   Menu utama terdiri atas menu buka file, simpan, setting, *pre-processing* dan *testing*.
2. List citra data uji
   Bagian ini menampilkan kata yang digunakan sebagai data uji. Citra berupa tulisan tangan hasil *scanning*. Pada bagian ini juga terdapat *button* yang dapat digunakan untuk membuka file lain untuk data uji.
3. Citra uji
   Pada bagian ini akan menampilkan citra data uji yang berupa tulisan tangan latin yang akan dilakukan proses pengenalan.
4. *Button* proses

Bagian ini terdapat 2 *button* yang digunakan untuk proses *preprocessing* dan pengenalan.

5. Tampilan hasil pengenalan
   a. Menampilkan citra karakter hasil segmentasi
   b. Menampilkan koordinat lokasi karakter
   c. Menampilkan hasil akhir pengenalan tulisan tangan



Gambar 3.15 Rancangan *interface* Fuzzy ART

Adapun penjelasan dari bagian-bagian *interface* Fuzzy ART adalah sebagai berikut :

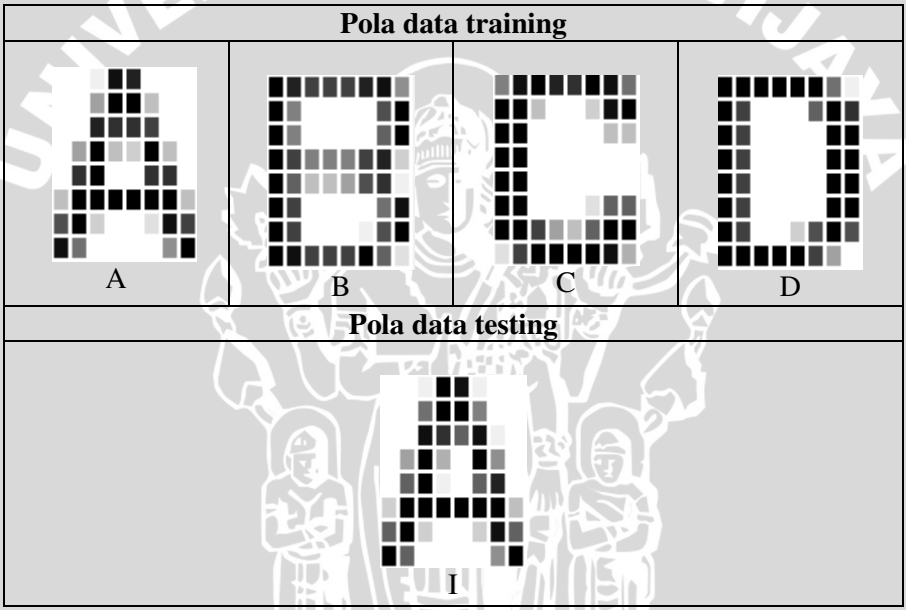
1. Pengaturan inisialisasi Fuzzy ART
   Bagian ini untuk mengatur inisialisasi awal Fuzzy ART yang berisi tentang nilai *vigilance paramter* (ρ), *choice parameter* (α) dan nilai *learning rate* (β).
2. Data karakter yang telah tersimpan dalam system
   Bagian ini menampilkan data bobot akhir yang telah tersimpan dalam sistem. Data yang ditampilkan yaitu berupa karakter beserta representasi vectornya.

3.4 Perhitungan Manual

Perhitungan manual akan digunakan untuk menunjukkan alur proses keseluruhan secara umum. Contoh berikut ini menjelaskan perhitungan algoritma Fuzzy ART untuk pengenalan karakter.

3.4.1 Perhitungan Fuzzy ART

Berikut diberikan contoh perhitungan manual pada Fuzzy ART dengan dimensi 8x8, vektor masukan berupa nilai yang berkisar antara 0 sampai 1 dengan tidak menambahkan bentuk *complement code*. Data *training* menggunakan 4 object karakter yang berbeda. Keempat vektor data training ditunjukkan pada Gambar 3.24. Dengan inisialisasi awal nilai *learning rate* (β) = 0.6 , *choice parameter* (α) = 0.01 dan *vigilance parameter* (ρ) = 0.9. Data testing memakai object karakter yang berbeda berdimensi 8x8. Data testing yang digunakan dapat dilihat pada Gambar 3.16.



Gambar 3.16 Kasifikasi pola dengan Fuzzy ART

Langkah 1 :

Menetukan nilai *learning rate* (β), *choice parameter* (α), dan *vigilance paramter* (ρ).

$\beta = 0.6$

$\alpha = 0.01$

$\rho = 0.9$

Langkah 2 :

Inisialisasi bobot awal. Inisialisasi bobot awal dilakukan pada *weight* (w_{ji}), seperti yang ditunjukkan pada Persamaan 2.7.

$$w_{ji}(0) = \cdots = w_{jM}(0) = 1$$

Keterangan :

$w_{ji} = 1$

j = 1, 2, 3, 4 i = 1,2,.....64

maka Tabel 3.4 menunjukkan inisialisasi awal w_{1i} = 1. Begitu pula nilai bobot awal untuk kelas yang lainnya. Jadi, $w_{1i} = w_{2i} = w_{3i} = w_{4i}$.

Tabel 3.4 Inisialisasi Bobot Awal

w_{1i}	1	2	3	4	5	6	7	8
1	1	1	1	1	1	1	1	1
2	1	1	1	1	1	1	1	1
3	1	1	1	1	1	1	1	1
4	1	1	1	1	1	1	1	1
5	1	1	1	1	1	1	1	1
6	1	1	1	1	1	1	1	1
7	1	1	1	1	1	1	1	1
8	1	1	1	1	1	1	1	1

Langkah 3 :

Memasukkan vektor input ’A’ ke lapis input. Vektor ’A’ terdiri dari elemen data riil dari 0 sampai 1. Vektor input ’A’ dapat dilihat pada Tabel 3.5.

Tabel 3.5 Input Vektor ’A’

Huruf A	1	2	3	4	5	6	7	8
1	0	0	0.06	0.94	0.87	0	0	0
2	0	0	0.37	1	1	0.25	0	0
3	0	0	0.87	0.81	0.81	0.75	0	0
4	0	0.37	1	0.25	0.19	1	0.25	0
5	0	0.81	1	0	0	0.81	0.81	0
6	0.25	1	1	1	1	1	1	0.25
7	0.69	1	0.19	0	0	0.12	0.94	0.75
8	0.94	0.56	0	0	0	0	0.44	1

Langkah 4 :

Menghitung nilai T_1 untuk semua *neuron* di lapis *output* yang aktif dengan fungsi *choice* pada Persamaan 2.8.

$$T_1 = \frac{|I \wedge w_1|}{\alpha + |w_1|}$$

Dengan ^ adalah operator *fuzzy* MIN sebagai (p ^ q) ≡ min (p,q) dan | . | didefinisikan sebagai norm.

Sebelumnya mencari nilai dari $I \wedge w_1^{(awal)}$ padamasing-masing vektor, dengan nilai *I* yang ditunjukkan pada Tabel 3.4 dan nilai $w_1^{(awal)}$ dari Tabel 3.5. Maka hasil pencarian nilai minimal dari kedua tabel tersebut, hasilnya dapat ditunjukkan pada Tabel 3.6.

Tabel 3.6 Nilai $I \wedge w_1^{(awal)}$

Huruf A	1	2	3	4	5	6	7	8
1	0	0	0.06	0.94	0.87	0	0	0
2	0	0	0.37	1	1	0.25	0	0
3	0	0	0.87	0.81	0.81	0.75	0	0
4	0	0.37	1	0.25	0.19	1	0.25	0
5	0	0.81	1	0	0	0.81	0.81	0
6	0.25	1	1	1	1	1	1	0.25
7	0.69	1	0.19	0	0	0.12	0.94	0.75
8	0.94	0.56	0	0	0	0	0.44	1

$$T_1 = {}^{27.35}/_{0.01 + 64} = 0.427277$$

Langkah 5 :

Memilih salah satu node yang memiliki nilai T_j terbesar sebagai nilai yang terbaik dari Persamaan 2.9. Karena hanya terdapat satu nilai *T* maka T_1 dianggap sebagai pemenang.

$T_1 = 0.427277$

Langkah 6 :

Menghitung nilai S_1 dengan Persamaan 2.10 untuk kemudian nilainya dibandingkan dengan nilai *vigilance* parameter.

$$S_1 = \frac{|I \wedge w_1|}{|I|}$$

Terlebih dahulu mencari nilai dari $I \wedge w_1^{(awal)}$ pada masing-masing vektor, dengan nilai I yang ditunjukkan pada Tabel 3.5 dan nilai $w_1^{(awal)}$dari Tabel 3.4. Maka hasil pencarian nilai minimal dari kedua Tabel tersebut, hasilnya dapat ditunjukkan pada Tabel 3.7.

Tabel 3.7 Nilai $I \wedge w_1^{(awal)}$

Huruf A	1	2	3	4	5	6	7	8
1	0	0	0.06	0.94	0.87	0	0	0
2	0	0	0.37	1	1	0.25	0	0
3	0	0	0.87	0.81	0.81	0.75	0	0
4	0	0.37	1	0.25	0.19	1	0.25	0
5	0	0.81	1	0	0	0.81	0.81	0
6	0.25	1	1	1	1	1	1	0.25
7	0.69	1	0.19	0	0	0.12	0.94	0.75
8	0.94	0.56	0	0	0	0	0.44	1

$S_1 = \frac{27.35}{27.35} = 1$ $\qquad$ $S_1 \geq 0.9$

Langkah 7 :

Karena nilai $S_1 \geq \rho$, maka dilakukan update bobot dengan menggunakan Persamaan 2.11.

$$w_1^{(baru)} = \beta\,(I \wedge w_1^{(lama)}) + (1 - \beta) w_1^{(lama)}$$

Dari nilai $w_1^{(lama)}$yang terdapat pada Tabel 3.5, kemudian dilakukan perhitungan menggunakan Persamaan 2.11 sehingga menghasilkan nilai $w_1^{(baru)}$ pada Tabel 3.8.

Tabel 3.8 Nilai $w_1^{(baru)}$

$w_1^{(baru)}$	1	2	3	4	5	6	7	8
1	0.4	0.4	0.436	0.964	0.922	0.4	0.4	0.4
2	0.4	0.4	0.622	1	1	0.55	0.4	0.4
3	0.4	0.4	0.922	0.886	0.886	0.85	0.4	0.4
4	0.4	0.622	1	0.55	0.514	1	0.55	0.4
5	0.4	0.886	1	0.4	0.4	0.886	0.886	0.4
6	0.55	1	1	1	1	1	1	0.55
7	0.814	1	0.514	0.4	0.4	0.472	0.964	0.85
8	0.964	0.736	0.4	0.4	0.4	0.4	0.664	1

Langkah 8 :

Memasukkan vektor input 'B' ke lapis input. Vektor 'B' terdiri dari elemen data riil dari 0 sampai 1. Vektor input 'B' dapat dilihat pada Tabel 3.9.

Tabel 3.9 Input Vektor 'B'

Huruf B	1	2	3	4	5	6	7	8
1	1	0.87	0.75	0.75	0.75	0.87	0.94	0.44
2	1	0.5	0	0	0	0	0.69	1
3	1	0.5	0	0	0	0	0.62	1
4	1	0.75	0.5	0.5	0.5	0.75	0.87	0.19
5	1	0.62	0.25	0.25	0.37	0.69	0.81	0.06
6	1	0.75	0	0	0	0	0.56	0.94
7	1	0.75	0	0	0	0.06	0.69	1
8	1	0.94	0.75	0.75	0.75	1	0.62	0.12

Langkah 9 :

Menghitung nilai T_j untuk semua *neuron* di lapis *output* yang aktif dengan fungsi *choice* pada Persamaan 2.8.

Mencari nilai dari $I \wedge w_1^{(baru)}$ padamasing-masing vektor, dengan nilai I yang ditunjukkan pada Tabel 3.9 dan nilai $w_1^{(baru)}$ dari Tabel 3.8. Maka hasil pencarian nilai minimal dari kedua tabel tersebut, hasilnya dapat ditunjukkan pada Tabel 3.10.

Tabel 3.10 Nilai $I \wedge w_1^{(baru)}$

Huruf B	1	2	3	4	5	6	7	8
1	0.4	0.4	0.436	0.75	0.75	0.4	0.4	0.4
2	0.4	0.4	0	0	0	0	0.4	0.4
3	0.4	0.4	0	0	0	0	0.4	0.4
4	0.4	0.622	0.5	0.5	0.5	0.75	0.55	0.19
5	0.4	0.62	0.25	0.25	0.37	0.69	0.81	0.06
6	0.55	0.75	0	0	0	0	0.56	0.55
7	0.814	0.75	0	0	0	0.06	0.69	0.85
8	0.964	0.736	0.4	0.4	0.4	0.4	0.62	0.12

$$T_1 = {}^{24.212}/_{0.01 + 64} = 0.378253$$

Menghitung nilai T_2 untuk semua *neuron* di lapis *output* yang aktif dengan fungsi *choice* pada Persamaan 2.8.

Mencari nilai dari $I \wedge w_2^{(awal)}$ padamasing-masing vektor, dengan nilai I yang ditunjukkan pada Tabel 3.9 dan nilai $w_2^{(awal)}$ dari Tabel 3.4. Maka hasil pencarian nilai minimal dari kedua tabel tersebut, hasilnya dapat ditunjukkan pada Tabel 3.11.

Tabel 3.11 Nilai $I \wedge w_2^{(awal)}$

Huruf B	1	2	3	4	5	6	7	8
1	1	0.87	0.75	0.75	0.75	0.87	0.94	0.44
2	1	0.5	0	0	0	0	0.69	1
3	1	0.5	0	0	0	0	0.62	1
4	1	0.75	0.5	0.5	0.5	0.75	0.87	0.19
5	1	0.62	0.25	0.25	0.37	0.69	0.81	0.06
6	1	0.75	0	0	0	0	0.56	0.94
7	1	0.75	0	0	0	0.06	0.69	1
8	1	0.94	0.75	0.75	0.75	1	0.62	0.12

$$T_2 = {}^{34.47}/_{0.01 + 64} = 0.53851$$

Langkah 10 :

Memilih salah satu node yang memiliki nilai T_j terbesar sebagai nilai yang terbaik dari Persamaan 2.9.

$T_1 = 0.378253$

$T_2 = 0.53851$

Dari kedua nilai T diatas, maka nilai T_2 terpilih sebagai pemenang.

Langkah 11 :

Menghitung nilai S_2 dengan Persamaan 2.10 untuk kemudian nilainya dibandingkan dengan nilai *vigilance* parameter.

Terlebih dahulu mencari nilai dari $I \wedge w_2^{(awal)}$ pada masing-masing vektor, dengan nilai I yang ditunjukkan pada Tabel 3.9 dan nilai $w_2^{(awal)}$ dari Tabel 3.4. Maka hasil pencarian nilai minimal dari kedua Tabel tersebut, hasilnya dapat ditunjukkan pada Tabel 3.12.

Tabel 3.12 Nilai $I \wedge w_2^{(awal)}$

Huruf B	1	2	3	4	5	6	7	8
1	1	0.87	0.75	0.75	0.75	0.87	0.94	0.44
2	1	0.5	0	0	0	0	0.69	1
3	1	0.5	0	0	0	0	0.62	1
4	1	0.75	0.5	0.5	0.5	0.75	0.87	0.19
5	1	0.62	0.25	0.25	0.37	0.69	0.81	0.06
6	1	0.75	0	0	0	0	0.56	0.94
7	1	0.75	0	0	0	0.06	0.69	1
8	1	0.94	0.75	0.75	0.75	1	0.62	0.12

$S_2 = \frac{34.47}{34.47} = 1$ $\quad S_2 \geq 0.9$

Langkah 12 :

Karena nilai $S_2 \geq \rho$, maka dilakukan update bobot dengan menggunakan Persamaan 2.11.

Dari nilai $w_2^{(lama)}$ yang terdapat pada Tabel 3.9, kemudian dilakukan perhitungan menggunakan Persamaan 2.11 sehingga menghasilkan nilai $w_2^{(baru)}$ pada Tabel 3.13.

Tabel 3.13 Nilai $w_2^{(baru)}$

$w_2^{(baru)}$	1	2	3	4	5	6	7	8
1	1	0.922	0.85	0.85	0.85	0.922	0.964	0.664
2	1	0.7	0.4	0.4	0.4	0.4	0.814	1
3	1	0.7	0.4	0.4	0.4	0.4	0.772	1
4	1	0.85	0.7	0.7	0.7	0.85	0.922	0.514
5	1	0.772	0.55	0.55	0.622	0.814	0.886	0.436
6	1	0.85	0.4	0.4	0.4	0.4	0.736	0.964
7	1	0.85	0.4	0.4	0.4	0.436	0.814	1
8	1	0.964	0.85	0.85	0.85	1	0.772	0.472

Langkah 13:

Memasukkan vektor input ’C’ ke lapis input. Vektor ’C’ terdiri dari elemen data riil dari 0 sampai 1. Vektor input ’C’ dapat dilihat pada Tabel 3.14.

Tabel 3.14 Input Vektor 'C'

Huruf C	1	2	3	4	5	6	7	8
1	0.5	0.94	1	0.87	0.81	1	0.94	0.56
2	1	1	0.25	0	0	0.19	0.94	1
3	1	1	0	0	0	0	0.25	0.25
4	1	1	0	0	0	0	0	0
5	1	1	0	0	0	0	0	0
6	1	1	0	0	0	0.12	0.62	0.62
7	1	1	0.75	0.37	0.25	0.62	1	1
8	0.12	0.75	1	1	1	1	0.94	0.31

Langkah 14 :

Menghitung nilai T_1 untuk semua *neuron* di lapis *output* yang aktif dengan fungsi *choice* pada Persamaan 2.8.

Mencari nilai dari $I \wedge w_1^{(baru)}$ padamasing-masing vektor, dengan nilai I yang ditunjukkan pada Tabel 3.14 dan nilai $w_1^{(baru)}$ dari Tabel 3.8. Maka hasil pencarian nilai minimal dari kedua tabel tersebut, hasilnya dapat ditunjukkan pada Tabel 3.15.

Tabel 3.15 Nilai $I \wedge w_1^{(baru)}$

Huruf C	1	2	3	4	5	6	7	8
1	0.4	0.4	0.436	0.87	0.81	0.4	0.4	0.4
2	0.4	0.4	0.25	0	0	0.19	0.4	0.4
3	0.4	0.4	0	0	0	0	0.25	0.25
4	0.4	0.622	0	0	0	0	0	0
5	0.4	0.886	0	0	0	0	0	0
6	0.55	1	0	0	0	0.12	0.62	0.55
7	0.814	1	0.514	0.37	0.25	0.472	0.964	0.85
8	0.12	0.736	0.4	0.4	0.4	0.4	0.664	0.31

$$T_1 = {}^{21.268}/_{0.01 + 64} = 0.332261$$

Menghitung nilai T_2 untuk semua *neuron* di lapis *output* yang aktif dengan fungsi *choice* pada Persamaan 2.8.

Mencari nilai dari $I \wedge w_2^{(baru)}$ padamasing-masing vektor, dengan nilai *I* yang ditunjukkan pada Tabel 3.14 dan nilai $w_2^{(baru)}$ dari Tabel 3.13. Maka hasil pencarian nilai minimal dari kedua tabel tersebut, hasilnya dapat ditunjukkan pada Tabel 3.16.

Tabel 3.16 Nilai $I \wedge w_2^{(baru)}$

Huruf C	1	2	3	4	5	6	7	8
1	0.5	0.922	0.85	0.85	0.81	0.922	0.94	0.56
2	1	0.7	0.25	0	0	0.19	0.814	1
3	1	0.7	0	0	0	0	0.25	0.25
4	1	0.85	0	0	0	0	0	0
5	1	0.772	0	0	0	0	0	0
6	1	0.85	0	0	0	0.12	0.62	0.62
7	1	0.85	0.4	0.37	0.25	0.436	0.814	1
8	0.12	0.75	0.85	0.85	0.85	1	0.772	0.31

$$T_2 = {}^{29.962}/_{0.01 + 64} = 0.468083$$

Menghitung nilai T_3 untuk semua *neuron* di lapis *output* yang aktif dengan fungsi *choice* pada Persamaan 2.8.

Mencari nilai dari $I \wedge w_3^{(awal)}$ padamasing-masing vektor, dengan nilai *I* yang ditunjukkan pada Tabel 3.14 dan nilai $w_3^{(awal)}$ dari Tabel 3.4. Maka hasil pencarian nilai minimal dari kedua tabel tersebut, hasilnya dapat ditunjukkan pada Tabel 3.17.

Tabel 3.17 Nilai $I \wedge w_3^{(awal)}$

Huruf C	1	2	3	4	5	6	7	8
1	0.5	0.94	1	0.87	0.81	1	0.94	0.56
2	1	1	0.25	0	0	0.19	0.94	1
3	1	1	0	0	0	0	0.25	0.25
4	1	1	0	0	0	0	0	0
5	1	1	0	0	0	0	0	0
6	1	1	0	0	0	0.12	0.62	0.62
7	1	1	0.75	0.37	0.25	0.62	1	1
8	0.12	0.75	1	1	1	1	0.94	0.31

$T_3 = {}^{32.97}/_{0.01 + 64} = 0.515076$

Langkah 15 :

Memilih salah satu node yang memiliki nilai T_j terbesar sebagai nilai yang terbaik dari Persamaan 2.9.

$T_1 = 0.332261$
$T_2 = 0.468083$
$T_3 = 0.515076$

Dari ketiga nilai T diatas, maka nilai T_3 terpilih sebagai pemenang.

Langkah 16 :

Menghitung nilai S_3 dengan Persamaan 2.10 untuk kemudian nilainya dibandingkan dengan nilai *vigilance* parameter.

Terlebih dahulu mencari nilai dari $I \wedge w_3^{(awal)}$ pada masing-masing vektor, dengan nilai I yang ditunjukkan pada Tabel 3.14 dan nilai $w_3^{(awal)}$dari Tabel 3.4. Maka hasil pencarian nilai minimal dari kedua Tabel tersebut, hasilnya dapat ditunjukkan pada Tabel 3.18.

Tabel 3.18 Nilai $I \wedge w_3^{(awal)}$

Huruf C	1	2	3	4	5	6	7	8
1	0.5	0.94	1	0.87	0.81	1	0.94	0.56
2	1	1	0.25	0	0	0.19	0.94	1
3	1	1	0	0	0	0	0.25	0.25
4	1	1	0	0	0	0	0	0
5	1	1	0	0	0	0	0	0
6	1	1	0	0	0	0.12	0.62	0.62
7	1	1	0.75	0.37	0.25	0.62	1	1
8	0.12	0.75	1	1	1	1	0.94	0.31

$S_3 = \frac{32.97}{32.97} = 1 \qquad S_3 \geq 0.9$

Langkah 17 :

Karena nilai $S_3 \geq \rho$, maka dilakukan update bobot dengan menggunakan Persamaan 2.11.

Dari nilai $w_3^{(lama)}$ yang terdapat pada Tabel 3.14, kemudian dilakukan perhitungan menggunakan Persamaan 2.11 sehingga menghasilkan nilai $w_3^{(baru)}$ pada Tabel 3.19.

Tabel 3.19 Nilai $w_3^{(baru)}$

$w_3^{(baru)}$	1	2	3	4	5	6	7	8
1	0.7	0.964	1	0.922	0.886	1	0.964	0.736
2	1	1	0.55	0.4	0.4	0.514	0.964	1
3	1	1	0.4	0.4	0.4	0.4	0.55	0.55
4	1	1	0.4	0.4	0.4	0.4	0.4	0.4
5	1	1	0.4	0.4	0.4	0.4	0.4	0.4
6	1	1	0.4	0.4	0.4	0.472	0.772	0.772
7	1	1	0.85	0.622	0.55	0.772	1	1
8	0.472	0.85	1	1	1	1	0.964	0.586

Langkah 18:

Memasukkan vektor input 'D' ke lapis input. Vektor 'D' terdiri dari elemen data riil dari 0 sampai 1. Vektor input 'D' dapat dilihat pada Tabel 3.20.

Tabel 3.20 Input Vektor 'D'

Huruf D	1	2	3	4	5	6	7	8
1	1	1	1	1	1	1	0.56	0.06
2	1	0.75	0	0	0	0.62	1	0.81
3	1	0.75	0	0	0	0	1	1
4	1	0.75	0	0	0	0	1	1
5	1	0.75	0	0	0	0	1	1
6	1	0.75	0	0	0	0	1	1
7	1	0.75	0	0	0.19	0.69	1	0.69
8	1	1	1	1	1	0.75	0.37	0

Langkah 19 :

Menghitung nilai T_I untuk semua *neuron* di lapis *output* yang aktif dengan fungsi *choice* pada Persamaan 2.8.

Mencari nilai dari $I \wedge w_1^{(baru)}$ padamasing-masing vektor, dengan nilai I yang ditunjukkan pada Tabel 3.20 dan nilai $w_1^{(baru)}$ dari Tabel

3.8. Maka hasil pencarian nilai minimal dari kedua tabel tersebut, hasilnya dapat ditunjukkan pada Tabel 3.21.

Tabel 3.21 Nilai $I \wedge w_1^{(baru)}$

Huruf D	1	2	3	4	5	6	7	8
1	0.4	0.4	0.436	0.964	0.922	0.4	0.4	0.06
2	0.4	0.4	0	0	0	0.55	0.4	0.4
3	0.4	0.4	0	0	0	0	0.4	0.4
4	0.4	0.622	0	0	0	0	0.55	0.4
5	0.4	0.75	0	0	0	0	0.886	0.4
6	0.55	0.75	0	0	0	0	1	0.55
7	0.814	0.75	0	0	0.19	0.472	0.964	0.69
8	0.964	0.736	0.4	0.4	0.4	0.4	0.37	0

$$T_1 = {}^{22.54}/_{0.01 + 64} = 0.352132$$

Menghitung nilai T_2 untuk semua *neuron* di lapis *output* yang aktif dengan fungsi *choice* pada Persamaan 2.8.

Mencari nilai dari $I \wedge w_2^{(baru)}$ padamasing-masing vektor, dengan nilai I yang ditunjukkan pada Tabel 3.20 dan nilai $w_2^{(baru)}$dari Tabel 3.13. Maka hasil pencarian nilai minimal dari kedua tabel tersebut, hasilnya dapat ditunjukkan pada Tabel 3.22.

Tabel 3.22 Nilai $I \wedge w_2^{(baru)}$

Huruf D	1	2	3	4	5	6	7	8
1	1	0.922	0.85	0.85	0.85	0.922	0.56	0.06
2	1	0.7	0	0	0	0.4	0.814	0.81
3	1	0.7	0	0	0	0	0.772	1
4	1	0.75	0	0	0	0	0.922	0.514
5	1	0.75	0	0	0	0	0.886	0.436
6	1	0.75	0	0	0	0	0.736	0.964
7	1	0.75	0	0	0.19	0.436	0.814	0.69
8	1	0.964	0.85	0.85	0.85	0.75	0.37	0

$$T_2 = {}^{32.432}/_{0.01 + 64} = 0.506671$$

Menghitung nilai T_3 untuk semua *neuron* di lapis *output* yang aktif dengan fungsi *choice* pada Persamaan 2.8.

Mencari nilai dari $I \wedge w_3^{(baru)}$ padamasing-masing vektor, dengan nilai I yang ditunjukkan pada Tabel 3.14 dan nilai $w_3^{(baru)}$ dari Tabel 3.19. Maka hasil pencarian nilai minimal dari kedua tabel tersebut, hasilnya dapat ditunjukkan pada Tabel 3.23.

Tabel 3.23 Nilai $I \wedge w_3^{(baru)}$

Huruf D	1	2	3	4	5	6	7	8
1	0.7	0.964	1	0.922	0.886	1	0.56	0.06
2	1	0.75	0	0	0	0.514	0.964	0.81
3	1	0.75	0	0	0	0	0.55	0.55
4	1	0.75	0	0	0	0	0.4	0.4
5	1	0.75	0	0	0	0	0.4	0.4
6	1	0.75	0	0	0	0	0.772	0.772
7	1	0.75	0	0	0.19	0.69	1	0.69
8	0.472	0.85	1	1	1	0.75	0.37	0

$$T_3 = {}^{31.136}/_{0.01 + 64} = 0.486424$$

Menghitung nilai T_4 untuk semua *neuron* di lapis *output* yang aktif dengan fungsi *choice* pada Persamaan 2.8.

Mencari nilai dari $I \wedge w_4^{(awal)}$ padamasing-masing vektor, dengan nilai I yang ditunjukkan pada Tabel 3.20 dan nilai $w_4^{(awal)}$ dari Tabel 3.4. Maka hasil pencarian nilai minimal dari kedua tabel tersebut, hasilnya dapat ditunjukkan pada Tabel 3.24.

Tabel 3.24 Nilai $I \wedge w_4^{(awal)}$

Huruf D	1	2	3	4	5	6	7	8
1	0.7	0.964	1	0.922	0.886	1	0.56	0.06
2	1	0.75	0	0	0	0.514	0.964	0.81
3	1	0.75	0	0	0	0	0.55	0.55
4	1	0.75	0	0	0	0	0.4	0.4
5	1	0.75	0	0	0	0	0.4	0.4
6	1	0.75	0	0	0	0	0.772	0.772
7	1	0.75	0	0	0.19	0.69	1	0.69
8	0.472	0.85	1	1	1	0.75	0.37	0

$$T_4 = {}^{36.24}/_{0.01 + 64} = 0.566162$$

Langkah 20 :

Memilih salah satu node yang memiliki nilai T_j terbesar sebagai nilai yang terbaik dari Persamaan 2.9.

$T_1 = 0.352132$
$T_2 = 0.506671$
$T_3 = 0.486424$
$T_4 = 0.566162$

Dari keempat nilai *T* diatas, maka nilai T_4 terpilih sebagai pemenang.

Langkah 21 :

Menghitung nilai S_4 dengan Persamaan 2.10 untuk kemudian nilainya dibandingkan dengan nilai *vigilance* parameter.

Terlebih dahulu mencari nilai dari $I \wedge w_4^{(awal)}$ pada masing-masing vektor, dengan nilai *I* yang ditunjukkan pada Tabel 3.20 dan nilai $w_4^{(awal)}$dari Tabel 3.4. Maka hasil pencarian nilai minimal dari kedua Tabel tersebut, hasilnya dapat ditunjukkan pada Tabel 3.25.

Tabel 3.25 Nilai $I \wedge w_4^{(awal)}$

Huruf D	1	2	3	4	5	6	7	8
1	1	1	1	1	1	1	0.56	0.06
2	1	0.75	0	0	0	0.62	1	0.81
3	1	0.75	0	0	0	0	1	1
4	1	0.75	0	0	0	0	1	1
5	1	0.75	0	0	0	0	1	1
6	1	0.75	0	0	0	0	1	1
7	1	0.75	0	0	0.19	0.69	1	0.69
8	1	1	1	1	1	0.75	0.37	0

$S_4 = \frac{36.24}{36.24} = 1$ $\quad S_4 \geq 0.9$

Langkah 22 :

Karena nilai $S_4 \geq \rho$, maka dilakukan update bobot dengan menggunakan Persamaan 2.11.
Dari nilai $w_4^{(lama)}$ yang terdapat pada Tabel 3.20, kemudian dilakukan perhitungan menggunakan Persamaan 2.11 sehingga menghasilkan nilai $w_4^{(baru)}$ pada Tabel 3.26.

Tabel 3.26 Nilai $w_4^{(baru)}$

$w_4^{(baru)}$	1	2	3	4	5	6	7	8
1	1	1	1	1	1	1	0.736	0.436
2	1	0.85	0.4	0.4	0.4	0.772	1	0.886
3	1	0.85	0.4	0.4	0.4	0.4	1	1
4	1	0.85	0.4	0.4	0.4	0.4	1	1
5	1	0.85	0.4	0.4	0.4	0.4	1	1
6	1	0.85	0.4	0.4	0.4	0.4	1	1
7	1	0.85	0.4	0.4	0.514	0.814	1	0.814
8	1	1	1	1	1	0.85	0.622	0.4

Langkah 23 :

Setelah mendapatkan masing-masing bobot baru setiap vektor, maka langkah selanjutnya adalah membandingkan vektor input *testing* dengan masing-masing setiap bobot baru yang telah didapat. Untuk nilai vektor input *testing* ditunjukkan pada Tabel 3.27.

Tabel 3.27 Input vektor *testing*

Testing	1	2	3	4	5	6	7	8
1	0	0	0.06	1	0.94	0.06	0	0
2	0	0	0.56	1	1	0.5	0	0
3	0	0	0.94	0.81	0.62	0.94	0.06	0
4	0	0.37	1	0.31	0	1	0.44	0
5	0	0.62	1	0.06	0	0.62	0.87	0
6	0.19	1	1	1	1	1	1	0.25
7	0.62	1	0.19	0	0	0.19	0.94	0.62
8	1	0.62	0	0	0	0	0.44	1

Langkah 24 :

Menghitung nilai T_1 untuk semua *neuron* di lapis *output* dengan fungsi *choice* pada Persamaan 2.8.

Membandingkan nilai vektor *testing* dengan nilai $w_1^{(baru)}$. Dengan terlebih dahulu mencari nilai dari $I \wedge w_1^{(baru)}$, nilai I ditunjukkan pada Tabel 3.27 dan nilai $w_1^{(baru)}$ ditunjukkan pada Tabel 3.8. Maka hasil pencarian nilai minimal dari kedua tabel tersebut, dapat ditunjukkan pada Tabel 3.28.

Tabel 3.28 Nilai $I \wedge w_1^{(baru)}$

$I \wedge w_1^{(baru)}$	1	2	3	4	5	6	7	8
1	0	0	0.06	0.964	0.922	0.06	0	0
2	0	0	0.56	1	1	0.5	0	0
3	0	0	0.922	0.81	0.62	0.85	0.06	0
4	0	0.37	1	0.31	0	1	0.44	0
5	0	0.62	1	0.06	0	0.62	0.87	0
6	0.19	1	1	1	1	1	1	0.25
7	0.62	1	0.19	0	0	0.19	0.94	0.62
8	0.964	0.62	0	0	0	0	0.44	1

$$T_1 = {}^{27.642}/_{0.01 + 42.01} = 0.65783$$

Menghitung nilai T_2 untuk semua *neuron* di lapis *output* dengan fungsi *choice* pada Persamaan 2.8.

Membandingkan nilai vektor *testing* dengan nilai $w_2^{(baru)}$. Dengan terlebih dahulu mencari nilai dari $I \wedge w_2^{(baru)}$, nilai I ditunjukkan pada Tabel 3.27 dan nilai $w_2^{(baru)}$ ditunjukkan pada Tabel

3.13. Maka hasil pencarian nilai minimal dari kedua tabel tersebut, dapat ditunjukkan pada Tabel 3.29.

Tabel 3.29 Nilai $I \wedge w_2^{(baru)}$

$I \wedge w_2^{(baru)}$	1	2	3	4	5	6	7	8
1	0	0	0.06	0.85	0.85	0.06	0	0
2	0	0	0.4	0.4	0.4	0.4	0	0
3	0	0	0.4	0.4	0.4	0.4	0.06	0
4	0	0.37	0.7	0.31	0	0.85	0.44	0
5	0	0.62	0.55	0.06	0	0.62	0.87	0
6	0.19	0.85	0.4	0.4	0.4	0.4	0.736	0.25
7	0.62	0.85	0.19	0	0	0.19	0.814	0.62
8	1	0.62	0	0	0	0	0.44	0.472

$$T_2 = {}^{19.912}/_{0.01 + 46.282} = 0.430139$$

Menghitung nilai T_3 untuk semua *neuron* di lapis *output* dengan fungsi *choice* pada Persamaan 2.8.

Membandingkan nilai vektor *testing* dengan nilai $w_3^{(baru)}$. Dengan terlebih dahulu mencari nilai dari $I \wedge w_3^{(baru)}$, nilai I ditunjukkan pada Tabel 3.27 dan nilai $w_3^{(baru)}$ ditunjukkan pada Tabel 3.19. Maka hasil pencarian nilai minimal dari kedua tabel tersebut, dapat ditunjukkan pada Tabel 3.30.

Tabel 3.30 Nilai $I \wedge w_3^{(baru)}$

$I \wedge w_3^{(baru)}$	1	2	3	4	5	6	7	8
1	0	0	0.06	0.922	0.886	0.06	0	0
2	0	0	0.55	0.4	0.4	0.5	0	0
3	0	0	0.4	0.4	0.4	0.4	0.06	0
4	0	0.37	0.4	0.31	0	0.4	0.4	0
5	0	0.62	0.4	0.06	0	0.4	0.4	0
6	0.19	1	0.4	0.4	0.4	0.472	0.772	0.25
7	0.62	1	0.19	0	0	0.19	0.94	0.62
8	0.472	0.62	0	0	0	0	0.44	0.586

$$T_3 = {}^{18.76}/_{0.01 + 45.382} = 0.413289$$

Menghitung nilai T_4 untuk semua *neuron* di lapis *output* dengan fungsi *choice* pada Persamaan 2.8.

Membandingkan nilai vektor *testing* dengan nilai $w_4^{(baru)}$. Dengan terlebih dahulu mencari nilai dari $I \wedge w_4^{(baru)}$, nilai I ditunjukkan pada Tabel 3.27 dan nilai $w_4^{(baru)}$ ditunjukkan pada Tabel 3.25. Maka hasil pencarian nilai minimal dari kedua tabel tersebut, dapat ditunjukkan pada Tabel 3.31.

Tabel 3.31 Nilai $I \wedge w_4^{(baru)}$

$I \wedge w_4^{(baru)}$	1	2	3	4	5	6	7	8
1	0	0	0.06	1	0.94	0.06	0	0
2	0	0	0.4	0.4	0.4	0.5	0	0
3	0	0	0.4	0.4	0.4	0.4	0.06	0
4	0	0.37	0.4	0.31	0	0.4	0.44	0
5	0	0.62	0.4	0.06	0	0.4	0.87	0
6	0.19	0.85	0.4	0.4	0.4	0.4	1	0.25
7	0.62	0.85	0.19	0	0	0.19	0.94	0.62
8	1	0.62	0	0	0	0	0.44	0.4

$$T_4 = {}^{19.45}/_{0.01 + 47.344} = 0.410736$$

Langkah 25 :

Memilih salah satu node yang memiliki nilai T_j terbesar sebagai nilai yang terbaik dari Persamaan 2.9.

$T_1 = 0.65783$
$T_2 = 0.430139$
$T_3 = 0.413289$
$T_4 = 0.410736$

Dari keempat nilai T diatas, maka nilai T_1 terpilih sebagai pemenang. Sehingga pola huruf 'A' pada data *training* dianggap paling mendekati dan dianggap mirip dengan data *testing* yang telah diujikan.

3.5 Perancangan Uji Coba

Tujuan dari uji coba sistem pengenalan tulisan tangan latin ini adalah untuk mencari nilai keakuratan proses segmentasi dan proses pengenalan karakter.

Proses pengujian segmentasi dilakukan dengan dengan menulis kata sebanyak 22 kata dengan masing-masing kata memiliki 5 varisasi tulisan tangan latin yang berbeda dan akan dilakukan uji segmentasi pada tiap kata tersebut.

Pada Tabel 3.32 akan ditunjukkan prosentase keberhasilan dari kata yang disegmentasi dibandingkan dengan jumlah total dari seluruh kata. Maka dari Tabel tersebut akan diperoleh kata yang berhasil dan tidak berhasil disegmentasi.

Tabel 3.32 Hasil Uji Keberhasilan Segmentasi

Jumlah Kata	Benar	Salah

Proses selanjutnya yang dilakukan adalah pengujian terhadap program Fuzzy ART untuk mengetahui nilai parameter yang optimal. Percobaan dilakukan terhadap beberapa faktor yang mempengaruhi, diantaranya adalah nilai *vigilance parameter* (ρ), *choice parameter* (α) dan *learning rate* (β). Pengujian parameter algoritma Fuzzy ART dapat dilihat pada Tabel 3.33.

Tabel 3.33 Hasil Uji Parameter Fuzzy ART

Vigilance Parameter	*Choice Parameter* (α)	*Learning rate*(β)	Prosentase Keakuratan (%)
ρ_1	α_1	β_1	
		β_2	
		β_3	
	α_2	β_1	
		β_2	
		β_3	
	α_3	β_1	
		β_2	
		β_3	
ρ_2	α_1	β_1	

		β_2	
		β_3	
	α_2	β_1	
		β_2	
		β_3	
	α_3	β_1	
		β_2	
		β_3	

Uji coba selanjutnya adalah proses pengenalan karakter dengan menggabungkan proses segmentasi dengan proses pengenalan oleh Fuzzy ART. Uji coba dilakukan dengan menggunakan data yang dapat disegmentasi dengan benar dan menggunakan parameter uji yang memiliki hasil pengenalan paling optimal. Perancangan hasil uji coba proses pengenalan karakter dapat dilihat pada Tabel 3.34. Tingkat keakuratan (*Recognition Rate*) dihitung dengan menggunakan Persamaan 2.12 yang telah dijelaskan pada bab dua.

Tabel 3.34 Hasil Uji Pengenalan Karakter

Jumlah Karakter	Benar	Salah

BAB IV
IMPLEMENTASI DAN PEMBAHASAN

Bab ini akan lebih lanjut menjelaskan tentang proses-proses yang telah dirancang pada bab tiga. Hal-hal yang akan dibahas meliputi implementasi proses-proses yang telah dirancang, tampilan *interface*, dan bagian-bagian *source code* yang ada dalam implementasi program. Pada bab ini juga dilakukan analisa terhadap hasil yang diperoleh dari pengenalan tulisan tangan latin.

4.1 Lingkungan Implementasi

Proses implementasi merupakan tahapan penerapan rancangan pada bahasa pemrograman yang dapat dimengerti oleh komputer. Lingkungan implementasi yang akan dijelaskan pada bab ini meliputi lingkungan implementasi perangkat keras dan perangkat lunak.

4.1.1 Lingkungan Perangkat Keras

Perangkat keras yang digunakan dalam pengembangan sistem pengenalan tulisan tangan latin ini adalah sebagai berikut :

1. *Processor* Intel Pentium Core 2 Duo 2,00 GHz.
2. RAM 4098 GB
3. *Harddisk* dengan kapasitas 250 GB
4. *Monitor* 14''
5. *Keyboard*
6. *Mouse*
7. *Scanner* Canon LIDE 110

4.1.2 Lingkungan Perangkat Lunak

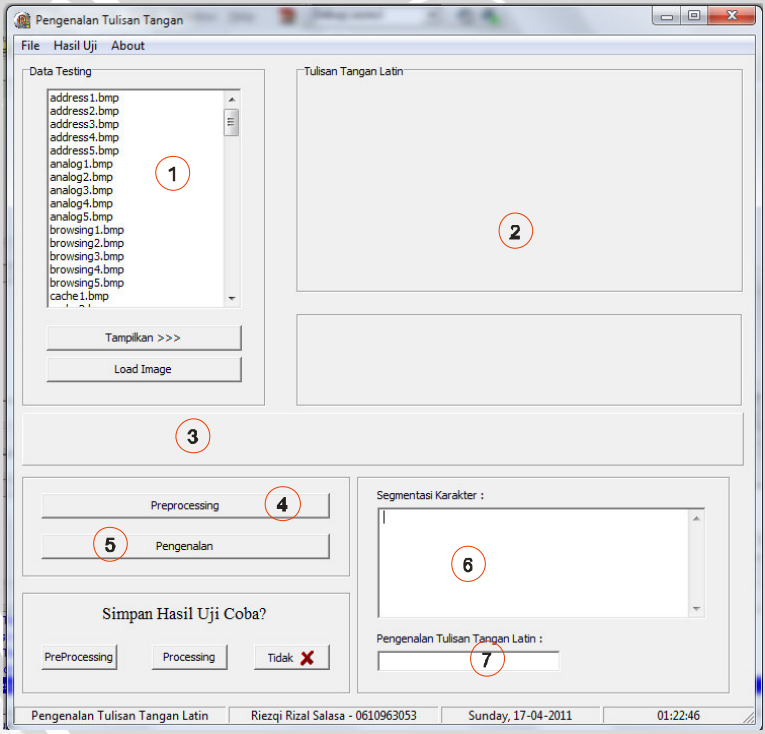
Perangkat lunak yang digunakan dalam pengembangan sistem pengenalan tulisan tangan latin ini adalah sebagai berikut :

1. Sistem Operasi *Microsoft Windows 7 Ultimate*
2. Turbo Delphi 2006 for Microsoft Windows™ sebagai *software development* dalam implementasi rancangan sistem.
3. *Notepad* sebagai media penyimpanan data latih pengenalan karakter.

4.2 Implementasi *Interface*

Seperti yang telah dijelaskan pada sub bab 3.2.2, *interface* pada sistem ini terdiri dari dua *interface* utama, yaitu *interface* pengenalan dan pelatihan.

Dalam *interface* pengenalan, citra *testing* hasil *scanning* didapatkan dari menekan tombol *Load Image* atau memilih daftar citra pada *List Image*. Pada *List Image* terdapat beberapa citra yang digunakan sebagai data testing dalam penelitian. Kemudian untuk mengetahui proses segmentasi atau *preprocessing* pada penelitian ini, maka disediakan tombol *Preprocessing*. Setelah itu menekan tombol Pengenalan, untuk melakukan pengenalan tulisan tangan latin yang terdapat pada citra hasil segmentasi. Hasil segmentasi karakter tulisan tangan latin diperlihatkan pada citra karakter dibawah citra awal tulisan tangan latin. Karakter hasil pengenalan dari citra segmentasi karakter menggunakan Fuzzy ART ditampilkan pada *teks edit*. Gambar 4.1 menunjukkan *interface* pengenalan.



Gambar 4.1*Interface* pengenalan

Keterangan Gambar :

1. Tombol *Load Image* dan Tampilkan dari *list box*
2. Citra awal tulisan tangan latin

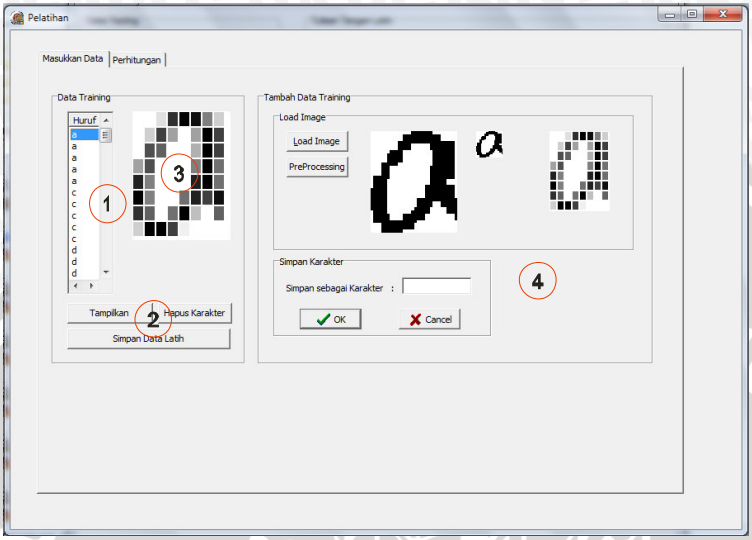
3. Citra karakter hasil segmentasi
4. Tombol *Preprocessing*
5. Tombol Pengenalan
6. Letak koordinat masing-masing karakter segmentasi
7. Karakter hasil pengenalan

Jika tombol *Load Image* atau tampilkanditekan, maka citra awal tulisan tangan latin hasil *scanning* akan muncul pada *group box* nomor 2 yang telah dijelaskan pada Gambar 4.1. Tahap *preprocessing* akan dilakukan setelah tombol *Preprocessing* ditekan. Proses *preprocessing* ini akan melewati beberapa proses, antara lain : Proses konversi citra ke bentuk biner, proses pemotongan kata dan prosessegmentasi. Pada tahap berikutnya untuk melakukan pengenalan karakter menggunakan Fuzzy ART, maka harus menekan tombol Pengenalan.

Pada program ini hasil dari proses konversi citra ke bentuk biner langsung ditampilkan pada *group box* nomor 2. Begitu juga untuk citra hasil pemotongan kata ditampilkan pada *group box* dibawahnya.

Citra tulisan tangan latin hasil pemotongan kata akan dilakukan proses segmentasi karakter. Citra karakter hasil proses segmentasi akan ditampilkan pada panel karakter, seperti yang diperlihatkan pada Gambar 4.1 keterangan nomor 3. Sedangkan untuk lokasi koordinat masing-masing karakter hasil segmentasi ditunjukkan pada Gambar 4.1 keterangan nomor 6. Citra karakter ini masing-masing akan dikenali menggunakan Fuzzy ART dan akan menghasilkan keluaran karakter hasil dari pengenalan. Hasil karakter ini akan ditampilkan pada *teks edit*, seperti yang diperlihatkan pada Gambar 4.1 pada keterangan nomor 7.

Interface pelatihan terdiri dari daftar karakter data latih, tombol tampilkan karakter data latih, hapus karakter data latih dan menu untuk menambah data latih. Data latih dapat ditambah dengan cara mengambil citra biner karakter dengan menekan tombol *Load Image*. Setelah itu, citra biner akan dilakukan tahap *preprocessing* dengan menekan tombol *Preprocessing.* Gambar 4.2 menunjukkan *interface* pelatihan.



Gambar 4.2*Interface* pelatihan

Keterangan Gambar :

1. *List* karakter data latih
2. Tombol tampilkan, hapus dan simpan karakter data latih
3. Tampilan karakter dari *list* data latih
4. Menu tambah data latih

List karakter pada *interface* pelatihan berisi daftar karakter data latih yang digunakan pengenalan karakter. Karakter data latih dapat dilihat dengan cara memilih karakter pada *list* dan menekan tombol Tampilkan karakter. Gambar data latih karakter akan muncul seperti pada Gambar 4.2 keterangan nomor 3. Tombol hapus karakter digunakan untuk menghapus karakter yang telah dipilih pada *list* karakter data latih.

Menu tambah karakter data latih pada Gambar 4.2 keterangan nomor 4, digunakan untuk menambah data latih yang sudah ada. Menu ini berisi tombol *Load Image*, tombol *Preprocessing, teks edit* untuk menulis nama karakter yang disimpan dan citra hasil konversi vektor dari citra tersebut. Proses penambahan karakter ini diawali dengan menekan tombol *Load Image*. Setelah tombol load ditekan akan muncul *open dialog* untuk memilih citra berekstensi *.bmp* mana yang akan dipakai sebagai data latih baru. Setelah dipilih satu citra,

maka citra tersebut akan ditampilkan pada *group box* tersebut. Kemudian untuk mendapatkan citra beserta hasil konversi vektornya maka ditekan tombol *Preprocessing*. Setelah itu *user* memasukkan nama karakter baru yang akan disimpan dalam data latih. Langkah terakhir penambahan karakter adalah menekan tombol *ok* pada menu dan secara otomatis karakter akan ditambah sebagai data latih dalam pengenalan.

4.3 Implementasi Program

Berdasarkan perancangan proses yang terdapat pada Bab 3. Maka pada subbab ini akan dijelaskan implementasi proses-proses tersebut. Secara garis besar proses dikelompokkan menjadi dua tahap, tahap pertama merupakan tahap *preprocessing* sedangkan tahap kedua merupakan tahap pengenalan (*processing*).

4.3.1 Implementasi Tahap *Preprocessing*

Tahap ini adalah proses yang mengolah citra masukan menjadi citra karakter. Proses binerisasi dilakukan terlebih dahulu untuk mendapatkan citra dalam bentuk biner. Kemudian dilakukanproses segmentasi karakter sehingga didapatkan citra karakter.

Struktur data yang digunakan pada implementasi tahap *preprocessing* ditunjukkan pada *Source code* 4.1.

```
...
type
  TKata = record
         ymax,ymin,xmax,xmin: Integer;
         end;
  TKarakter = record
            ymax,ymin,xmax,xmin: Integer;
            prototype : array [10..80] of integer;
           end;
  TCharKata = array [1..100] of TKarakter;

var
  kata : TKata;
  jum_karakter : integer;
  CharKata : TCharKata;                    max,min :integer;
...
```

Source code 4.1 Struktur data *pre-processing*

4.3.1.1 Binerisasi

Pada tahap binerisasi, citra awal pertama kali akan diproses dalam konversi *greyscale*. Citra keabuan (*greyscale*) selanjutnya diubah menjadi citra biner dengan menggunakan nilai *threshold* sebesar 128. Proses konversi dari citra awal menjadi citra *biner* ini dapat dilihat pada *Source code* 4.2.

```
procedure Binerisasi(GamAwal,GamBiner:Timage);
var i, j : integer;
    r, g, b, abu : byte;
    clr : TColor;
begin
  for i:= 0 to GamAwal.Picture.Width do
  begin
    for j:= 0 to GamAwal.Picture.Height do
    begin
      clr:=GamAwal.Canvas.Pixels[i,j];
      r := getRValue(clr);
      g := getGValue(clr);
      b := getBValue(clr);
      abu := round((r+g+b)/3);

      if abu <= 128 then
      begin
        GamBiner.Canvas.Pixels[i,j] := RGB(0,0,0);
      end
        else
      begin
        GamBiner.Canvas.Pixels[i,j]:=RGB(255,255,255);
      end;

    end;
  end;
end;
```

Source code 4.2 Proses konversi *binerisasi*

4.3.1.2 *Character Segmentation*

Proses pemisahaan karakter merupakan proses untuk memisahkan tiap karakter dari setiap kata tulisan latin yang ditulis secara bersambung. Terdapat dua pemisahan (segmentasi) pada proses ini. Pertama secara vertikal sehingga didapatkan batas kiri dan kanan karakter. Kemudian dilanjutkan pemisahaan horisontal dari batas vertikal tiap karakter sehingga dihasilkan batas atas dan bawah tiap karakter. *Source code* 4.3 menjelaskan proses pemisahaan karakter pada tulisan latin yang ditulis secara bersambung.

```
…
//segmentasi vertikal
index_karakter_kand :=0;
for x:=0 to w-1 do
begin
  jum_hitam[x]:=0;
  for y:=0 to h-1 do
    if GamOlah.Canvas.Pixels[x,y]<> clWhite then
     inc (jum_hitam[x]);
  if jum_hitam[x] > 0.08*h then
  begin
    inc(index_karakter_kand);
    karakter_kand[index_karakter_kand]:=x;
  end;
end;

jum_kand_karakter:=0;
for i:=1 to index_karakter_kand do
begin
  if karakter_kand[i]-1 <> karakter_kand[i-1] then
  begin
    inc(jum_kand_karakter);
    x_min_karakter[jum_kand_karakter] := karakter_kand[i];
  end;
  if karakter_kand[i]+1 <> karakter_kand[i+1] then
  begin
    x_max_karakter[jum_kand_karakter] := karakter_kand[i];
  end;
end;

//cek noise
jum_karakter:=0;
for i:=1 to jum_kand_karakter do
begin
  jum_hitam_ver[i]:=0;
  for x:=x_min_karakter[i] to x_max_karakter[i] do
    for y := 0 to h-1 do
    begin
      if GamOlah.Canvas.Pixels[x,y] <> clWhite then
        inc(jum_hitam_ver[i]);
    end;
  //threshold kandidat karakter
 thres_kand_kar[i]:=5;
  if(jum_hitam_ver[i]>thres_kand_kar[i])and((x_max_karakter[i]-x_min_karakter[i])>= 5) then
  begin
    inc(jum_karakter);
    CharKata[jum_karakter].xmin := x_min_karakter[i];
    CharKata[jum_karakter].xmax := x_max_karakter[i];
  end;
end;
//END segmentasi vertikal

//segmentasi horizontal
for i:=0 to jum_karakter do   //i:=1
```

```
begin
  index_kar_ver[i] := 0;
  for y:=0 to h-1 do
  begin
    jum_hitam_hor[i][y]:=0;
    for x:= CharKata[i].xmin to CharKata[i].xmax do
    begin
      if GamOlah.Canvas.Pixels[x,y]<> clWhite then
        inc (jum_hitam_hor[i][y]);
    end;
    if jum_hitam_hor[i][y] > 2 then
    begin
      inc(index_kar_ver[i]);
      kar_ver[i][index_kar_ver[i]]:=y;
    end;
  end;

  jum_kand_ver_atas[i]:=0;
  jum_kand_ver_bawah[i]:=0;
  for a:=1 to index_kar_ver[i] do
  begin
    if kar_ver[i][a]-1 <> kar_ver[i][a-1] then
    begin
      Inc(jum_kand_ver_atas[i]);
      kand_min_y_kar[i][jum_kand_ver_atas[i]] := kar_ver[i][a];
    end;
    if kar_ver[i][a]+1 <> kar_ver[i][a+1] then
    begin
      Inc(jum_kand_ver_bawah[i]);
      kand_max_y_kar[i][jum_kand_ver_bawah[i]]:= kar_ver[i][a];
    end;
  end;

  CharKata[i].ymin:=kand_min_y_kar[i][jum_kand_ver_atas[i]];
  CharKata[i].ymax:=kand_max_y_kar[i][jum_kand_ver_bawah[i]];
end;
//END segmentasi horizontal
...
```

Source code 4.3 Segmentasi karakter

Dari proses segmentasi karakter akan didapat citra huruf tiap karakter dari tulisan tangan latin.

4.3.2 Implementasi Tahap Pengenalan (*Processing*)

Dalam proses pengenalan, citra tiap karakter yag didapat dari proses segmentasi akan dilakukan pengenalan dengan algoritma Fuzzy ART terhadap data latih yang telah dibuat. Struktur data yang digunakan pada implementasi tahap *processing* ditunjukkan pada *Source code* 4.4.

```
…
//struktur data untuk vektor bobot dan vektor input
const W_sampel = 8;
      H_sampel = 8;
Type
  Tdata = array[0..(W_sampel*H_sampel*2)-1]of Real;
  TKarakter = record
      huruf : String;
      data : Tdata;
      end;
var
  FormPengenalan: TFormPengenalan; MenuProgram : TMainMenu;
  vektorTesting : array[0..20] of TKarakter;
…

//struktur data untuk Fuzzy ART
var
  FormFuzzyArt: TFormFuzzyArt;
  choiceFunction : Tdata;
  vigilance, choiceParam, learningParam : real;
  inputNode : integer;
  selectedCategory, categoryNode, epoch  : integer;
  currentInput : Tdata;
  categoryWeight : array[0..200] of Tdata;
  kelasCategory: array of String;
  indexKelasCategory : integer;
…
```

Source code 4.4 Struktur data *processing*

Dalam proses pengenalan, citra karakter akan diubah terlebih dahulu menjadi vektor input yang bernilai antara 0 sampai dengan 1 dan ditambahkan bentuk *complement code*. Citra karakter dibagi menjadi blok-blok daerah berukuran 4x4. Masing-masing blok akan dicari nilai rata-rata sehingga mendapatkan nilai vektor dengan rentang nilai antara 0 sampai dengan 1. Setelah didapatkan nilai vektor input tersebut, selanjutnya ditambahkan dengan bentuk *complement code* dari vektor input tersebut. *Source code* 4.5 akan menjelaskan proses konversi blok, sedangkan *Source code* 4.6 menjelaskan proses konversi citra menjadi vektor input.

```
…
lbr_blok := (w/W_sampel);
pjg_blok := (h/H_sampel);
…
function benar(xa,ya:integer):boolean;
var i,j,yawal,xawal,yakhir,xakhir :integer;
  res:boolean;
  jum_piksel_putih : integer;
```

```
begin
  yawal:= round((ya-1)*pjg_blok);
  xawal:= round((xa-1)*lbr_blok);
  yakhir := round(yawal+pjg_blok)-1;
  xakhir := round(xawal+lbr_blok)-1;
  jum_piksel_putih := 0;
  for j:=yawal to yakhir do
    for i:=xawal to xakhir do
    begin
      if img.Canvas.Pixels[i,j]<>clWhite then
          inc(jum_piksel_putih);
    end;
  rata := RoundTo(jum_piksel_putih / 16, -2);
  res := true;
  result:=res;
end;
…
```

Source code 4.5 Konversi blok

```
…
begin
w := img.Picture.Width;
h := img.Picture.Height;
ind :=0;
for i := 0 to 1 do
  for y := 1 to H_sampel do
    for x := 1 to W_sampel do
    begin
      if benar(x,y)then
      begin
        if i = 0 then
          vektorTesting[index-1].data[ind] := Rata
        else
          vektorTesting[index-1].data[ind] := 1-Rata;
        inc(ind);
      end;
    end;
end;

…
```

Source code 4.6 Konversi vektor input

Vektor input yang didapat akan dikenali dengan menggunakan Fuzzy ART. Hasil dari pengenalan ini adalah karakter dimana vektor bobot karakter tersebut yang paling mendekati vektor input setelah dilakukan proses pengenalan dengan metode Fuzzy ART. Implementasi perhitungan yang terdapat pada Fuzzy ART dapat dilihat pada *Source code* 4.7. Sedangkan implementasi proses

pengenalan menggunakan Fuzzy ART dapat dilihat pada *Source code* 4.8.

```
...
Procedure art(inputNode1:integer;categoryNode1:integer;
      choiceParam1:real;learningParam1:real;vigilance1:real);
var i,j:integer;
begin
  SetLength(kelasCategory, categoryNode);
  for i := 0 to categoryNode - 1 do
    kelasCategory[i] := '';
  for i := 0 to categoryNode-1 do
    for j := 0 to inputNode do
    begin
      categoryWeight[i][j] := 1;
    end;
  selectedCategory := 0;
end;

Procedure setCurrentInput(inputData:TData);
var i:integer;
begin
  for i := 0 to 127 do
    currentInput[i] := inputData[i];
end;

function fuzzyNorm(input:Tdata):real;
var i: integer; dRet: real;
begin
  dRet := 0;
  for i := 0 to 127 do
  begin
    dRet := dRet + Abs(input[i]);
  end;
  result := dRet;
end;

function fuzzyAnd(inputA:Tdata;inputB:Tdata):Tdata;
var i: integer; dRet: Tdata;  nilaiMin: real;
begin
  for i := 0 to 127 do
  begin
    if inputA[i] < inputB[i] then
      nilaiMin := inputA[i]
    else
      nilaiMin := inputB[i];
    dRet[i] := nilaiMin;
  end;
  result := dRet;
end;

procedure setChoiceFunction;
var i: integer;
```

```
begin
  for i := 0 to categoryNode-1  do
  begin
choiceFunction[i]:=fuzzyNorm(fuzzyAnd(currentInput,categoryWeight[i]))/(choiceParam+fuz
zyNorm(categoryWeight[i]));
  end;
end;

procedure pilihCategory;
var dMax : real; i : integer;
begin
  dMax := 0;
  for i := 0 to categoryNode-1 do
  begin
    if dMax < choiceFunction[i] then
    begin
      dMax := choiceFunction[i];
      selectedCategory := i;
    end;
  end;
end;

function cekVigilance:boolean;
var dTemp : real;
begin
  pilihCategory;
  if choiceFunction[selectedCategory] = -1 then
  begin
    selectedCategory := -1;
  end;
dTemp:=fuzzyNorm(fuzzyAnd(currentInput,categoryWeight[selectedCategory]))/fuzzyNorm(c
urrentInput);
  if dTemp >= vigilance then
  begin
    result := true;
  end
  else
  begin
    choiceFunction[selectedCategory] := -1;
    result := cekVigilance;
  end;
end;

procedure learn;
var i : integer; dTemp : Tdata;
begin dTemp:=fuzzyAnd(currentInput,categoryWeight[selectedCategory]);
  for i := 0 to 127 do
categoryWeight[selectedCategory][i]:=learningParam*dtemp[i]+(1-
learningParam)*categoryWeight[selectedCategory][i];
end;

procedure addCategory;
var i,j : integer; weightTemp : array of Tdata;
begin
  SetLength(weightTemp, categoryNode+1);
```

```
  for i := 0 to categoryNode-1 do
    for j := 0 to inputNode do
    begin
      weightTemp[i][j] := categoryWeight[i][j];
    end;
  for j := 0 to inputNode do
    weightTemp[categoryNode+1][j] := currentInput[j];
  selectedCategory := categoryNode + 1;
  categoryNode := categoryNode + 1;
  for i := 0 to categoryNode - 1 do
    for j := 0 to inputNode do
    begin
      categoryWeight[i][j] := weightTemp[i][j];
    end;
end;

procedure prosesCurrentInput;
begin
  setChoiceFunction;
  if cekVigilance then
  begin
    learn;
  end
  else
    addCategory;
  if kelasCategory[selectedCategory] = '' then
kelasCategory[selectedCategory]:=sampel[indexKelasCategory].huruf;
end;

function TFormFuzzyArt.operate:String;
begin
  setChoiceFunction;
  pilihCategory;
  Result := kelasCategory[selectedCategory];
end;
...
```

Source code 4.7 Fungsi-fungsi perhitungan Fuzzy ART

```
...
procedure TFormPengenalan.BtPengenalanClick(Sender: TObject);
var testOutput: array[0..20] of String; i,j: integer;
    pengenalan : string; tempVektorTesting : UFuzzyArt.Tdata;
begin
  pengenalan := '';
  for i := 0 to jum_karakter-1 do
  begin
    for j := 0 to 127 do
      tempVektorTesting[j] := vektorTesting[i].data[j];
    FormFuzzyArt.setCurrentInput(tempVektorTesting);
    testOutput[i] := FormFuzzyArt.operate;
    pengenalan := pengenalan + testOutput[i] ;
  end;
  edKarakter.Text := pengenalan;
```

```
end;
...
```

Source code 4.8 Pengenalan Fuzzy ART

4.4 Skenario Pengujian

Pada sistem pengenalan tulisan tangan latin dengan metode Fuzzy ART akan dilakukan proses pengujian. Pengujian digunakan untuk mencari nilai keakuratan proses segmentasi dan proses pengenalan karakter. Proses pengujian dilakukan dengan menggunakan beberapa data *testing* dengan beberapa kondisi parameter tertentu, untuk mengetahui tingkat akurasi paling optimal yang dihasilkan oleh sistem.

4.4.1 Data Pengujian

Proses pengujian dilakukan dengan menggunakan tulisan tangan latin hasil *scanning* yang terdiri dari 110 kata. Masing-masing kata memiliki variasi tulisan yang berbeda-beda dan akan dilakukan pengujian pada setiap kata tersebut.

Data uji yang digunakan merupakan data citra berformat *.bmp dan memiliki ukuran 350x150 piksel, serta berbentuk citra *grayscale*. Citra uji berasal dari citra tulisan tangan latin hasil *scanning* dari 22 macam kata, yang masing-masing kata ditulis sebanyak 5 kali, sehingga mendapatkan variasi tulisan tangan latin yang berbeda-beda. Sehingga keseluruhan data testing berjumlah 110 kata citra tulisan tangan latin.

4.4.2 Lingkungan Pengujian

Lingkungan pengujian dalam system pengenalan tulisan tangan latin menggunakan metode Fuzzy ART akan dilakukan dengan beberapa kali pengujian terhadap citra data uji dengan lingkungan pengujian atau settingan paramater sebagai berikut :

a. *vigilance parameter* (ρ) : 0.80 dan 0.85
b. *choice parameter* (α) : 0.01, 0.1 dan 1.00
c. *learning rate* (β) : 0.80, 0.90 dan 1.00

Daftar pengujian berdasarkan settingan nilai *vigilance parameter* (ρ), *choice parameter* (α) dan *learning rate* (β) yang diatur dengan semua kombinasi dari ketiga parameter tersebut.

4.4.3 Hasil Pengujian

4.4.3.1 Hasil Pengujian *Preprocessing*

Dalam tahap pengenalan karakter tulisan tangan latin, pertama kali diujikan adalah proses pemisahan (segmentasi) karakter pada citra awal. Pengujian ini dilakukan dengan menggunakan 110 kata citra tulisan tangan latin hasil *scanning* dan dilakukan pada setiap kata citra tulisan tangan latin tersebut. Tujuan dari proses ini adalah untuk mendapatkan citra tiap karakter dari masing-masing kata tulisan tangan latin.

Pada proses segmentasi ini terdapat dua kemungkinan hasil, benar atau salah. Kondisi segmentasi benar merupakan kondisi dimana semua karakter pada data uji tulisan tangan latin dapat disegmentasi dengan benar. Sedangkan kondisi segmentasi salah jika salah satu atau lebih karakter pada tulisan tulisan tangan latin tidak dapat melakukan segmentasi dengan benar. Tabel 4.1 berikut merupakan tabel hasil dari proses segmentasi.

Tabel 4.1 Hasil Segmentasi Tulisan Tangan Latin

Jumlah Kata	Benar	Salah
110	71 (64,54%)	39 (35,46%)

Dari Tabel 4.1 didapatkan hasil keakuratan proses segmentasi karakter pada 110 kata citra tulisan tangan latin adalah sebesar 64,54%, yaitu sebanyak 71 kata citra tulisan tangan latin berhasil dilakukan segmentasi dengan benar, sedangkan sisanya sebanyak 39 kata citra tulisan tangan latin tidak dapat dilakukan segmentasi dengan benar. Data hasil uji segmentasi tulisan tangan latin dapat dilihat pada Lampiran 2.

4.4.3.2 Hasil Pengujian Pengenalan dengan Fuzzy ART

Citra karakter hasil segmentasi dengan benar akan dilakukan proses pengenalan menggunakan Fuzzy ART. Untuk itu diperlukan pemilihan parameter Fuzzy ART yang optimal dalam pengenalan karakter tersebut. Pemilihan parameter Fuzzy ART ini meliputi pemilihan nilai *vigilance parameter* (ρ), *choice parameter* (α) dan *learning rate* (β).

Nilai yang diujikan adalah nilai *vigilance parameter* (ρ), *choice parameter* (α) dan *learning rate* (β) . Akurasi hasil pengujian dihitung dengan menggunakan rumus seperti pada Persamaan 2.12.

Pengujian parameter Fuzzy ART ini dilakukan dengan menggunakan 443 citra karakter hasil dari proses segmentasi yang

dinyatakan benar. Hasil dari uji coba parameter Fuzzy ART dapat dilihat pada Tabel 4.2.

Tabel 4.2 Hasil Uji Parameter Fuzzy ART

Vigilance Parameter	***Choice Parameter*** (α)	***Learning rate*(β)**	**Prosentase Keakuratan (%)**
0.80	0.01	0.80	61,40 %
		0.90	**75,17 %**
		1.00	72,91 %
	0.1	0.80	61,17 %
		0.90	74,94 %
		1.00	72,68 %
	1	0.80	60,95 %
		0.90	74,71 %
		1.00	72,46 %
0.85	0.01	0.80	68,62 %
		0.90	74,27 %
		1.00	68,84 %
	0.1	0.80	68,39 %
		0.90	74,04 %
		1.00	68,62 %
	1	0.80	68,17 %
		0.90	73,81 %
		1.00	68,40 %

Dari percobaan di atas dipilih nilai parameter dengan hasi keakuratan tertinggi, yaitu dengan kombinasi nilai *vigilance parameter* (ρ) sebesar 0.80, *learning rate* (β) sebesar 0.90 dan *choice parameter* (α) sebesar 0.01. Nilai parameter ini digunakan untuk mengenali citra karakter tulisan tangan latin dari hasil proses segmentasi. Tabel 4.3 menunjukkan tingkat keberhasilan pengenalan karakter menggunakan Fuzzy ART.

Tabel 4.3 Hasil Uji Pengenalan Karakter

Jumlah Karakter	Benar	Salah
443	333 (75,17%)	110 (24,83%)

Tabel 4.3 menunjukkan bahwa dari jumlah 443 citra karakter yang dihasilkan dengan benar oleh proses segmentasi, didapatkan 333 karakter yang berhasil dikenali dengan benar dan sisanya sebanyak 110 karakter tidak dapat dikenali dengan benar. Dari Persamaan 2.12, maka didapat tingkat keberhasilan pengenalan tulisan tangan latin menggunakan Fuzzy ART adalah sebesar 75,17% dan tingkat kesalahannya sebesar 24,83%. Detail data hasil uji pengenalan karakter menggunakan Fuzzy ART ditunjukkan pada Lampiran 3.

4.5 Analisa Hasil

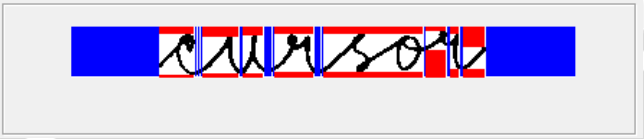
4.5.1 Analisa Hasil *Preprocessing*

Dari Tabel 4.1 menunjukkan bahwa pada proses segmentasi karakter didapatkan hasil keakuratan segmentasi sebesar 64,54% dan nilai kesalalahan segmentasi sebesar 35,46%. Besarnya nilai kesalahan pada segmentasi disebabkan oleh beberapa hal. Salah satu penyebab kesalahan ini adalah kesalahan pada segmentasi karakter secara vertikal, yang terdapat dua jenis kesalahan. Kesalahan yang pertama, beberapa karakter terpotong tidak sempurna karena pada bagian karakter tersebut jumlah piksel hitamnya lebih kecil dari nilai *threshold* sehingga tidak dianggap sebagai salah satu kesatuan karakter. Gambar 4.3 menunjukkan contoh kesalahan pada segmentasi karakter.

Gambar 4.3 Contoh Kesalahan Segmentasi Karakter

Kesalahan segmentasi yang kedua disebabkan oleh persambungan antar karakter. Nilai jumlah piksel hitam pada persambungan antar karakter lebih besar jika dibandingkan dengan nilai *threshold* sehingga persambungan antar karakter tersebut

dianggap sebagai bagian dari karakter.Untuk lebih jelasnya dapat dilihat contoh kesalahan segmentasi karakter pada Gambar 4.4.

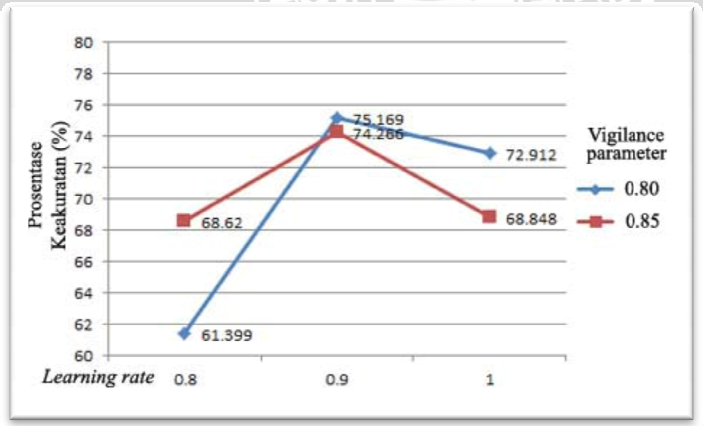


Gambar 4.4 Contoh Kesalahan Segmentasi Karakter

Dari hasil pengujian yang diperoleh, dapat disimpulkan bahwa proses segmentasi dapat dilakukan pada tulisan tangan latin. Tingkat akurasi pada proses segmentasi membuktikan bahwa metode segmentasi yang dilakukan pada penelitian ini masih perlu dilakukan penambahan atau penggabungan metode lain yang mungkin dapat mengoptimalkan proses segmentasi pada tulisan tangan latin.

4.5.2 Analisa Hasil Pengenalan Karakter

Berdasarkan data pengujian yang dilakukan, maka akan didapatkan nilai perubahan tingkat akurasi terhadap perubahan nilai-nilai parameter yang diberikan. Sehingga setiap nilai parameter dapat mengakibatkan perubahan nilai akurasi pada pengenalan karakter. Pada Tabel 4.2 menunjukkan bahwa pemilihan kombinasi parameter Fuzzy ART mempengaruhi tingkat keakuratan pengenalan. Dari hasil Tabel 4.2 dapat dibuat grafik seperti pada Gambar 4.5.



Gambar 4.5 Grafik pengaruh nilai parameter Fuzzy ART

Dengan arsitektur Fuzzy ART yang digunakan, maka didapatkan nilai keakuratan tertinggi dari jumlah 443 citra karakter sebesar 75,17%. Pada hasil pengujian terdapat nilai-nilai yang paling berpengaruh terhadap akurasi pengenalan tulisan tangan latin. Gambar 4.5 menunjukkan bahwa pada kenaikan nilai *learning rate* yang pertama menyebabkan hasil akurasi meningkat sekitar 5% sampai 14%. Peningkatan parameter *learning rate* selanjutnyamengakibatkan nilai akurasi menurun sekitar 2% sampai 6%. Perubahan nilai *vigilance parameter* berpengaruh terhadap jumlah kategori pada ART. Semakin besar nilai *vigilance parameter* maka mengakibatkan jumlah kategori pada ART semakin banyak, ini baik terhadap pengenalan data training, namun tidak baik terhadap data yang belum di training. Semakin besar nilai *vigilance parameter* maka jaringan semakin selektif dan spesifik terhadap pola masukan. Sedangkan untuk perubahan nilai *choice parameter,* hanya terdapat perubahan tingkat akurasi yang tidak signifikan.

Dari pengujian yang telah dilakukan didapatkan masing-masing parameter (*vigilance parameter, learning rate* dan *choice parameter*) memiliki pengaruh akurasi yang berbeda terhadap pengenalan. Berdasarkan uji coba tersebut didapat parameter Fuzzy ART yang memiliki nilai keakuratan tertinggi, yaitu dengan kombinasi nilai *vigilance parameter* (ρ) sebesar 0.80, *choice parameter* (α) sebesar 0.01 dan *learning rate* (β) sebesar 0.90. Nilai parameter ini digunakan untuk mengenali citra karakter tulisan tangan latin hasil dari proses segmentasi.

Pada Tabel 4.3 didapat tingkat keberhasilan pengenalan karakter sebesar 75,17 %. Hal ini membuktikan bahwa metode Fuzzy ART mampu mengenali karakter pada tulisan tangan latin. Tabel 4.3 juga menunjukkan bahwa terdapat kesalahan dalam pengenalan karakter yaitu dengan nilai kesalahan sebesar 24,83%. Contoh dari kesalahan-kesalahan pengenalan karakter tulisan tangan latin ditunjukkan pada Gambar 4.6.

Gambar 4.6 Contoh Kesalahan Pengenalan Karakter

Kesalahan pada pengenalan karakter ini disebabkan oleh beberapa hal. Salah satu penyebab kesalahan ini adalah adanya penulisan karakter huruf latin yang memiliki bentuk mirip dengan karakter lainnya. Seperti huruf “e” dengan huruf “l”, huruf “u” dengan “v”, dan sebagainya. Kesalahan juga dapat terjadi akibat kondisi citra karakter yang tidak jelas atau terdapat bagian yang hilang pada citra. Hal ini menyebabkan nilai kedekatan dengan karakter bobot lain lebih besar dibandingkan dengan karakter yang sesungguhnya dan mengakibatkan kesalahan pengenalan karakter tersebut. Kesalahan pengenalan juga dipengaruhi oleh kondisi dari jenis tulisan tangan latin. Tulisan tangan latin dengan garis karakter yang tipis juga tidak dapat dikenali dengan baik.

Keberhasilan pengenalan karakter juga tidak lepas dari proses segmentasi karakter. Semakin jelas citra karakter hasil segmentasi akan membuat pengenalan karakter menjadi lebih baik.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dari percobaan dan analisa yang telah dilakukan, dapat disimpulkan bahwa :

1. Sistem ini dapat melakukan proses segmentasi karakter pada tulisan tangan latin dengan hasil keakuratan sebesar 64,54 % dari 110 citra uji tulisan tangan latin.
2. Permasalahan pengenalan tulisan tangan latin dapat diaplikasikan dengan menggunakan metode Fuzzy ART *Neural Network.*
3. Didapatkan tingkat keberhasilan pengenalan dari 443 karakter tulisan tangan latin menggunakan Fuzzy ART sebesar 75,17%.

5.2 Saran

Sebagai penelitian lebih lanjut, beberapa saran yang dapat menjadi pertimbangan untuk dilakukan adalah :

1. Memperbaiki proses segmentasi karakter dengan menambahkan proses pada *preprocessing* seperti *threshold* deteksi karakter agar didapatkan citra karakter yang lebih baik sebelum masuk ke tahap proses pengenalan karakter.
2. Memperbaiki proses *normalization* sebelum memasukkan citra segmentasi ke dalam proses pengenalan dengan menggunakan rasio perbandingan tinggi dan lebar karakter.
3. Meningkatkan akurasi pengenalan karakter dengan menggunakan metode Fuzzy ART yang sudah dikembangkan, yaitu metode Fuzzy ARTMAP yang merupakan jaringan syaraf tiruan dengan proses pembelajaran *supervised learning*.


UNIVERSITAS BRAWIJAYA

DAFTAR PUSTAKA

Achmad, Balza dan Firdausy, Kartika. 2005. *Teknik Pengolahan Citra Digital Menggunakan Delphi,* Ardi Publishing. Yogyakarta.

Arica, Nafiz dan Fatos, T. 2001. *An Overview of Character Recognition Focused on Off-line Handwrtiting*. Yarman-Vural.

Budhi, G.S, Gunawan, & I, Jaowry S,. 2006. *Metode Jaringan Syaraf Tiruan Backpropagation untuk Pengenalan Huruf Cetak pada Citra Digital*.

Carpenter. G.A., Grossberg. S., & Rosen. D.B. 1991. *Fuzzy ART: Fast Stable Learning and Categorization of Analog Pattern by an Adaptive Resonance System*, Neural Networks. Boston Univercity.

Diaz, Hartadi, dkk. *Simulasi Penghitungan Jumlah Sel darah Merah*. http://www.geocities.com/transmisi_ecundip/diazhartadi.pdf. Tanggal akses: 31 Agustus 2010.

Fauset, L. 1994. *Fundamentals of Neural Network.* Precentice Hall, New Jersey.

Gader. P.D., James M. Keller, Raghu Krishnapuram, Jung-Hsien Chiang, Magdi A. Mohamed. 1997. *Neural and Fuzzy Methods in Handwriting Recognition.*Computer.

Gonzales, R.C. & R. E. Woods. 1993. *Digital Image Processing.* Addison Wasley, Massachussets.

Ibrahim, Arif. 2003. *Perancangan Algoritma Kriptografi Kunci Publik menggunakan Arsitektur Jaringan Syaraf Tiruan*. Program Studi Teknik Informatika, ITB. Bandung.

Jain, A.K. 1995. *Fundamentals of Digital Image Processing.* Practice Hall, New Delhi

Jean, Georges. 1997. *Writing: The story of alphabets and scripts*. London: Thames and Hudson Ltd.

Kausari, Mulki. 2009. *Komputerisasi Iridologi untuk Mendeteksi Kondisi Ginjal menggunakan PCA dan KNN*. IT Telkom.

Koerich. A. L. dkk . 2003. *Large Vocabulary Off-line Handwriting Recognition : A Survey*.

Kristanto, Andi. 2004. *Jaringan Syaraf Tiruan (Konsep Dasar, Algoritma dan Aplikasi).* Graha Media, Yogyakarta.

Lecoinet, E. and Baret, O. 1994. *Cursive Word Recognition: Methods and Strategies,* in Fundamentals in Handwriting Recognition, Springer Verlag.

Leila, C. and Mohammed, B. 2008. *ART network for Arabic Handwritten Recognition Syste*. The International Arab Conference on Information Technology.

Lim, Resmana, dkk. 2003. *Sistem Pengenalan Plat Nomor Mobil dengan Metode Principal Components Analysis*. Universitas Kristen Petra. Surabaya.

Munir, Rinaldi. 2004. *Pengolahan Citra Digital dengan Pendekatan Algoritmik*. Informatika, Bandung.

Rodrigues, R., Thome, G. and Carlos, A. 2000. *Cursive character recocnition – a character segmentation method using projection profile-based technique*. Brasil.

Sediyono, Eko.2000.*Metoda Jaringan Saraf Tiruan dengan Arsitektur Jaringan Radial Basis Kooperatif dan Kompetitif untuk Pengenalan Tulisan Tangan*. Fakultas Teknik Jurusan Teknik Elektro, Univ. Kristen Satya Wacana Salatiga. Jakarta.

Tay, Y.H. & Khalid, M. 2002. *Comparison of Fuzzy ARTMAP and MLP Neural Network for Hand-written Character Recognition.*

Wasserman, Philip D. 1989. Neural computing: theory and practice, New York: Van Nostrand Reinhold, ISBN 0-442-20743-3

LAMPIRAN

Lampiran 1 Karakter Data Latih

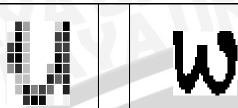
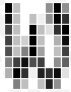
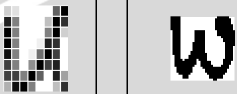
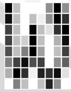
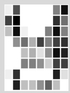
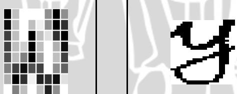
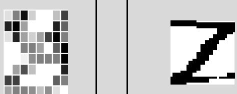
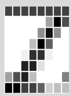
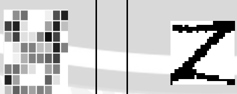
Gambar Karakter	Karakter	Vektor 8x8		Gambar Karakter	Karakter	Vektor 8x8
	a				b	
	a				b	
	a				c	
	a				c	
	a				c	
	b				c	
	b				c	
	b				d	

	d				f	
	d				f	
	d				f	
	d				g	
	e				g	
	e				g	
	e				g	
	e				g	
	e				h	
	f				h	
	f				h	

	h				j	
	h				j	
	i				k	
	i				k	
	i				k	
	i				k	
	i				k	
	j				l	
	j				l	
	j				l	
	l				n	

	l				o	
	m				o	
	m				o	
	m				o	
	m				o	
	m				p	
	n				p	
	n				p	
	n				p	
	n				p	
	q				s	

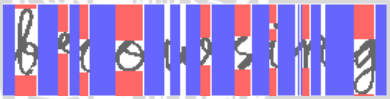
	q			s	
	q			s	
	q			s	
	q			t	
	r			t	
	r			t	
	r			t	
	r			t	
	r			u	
	s			u	
	u			w	

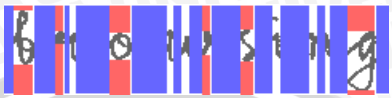
	u				w	
	u				w	
	v				x	
	v				x	
	v				x	
	v				x	
	v				x	
	w				y	
	w				y	
	y				z	
	y				z	

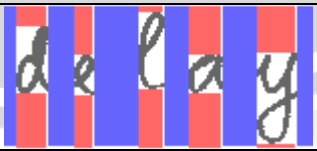
	y				z	
	z				z	

Lampiran 2 Data Uji Segmentasi Karakter

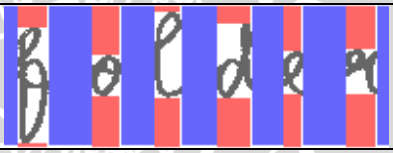
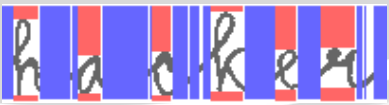
No.	Nama File	Segmentasi
1	address1.bmp	
2	address2.bmp	
3	address3.bmp	
4	address4.bmp	
5	address5.bmp	
6	analog1.bmp	
7	analog2.bmp	
8	analog3.bmp	

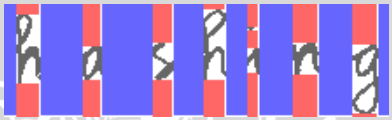
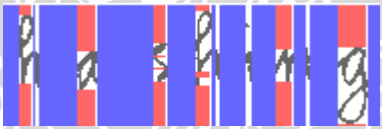
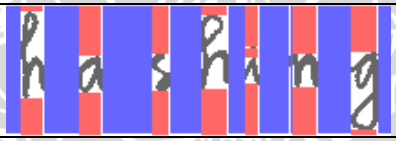
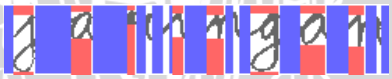
9	analog4.bmp	
10	analog5.bmp	
11	artificial1.bmp	
12	artificial2.bmp	
13	artificial3.bmp	
14	artificial4.bmp	
15	artificial5.bmp	
16	browsing1.bmp	
17	browsing2.bmp	
18	browsing3.bmp	
19	browsing4.bmp	

20	browsing5.bmp	
21	chace1.bmp	
22	chace2.bmp	
23	chace3.bmp	
24	chace4.bmp	
25	chace5.bmp	
26	cookie1.bmp	
27	cookie2.bmp	
28	cookie3.bmp	
29	cookie4.bmp	
30	cookie5.bmp	

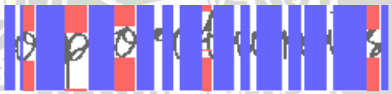
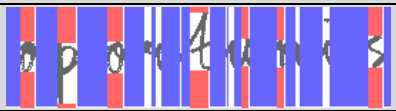
31	cursor1.bmp	
32	cursor2.bmp	
33	cursor3.bmp	
34	cursor4.bmp	
35	cursor5.bmp	
36	delay1.bmp	
37	delay2.bmp	
38	delay3.bmp	
39	delay4.bmp	
40	delay5.bmp	

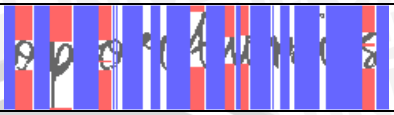
41	desktop1.bmp	
42	desktop2.bmp	
43	desktop3.bmp	
44	desktop4.bmp	
45	desktop5.bmp	
46	exodus1.bmp	
47	exodus2.bmp	
48	exodus3.bmp	
49	exodus4.bmp	

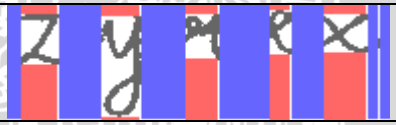
50	exodus5.bmp	
51	folder1.bmp	
52	folder2.bmp	
53	folder3.bmp	
54	folder4.bmp	
55	folder5.bmp	
56	hacker1.bmp	
57	hacker2.bmp	
58	hacker3.bmp	

59	hacker4.bmp	
60	hacker5.bmp	
61	hashing1.bmp	
62	hashing2.bmp	
63	hashing3.bmp	
64	hashing4.bmp	
65	hashing5.bmp	
66	jaringan1.bmp	
67	jaringan2.bmp	
68	jaringan3.bmp	

69	jaringan4.bmp	
70	jaringan5.bmp	
71	localhost1.bmp	
72	localhost2.bmp	
73	localhost3.bmp	
74	localhost4.bmp	
75	localhost5.bmp	
76	mandrake1.bmp	
77	mandrake2.bmp	
78	mandrake3.bmp	

79	mandrake4.bmp	
80	mandrake5.bmp	
81	matrix1.bmp	
82	matrix2.bmp	
83	matrix3.bmp	
84	matrix4.bmp	
85	matrix5.bmp	
86	oportunis1.bmp	
87	oportunis2.bmp	
88	oportunis3.bmp	
89	oportunis4.bmp	

90	oportunis5.bmp	
91	processor1.bmp	
92	processor2.bmp	
93	processor3.bmp	
94	processor4.bmp	
95	processor5.bmp	
96	switch1.bmp	
97	switch2.bmp	
98	switch3.bmp	
99	switch4.bmp	
100	switch5.bmp	
101	virtual1.bmp	

102	virtual2.bmp	
103	virtual3.bmp	
104	virtual4.bmp	
105	virtual5.bmp	
106	zyrex1.bmp	
107	zyrex2.bmp	
108	zyrex3.bmp	
109	zyrex4.bmp	
110	zyrex5.bmp	

Lampiran 3 Data Uji Pengenalan Karakter

No.	Nama File	Karakter yg diuji	Jum Karakter	Karakter Benar	Jum Benar
1	address2.bmp	a,d,d,r,e,s,s	7	d,d,r,s,s	5
2	address3.bmp	a,d,d,r,e,s,s	7	d,r,s,s	4
3	address4.bmp	a,d,d,r,e,s,s	7	a,d,r,s	4
4	address5.bmp	a,d,d,r,e,s,s	7	a,d,r,s,s	5
5	analog1.bmp	a,n,a,l,o,g	6	a,n,a,o,g	5
6	analog2.bmp	a,n,a,l,o,g	6	a,n,a,o	4
7	analog3.bmp	a,n,a,l,o,g	6	a,n,a,o	4
8	analog4.bmp	a,n,a,l,o,g	6	a,n,a,o	4
9	analog5.bmp	a,n,a,l,o,g	6	a,n,a,o	4
10	chace1.bmp	c,h,a,c,e	5	c,a,c,h,e	5
11	chace2.bmp	c,h,a,c,e	5	c,a,c,h,	4
12	chace3.bmp	c,h,a,c,e	5	c,a,c	3
13	chace4.bmp	c,h,a,c,e	5	c,a,c,e	4
14	chace5.bmp	c,h,a,c,e	5	c,a,c,h	4
15	cookie1.bmp	c,o,o,k,i,e	6	c,o,o,k,i	5
16	cookie2.bmp	c,o,o,k,i,e	6	o,o,k,i	4
17	cookie3.bmp	c,o,o,k,i,e	6	c,o,k,i	4
18	cookie4.bmp	c,o,o,k,i,e	6	c,o,k,i	4
19	cookie5.bmp	c,o,o,k,i,e	6	c,o,k,i,e	5
20	cursor1.bmp	c,u,r,s,o,r	6	c,u,r,s,o,r	6
21	cursor3.bmp	c,u,r,s,o,r	6	c,u,r,s,r	5
22	cursor4.bmp	c,u,r,s,o,r	6	c,r,s,o	4
23	cursor5.bmp	c,u,r,s,o,r	6	c,u,r,s,o,r	6
24	delay1.bmp	d,e,l,a,y	5	d,e,a,y	4
25	delay2.bmp	d,e,l,a,y	5	d,a,y	3
26	delay3.bmp	d,e,l,a,y	5	d,e,a,y	4
27	delay4.bmp	d,e,l,a,y	5	d,a,y	3
28	delay5.bmp	d,e,l,a,y	5	d,a,y	3
29	desktop1.bmp	d,e,s,k, t,o,p	7	d,s,k,t,o,p	6
30	desktop2.bmp	d,e,s,k, t,o,p	7	d,s,k,o,p	5
31	desktop3.bmp	d,e,s,k, t,o,p	7	d,e,s,k, t,o,p	7

32	desktop4.bmp	d,e,s,k, t,o,p	7	d,s,k,o,p	5
33	desktop5.bmp	d,e,s,k, t,o,p	7	d,e,s,o,p	5
34	exodus1.bmp	e,x,o,d,u,s	6	x,o,u,s	4
35	exodus2.bmp	e,x,o,d,u,s	6	x,d,u,s	4
36	exodus3.bmp	e,x,o,d,u,s	6	x,o,d,u,s	5
37	exodus4.bmp	e,x,o,d,u,s	6	x,d,u,s	4
38	exodus5.bmp	e,x,o,d,u,s	6	x,o,d,u,s	5
39	folder1.bmp	f,o,l,d,e,r	6	o,d,e,r	4
40	folder2.bmp	f,o,l,d,e,r	6	o,d,r	3
41	folder3.bmp	f,o,l,d,e,r	6	o,d,r	3
42	folder4.bmp	f,o,l,d,e,r	6	o,d,r	3
43	folder5.bmp	f,o,l,d,e,r	6	f,o,d,r	4
44	hacker1.bmp	h,a,c,k,e,r	6	h,a,c,k,e,r	6
45	hacker3.bmp	h,a,c,k,e,r	6	a,c,k,r	4
46	hacker4.bmp	h,a,c,k,e,r	6	a,c,k,e,r	5
47	hacker5.bmp	h,a,c,k,e,r	6	a,c,k,r	4
48	hashing1.bmp	h,a,s,h,i,n,g	7	a,s,i,n,g	5
49	hashing2.bmp	h,a,s,h,i,n,g	7	a,s,i,n,g	5
50	hashing4.bmp	h,a,s,h,i,n,g	7	a,s,i,n	4
51	hashing5.bmp	h,a,s,h,i,n,g	7	a,s,i,n,g	5
52	matrix1.bmp	m,a,t,r,i,x	6	m,a,t,r,i,x	6
53	matrix2.bmp	m,a,t,r,i,x	6	m,a,t,r,i,x	6
542	matrix3.bmp	m,a,t,r,i,x	6	m,t,r,i,x	5
55	matrix4.bmp	m,a,t,r,i,x	6	m,a,t,r,x	5
56	matrix5.bmp	m,a,t,r,i,x	6	m,a,t,r,i,x	6
57	processor1.bmp	p,r,o,c,e, s,s,o,r	9	p,o,c, s,s,o	6
58	processor2.bmp	p,r,o,c,e, s,s,o,r	9	p,r,c,s, s,o,r	7
59	processor3.bmp	p,r,o,c,e, s,s,o,r	9	p,r,o,c, s,s,o,r	8
60	processor4.bmp	p,r,o,c,e, s,s,o,r	9	p,r,o,c, s,s,o,r	8
61	processor5.bmp	p,r,o,c,e, s,s,o,r	9	p,o,s, s,o,r	6
62	switch4.bmp	s,w,i,t,c,h	6	s,w,t,c,h	5
63	virtual1.bmp	v,i,r,t,u,a,l	7	v,i,r,a	4

64	virtual2.bmp	v,i,r,t,u,a,l	7	v,i,r,t,u,a	6
65	virtual3.bmp	v,i,r,t,u,a,l	7	v,r,u,a	4
66	virtual4.bmp	v,i,r,t,u,a,l	7	v,r,t,u,a	5
67	zyrex1.bmp	z,y,r,e,x	5	z,y,r,e,x	5
68	zyrex2.bmp	z,y,r,e,x	5	z,y,r,x	4
69	zyrex3.bmp	z,y,r,e,x	5	z,y,r,x	4
70	zyrex4.bmp	z,y,r,e,x	5	z,y,r,x	4
71	zyrex5.bmp	z,y,r,e,x	5	z,y,r,ex	5
Total			**443**		**333**


UNIVERSITAS BRAWIJAYA