BIDANG ILMU: 538/REKAYASA PERANGKAT LUNAK

LAPORAN AKHIR

JUDUL:

IMPLEMENTASI ALGORITMA CHAIN CODE DAN LVQ PADA PENGENALAN HURUF HIJAIYAH

TIM PENGUSUL :

Uung Ungkawa, M.T (NIDN: 0411105902)

Maulana Nur Muhamad. (NRP: 15-2011-049)

Dibiayai oleh **Pribadi**

INSTITUT TEKNOLOGI NASIONAL

Januari 2016

Halaman Pengesahan

Judul Penelitian	:	IMPLEMENTASI ALGORITMA CHAIN CODE DAN LVQ PADA PENGENALAN HURUF HIJAIYAH
Kode/ Nama Rumpun Ilmu	:	538/ Rekayasa Perangkat Lunak
Ketua Peneliti	:	
a. Nama Lengkap	:	Ir Uung Ungkawa, M.T.
b. NIDN	:	0411105902
c. Jabatan Fungsional	:	Lektor
d. Program Studi	:	Teknik Informatika
e. Nomor HP	:	08121443095
f. Alamat surel (e-mail)	:	uung@itenas.ac.id
Anggota Peneliti (1)		
a. Nama Lengkap	:	Maulana Nur Muhamad
b. NRP	:	15-2011-049
c. Instansi	:	Mahasiswa Institut Teknologi Nasional (ITENAS) Bandung
Anggota Peneliti (2)		
a. Nama Lengkap	:	
b. NIDN	:	
c. Perguruan Tinggi	:	
Lama Penelitian Keseluruhan	:	6 bulan
Penelitian Tahun ke	:	1 (Satu)
Biaya Penelitian Keseluruhan	:	Rp. 5.000.000,-
Biaya Tahun Berjalan	:	- diusulkan ke DIKTI Rp. 0,- - dana internal PT Rp.0 - dana institusi lain Rp.0 - inkind sebutkan

Mengetahui

Dekan

Dani Rusirawan, S.T, M.T., Ph.D

NIP/NIK

Bandung, 24 – 1 - 2016

Ir Uung Ungkawa, M.T

NIP/NIK. 120071201

Menyetuji
Ketua LPPM Itenas

Dr. Tarsisius Kristyadi, S.T., M.T
NIP/NIK

IMPLEMETASI ALGORITMA CHAIN CODE DAN LVQ PADA PENGENALAN HURUF HIJAIYAH

Maulana Nur Muhamad, Jasman Pardede, Uung Ungkawa

Jurusan Teknik Informatika
Fakultas Teknologi Industri
Institut Teknologi Nasional

Intisari

Bahasa Arab merupakan pengantar dalam berbagai ilmu ke wilayah Eropa sehingga menjadi pondasi peradaban Eropa modern. Bahasa Arab oleh PBB juga akhirnya resmi dijadikan sebagai bahasa internasional ketujuh di dunia dengan penutur lebih dari dua ratus juta orang Arab. Bahasa Arab yang terbentuk dari huruf Hijaiyah juga merupakan huruf penyusun kata dalam Al-Qur'an. Maka dalam perkembangannya, akan terdapat banyak dokumen yang menggunakan bahasa Arab. *Chararacter Recognition* merupakan salah satu area studi dalam bidang pengenalan pola yang bisa menjadi salah satu referensi dalam memecahkan masalah diatas. Teknologi *Optical Character Recognition* (OCR) digunakan untuk membuat sebuah *softcopy* dokumen dari *hardcopy* dokumen. Hal tersebut mengurangi waktu pengerjaan dibandingkan dengan metode konvensional yang pengerjaannya memakan waktu yang relatif lama dengan mengetik ulang sebuah dokumen dari dokumen aslinya. Dalam penelitian ini, teknik ekstraksi ciri menggunakan algoritma berbasis *Chain Code* serta teknik pencocokan dengan menggunakan algoritma *Learning Vector Quantization* (LVQ). Dengan mekanisme melakukan penelusuran per-piksel, teknik *Chain Code* dapat digunakan untuk menemukan struktur pembentuk dari suatu objek. Hasil dari pengujian sistem diperoleh prosentase keberhasilan sebesar 78,5% dari 28 citra huruf Hijaiyah yang diuji.

Kata Kunci : karakter hijaiyah, ocr, kode rantai, lvq

IMPLEMENTATION OF CHAIN CODE AND LVQ ALGORITHM IN HIJAIYAH CHARACTER RECOGNITION

Maulana Nur Muhamad, Jasman Pardede, Uung Ungkawa

Department of Informatics Engineering
Faculty of Industrial Technology
National Institute of Technology

Abstract

The Arabic language is an introductory in various of science to the regions of Europe that it became the foundation of modern European civilization. Arabic by the United Nations also eventually officially serve as the seventh international language in the world with the speakers more than two hundred million Arabian. Arabic is formed of Hijaiyah letters is also the word composer letters in the Qur'an. Then in its development, there will be a lot of documents in Arabic. Character Recognition is one of the study areas in the field of pattern recognition that could be one of the references in solving the above problems. Optical Character Recognition (OCR) technology is used to create a softcopy of document from hardcopy of document. This matter reduces the work time as compared to conventional methods that the process takes a relatively long time to retype a document of the original document. In this research, the technique of feature extraction uses algorithm based on Chain Code as well as matching technique which use Learning Vector Quantization (LVQ) algorithms. By performing searches mechanism for per-pixel, Chain Code technique can be used to locate the structure forming of an object. The result of the system testing obtained the successful percentage 78.5% of the 28 Hijaiyah letter was tested.
Keywords : *hijaiyah character, ocr, chain code, lvq*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR RUMUS

BAB I
PENDAHULUAN

Pada bagian ini, pembahasan yang dijelaskan mengenai latar belakang dari penelitian yang dilakukan, rumusan masalah, tujuan penelitian, batasan masalah, metodologi yang digunakan, tinjauan pustaka serta sistematika dari penulisan laporan penelitian ini.

1.1 Latar Belakang

Peradaban Islam yang sejak zaman Daulah Abbasiyah mulai berkembang, menjadikan Bahasa Arab sebagai bahasa internasional. Semenjak itu, bahasa Arab menjadi bahasa peradaban yang ditandai dengan diterjemahkannya berbagai buku dari bahasa Yunani dan Persia ke dalam bahasa Arab. Dengan bahasa Arab pula, para sarjana Islam menulis berbagai karya dalam bidang kedokteran, teknik, matematika, sains, dan berbagai bidang ilmu yang lain. Bahasa Arab merupakan pengantar ilmu-ilmu tadi ke wilayah Eropa sehingga menjadi pondasi peradaban Eropa modern. Bahasa Arab oleh PBB akhirnya resmi dijadikan sebagai bahasa internasional ketujuh di dunia dengan penutur lebih dari dua ratus juta orang Arab. Dengan bahasa ini pula, lebih dari satu milyar umat Islam menunaikan ibadah.

Maka dalam perkembangannya, akan terdapat banyak dokumen yang menggunakan bahasa Arab. *Chararacter Recognition* merupakan salah satu area studi dalam bidang pengenalan pola yang bisa menjadi salah satu referensi dalam memecahkan masalah diatas. Salah satu sistem yang menggunakan teknologi *character recognition* adalah *Optical Character Recognition* (OCR). Teknologi OCR digunakan untuk membuat sebuah *softcopy* dokumen dari *hardcopy* dokumen. Hal tersebut mengurangi waktu pengerjaan dibandingkan dengan metode konvensional yang pengerjaannya memakan waktu yang relatif lama dengan mengetik ulang sebuah dokumen dari dokumen aslinya.

Secara umum terdapat dua hal utama yang mempengaruhi proses pengenalan huruf atau pola yaitu mekanisme ekstraksi ciri dan mekanisme klasifikasi atau inferensi. Dalam melakukan proses pengenalan huruf tersebut, perlu

diperhatikan mengenai teknik yang dapat mengenali berbagai jenis huruf dengan ukuran, ketebalan serta bentuk yang berbeda.

Hal itu dapat diselesaikan dengan menerapkan teknik ekstraksi ciri menggunakan algoritma berbasis *Chain Code* serta pencocokkan pola dengan menggunakan algoritma *Learning Vector Quantization* (LVQ). Algoritma *Chain Cod*e akan digunakan untuk membangun vektor ciri yang berisi informasi kode *chain code* pembentuk huruf. Kemudian akan dilakukan mekanisme inferensi dengan menggunakan algoritma LVQ dari nilai *chain code* yang didapat.

1.2 Rumusan Masalah

Dari penjelasan latar belakang yang telah dikemukakan, maka dapat dirumuskan beberapa masalah sebagai berikut :

1. Bagaimana mekanisme ekstraksi ciri dengan menggunakan algoritma *Chain Code* pada huruf Hijaiyah.
2. Bagaimana teknik inferensi dengan algoritma LVQ.

1.3 Tujuan

Tujuan dari penelitian ini adalah mengenali jenis-jenis huruf Hijaiyah dengan ekstraksi ciri menggunakan algoritma *Chain Code* dan teknik inferensi menggunakan algoritma LVQ.

1.4 Batasan Masalah

Adapun batasan masalah dalam penelitian ini yaitu :

1. Huruf Hijaiyah yang diteliti yaitu sebanyak 28 buah (tanpa menggunakan baris (gundul)).
2. Jenis *font* yang dipilih untuk citra huruf pada data latih dan data uji yaitu KFGQPC Uthman Taha Naskh serta ukuran *font* yang digunakan yaitu 120 pt.
3. Menggunakan metode *Chain Code* dalam proses ekstraksi ciri.
4. Menggunakan metode LVQ dalam proses klasifikasi.

1.5 Metodologi

Pada bagian ini dibagai menjadi dua bagian, yaitu Metodologi Penelitian dan Metodologi Pengembangan Sistem.

1.5.1 Metodelogi Penelitian

a. Studi Literatur

Literatur yang digunakan adalah yang terkait dengan algoritma *Chain Code* serta LVQ. Pembelajaran tersebut dilakukan dengan cara mencari referensi dan pengumpulan data yang diperoleh dari buku, jurnal serta artikel sejenis yang telah dilakukan sebelumnya.

b. Analisis dan Perancangan

Pada tahap ini dilakukan analisis terhadap masalah untuk mencari solusi dalam perancangan *prototype* sistem berdasarkan studi literatur yang diperoleh.

c. Pembangunan *Prototype* Sistem

Dari hasil analisis dan perancangan sistem yang telah dilakukan pada tahap sebelumnya, maka dilakukan pembangunan *prototype* sistem.

d. Implementasi dan Pengujian *Prototype* Sistem

Setelah *prototype* sistem selesai dibuat, maka pada tahap selanjutnya dilakukan pengujian-pengujian terhadap *prototype* sistem.

e. Penyusunan Laporan Tugas Akhir

Pada tahap ini dilakukan penulisan laporan dari aplikasi yang dibangun mulai dari perancangan hingga hasil dari pengujian *prototype* sistem.

1.5.2 Metodelogi Pengembangan Sistem

Metode pengembangan sistem yang digunakan dalam membangun implementasi algoritma *Chain Code* dan LVQ pada pengenalan huruf Hijaiyah ini adalah metodologi *Prototype*. Metode ini menyajikan gambaran yang lengkap

tentang sistemnya, pemesan dapat melihat pemodelan sistem dari sisi tampilan maupun teknik prosedural yang akan dibangun. Pada sisi *development* mencoba efisiensi algoritma, interaksi dengan OS dan *user*. Metode ini dapat mengidentifikasi kebutuhan pemakai, analisis sistem akan melakukan studi kelayakan dan studi terhadap kebutuhan pemakai, meliputi model *interface*, teknik prosedural dan teknologi yang akan digunakan. Aktifitas *Prototype* terdiri dari :

1. Mengidentifikasi kebutuhan : analisa terhadap kebutuhan calon *user*.
2. *Quick design* : pembuatan desain global untuk membentuk *software*.
3. *Build prototype* : pembuatan *software prototype* termasuk pengujian.
4. Evaluasi hasil *prototype* : mengevaluasi *prototype* dengan analisis kebutuhan calon pemakai.
5. Menyempurnakan *prototype* sesuai dengan kebutuhan dan desain awal sistem.
6. Melakukan pengembangan terhadap *prototype*.

1.6 Tinjauan Pustaka

Penelitian ini menggunakan pustaka sebagai referensi yaitu :

a. Jurnal pertama adalah tulisan **H. Izakian, S. A. Monadjemi, B. Tork Ladani dan K. Zamanifar** dari **Department of Computer Engineering, Faculty of Engineering, University of Isfahan Iran** pada tahun 2008 dengan judul ***"Multi-Font Farsi/Arabic Isolated Character Recognition Using Chain Codes".*** Pada penelitian ini dibahas mengenai metode pengenalan huruf Arab/Persia dengan teknik *Chain Code*. Seperti pengaplikasian lain dari pengenalan pola, kesulitan dan poin dasar dalam sistem OCR yaitu memilih ekstraksi fitur yang cepat dan kuat. Metode *Chain Code* melengkapi beberapa algoritma sebelumnya dengan menambahkan bagian penting dalam proses ekstraksi fitur dalam mengenali jumlah lubang dan posisi lubang dari setiap karakter huruf Arab.
b. **Tjokorda Agung Budi Wirayuda, Syilvia Vaulin, Retno Novi Dayawati** dari **Fakultas Teknik Informatika Institut Teknologi Telkom Bandung** pada tahun 2009 dengan judul ***"Pengenalan Huruf Komputer Menggunakan Algoritma Berbasis Chain Code dan Algoritma Sequence Alignment"***. Pada

jurnal ini dirancang sebuah sistem pengenalan huruf komputer dengan dua algoritma. Dua algoritma tersebut mempunya fungsi masing-masing, diantaranya algoritma *Chain Code* untuk mekanisme ekstraksi ciri serta algoritma *Sequence Alignment* sebagai mekanisme klasifikasi atau inferensi. Penelitian ini sebagai lanjutan dari penelitian sebelumnya yang menggunakan mekanisme k-NN yang masih terdapat beberapa kekurangan untuk proses klasfikasi/inferensinya.

c. **Akhmad Robit M., Rizky Bangkit S., Akhmad Fikri H., Ach. Dwi Ardian dan Nuri Nikhmawati Anis U.** dengan judul ***"Optical Character Recognition dengan Metode Naïve Bayes".*** Pada penelitian ini dibahas mengenai teknik OCR menggunakan metode *Naïve Bayes* dan *Chain Code*. OCR yang mempunyai karakteristik yang unik dibandingkan dengan *optical recognize* lainnya akan mengidentifikasi huruf untuk kemudian dianalisa oleh metode *Chain Code* dimana biner-biner nanti akan mencocokkan dan menyesuaikan dengan perbedaan karakteristik setiap huruf. Sehingga untuk keakuratan data OCR terbilang cukup akurat. Sedangkan metode *Naïve Bayes* akan digunakan sebagai metode pengklasifikasian. Model ini merupakan model paling sederhana dari model pengklasifikasian dengan peluang, dimana diasumsikan bahwa setiap atribut contoh bersifat saling lepas satu sama lain berdasarkan atribut kelas.

d. **Asep Nana Hermana, Irma Amelia, Andri Pramana** dengan judul ***"Verifikasi Tanda Tangan dengan Metode Edge Detection dan Learning Vector Quantization".*** Pada penelitian ini, dibahas mengenai pencocokan tanda tangan dengan metode LVQ. Metode Jaringan Syaraf Tiruan (JST) LVQ digunakan untuk mencari kecocokan pola tanda tangan berdasarkan nilai bobot terkecil.

Perbandingan dari tinjauan pustaka diatas, dapat dilihat pada Tabel 1.

Tabel 1. Perbandingan Tinjauan Pustaka

Judul	Nama	Metode	Keterangan
Multi-Font Farsi/Arabic Isolated Character Recognition Using Chain Codes (2008)	H. Izakian, S. A. Monadjemi, B. Tork Ladani dan K. Zamanifar	*Chain Codes*	Dari hasil penelitian yang dilakukan, metode *Chain Code* diperkenalkan dalam mengidentifikasi karakter huruf Arab. Hasil percobaan menggunakan *font* Farsi standar menunjukkan bahwa akurasi metode identifikasi karakter yang diusulkan rata-rata mencapai 97.4 % yang benar.
Pengenalan Huruf Komputer Menggunakan Algoritma Berbasis *Chain Code* dan Algoritma *Sequence Alignment* (2009)	Tjokorda Agung Budi Wirayuda, Syilvia Vaulin, Retno Novi Dayawati	*Chain Code* dan *Sequence Alignment*	Dari hasil penelitian menunjukkan bahwa penerapan mekanisme *Sequence Alignment* dapat mengatasi masalah perbedaan titik awal proses *chain code*. Hal tsb. ditunjukkan oleh hasil akurasi pengenalan yang baik dalam mengenali huruf asing yang tidak menjadi basis pengetahuan.
Optical Character Recognition dengan Metode *Naïve Bayes*	Akhmad Robit M., Rizky Bangkit S., Akhmad Fikri H., Ach. Dwi Ardian dan Nuri Nikhmawati Anis U.	*Naïve Bayes* dan *Chain Code*	Dari hasil penelitian yang dilakukan, terlihat bahwa penerapan mekanisme metode *chain code* dan *naïve bayes* mampu melakukan ekstraksi fitur yang tepat. Aplikasi ini memiliki keakuratan yang baik, terlihat pada hasil *data train* yang akurasinya mencapai 75% dan data uji sebesar 85 %.
Verifikasi Tanda Tangan dengan Metode *Edge*	Asep Nana Hermana, Irma	*Edge Detection* dan	Penelitian yang dilakukan, difokuskan untuk mencari kecocokan tanda tangan

Tabel 1. Perbandingan Tinjauan Pustaka (Lanjutan)

Judul	Nama	Metode	Keterangan
Detection dan *Learning Vector Quantization* (2015)	Amelia, Andri Pramana	*Learning Vector Quantization*	asli serta palsu dengan menggunakan metode LVQ. Hasil pencocokan didapat persentase 70% tanda tangan asli teridentifikasi dengan benar.
Implementasi Algoritma *Chain Code* dan LVQ Pada Pengenalan Huruf Hijaiyah (2015)	Maulana Nur Muhamad, Jasman Pardede, Uung Ungkawa	*Chain Code* dan *Sequence Alignment*	Penelitian difokuskan pada pengenalan huruf Hijaiyah menggunakan algoritma *Chain Code* sebagai ektraksi ciri dan algoritma *LVQ* sebagai teknik inferensinya. Teknik *Chain Code* dapat digunakan untuk menemukan struktur pembentuk dari suatu objek. Hasil dari pengujian sistem diperoleh prosentase keberhasilan sebesar 78,5% dari 28 citra huruf Hijaiyah yang diuji

1.7 Sistematika Penulisan

Untuk memberikan kemudahan bagi penulis dalam menyusun laporan Tugas Akhir ini, maka sistematika penulisan laporan ini adalah sebagai berikut :

BAB I PENDAHULUAN

Bab ini berisikan latar belakang masalah, rumusan masalah, batasan masalah, tujuan, batasan masalah, tinjauan pustaka, metodologi penelitian, dan sistematika penulisan.

BAB II DASAR TEORI

Dalam bab ini akan diuraikan mengenai pengertian dari istilah-istilah maupun pengetahuan-pengetahuan yang digunakan sebagai penjelasan lebih dalam dari tinjauan pustaka yang ada.

BAB III ANALISIS DAN PERANCANGAN

Bab ini berisikan penjelasan mengenai analisis dan perancangan dari sistem.

BAB IV IMPLEMENTASI DAN PENGUJIAN

Bab ini berisikan mengenai implementasi dari sistem yang telah diciptakan dan hasil pengujian terhadap fungsionalitas sistem.

BAB V PENUTUP

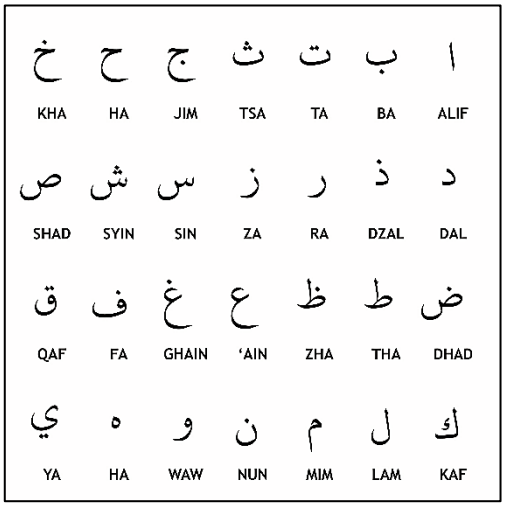
Dalam bab ini akan diuraikan mengenai kesimpulan dari hasil pengujian sistem.

BAB II
DASAR TEORI

Pada bagian ini, pembahasan yang dijelaskan mengenai teori dasar yang berkaitan dengan penelitian yang akan dilakukan. Diantaranya mengenai huruf Hijaiyah, pengenalan pola, *Optical Character Recognition*, algoritma *Chain Code*, algoritma LVQ, ekstraksi ciri huruf hijaiyah serta contoh studi kasus untuk penelitian ini.

2.1 Huruf Hijaiyah [1]

Huruf Hijaiyah merupakan huruf penyusun kata dalam Al Qur'an. Seperti halnya di Indonesia yang memiliki huruf alfabet dalam menyusun sebuah kata menjadi kalimat, huruf hijaiyah juga memiliki peran yang sama. Secara umum, huruf hijaiyah berjumlah 28 huruf (Gambar 1).



Gambar 1. Huruf Hijaiyah

2.2 Pengenalan Pola [10]

Pengenalan pola dapat dikatakan sebagai kemampuan mengenali objek berdasarkan ciri-ciri dan pengetahuan yang pernah diamatinya dari objek-objek tersebut. Tujuan dari pengenalan pola adalah mengklasifikasi dan mendeskripsikan pola atau objek kompleks melalui pengetahuan sifat-sifat atau ciri-ciri objek tersebut.

Ada tiga pendekatan dalam pengenalan pola yaitu secara sintaks, statistik, dan semantik. Pengenalan pola secara sintaks dilakukan berdasarkan ciri-ciri objek. Pengenalan pola secara statistik dilakukan berdasarkan komputasi matematis. Pendekatan dengan semantik berarti pola dikenali dalam tataran yang lebih abstrak.

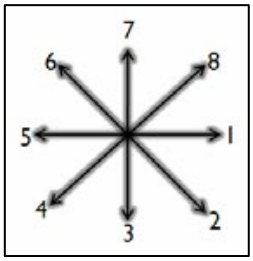
2.3 *Optical Character Recognition* [7]

OCR adalah sebuah sistem komputer yang dapat membaca huruf, baik yang berasal dari sebuah pencetak (*printer* atau mesin ketik) maupun yang berasal dari tulisan tangan. Adanya sistem pengenal huruf ini akan meningkatkan fleksibilitas ataupun kemampuan dan kecerdasan sistem komputer. Dengan adanya sebuah sistem OCR tersebut akan mempermudah seseorang untuk mengetahui keakuratan data apabila terjadi sebuah problematika atau masalah yang timbul diakibatkan oleh ketidak validan sebuah susunan kalimat yang terdiri dari beberapa huruf.

OCR sendiri biasa digunakan sebagai alat bantu yang praktis dalam kehidupan sehari-hari, OCR juga memiliki karakteristik yang unik dibandingkan dengan *optical recognize* lainya, tidak hanya dapat mengenali tulisan tangan akan tetapi juga dapat mempermudah pengguna atau *user* untuk memasukan data tidak harus dengan papan ketik melainkan dengan pena elektronik.

2.4 Algoritma *Chain Code* [3]

Chain Code adalah metode yang melakukan penelusuran piksel- piksel objek dengan panduan arah mata angin[2], seperti yang ditunjukkan pada `Gambar 2`.



`Gambar 2. Arah Mata Angin Sebagai Panduan`

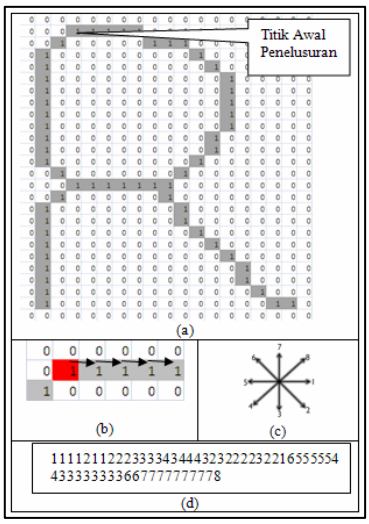
Chain Code juga merupakan contoh representasi kontur yang mula-mula diperkenalkan oleh Freeman pada tahun 1961. Representasi bentuk dilakukan dengan menggunakan pendekatan piksel 8-ketetanggaan[5].

Dengan mekanisme melakukan penelusuran per-piksel, teknik *Chain Code* dapat digunakan untuk menemukan struktur pembentuk dari suatu objek. Untuk memperoleh *Chain Code*, pencarian difokuskan pada bagian utama (tubuh) dari karakter suatu gambar. Dari atas gambar, pencarian dilakukan baris per baris menuju ke bawah dan mempertimbangkan piksel pertama dari gambar tersebut yang telah memiliki satu tetangga, sebagai titik awal dari *Chain Code*. Jika karakter tidak memiliki titik awal, nilai pencarian akan dianggap sebagai nol (misalnya karakter 'ه').[4]

Setelah menemukan titik permulaan *Chain Code* pada karakter suatu citra, pencarian akan bergerak untuk piksel berikutnya yang berdekatan yang juga menjadi bagian dari tubuh pembentuk huruf tersebut. Jika dalam kasus memiliki dua atau lebih piksel tetangga dengan kondisi di atas maka pencarian menggunakan prioritas arah seperti yang ditunjukkan pada Gambar 2.

Setelah mendapat *Chain Code* untuk semua karakter, *Chain Code* untuk karakter yang berbeda memiliki panjang yang berbeda dan panjang masing-masing *Chain Code* tergantung pada ukuran dari karakter yang diinginkan. Biasanya panjang suatu *Chain Code* akan panjang; oleh karena itu perlunya normalisasi *Chain Code* yang panjang ini menjadi nilai yang tetap dan terbatas (misalnya 10 angka).[4]

Ilustrasi proses *Chain Code* dapat dilihat pada Gambar 3.



Gambar 3. (a)Huruf R Dalam Bentuk Biner(b)Arah Penelsuran Awal(c)Acuan Arah Mata Angin(d)Hasil Chain Code untuk Huruf R

Hasil akhir dari proses ekstraksi ciri berbasis *Chain Code* yang dilakukan adalah sebuah *vector* ciri yang berisi informasi urutan kode *Chain Code* pembentuk huruf. Dari mekanisme yang dilakukan oleh *Chain Code* maka urutan *chain code* yang dihasilkan untuk setiap huruf dapat memiliki panjang yang berbeda, sehingga diperlukan sebuah mekanisme normalisasi untuk menyamakan panjang *Chain Code* agar dapat digunakan sebagai masukan pada proses klasifikasi.

Langkah-langkah normalisasi yang dilakukan yaitu mengubah *Chain Code* menjadi matriks berisi nilai dan frekuensinya, menghapus nilai dengan frekuensi 1, menerapkan rumus normalisasi untuk mencari frekuensi sesuai yang diinginkan, dan membangun ulang *Chain Code* sesuai frekuensi yang baru. Nilai frekuensi baru kode ke-i didapatkan dengan rumus normalisasi yang diterapkan pada Rumus 1.

$$F_i^n = \frac{F_i}{\sum F_i} \times N \quad \text{..... (1)}$$

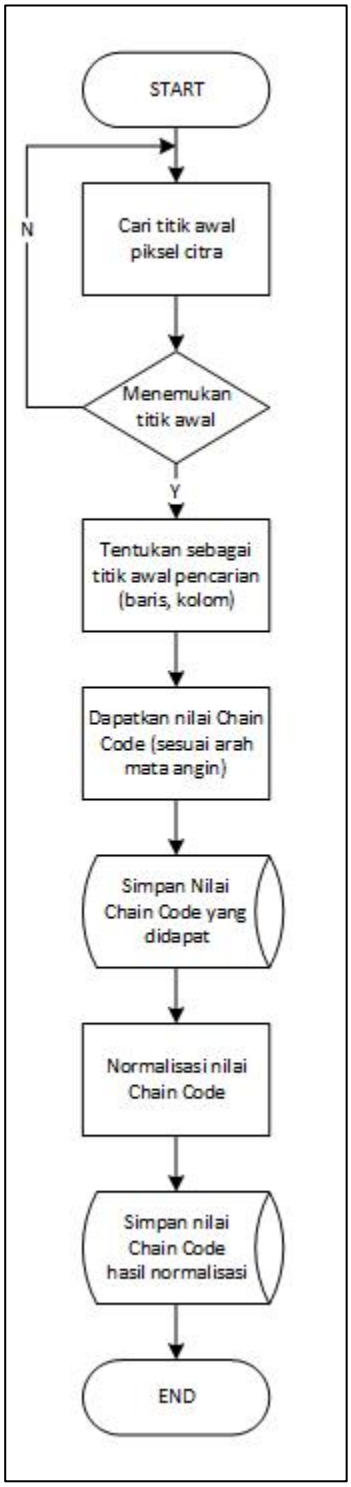
dimana :

Fi adalah frekuensi kode ke-i

$\sum Fi$ adalah total frekuensi semua kode

N adalah nilai frekuensi yang diinginkan

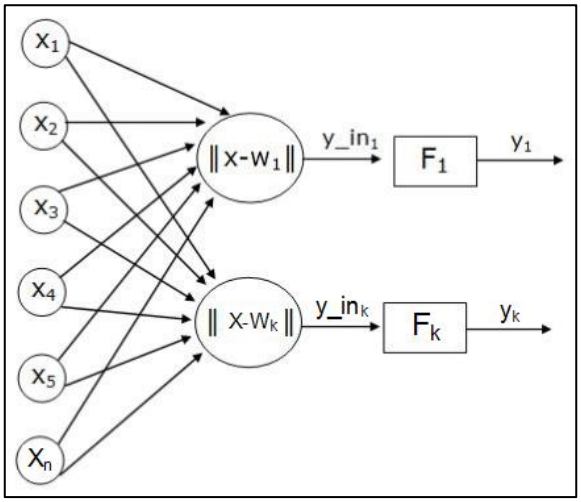
Alur kerja dari *Chain Code* dapat dilihat pada Gambar 4.



Gambar 4. Alur kerja Chain Code secara umum

2.5 *Learning Vector Quantization* [6]

Learning Vector Quantization (LVQ) adalah suatu metode untuk melakukan pembelajaran pada lapisan kompetitif yang terawasi. Suatu lapisan kompetitif akan secara otomatis belajar untuk mengklasifikasi vektor-vektor *input*. Kelas-kelas yang didapatkan sebagai hasil dari lapisan kompetitif ini hanya tergantung pada jarak antara vektor-vektor *input*. Jika dua vektor *input* mendekati sama, maka lapisan kompetitif akan meletakkan kedua vektor *input* tersebut ke dalam kelas yang sama. Contoh arsitektur LVQ seperti terlihat pada Gambar 5.



Gambar 5. Arsitektur LVQ

Pada Gambar 5, notasi X_1 ... X_n menunjukkan sebagai nilai vektor *input*, sedangkan notasi X - W_1 ... X - W_k menunjukkan sebagai perhitungan nilai jarak dari nilai vektor *input* dan vektor bobot. Penghitungan jarak *Euclidean* tersebut dilakukan dengan menggunakan Rumus 2.

$$C_j = \sqrt{\sum_{i=1}^{n} (x_i - w_j)^2} \quad \text{.....(2)}$$

dimana :

C_j adalah nilai jarak *Euclidean*

x_i adalah nilai vektor *input*

w_j adalah nilai vektor bobot

Selain itu LVQ selalu memperbaharui bobotnya sesuai jumlah maksimal *epoch* yang ditetapkan. *Epoch* yaitu satu siklus pelatihan yang melibatkan semua pola. Rumus 3 digunakan untuk memperbaharui nilai bobot.

$$\vec{W_m} \leftarrow \vec{W_m} + \alpha \cdot (\vec{X} - \vec{W_m}) \quad \text{..... (3)}$$

dimana :

$\overrightarrow{W_m}$ adalah bobot

α adalah *learning rate*

$\overrightarrow{X}$ adalah *input*

Secara garis besar, LVQ akan mencari unit keluaran yang paling mirip dengan vektor masukan. Jika vektor pelatihan adalah bagian dari kelas yang sama, maka vektor bobot digeser mendekati vektor masukan tersebut. Sebaliknya jika vektor pelatihan bukan bagian dari kelas yang sama, maka vektor bobot digeser menjauhi vektor masukan tersebut.

2.6 Ekstraksi Ciri Huruf Hijaiyah [4]

Pada bagian ini akan dibahas mengenai jumlah dan lokasi bagian huruf Hijaiyah, serta analisis lubang pada huruf Hijaiyah yang akan didefinisikan dalam sebuah formula.

2.6.1 Jumlah dan Lokasi Bagian Huruf

Karakter bagian huruf (objek terbesar yang tersaji dalam gambar) akan dipertimbangkan sebagai objek utama dalam gambar dan objek lain yang ada akan diasumsikan sebagai bagian pelengkap. Fitur ini akan dipertimbangakan menjadi tiga urutan, F: (P1, P2, P3). Langkah ini dimulai dari bagian atas gambar, dilanjutkan baris per baris dan menetapkan P1 sebagai objek pertama pada gambar. P2 dan P3 juga ditentukan untuk mengalokasikan objek pada bagian terbawah karakter suatu gambar. Setiap nilai P1, P2 dan P3 mempunyai nilai integer antara 0 sampai 3.

Jika objek yang dimaksud merupakan bagian karakter utama, makan nilai P*i* akan bernilai 3. Jika objek tersebut mempunyai dua titik (contohnya seperti pada huruf ق), maka nilai P*i* adalah 2. Sedangkan jika objek selanjutnya memiliki satu titik, nilai P*i* adalah 1. Contohnya seperti terlihat pada Gambar 6.

پ	ث	ب	ف	ت
3 2 1	1 2 3	3 1 0	1 3 0	2 3 0
چ	گ	س	ر	ن
3 2 1	2 3 0	3 0 0	3 0 0	1 3 0

Gambar 6. Tiga Kode Khusus pada Karakter Huruf Persia

2.6.2 Analisis Lubang pada Karakter Huruf Hijaiyah dan Nilai *Chain Code*

Selanjutnya yaitu menentukan jumlah lubang pada setiap karakter huruf Hijaiyah. Setiap karakter huruf mempunyai maksimal satu lubang pada setiap hurufnya. Contohnya untuk huruf *ba, ra, dal* yang tidak mempunyai lubang. Sedangkan untuk karakter seperti huruf *fa* dan *sha* mempunyai satu lubang. Dari hasil analisis dua fitur diatas, maka vektor fitur masing-masing karakter yaitu sebagai berikut :

$$F = [P1, P2, P3, Holes, Chain\ Code]$$

Maka dengan menggunakan persamaan vektor fitur diatas, karakter huruf Hijaiyah dapat dibedakan menjadi beberapa jenis, seperti terlihat pada Tabel 2.

Tabel 2. Karakter Huruf Hijaiyah

No.	F=(P1,P2,P3,Holes, Chain Code)	Huruf	No.	F=(P1,P2,P3,Holes, Chain Code)	Huruf
1	F=(3,0,0,0,3333333333)	ا	15	F=(1,3,0,1,1235534567)	ض
2	F=(3,1,0,0,2234555577)	ب	16	F=(3,0,0,1,3332114557)	ط
3	F=(2,3,0,0,1123455557)	ت	17	F=(3,1,0,1,3332114557)	ظ
4	F=(1,2,3,0,1112345557)	ث	18	F=(3,0,0,0,5431843321)	ع
5	F=(3,1,0,0,1111154332)	ج	19	F=(1,3,0,0,2431843321)	غ
6	F=(3,0,0,0,1111445332)	ح	20	F=(1,3,0,1,1234555577)	ف
7	F=(1,3,0,0,3555143321)	خ	21	F=(2,3,0,1,1123445577)	ق
8	F=(3,0,0,0,2223333555)	د	22	F=(3,0,0,0,4532242222)	ك
9	F=(1,3,0,0,2222333555)	ذ	23	F=(3,0,0,0,3333334567)	ل
10	F=(3,0,0,0,3333444555)	ر	24	F=(3,0,0,1,2224533333)	م
11	F=(1,3,0,0,3322334455)	ز	25	F=(1,3,0,0,3333245577)	ن
12	F=(3,0,0,0,3574345577)	س	26	F=(3,0,0,1,2333445577)	و
13	F=(1,2,3,0,3574345577)	ش	27	F=(3,0,0,1,1115555777)	ه
14	F=(3,0,0,1,1235534567)	ص	28	F=(3,2,0,0,5443156777)	ي

2.7 Contoh Studi Kasus

Pembahasan studi kasus ini meliputi contoh studi kasus *Chain Code* dan contoh studi kasus LVQ.

2.7.1 Studi Kasus *Chain Code*

Pada contoh studi kasus ini, yang akan dibahas adalah proses ekstraksi ciri menggunakan algoritma *Chain Code* untuk mendapatkan nilai huruf pada data latih yang akan disimpan pada basis pengetahuan. Huruf yang akan dijadikan contoh yaitu huruf Ha (ح). Berikut merupakan langkah-langkah pengerjaannya :

- Langkah pertama yaitu menetapkan huruf yang akan di ektraksi. Huruf yang akan dicari nilai *Chain Code* nya yaitu seperti pada Gambar 7.

Gambar 7. Huruf Ha yang diekstraksi

- Setelah itu, lakukanlah langkah *Preprocessing*. Pada tahap ini, huruf akan melalui proses *thinning* (penipisan huruf).
- Setelah itu huruf ditelusuri piksel per piksel (Gambar 8) dalam bentuk biner, dimulai dari bagian paling atas sampai bagian paling bawah.

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	1	0	0
0	0	0	0	0	1	1	1	0	0
0	0	0	0	1	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0
0	0	0	0	0	1	1	1	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

Gambar 8. Huruf Ha dalam bentuk biner

Proses penelususan tersebut menggunakan arah mata angin sebagai panduan seperti pada Gambar 2. Hasil akhir dari proses ektrasi ciri berbasis *Chain Code* adalah sebuah vektor ciri yang berisi informasi urutan kode *Chain Code* pembentuk huruf[3]. Maka didapatlah nilai ektrasi ciri tersebut seperti di bawah ini :

11111114554443333322211 1

- Lalu nilai tersebut dikonversi menjadi sebuah matriks berukuran 2 x n, dimana pada baris pertama berisi nilai dari *Chain Code* dan baris kedua merupakan nilai frekuensi dari *Chain Code* tersebut. Nilai *n* merupakan banyaknya jumlah nilai *Chain Code* yang dimaksud[4]. Pada kasus ini, matriks yang terbentuk yaitu matriks berukuran 2 x 7 :

Nilai *Chain Code*	:	1	4	5	4	3	2	1
Jumlah Frekuensi	:	7	1	2	3	5	3	3

- Setelah memindahkan nilai *Chain Code* ke dalam matriks tersebut, langkah selanjutnya yaitu menghilangkan nilai dengan jumlah frekuensi 1 dan melakukan penggabungan nilai frekuensi jika terdapat nilai yang sama[4]. Maka matriks *Chain Code* yang baru akan terbentuk seperti di bawah ini :

Nilai *Chain Code*	:	1	4	5	3	2
Jumlah Frekuensi	:	10	4	2	5	3

- Setelah membangun ulang *Chain Code* sesuai frekuensi yang baru, maka langkah selanjutnya yaitu melakukan normalisasi nilai frekuensi baru kode ke-i dengan menerapkan `Rumus 2` yang sudah disebutkan pada bagian sebelumnya. $F_i^n = \frac{F_i}{\sum F_i} \times N$ dimana F_i adalah frekuensi kode ke-i, $\sum F_i$ adalah total frekuensi semua kode dan N adalah nilai frekuensi yang diinginkan. Berikut merupakan perthitungannya :

$$F = \frac{10}{24} \times 10 = \mathbf{4,17} \; ; F = \frac{4}{24} \times 10 = \mathbf{1,67} \; ; F = \frac{2}{24} \times 10 = \mathbf{0,83}$$

$$F = \frac{5}{24} \times 10 = \mathbf{2,08} \; ; F = \frac{3}{24} \times 10 = \mathbf{1,25}$$

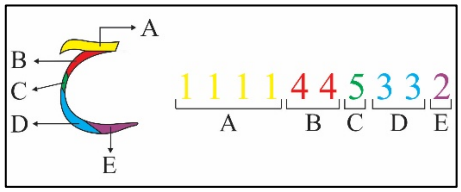
Maka akan di dapat hasil sebagai berikut :

Nilai *Chain Code*	:	1	4	5	3	2
Jumlah Frekuensi	:	4,17	1,67	0.83	2,08	1,25

- Dengan menggunakan aturan pembulatan angka penting[12], nilai jumlah frekuensi yang didapat akan dibulatkan sesuai dengan aturan yang ada. Maka hasil normalisasi frekuensi dengan panjang ukuran normalisasi 10 yaitu sebagai berikut:

1111445332

- Pada kasus lain, jika panjang dari *Chain Code* kurang dari 10, maka nilai pertama atau terakhir dapat diulang[4]. `Gambar 9` menunjukkan karakter huruf Ha (ح) setelah dilakukan ekstraksi ciri.

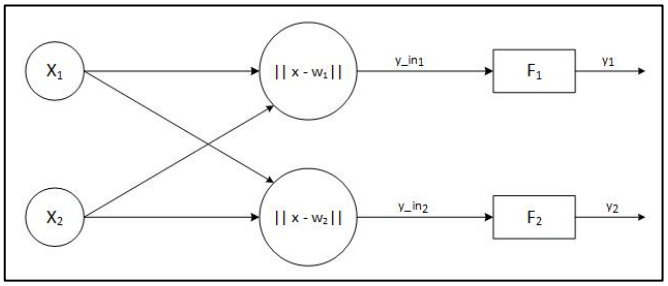


Gambar 9. Nilai Chain Code untuk huruf ha

2.7.2 Studi Kasus LVQ

Pada studi kasus algoritma LVQ ini, yang dibahas yaitu proses penghitungan LVQ menggunakan citra huruf dengan ukuran yang berbeda (120 pt dan 100 pt). Nilai vektor masukan didapat dari nilai *Chain Code* pembentuk huruf.

Untuk studi kasus ini, arsitektur LVQ yang digunakan seperti terlihat pada Gambar 10.



Gambar 10. Arsitektur LVQ untuk studi kasus

Dari Gambar 10, X_1 dan X_2 menunjukkan nilai masukan berupa nilai *Chain Code*. Nilai tersebut kemudian dilakukan penghitungan jarak menggunakan Rumus 2 (Rumus *Euclidean Distance*) terhadap masing-masing nilai bobot. Kemudian akan didapatkan nilai bobot terkecil (F_1 dan F_2) lalu dihitung sebagai nilai bobot baru (y_1 dan y_2). Berikut merupakan langkah pengerjaannya.

Diasumsikan bahwa nilai vektor *Chain Code* dari masing-masing citra huruf sudah ditemukan (Tabel 3).

Tabel 3. Nilai *Chain Code* untuk contoh citra huruf uji

No.	Nama Huruf	Bentuk Huruf	Ukuran Huruf	Nilai *Chain Code*
1	Ha	ح	120 pt.	1 1 1 1 4 4 5 3 3 2
2	Ha	ح	100 pt.	1 1 1 1 1 4 4 3 3 2
3	Lam	ل	120 pt.	3 3 3 3 3 3 4 5 6 7
4	Lam	ل	100 pt.	3 3 3 3 3 4 5 7 7 7

Langkah 1 : Kelompokkan vektor Chain Code ke dalam bentuk kelas

Masing-masing citra huruf pada Tabel 3 dikelompokkan menjadi dua kelas sesuai dengan bentuk hurufnya. Pada setiap nilai vektor, tahap awal yaitu dengan menginisialisasikan kelas untuk mengklasifikasikan setiap masukan (Tabel 4).

Tabel 4. Nilai *Chain Code* untuk input vektor LVQ

No.	Vektor *Chain Code*	Kelas
1	1 1 1 1 4 4 5 3 3 2	1
2	1 1 1 1 1 4 4 3 3 2	1
3	3 3 3 3 3 3 4 5 6 7	2
4	3 3 3 3 3 4 5 7 7 7	2

Langkah 2 : Tentukan nilai vektor yang akan dijadikan bobot (W)

Masukan pertama dari masing-masing kelas akan dijadikan sebagai inisialisasi bobot (Tabel 5).

Tabel 5. Inisialisasi vektor bobot LVQ

No.	Vektor Bobot (W)	Vektor input ke-	Kelas
1	1, 1, 1, 1, 4, 4, 5, 3, 3, 2	1	1
2	3, 3, 3, 3, 3, 3, 4, 5, 6, 7	3	2

Langkah 3 : Tentukan nilai vektor yang akan dijadikan input (X)

Dua masukan sisanya akan dijadikan sebagai data yang akan dilatih (Tabel 6).

Tabel 6. Inisialisasi vektor latih LVQ

No.	Vektor (X)	Vektor input ke-	Kelas
1	1, 1, 1, 1, 1, 4, 4, 3, 3, 2	2	1
2	3, 3, 3, 3, 3, 4, 5, 7, 7, 7	4	2

Langkah 4 : Tentukan learning rate serta maksimum epoch

Sebagai nilai awal dipilih *learning rate* ($\alpha = 0.01$) dan maksimum *epoch* (*MaxEpoch* = 1).

Langkah 5 : Hitung nilai jarak dari nilai vektor input ke-2 (data ke-1) terhadap masing-masing bobot

- **Data Ke-1 → (1, 1, 1, 1, 1, 4, 4, 3, 3, 2)**
  - ✓ Jarak pada bobot ke-1 =

    $= \sqrt{(1-1)^2 + (1-1)^2 + (1-1)^2 + (1-1)^2 + (1-4)^2 + (4-4)^2 + (4-5)^2 + (3-3)^2 + +(3-3)^2 + (2-2)^2}$

    $= \sqrt{(0)^2 + (0)^2 + (0)^2 + (0)^2 + (-3)^2 + (0)^2 + (-1)^2 + (0)^2 + +(0)^2 + (0)^2} = 10$
  - ✓ Jarak pada bobot ke-2 =

    $= \sqrt{(1-3)^2 + (1-3)^2 + (1-3)^2 + (1-3)^2 + (1-3)^2 + (4-3)^2 + (4-4)^2 + (3-5)^2 + +(3-6)^2 + (2-6)^2}$

    $= \sqrt{(-2)^2 + (-2)^2 + (-2)^2 + (-2)^2 + (-2)^2 + (1)^2 + (0)^2 + (-2)^2 + +(-3)^2 + (-4)^2} = 50$

  Dari hasil perhitungan jarak diatas, jarak terkecil terdapat dalam perhitungan jarak pada bobot ke-1 dengan nilai 10.

Langkah 6 : Hitung nilai bobot baru untuk bobot kelas 1

Karena target data ke-1 adalah kelas 1, maka bobot (w) baru untuk kelas 1 adalah :

$$W_{j\,(baru)} = W_{j\,(lama)} + \alpha\left(X_i - W_{j\,(lama)}\right)$$

$$= 1 + 0.01 * (1 - 1) = 1$$
$$= 1 + 0.01 * (1 - 1) = 1$$
$$= 1 + 0.01 * (1 - 1) = 1$$
$$= 1 + 0.01 * (1 - 1) = 1$$
$$= 4 + 0.01 * (1 - 4) = 3.97$$
$$= 4 + 0.01 * (4 - 4) = 4$$
$$= 5 + 0.01 * (4 - 5) = 4.99$$
$$= 3 + 0.01 * (3 - 3) = 3$$
$$= 3 + 0.01 * (3 - 3) = 3$$
$$= 2 + 0.01 * (2 - 2) = 2$$

$\therefore$ $W_{j\,(baru)}$ adalah $(1, 1, 1, 1, 3.97, 4, 4.99, 3, 3, 2)$

Langkah 7 : Hitung nilai jarak dari nilai vektor input ke-4 (data ke-2) terhadap masing-masing bobot

- **Data Ke-2 → (3, 3, 3, 3, 3, 4, 5, 7, 7, 7)**
  - ✓ Jarak pada Bobot ke-1 =

    $= \sqrt{(3-1)^2 + (3-1)^2 + (3-1)^2 + (3-1)^2 + (3-3.97)^2 + (4-4)^2 + (5-4.99)^2 + (7-3)^2 + +(7-3)^2 + (7-2)^2}$

    $= \sqrt{(2)^2 + (2)^2 + (2)^2 + (2)^2 + (0.97)^2 + (0)^2 + (0.01)^2 + (4)^2 + +(4)^2 + (5)^2} = 73.941$
  - ✓ Jarak pada bobot ke-2 =

    $= \sqrt{(3-3)^2 + (3-3)^2 + (3-3)^2 + (3-3)^2 + (3-3)^2 + (4-3)^2 + (5-4)^2 + (7-5)^2 + +(7-6)^2 + (7-7)^2}$

    $= \sqrt{(0)^2 + (0)^2 + (0)^2 + (0)^2 + (0)^2 + (1)^2 + (1)^2 + (2)^2 + +(1)^2 + (0)^2} = 7$

  Dari hasil perhitungan jarak diatas, jarak terkecil terdapat dalam perhitungan jarak pada bobot ke-2 dengan nilai 7.

Langkah 8 : Hitung nilai bobot baru untuk bobot kelas 2

Karena target data ke-2 adalah kelas 2, maka bobot (w) baru untuk kelas 2 adalah :

$$W_{j\,(baru)} = W_{j\,(lama)} + \alpha\left(X_i - W_{j\,(lama)}\right)$$

$$= 3 + 0.01 * (3 - 3) = 3$$
$$= 3 + 0.01 * (3 - 3) = 3$$
$$= 3 + 0.01 * (3 - 3) = 3$$
$$= 3 + 0.01 * (3 - 3) = 3$$
$$= 3 + 0.01 * (3 - 3) = 3$$
$$= 3 + 0.01 * (4 - 3) = 3.01$$
$$= 4 + 0.01 * (5 - 4) = 4.01$$
$$= 5 + 0.01 * (7 - 5) = 5.02$$
$$= 6 + 0.01 * (7 - 6) = 6.01$$
$$= 7 + 0.01 * (7 - 7) = 7$$

$\therefore$ $W_{j\,(baru)}$ adalah $(3, 3, 3, 3, 3, 3.01, 4.01, 5.02, 6.01, 7)$

Langkah 9 : Menyimpan nilai bobot baru (W_1 dan W_2)

Maka, diperoleh bobot akhir yaitu :

- $W_1 = (1, 1, 1, 1, 3.97, 4, 4.99, 3, 3, 2)$
- $W_2 = (3, 3, 3, 3, 3, 3.01, 4.01, 5.02, 6.01, 7)$

Contoh proses pencocokan dengan citra data uji

Dimisalkan ada sebuah citra uji dari huruf Lam (ل) dengan vektor nilai *Chain Code* sebagai berikut :

3 3 3 3 3 4 5 7 7 7

Maka dengan nilai bobot yang sudah dimasukkan ke dalam proses data latih sebelumnya, berikut merupakan perhitungannya (hasil dari bobot dengan jarak terpendek akan menjadi kelas dari citra huruf tersebut) :

- ✓ Jarak pada bobot ke-1 =

$$= \sqrt{(3-1)^2 + (3-1)^2 + (3-1)^2 + (3-1)^2 + (3-3.97)^2 + (4-4)^2 + (5-4.99)^2 + (7-3)^2 + (7-3)^2 + (7-2)^2}$$

$$= \sqrt{(2)^2 + (2)^2 + (2)^2 + (2)^2 + (0.97)^2 + (0)^2 + (0.01)^2 + (4)^2 + +(4)^2 + (5)^2} = 73.941$$

- ✓ Jarak pada bobot ke-2 =

$$= \sqrt{(3-3)^2 + (3-3)^2 + (3-3)^2 + (3-3)^2 + (3-3)^2 + (4-3.01)^2 + (5-4.01)^2 + (7-5.02)^2 + (7-6.01)^2 + (7-7)^2}$$

$$= \sqrt{(0)^2 + (0)^2 + (0)^2 + (0)^2 + (0)^2 + (0.99)^2 + (0.99)^2 + (1.98)^2 + +(0.99)^2 + (0)^2} = 6.8607$$

Dari hasil perhitungan jarak diatas, nilai jarak paling kecil yaitu nilai jarak pada bobot ke-2 yaitu dengan nilai 6.8607. Maka dapat disimpulkan bahwa masukan vektor nilai *Chain Code* yang diuji tersebut termasuk ke dalam Kelas 2 (Huruf Lam).

Contoh proses pencocokan satu citra data uji terhadap 28 citra latih

Dimisalkan terdapat sebuah citra uji dari huruf Lam (ل) dengan vektor nilai *Chain Code* sebagai berikut :

3 3 3 3 3 4 5 7 7 7

Tabel 7 menunjukan nilai vektor bobot dari 28 citra huruf pada data latih.

Tabel 7. Inisialisasi Vektor Bobot

No.	Vektor Bobot	Huruf	Kelas
1	3 3 3 3 3 3 3 3 3 3	ا	1
2	2 2 3 4 5 5 5 5 7 7	ب	2
3	1 1 2 3 4 5 5 5 5 7	ت	3
4	1 1 1 2 3 4 5 5 5 7	ث	4
5	1 1 1 1 1 5 4 3 3 2	ج	5

Tabel 7. Inisialisasi Vektor Bobot(Lanjutan)

No.	Vektor Bobot	Huruf	Kelas
6	1 1 1 1 4 4 5 3 3 2	ح	6
7	3 5 5 5 1 4 3 3 2 1	خ	7
8	2 2 2 3 3 3 3 5 5 5	د	8
9	2 2 2 2 3 3 3 5 5 5	ذ	9
10	3 3 3 3 4 4 4 5 5 5	ر	10
11	3 3 2 2 3 3 4 4 5 5	ز	11
12	3 5 7 4 3 4 5 5 7 7	س	12
13	3 5 7 4 3 4 5 5 7 7	ش	13
14	1 2 3 5 5 3 4 5 6 7	ص	14
15	1 2 3 5 5 3 4 5 6 7	ض	15
16	3 3 3 2 1 1 4 5 5 7	ط	16
17	3 3 3 2 1 1 4 5 5 7	ظ	17
18	5 4 3 1 8 4 3 3 2 1	ع	18
19	2 4 3 1 8 4 3 3 2 1	غ	19
20	1 2 3 4 5 5 5 5 7 7	ف	20
21	1 1 2 3 4 4 5 5 7 7	ق	21
22	4 5 3 2 2 4 2 2 2 2	ك	22
23	3 3 3 3 3 4 5 7 7 7	ل	23
24	2 2 2 4 5 3 3 3 3 3	م	24
25	3 3 3 3 2 4 5 5 7 7	ن	25
26	2 3 3 3 4 4 5 5 7 7	و	26
27	1 1 1 5 5 5 5 7 7 7	ه	27
28	5 4 4 3 1 5 6 7 7 7	ي	28

Berikut merupakan perhitungan untuk mencari nilai terdekat dengan bobot untuk dijadikan kelas dari citra huruf tersebut :

- ✓ Jarak pada bobot ke-1 =

$$= \sqrt{(3-3)^2 + (3-3)^2 + (3-3)^2 + (3-3)^2 + (3-3)^2 + (4-3)^2 + (5-3)^2 + (7-3)^2 + (7-3)^2 + (7-3)^2}$$

$$= \sqrt{(0)^2 + (0)^2 + (0)^2 + (0)^2 + (0)^2 + (1)^2 + (2)^2 + (4)^2 + (4)^2 + (4)^2} = \sqrt{53} = 7.280$$

- ✓ Jarak pada bobot ke-2 =

$$= \sqrt{(3-2)^2 + (3-2)^2 + (3-3)^2 + (3-4)^2 + (3-5)^2 + (4-5)^2 + (5-5)^2 + (7-5)^2 + (7-7)^2 + (7-7)^2}$$

$$= \sqrt{(1)^2 + (1)^2 + (0)^2 + (-1)^2 + (-2)^2 + (-1)^2 + (0)^2 + (2)^2 + (0)^2 + (0)^2} = \sqrt{12} = 3.464$$

- ✓ Jarak pada bobot ke-3 =

$$= \sqrt{(3-1)^2 + (3-1)^2 + (3-2)^2 + (3-3)^2 + (3-4)^2 + (4-5)^2 + (5-5)^2 + (7-5)^2 + (7-5)^2 + (7-7)^2}$$

$$= \sqrt{(2)^2 + (2)^2 + (1)^2 + (0)^2 + (-1)^2 + (-1)^2 + (0)^2 + (2)^2 + (2)^2 + (0)^2} = \sqrt{19} = 4.359$$

- ✓ Jarak pada bobot ke-4 =

$$= \sqrt{(3-1)^2 + (3-1)^2 + (3-1)^2 + (3-2)^2 + (3-3)^2 + (4-4)^2 + (5-5)^2 + (7-5)^2 + (7-5)^2 + (7-7)^2}$$

$$= \sqrt{(2)^2 + (2)^2 + (2)^2 + (1)^2 + (0)^2 + (0)^2 + (0)^2 + (2)^2 + (2)^2 + (0)^2} = \sqrt{21} = 4.583$$

- ✓ Jarak pada bobot ke-5 =

$$= \sqrt{(3-1)^2 + (3-1)^2 + (3-1)^2 + (3-1)^2 + (3-1)^2 + (4-5)^2 + (5-4)^2 + (7-3)^2 + (7-3)^2 + (7-2)^2}$$

$$= \sqrt{(2)^2 + (2)^2 + (2)^2 + (2)^2 + (2)^2 + (-1)^2 + (1)^2 + (4)^2 + (4)^2 + (5)^2} = \sqrt{79} = 8.888$$

- ✓ Jarak pada bobot ke-6 =

$$= \sqrt{(3-1)^2 + (3-1)^2 + (3-1)^2 + (3-1)^2 + (3-4)^2 + (4-4)^2 + (5-5)^2 + (7-3)^2 + (7-3)^2 + (7-2)^2}$$

$$= \sqrt{(2)^2 + (2)^2 + (2)^2 + (2)^2 + (-1)^2 + (0)^2 + (0)^2 + (4)^2 + (4)^2 + (5)^2} = \sqrt{74} = 8.602$$

- ✓ Jarak pada bobot ke-7 =

$$= \sqrt{(3-3)^2 + (3-5)^2 + (3-5)^2 + (3-5)^2 + (3-1)^2 + (4-4)^2 + (5-3)^2 + (7-3)^2 + (7-2)^2 + (7-1)^2}$$

$$= \sqrt{(0)^2 + (-2)^2 + (-2)^2 + (-2)^2 + (2)^2 + (0)^2 + (2)^2 + (4)^2 + (5)^2 + (6)^2} = \sqrt{97} = 9.849$$

- ✓ Jarak pada bobot ke-8 =

$$= \sqrt{(3-2)^2 + (3-2)^2 + (3-2)^2 + (3-3)^2 + (3-3)^2 + (4-3)^2 + (5-3)^2 + (7-5)^2 + (7-5)^2 + (7-5)^2}$$

$$= \sqrt{(1)^2 + (1)^2 + (1)^2 + (0)^2 + (0)^2 + (1)^2 + (2)^2 + (2)^2 + (2)^2 + (2)^2} = \sqrt{20} = 4.472$$

- ✓ Jarak pada bobot ke-9 =

$$= \sqrt{(3-2)^2 + (3-2)^2 + (3-2)^2 + (3-2)^2 + (3-3)^2 + (4-3)^2 + (5-3)^2 + (7-5)^2 + (7-5)^2 + (7-5)^2}$$

$$= \sqrt{(1)^2 + (1)^2 + (1)^2 + (1)^2 + (0)^2 + (1)^2 + (2)^2 + (2)^2 + (2)^2 + (2)^2} = \sqrt{21} = 4.583$$

- ✓ Jarak pada bobot ke-10 =

$$= \sqrt{(3-3)^2 + (3-3)^2 + (3-3)^2 + (3-3)^2 + (3-4)^2 + (4-4)^2 + (5-4)^2 + (7-5)^2 + (7-5)^2 + (7-5)^2}$$

$$= \sqrt{(0)^2 + (0)^2 + (0)^2 + (0)^2 + (-1)^2 + (0)^2 + (-1)^2 + (2)^2 + (2)^2 + (2)^2} = \sqrt{14} = 3.742$$

- ✓ Jarak pada bobot ke-11 =

$$= \sqrt{(3-3)^2 + (3-3)^2 + (3-2)^2 + (3-2)^2 + (3-3)^2 + (4-3)^2 + (5-4)^2 + (7-4)^2 + (7-5)^2 + (7-5)^2}$$

$$= \sqrt{(0)^2 + (0)^2 + (1)^2 + (1)^2 + (0)^2 + (1)^2 + (1)^2 + (3)^2 + (2)^2 + (2)^2} = \sqrt{21} = 4.583$$

✓ Jarak pada bobot ke-12 =

$= \sqrt{(3-3)^2 + (3-5)^2 + (3-7)^2 + (3-4)^2 + (3-3)^2 + (4-4)^2 + (5-5)^2 + (7-5)^2 + (7-7)^2 + (7-7)^2}$

$= \sqrt{(0)^2 + (2)^2 + (-4)^2 + (-1)^2 + (0)^2 + (0)^2 + (0)^2 + (2)^2 + (0)^2 + (0)^2} = \sqrt{25} = 5.000$

✓ Jarak pada bobot ke-13 =

$= \sqrt{(3-3)^2 + (3-5)^2 + (3-7)^2 + (3-4)^2 + (3-3)^2 + (4-4)^2 + (5-5)^2 + (7-5)^2 + (7-7)^2 + (7-7)^2}$

$= \sqrt{(0)^2 + (2)^2 + (-4)^2 + (-1)^2 + (0)^2 + (0)^2 + (0)^2 + (2)^2 + (0)^2 + (0)^2} = \sqrt{25} = 5.000$

✓ Jarak pada bobot ke-14 =

$= \sqrt{(3-1)^2 + (3-2)^2 + (3-3)^2 + (3-5)^2 + (3-5)^2 + (4-3)^2 + (5-4)^2 + (7-5)^2 + (7-6)^2 + (7-7)^2}$

$= \sqrt{(2)^2 + (1)^2 + (0)^2 + (-2)^2 + (-2)^2 + (1)^2 + (1)^2 + (2)^2 + (1)^2 + (0)^2} = \sqrt{20} = 4.472$

✓ Jarak pada bobot ke-15 =

$= \sqrt{(3-1)^2 + (3-2)^2 + (3-3)^2 + (3-5)^2 + (3-5)^2 + (4-3)^2 + (5-4)^2 + (7-5)^2 + (7-6)^2 + (7-7)^2}$

$= \sqrt{(2)^2 + (1)^2 + (0)^2 + (-2)^2 + (-2)^2 + (1)^2 + (1)^2 + (2)^2 + (1)^2 + (0)^2} = \sqrt{20} = 4.472$

✓ Jarak pada bobot ke-16 =

$= \sqrt{(3-3)^2 + (3-3)^2 + (3-3)^2 + (3-2)^2 + (3-1)^2 + (4-1)^2 + (5-4)^2 + (7-5)^2 + (7-5)^2 + (7-7)^2}$

$= \sqrt{(0)^2 + (0)^2 + (0)^2 + (1)^2 + (2)^2 + (3)^2 + (1)^2 + (2)^2 + (2)^2 + (0)^2} = \sqrt{23} = 4.796$

✓ Jarak pada bobot ke-17 =

$= \sqrt{(3-3)^2 + (3-3)^2 + (3-3)^2 + (3-2)^2 + (3-1)^2 + (4-1)^2 + (5-4)^2 + (7-5)^2 + (7-5)^2 + (7-7)^2}$

$= \sqrt{(0)^2 + (0)^2 + (0)^2 + (1)^2 + (2)^2 + (3)^2 + (1)^2 + (2)^2 + (2)^2 + (0)^2} = \sqrt{23} = 4.796$

✓ Jarak pada bobot ke-18 =

$= \sqrt{(3-5)^2 + (3-4)^2 + (3-3)^2 + (3-1)^2 + (3-8)^2 + (4-4)^2 + (5-3)^2 + (7-3)^2 + (7-2)^2 + (7-1)^2}$

$= \sqrt{(-2)^2 + (-1)^2 + (0)^2 + (2)^2 + (-5)^2 + (0)^2 + (2)^2 + (4)^2 + (5)^2 + (6)^2} = \sqrt{115} = 10.724$

✓ Jarak pada bobot ke-19 =

$= \sqrt{(3-2)^2 + (3-4)^2 + (3-3)^2 + (3-1)^2 + (3-8)^2 + (4-4)^2 + (5-3)^2 + (7-3)^2 + (7-2)^2 + (7-1)^2}$

$= \sqrt{(1)^2 + (-1)^2 + (0)^2 + (2)^2 + (-5)^2 + (0)^2 + (2)^2 + (4)^2 + (5)^2 + (6)^2} = \sqrt{112} = 10.583$

✓ Jarak pada bobot ke-20 =

$= \sqrt{(3-1)^2 + (3-2)^2 + (3-3)^2 + (3-4)^2 + (3-5)^2 + (4-5)^2 + (5-5)^2 + (7-5)^2 + (7-7)^2 + (7-7)^2}$

$= \sqrt{(2)^2 + (1)^2 + (0)^2 + (-1)^2 + (-2)^2 + (-1)^2 + (0)^2 + (2)^2 + (0)^2 + (0)^2} = \sqrt{15} = 3.873$

✓ Jarak pada bobot ke-21 =

$= \sqrt{(3-1)^2 + (3-1)^2 + (3-2)^2 + (3-3)^2 + (3-4)^2 + (4-4)^2 + (5-5)^2 + (7-5)^2 + (7-7)^2 + (7-7)^2}$

$= \sqrt{(2)^2 + (2)^2 + (1)^2 + (0)^2 + (-1)^2 + (0)^2 + (0)^2 + (2)^2 + (0)^2 + (0)^2} = \sqrt{14} = 3.742$

✓ Jarak pada bobot ke-22 =

$= \sqrt{(3-4)^2 + (3-5)^2 + (3-3)^2 + (3-2)^2 + (3-2)^2 + (4-4)^2 + (5-2)^2 + (7-2)^2 + (7-2)^2 + (7-2)^2}$

$= \sqrt{(-1)^2 + (-2)^2 + (0)^2 + (1)^2 + (1)^2 + (0)^2 + (3)^2 + (5)^2 + (5)^2 + (5)^2} = \sqrt{91} = 9.539$

✓ Jarak pada bobot ke-23 =

$= \sqrt{(3-3)^2 + (3-3)^2 + (3-3)^2 + (3-3)^2 + (3-3)^2 + (4-4)^2 + (5-5)^2 + (7-7)^2 + (7-7)^2 + (7-7)^2}$

$= \sqrt{(0)^2 + (0)^2 + (0)^2 + (0)^2 + (0)^2 + (1)^2 + (1)^2 + (2)^2 + (1)^2 + (0)^2} = \sqrt{0} = 0$

✓ Jarak pada bobot ke-24 =

$= \sqrt{(3-2)^2 + (3-2)^2 + (3-2)^2 + (3-4)^2 + (3-5)^2 + (4-3)^2 + (5-3)^2 + (7-3)^2 + (7-3)^2 + (7-3)^2}$

$= \sqrt{(1)^2 + (1)^2 + (1)^2 + (1)^2 + (-2)^2 + (1)^2 + (2)^2 + (4)^2 + (4)^2 + (4)^2} = \sqrt{61} = 7.810$

✓ Jarak pada bobot ke-25 =

$= \sqrt{(3-3)^2 + (3-3)^2 + (3-3)^2 + (3-3)^2 + (3-2)^2 + (4-4)^2 + (5-5)^2 + (7-5)^2 + (7-7)^2 + (7-7)^2}$

$= \sqrt{(0)^2 + (0)^2 + (0)^2 + (0)^2 + (1)^2 + (0)^2 + (0)^2 + (2)^2 + (0)^2 + (0)^2} = \sqrt{5} = 2.236$

✓ Jarak pada bobot ke-26 =

$= \sqrt{(3-2)^2 + (3-3)^2 + (3-3)^2 + (3-3)^2 + (3-4)^2 + (4-4)^2 + (5-5)^2 + (7-5)^2 + (7-7)^2 + (7-7)^2}$

$= \sqrt{(1)^2 + (0)^2 + (0)^2 + (0)^2 + (-1)^2 + (0)^2 + (0)^2 + (2)^2 + (0)^2 + (0)^2} = \sqrt{6} = 2.449$

✓ Jarak pada bobot ke-27 =

$= \sqrt{(3-1)^2 + (3-1)^2 + (3-1)^2 + (3-5)^2 + (3-5)^2 + (4-5)^2 + (5-5)^2 + (7-7)^2 + (7-7)^2 + (7-7)^2}$

$= \sqrt{(2)^2 + (2)^2 + (2)^2 + (-2)^2 + (-2)^2 + (-1)^2 + (0)^2 + (0)^2 + (0)^2 + (0)^2} = \sqrt{21} = 4.583$

✓ Jarak pada bobot ke-28 =

$= \sqrt{(3-5)^2 + (3-4)^2 + (3-4)^2 + (3-3)^2 + (3-1)^2 + (4-5)^2 + (5-6)^2 + (7-7)^2 + (7-7)^2 + (7-7)^2}$

$= \sqrt{(-2)^2 + (-1)^2 + (-1)^2 + (0)^2 + (2)^2 + (-1)^2 + (-1)^2 + (0)^2 + (0)^2 + (0)^2} = \sqrt{12} = 3.464$

Dari hasil perhitungan jarak diatas, nilai jarak paling kecil yaitu nilai jarak pada bobot ke-23 yaitu dengan nilai 0 seperti terlihat pada Tabel 8.

Tabel 8. Nilai bobot huruf

No.	Nama Huruf	Bentuk Huruf	Kelas	Nilai Bobot
1	Alif	ا	1	7.280
2	Ba	ب	2	3.464
3	Ta	ت	3	4.359
4	Tsa	ث	4	4.583
5	Jim	ج	5	8.888
6	Ha	ح	6	8.602
7	Kha	خ	7	9.849
8	Dal	د	8	4.472
9	Dzal	ذ	9	4.583

Tabel 8. Nilai bobot huruf (Lanjutan)

No.	Nama Huruf	Bentuk Huruf	Kelas	Nilai Bobot
10	Ra	ر	10	3.742
11	Za	ز	11	4.583
12	Sin	س	12	5.000
13	Syin	ش	13	5.000
14	Shad	ص	14	4.472
15	Dha	ض	15	4.472
16	Tha	ط	16	4.796
17	Zha	ظ	17	4.796
18	‘Ain	ع	18	10.724
19	Gha	غ	19	10.583
20	Fa	ف	20	3.873
21	Qaf	ق	21	3.742
22	Kaf	ك	22	9.539
23	Lam	ل	23	0.000
24	Mim	م	24	7.810
25	Nun	ن	25	2.236
26	Waw	و	26	2.449
27	Haa	ه	27	4.583
28	Ya	ي	28	3.464

= Huruf yang mempunyai nilai bobot terkecil

Maka dapat disimpulkan bahwa masukan vektor nilai *Chain Code* yang diuji tersebut termasuk ke dalam Kelas 23 (Huruf Lam) seperti terlihat pada Tabel 9.

Tabel 9. Nilai bobot huruf (diurutkan berdasarkan nilai bobot terkecil)

No.	Nama Huruf	Bentuk Huruf	Kelas	Nilai Bobot
1	Lam	ل	23	0.000
2	Nun	ن	25	2.236
3	Waw	و	26	2.449
4	Ba	ب	2	3.464

Tabel 9. Nilai bobot huruf (diurutkan berdasarkan nilai bobot terkecil)(Lanjutan)

No.	Nama Huruf	Bentuk Huruf	Kelas	Nilai Bobot
5	Ya	ي	28	3.464
6	Ra	ر	10	3.742
7	Qaf	ق	21	3.742
8	Fa	ف	20	3.873
9	Ta	ت	3	4.359
10	Dal	د	8	4.472
11	Shad	ص	14	4.472
12	Dha	ض	15	4.472
13	Tsa	ث	4	4.583
14	Dzal	ذ	9	4.583
15	Za	ز	11	4.583
16	Haa	ه	27	4.583
17	Tha	ط	16	4.796
18	Zha	ظ	17	4.796
19	Sin	س	12	5.000
20	Syin	ش	13	5.000
21	Alif	ا	1	7.280
22	Mim	م	24	7.810
23	Ha	ح	6	8.602
24	Jim	ج	5	8.888
25	Kaf	ك	22	9.539
26	Kha	خ	7	9.849
27	Gha	غ	19	10.583
28	‘Ain	ع	18	10.724

= Huruf yang mempunyai nilai bobot terkecil

BAB III
ANALISIS DAN PERANCANGAN

Pada bagian ini, pembahasan yang dijelaskan mengenai analisis serta perancangan dari aplikasi yang dibangun serta penelitian yang dilakukan. Diantaranya analisis sistem, cara kerja sistem, *workflow* sistem, tahap perancangan sistem dengan pemodelan UML serta perancangan antar muka aplikasi yang dibuat.

3.1 Analisis

Dalam pembangunan aplikasi identifikasi huruf Hijaiyah dengan menggunakan algoritma *Chain Code* dan LVQ ini diperlukan terlebih dahulu analisis terhadap sistem yang akan dibuat. Algoritma *Chain Code* yang akan menentukan struktur pembentuk huruf, termasuk ke dalam proses ekstraksi ciri. Sedangkan algoritma LVQ menentukan kecocokan citra huruf uji dengan latih sesuai dengan nilai bobot yang terkecil yang kemudian dimasukkan ke dalam kelas huruf yang bersangkutan.

Citra huruf yang diuji merupakan citra huruf Hijaiyah tanpa menggunakan baris yang berjumlah 28 huruf. Jenis huruf yang digunakan yaitu KFGQPC Uthman Taha Naskh dengan ukuran 120 pt. Citra huruf Hijaiyah yang diambil sudah dalam bentuk gambar dengan format *Portable Network Graphics* (PNG) yang sudah dibuat sebelumnya. Citra huruf yang dilatih, disimpan nilai *Chain Code* nya ke dalam basis pengetahuan data latih. Nilai *Chain Code* yang didapat mempunyai panjang berbeda dari setiap hurufnya. Hal tersebut disesuaikan dengan panjang vektor huruf yang dipilih.

Hasil akhir dari sistem aplikasi yang dibangun yaitu berupa hasil kecocokan huruf yang diuji dengan huruf yang dilatih. Algoritma LVQ berfungsi untuk mencari kedekatan kelas dari citra huruf uji dengan citra huruf latih.

3.2 Cara Kerja Sistem Identifikasi Huruf Hijaiyah

Cara kerja sistem dibagi ke dalam dua bagian utama, yaitu proses mendapatkan data latih dan proses data uji. Proses data latih bertujuan untuk mendapatkan citra huruf Hijaiyah yang tersimpan pada basis pengetahuan. Sedangkan proses data uji untuk memeriksa kecocokan huruf yang diuji dengan huruf yang ada pada basis pengetahuan.

Berikut merupakan proses mendapatkan data latih :

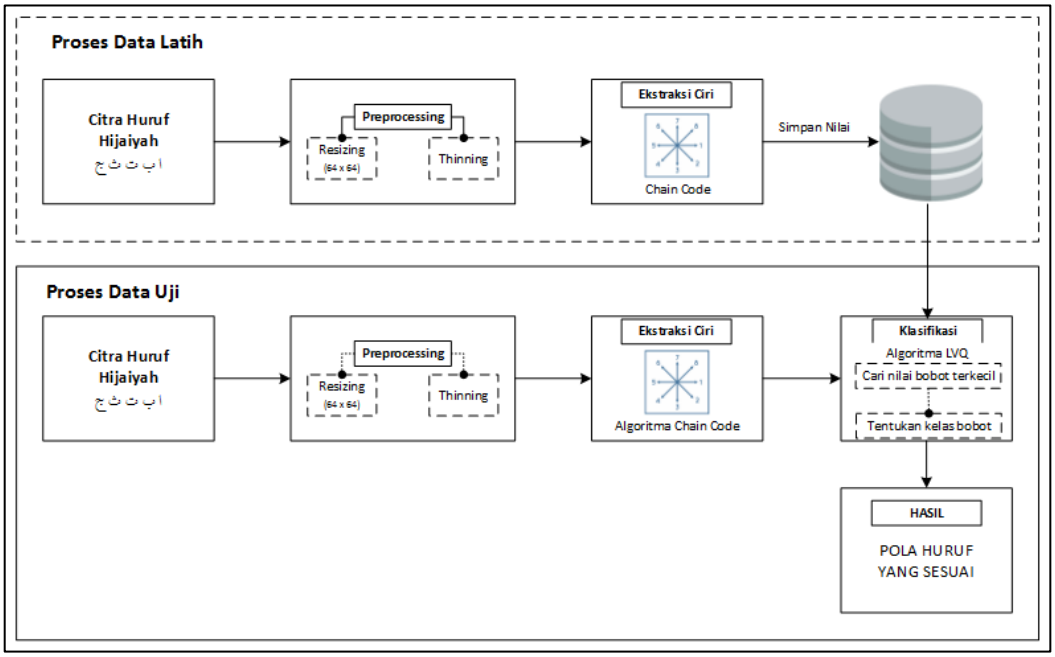
1. Memasukkan citra huruf Hijaiyah
2. Melakukan tahap *Preprocessing* pada citra huruf Hijaiyah tersebut
3. Melakukan proses ektraksi ciri dengan menggunakan algoritma *Chain Code* untuk dijadikan vektor bobot pada algoritma LVQ
4. Simpan hasil ke basis pengetahuan

Sedangkan untuk proses data uji, urutan langkahnya sebagai berikut :

1. Memasukkan citra huruf Hijaiyah
2. Melakukan tahap *Preprocessing* pada citra huruf Hijaiyah tersebut
3. Melakukan proses ektraksi ciri dengan menggunakan algoritma *Chain Code* untuk dijadikan vektor masukan (vektor latih) pada algoritma LVQ
4. Melakukan proses klasifikasi sistem dengan mencari nilai bobot terkecil antara vektor masukan dengan vektor bobot menggunakan algoritma LVQ
5. Sistem memberikan pola huruf yang sesuai antara data uji dan data latih

3.3 *Workflow* Sistem

Setelah dibahas mengenai cara kerja sistem pada penjelasan sebelumnya, *workflow* (alur kerja) dari sistem yang dibuat seperti terlihat pada `Gambar 11`.

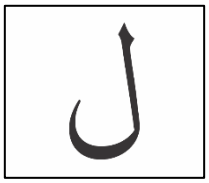


Gambar 11. Workflow Sistem

Penjelasan dari *workflow* sistem (Gambar 10) dijelaskan pada sub bab 3.3.1 sampai dengan sub bab 3.3.4.

3.3.1 Citra Huruf Hijaiyah

Citra huruf Hijaiyah (Gambar 12) dipilih sebagai langkah awal cara kerja sistem. Citra huruf Hijaiyah berupa citra digital.



Gambar 12. Citra Huruf Hijaiyah

3.3.2 Tahap *Preprocessing*

Sesuai dengan perancangan awal, tahap *Preprocessing* terdiri dari dua bagian yaitu tahap *Resize* dan *Thinning*.

3.3.2.1 Tahap *Resize*

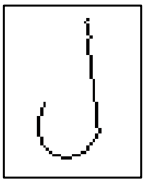
Pada tahap *Resize*, ukuran citra yang dipilih diubah menjadi citra berukuran 64x64 piksel. Hal ini dilakukan agar citra yang diproses memiliki ukuran piksel

yang sama. Proses *Resize* menggunakan fitur yang disediakan oleh MATLAB yaitu `imresize`. Contoh penerapannya pada MATLAB yaitu :

```
img = imresize(img,[64 ,64]);
```

3.3.2.2 Tahap Thinning

Selanjutnya sistem melakukan proses *Thinning*. Pada proses ini, citra huruf yang dipilih dilakukan penipisan untuk mendapatkan garis pada citra huruf menjadi 1 piksel. Citra huruf setelah melalui proses *Thinning* ditunjukkan oleh `Gambar 13`.



Gambar 13. Citra huruf hasil Thinning

Fitur MATLAB yang digunakan yaitu `bwmorph`. Contoh penerapannya pada MATLAB yaitu :

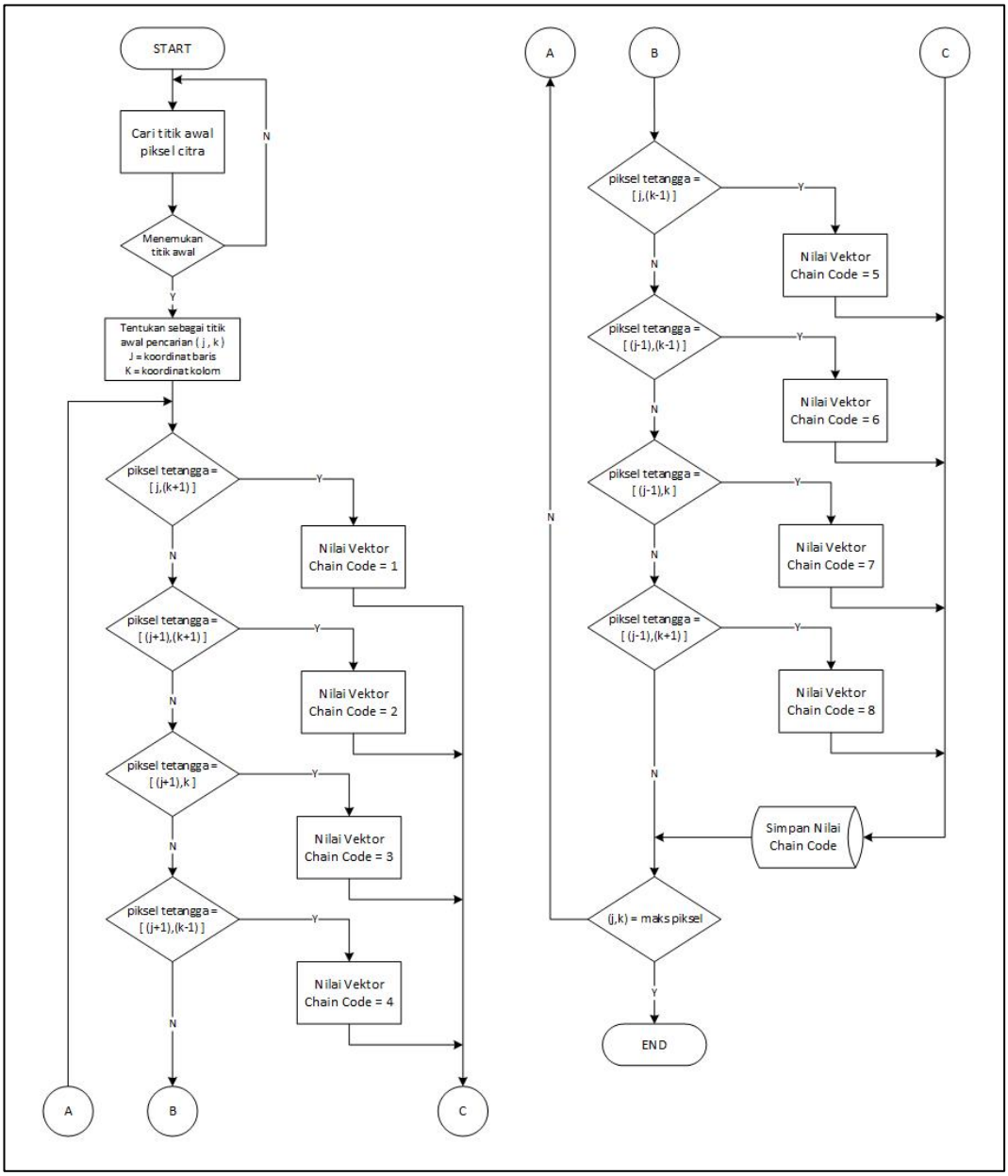
```
img = bwmorph(~img3,'thin',inf);
```

3.3.3 Tahap Ekstraksi Ciri dengan *Chain Code*

Pada tahap ini, citra huruf yang sudah melalui tahap *Preprocessing (Resize* dan *Thinning)* dilakukan proses ekstraksi ciri dengan menggunakan algoritma *Chain Code*. Hasil akhir yang didapat dari hasil ekstraksi ini yaitu berupa nilai vektor pembentuk citra huruf yang dipilih. Contoh hasil akhir dari nilai *Chain Code* pembentuk huruf yaitu (Huruf ل) :

```
2 4 2 3 3 3 2 4 3 3 3 3 3 2 3 3 3 3 3 3 3 2
3 3 3 3 3 3 3 2 3 3 3 3 3 3 3 3 2 3 4 3 4 4
4 3 4 5 4 5 5 4 5 5 5 6 5 5 6 6 6 6 7 7 6 7
7 7 7 7 7 8 7 7 8 7
```

Seperti telah dijelaskan pada Sub Bab 3.1 Analisis, bahwa nilai *Chain Code* yang didapat berbeda setiap hurufnya. Untuk nilai *Chain Code* huruf Lam diatas, panjang nilai *Chain Code* yaitu sebanyak 38. Alur kerja dari *Chain Code* sesuai sistem yang dibangun, seperti terlihat pada `Gambar 14`.



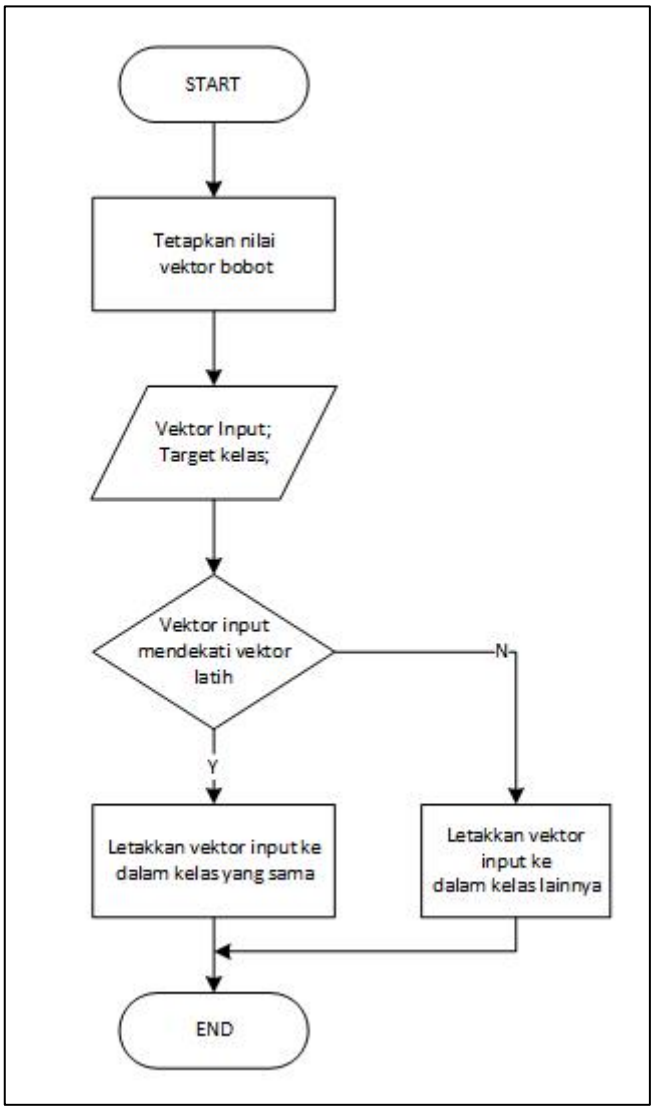
Gambar 14. Alur kerja Chain Code sesuai sistem yang dibangun

3.3.4 Tahap Klasifikasi dengan LVQ

Pada tahap ini, klasifikasi dilakukan dengan menggunakan metode LVQ. Masukan dari metode LVQ ini yaitu berupa nilai vektor *Chain Code* pembentuk suatu citra huruf. Vektor masukan tersebut kemudian diklasifikasikan oleh lapisan kompetitif *(Competitive Layer)*. Kelas-kelas yang didapatkan sebagai hasil dari lapisan kompetitif ini hanya tergantung pada jarak antara vektor-vektor *input*. Jika

dua vektor *input* mendekati sama, maka lapisan kompetitif meletakkan kedua vektor *input* tersebut ke dalam kelas yang sama[6].

Alur kerja dari metode LVQ sesuai sistem yang dibangun, seperti terlihat pada Gambar 15.



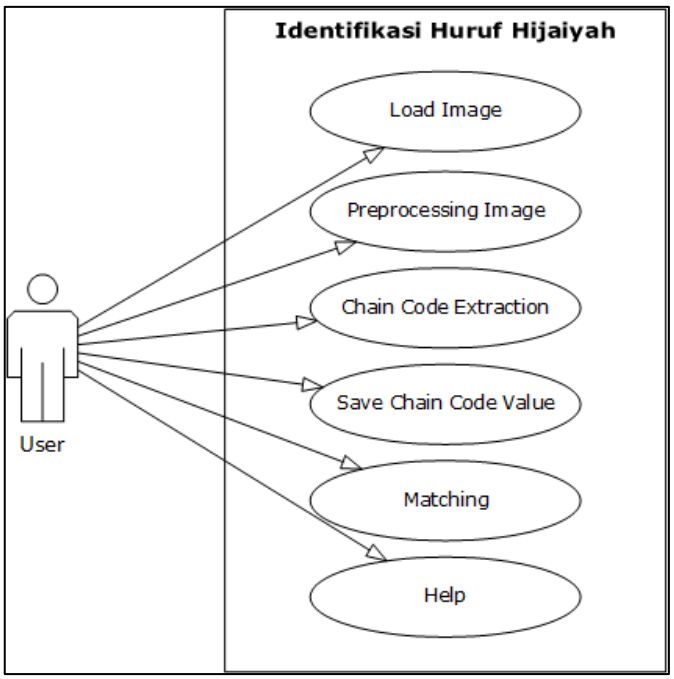
Gambar 15. Alur Kerja LVQ sesuai sistem yang dibangun

3.4 Perancangan Sistem

Pada perancangan sistem identifikasi huruf Hijaiyah ini menggunakan UML sebagai metode pendekatannya.

3.4.1 Use Case Diagram

Berdasarkan analisis yang sudah dilakukan, maka dapat didefinisikan kebutuhan fungsionalitas-fungsionalitas yang meliputi fungsionalitas *Load Image*, fungsionalitas *Preprocessing Image*, fungsionalitas *Chain Code Extraction*, fungsionalitas *Save Chain Code Value*, fungsionalitas *Matching* serta fungsionalitas *Help*. Seluruh fungsionalitas aplikasi dinyatakan pada diagram *Use Case* Gambar 16.



Gambar 16. Use Case Diagram

Pada diagram *Use Case* yang ditunjukkan oleh Gambar 15, terdapat satu aktor yaitu *User*. Berikut ini adalah deskripsi dari masing-masing fungsionalitas yang terdapat pada diagram *Use Case* pada Gambar 15.

- Fungsionalitas *Load Image*, pada fungsionalitas ini *User* memilih citra atau gambar yang dimasukan ke dalam data latih atau citra yang diuji.

- Fungsionalitas *Preprocessing Image*, pada fungsionalitas ini sistem melakukan tahap *resizing* serta penipisan citra *(thinning)* yang dimasukkan untuk kemudian diolah pada tahap selanjutnya.
- Fungsionalitas *Chain Code Extraction*, pada fungsionalitas ini sistem mendapatkan nilai *Chain Code* dari proses ektraksi ciri pada citra yang dimasukkan.
- Fungsionalitas *Save Chain Code Value*, pada fungsionalitas ini sistem menyimpan nilai *Chain Code* dari citra huruf ke basis pengetahuan.
- Fungsionalitas *Matching*, pada fungsionalitas ini sistem menampilkan karakter citra yang sesuai dengan yang ada pada basis pengetahuan.
- Fungsionalitas *Help*, pada fungsionalitas ini ditampilkan mengenai cara kerja dari aplikasi ini.

3.4.1.1 Skenario *Use Case*

Untuk memberikan penjelasan dari fungsionalitas yang sudah didefinisikan sebelumnya, maka perlu adanya skenario *Use Case* dari masing-masing fungsionalitas tersebut. Hal tersebut dijelaskan dalam bentuk tabel skenario yaitu dari `Tabel 10` sampai dengan `Tabel 15`.

`Tabel 10. Skenario Use Case Fungsionalitas Load Image`

IDENTIFIKASI	
Nomor	IH-01
Nama	*Load Image*
Tujuan	Aktor dapat memilih citra yang dimasukkan ke data latih dan data uji
Aktor	*User*
SKENARIO	
Kondisi Awal	- Aktor berada pada *form Load Image* - *Axes* untuk citra yang dipilih, masih dalam keadaan kosong
Aksi Aktor	**Reaksi Sistem**
Skenario Normal	
1. Aktor menekan tombol *'Load Image'* untuk memilih citra huruf	

Tabel 10. Skenario Use Case Fungsionalitas Load Image (Lanjutan)

	2. Sistem menampilkan *dialog box* lokasi *path* untuk memilih citra yang dimasukkan 3. Sistem menampilkan citra yang telah dipilih ke dalam *axes*
4. Aktor menekan tombol *'Next'* untuk masuk ke *form* berikutnya	
Skenario Alternatif	
5. Aktor menekan tombol *'Load Image'* untuk memilih citra huruf	
	6. Sistem menampilkan *dialog box* lokasi *path* untuk memilih citra yang dimasukkan 7. Sistem menampilkan citra yang telah dipilih ke dalam *axes*
8. Aktor menekan tombol '*Reset'* untuk memasukkan citra huruf yang lainnya atau jika aktor salah memasukkan citra huruf	
	9. Sistem menghapus citra huruf pada *Axes*
Kondisi Akhir	Sistem menampilkan citra yang dipilih

Tabel 11. Skenario Use Case Fungsionalitas Preprocessing

IDENTIFIKASI	
Nomor	IH-02
Nama	*Preprocessing Image*
Tujuan	Aktor dapat melihat hasil dari tahap *Preprocessing* yang terdiri dari proses *resizing* dan *thinning*.
Aktor	*User*
SKENARIO	
Kondisi Awal	- Aktor berada pada *form Preprocessing* - *Axes* masih dalam keadaan kosong, serta tabel hasil binerisasi belum ditampilkan
Aksi Aktor	**Reaksi Sistem**

Tabel 11. Skenario Use Case Fungsionalitas Preprocessing

1. Aktor menekan tombol *'Preprocessing'*	
	2. Sistem melakukan tahap *Preprocessing* yang dimulai dengan proses *resizing* pada citra latih yang dipilih 3. Sistem melakukan proses penipisan *(thinning)* pada citra latih yang dipilih 4. Sistem menampilkan hasil citra yang sudah dilakukan proses penipisan 5. Sistem menampilkan tabel hasil binerisasi dari vektor yang sudah melalui proses *thinning* tersebut
6. Aktor menekan tombol *'Next'* untuk masuk ke *form* berikutnya	
Kondisi Akhir	Sistem menampilkan citra yang sudah dilakukan penipisan *(thinning)* serta tabel hasil binerisasi.

Tabel 12. Skenario Use Case Chain Code Extraction

IDENTIFIKASI	
Nomor	IH-03
Nama	*Chain Code Extraction*
Tujuan	Aktor dapat melihat hasil akhir dari proses ekstraksi *Chain Code* berupa nilai vektor dari *Chain Code* citra yang dipilih.
Aktor	*User*
SKENARIO	
Kondisi Awal	- Aktor berada pada *form Chain Code Extraction* - Kolom keterangan untuk nilai *Chain Code* dalam keadaan tidak ada nilai
Aksi Aktor	**Reaksi Sistem**
1. Aktor menekan tombol *'Extraction'*	
	2. Sistem menjalankan algoritma *Chain Code*

Tabel 11. Skenario Use Case Fungsionalitas Preprocessing (Lanjutan)

	3. Sistem menampilkan nilai ekstraksi ciri dari citra yang diambil pada kolom yang tersedia
Kondisi Akhir	Sistem menampilkan nilai *Chain Code* dari citra yang dipilih

Tabel 13. Skenario Use Case Save Chain Code Value

IDENTIFIKASI	
Nomor	IH-04
Nama	*Save Chain Code Value*
Tujuan	Aktor dapat menyimpan nilai *Chain Code* dari citra yang dipilih ke basis pengetahuan.
Aktor	*User*
SKENARIO	
Kondisi Awal	- Aktor berada pada *form Chain Code Extraction* - Nilai *Chain Code* sudah berhasil di tampilkan oleh sistem
Aksi Aktor	**Reaksi Sistem**
1. Aktor menekan tombol *'Save'*	
	2. Sistem menampilkan *dialog box* lokasi *path* untuk meyimpan nilai *Chain Code*
3. Aktor memasukkan nama *file* nilai *Chain Code* sesuai citra huruf yang dipilih	
	4. Sistem menyimpan nilai akhir *Chain Code* ke dalam data latih
Kondisi Akhir	Nilai *Chain Code* dari citra yang dipilih berhasil disimpan ke dalam data latih

Tabel 14. Skenario Use Case Matching

IDENTIFIKASI	
Nomor	IH-05
Nama	*Matching*
Tujuan	Aktor dapat melihat citra huruf yang cocok dengan citra huruf yang ada pada data latih.
Aktor	*User*
SKENARIO	
Kondisi Awal	- Aktor berada pada *form Matching* - *Axes* serta keterangan kecocokan huruf dalam keadaan kosong
Aksi Aktor	**Reaksi Sistem**
1. Aktor menekan tombol *'Load Image'* untuk memilih citra huruf	
	2. Sistem menampilkan dialog box lokasi *path* untuk memilih citra yang dimasukkan 3. Sistem menampilkan citra yang telah dipilih ke dalam *axes*
4. Aktor menekan tombol *'Matching'* untuk mendapatkan pola huruf yang sesuai	
	5. Sistem melakukan proses *resizing* pada citra uji yang dipilih 6. Sistem melakukan proses penipisan *(thinning)* pada citra uji yang dipilih 7. Sistem menjalankan algoritma *Chain Code* untuk mendapatkan nilai vektor pembentuk huruf 8. Sistem mencocokkan pola huruf dengan basis pengetahuan yang dimiliki dengan mencari kedekatan nilai *Chain Code* menggunakan algoritma LVQ 9. Sistem menampilkan citra huruf yang sesuai
Kondisi Akhir	Sistem menampilkan citra huruf yang sesuai.

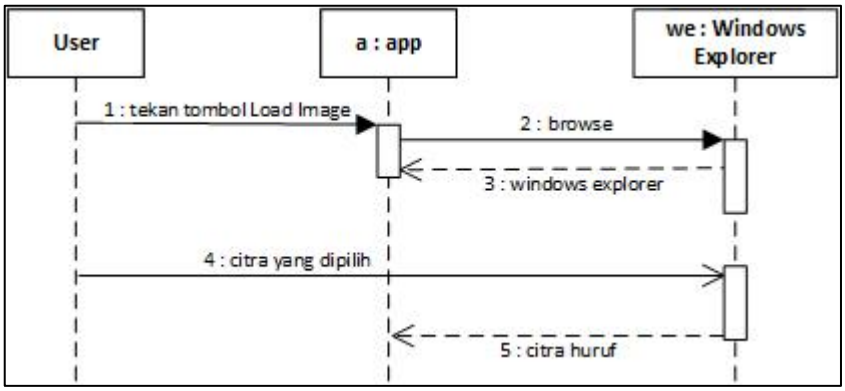
Tabel 15. Skenario Use Case Help

IDENTIFIKASI	
Nomor	IH-06
Nama	*Help*
Tujuan	Aktor dapat mengetahui cara kerja aplikasi serta informasi mengenai aplikasi.
Aktor	*User*
SKENARIO	
Kondisi Awal	Aktor berada pada *form* aplikasi.
Aksi Aktor	**Reaksi Sistem**
1. Aktor menekan tombol *'Help'*	
	2. Sistem menampilkan *form Help* yang berisi mengenai gambaran singkat aplikasi
Kondisi Akhir	Sistem menampilkan *form Help*

3.4.2 *Sequence Diagram*

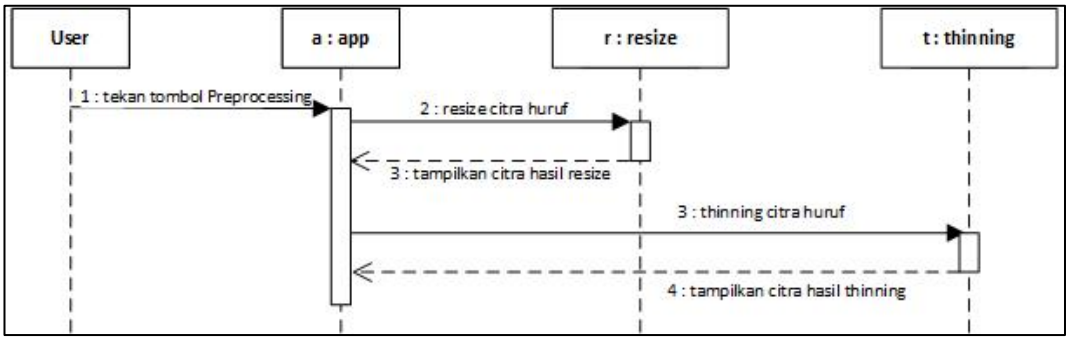
Sequence Diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek[9]. *Sequence Diagram* dari sistem yang dibangun ditunjukkan oleh Gambar 17 sampai Gambar 22.

1. *Sequence Diagram* untuk *Use Case* : *Load Image* (Gambar 17)



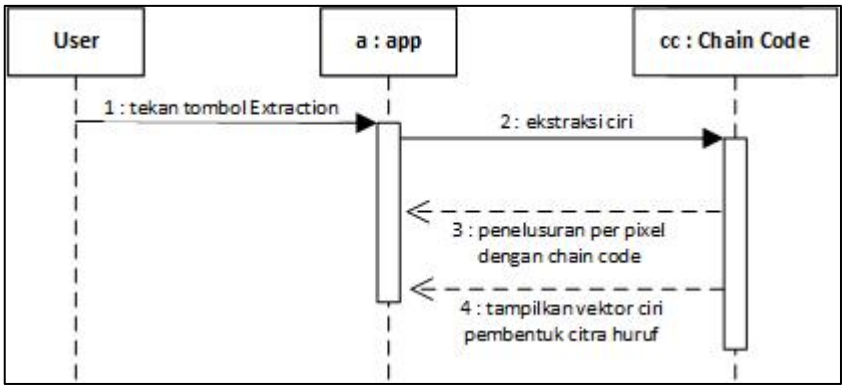
Gambar 17. Sequence Diagram Load Image

2. *Sequence Diagram* untuk *Use Case* : *Preprocessing* (Gambar 18)



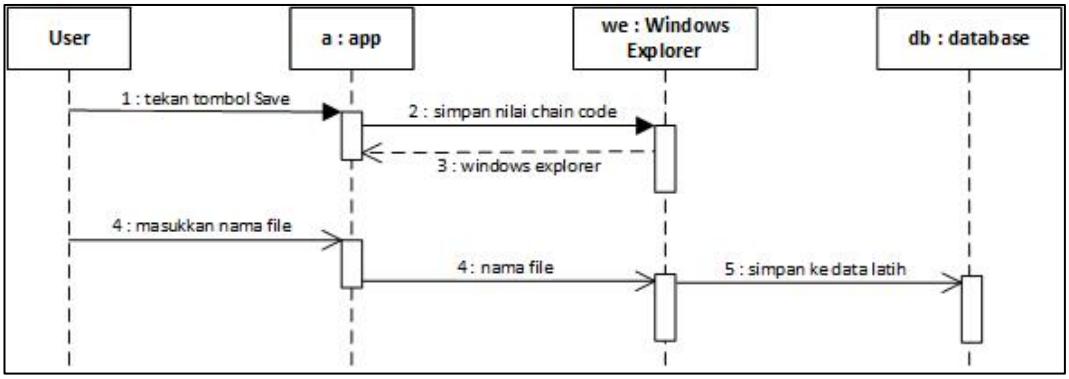
Gambar 18. Sequence Diagram Preprocessing

3. *Sequence Diagram* untuk *Use Case* : *Chain Code Extraction* (Gambar 19)



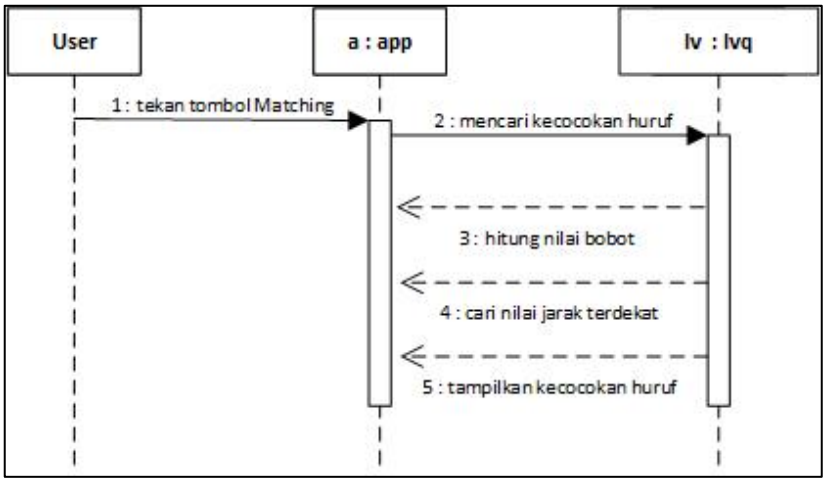
Gambar 19. Sequence Diagram Chain Code Extraction

4. *Sequence Diagram* untuk *Use Case* : *Save Chain Code Value* (Gambar 20)



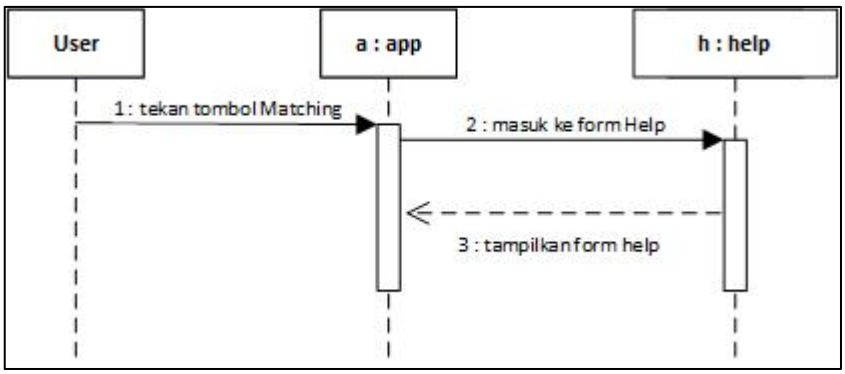
Gambar 20. Sequence Diagram Save Chain Code Value

5. *Sequence Diagram* untuk *Use Case* : *Matching* (Gambar 21)



Gambar 21. Sequence Diagram Matching

6. *Sequence Diagram* untuk *Use Case* : *Help* (Gambar 22)

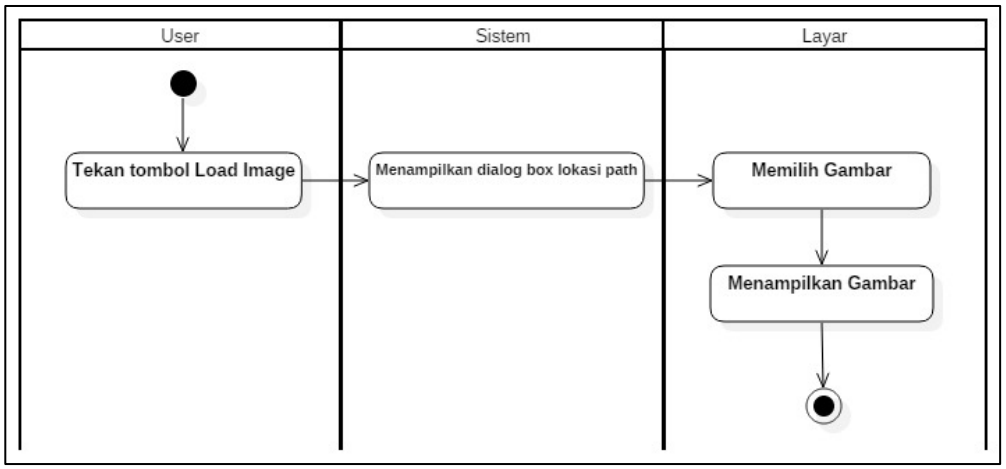


Gambar 22. Sequence Diagram Help

3.4.3 *Activity Diagram*

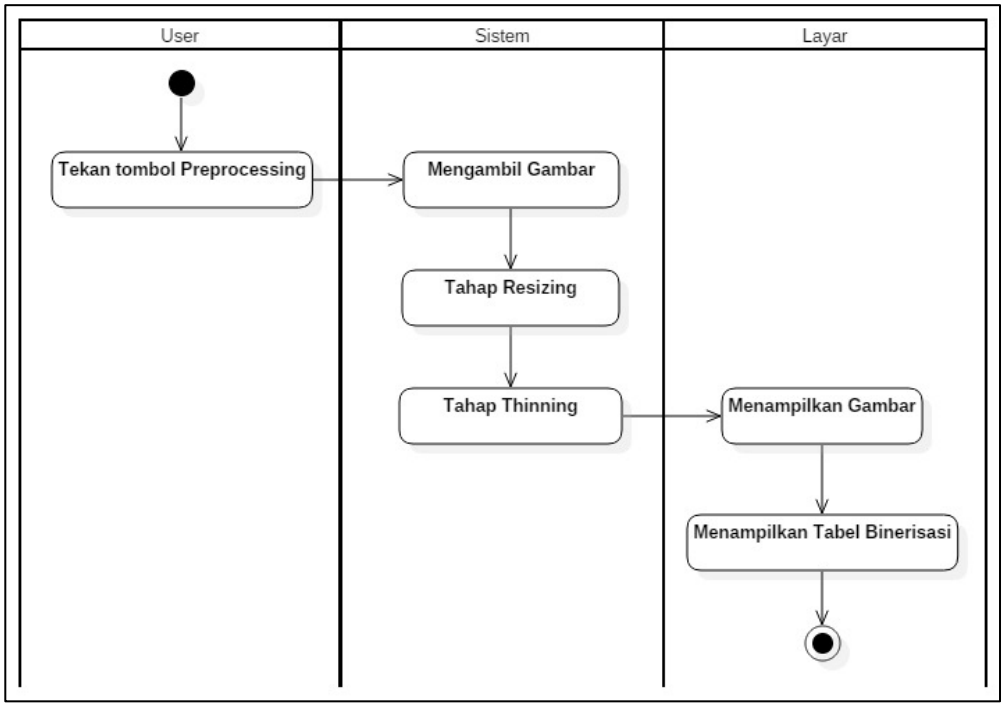
Activiy Diagram atau diagram aktifitas menggambarkan *workflow* (aliran kerja) atau aktifitas dari sebuah sistem yang ada pada perangkat lunak[9]. *Activity Diagram* dari sistem yang dibangun ditunjukkan oleh Gambar 23 sampai Gambar 28.

1. *Activity Diagram* untuk *Use Case* : *Load Image* (Gambar 23)



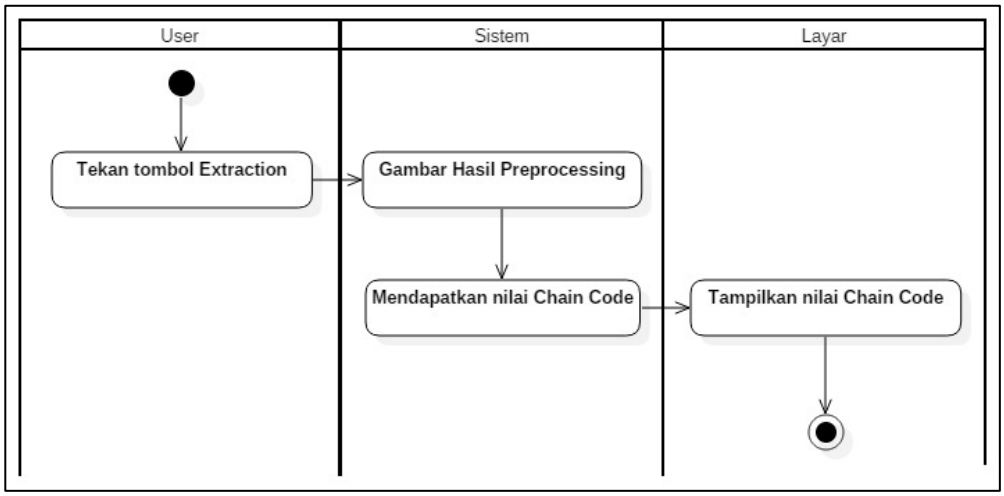
Gambar 23. Activity Diagram Load Image

2. *Activity Diagram* untuk *Use Case* : *Preprocessing* (Gambar 24)



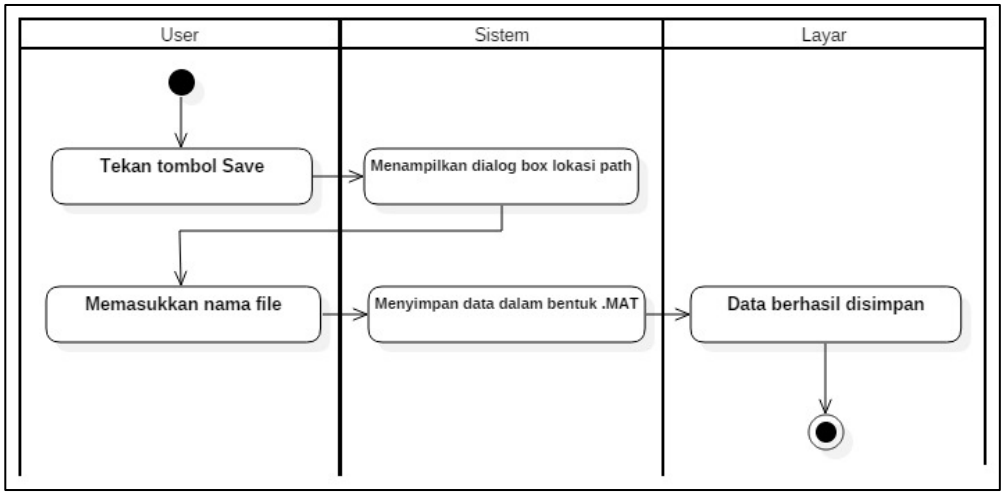
Gambar 24. Activity Diagram Preprocessing

3. *Activity Diagram* untuk *Use Case* : *Chain Code Extraction* (Gambar 25)



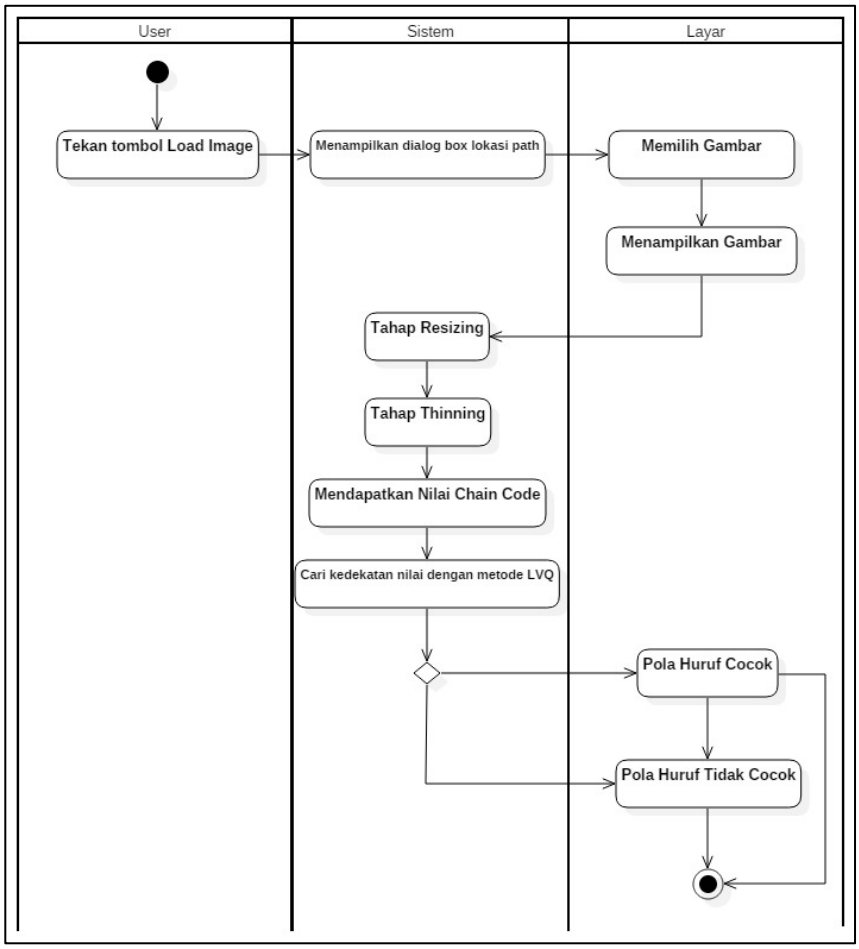
Gambar 25. Activity Diagram Chain Code Extraction

4. *Activity Diagram* untuk *Use Case* : *Save Chain Code Value* (Gambar 26)



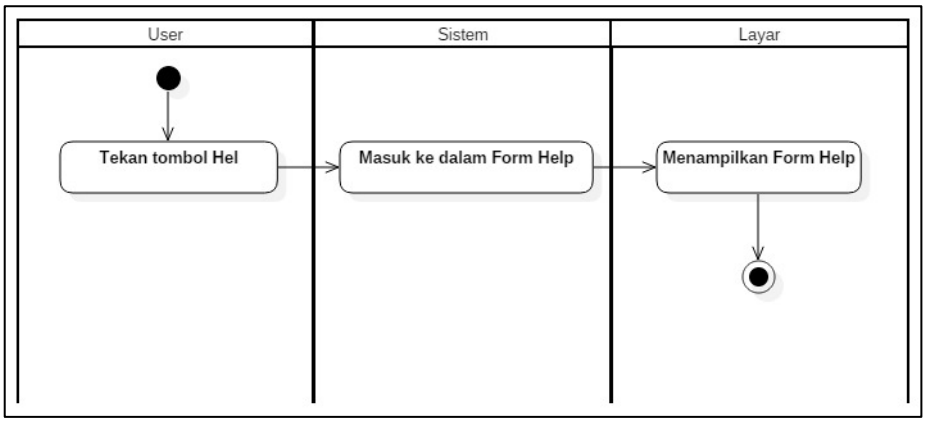
Gambar 26. Activity Diagram Save Chain Code Value

5. *Activity Diagram* untuk *Use Case* : *Matching* (Gambar 27)



Gambar 27. Activity Diagram Matching

6. *Activity Diagram* untuk *Use Case* : *Help* (Gambar 28)



Gambar 28. Activity Diagram Help

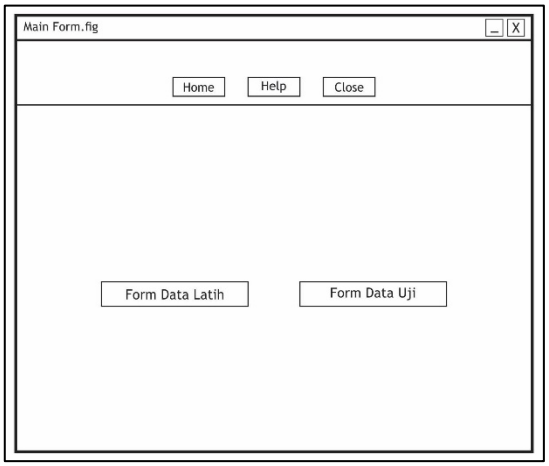
3.5 Perancangan Antar Muka

1. Rancangan Antar Muka *Form* Utama

   Nama Dialog : *Form Main Form*

   Fungsi : Menampilkan halaman awal aplikasi

   Bentuk : <<Gambar 29>>



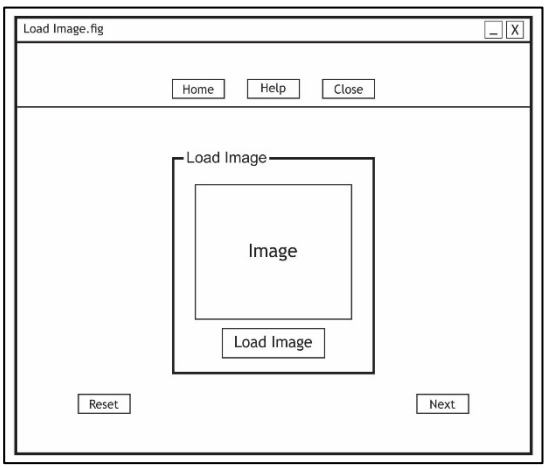
Gambar 29. Rancangan Antar Muka Form Utama

2. Rancangan Antar Muka *Form Load Image*

   Nama Dialog : *Form Load Image*

   Fungsi : Memasukkan citra latih dan citra uji

   Bentuk : <<Gambar 30>>



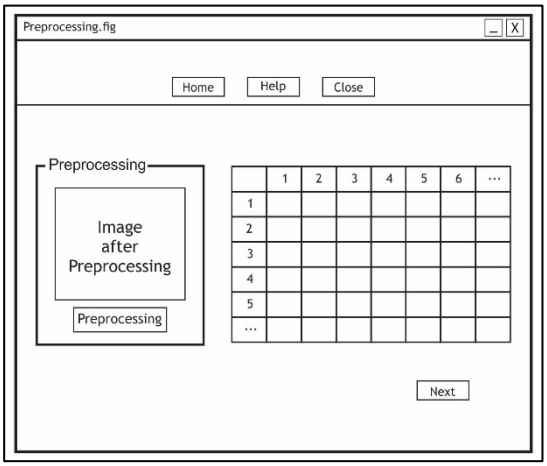
Gambar 30. Rancangan Antar Muka Form Load Image

3. Rancangan Antar Muka *Form Preprocessing*

   Nama Dialog : *Form Preprocessing*

   Fungsi : Mengeksekusi tahap *Preprocessing* citra dan menampilkan nilai vektor biner hasil *Preprocessing*

   Bentuk : <<Gambar 31>>



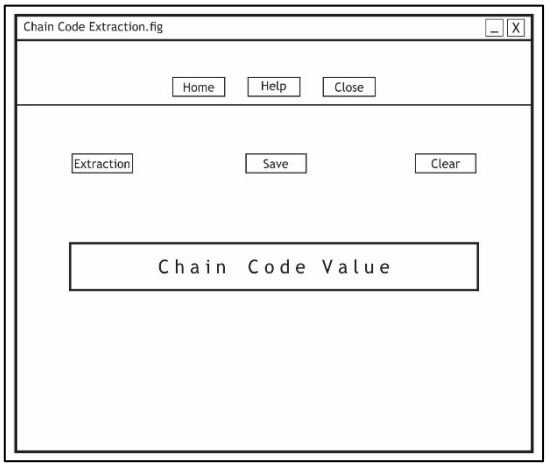
Gambar 31. Rancangan Antar Muka Form Preprocessing

4. Rancangan Antar Muka *Form Chain Code Extraction*

   Nama Dialog : *Form Chain Code Extraction*

   Fungsi : Mendapatkan nilai vektor *Chain Code* dari citra huruf yang dipilih

   Bentuk : <<Gambar 32>>



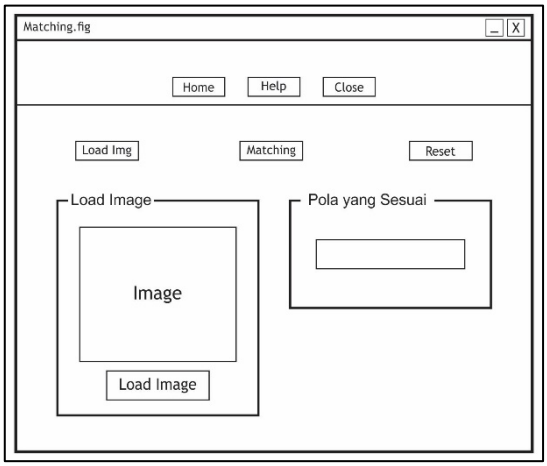
Gambar 32. Rancangan Antar Muka Form Chain Code Extraction

5. Rancangan Antar Muka *Form Matching*

   Nama Dialog : *Form Matching*

   Fungsi : Mendapatkan pola huruf yang sesuai antara citra yang ada pada data latih dengan citra yang ada pada data uji

   Bentuk : <<Gambar 33>>



Gambar 33. Rancangan Antar Muka Form Matching

BAB IV
IMPLEMENTASI DAN PENGUJIAN

Pada bagian ini, pembahasan yang dijelaskan mengenai implementasi dan pengujian dari aplikasi yang dibangun. Diantaranya yaitu mengenai lingkungan pengembangan sistem, implementasi modul program, pengujian hasil implementasi perangkat lunak serta hasil pengujian.

4.1 Lingkungan Pengembangan

Dalam lingkungan pengembangan, dijelaskan mengenai perangkat-perangkat yang digunakan dalam pengembangan aplikasi algoritma *Chain Code* serta LVQ pada pengenalan huruf Hijaiyah. Berikut spesifikasi lingkungan pengembangan yang digunakan.

4.1.1 Perangkat Keras

Spesifikasi yang digunakan dalam pembangunan aplikasi yaitu *Notebook* ASUS A44H dengan rincian sebagai berikut :

1. *Processor* Intel® Core™ i3-2330M CPU @ 2.20 GHz
2. *Internal Memory* dengan kapasitas maksimum 2048MB RAM
3. *Internal Storage* HDD 250GB
4. *Display* Intel® HD Graphics 3000
5. Sistem Operasi Windows 10 Pro 64-bit (10.0, Build 10240)

4.1.2 Perangkat Lunak

Sedangkan kebutuhan perangkat lunak yang diperlukan dalam pembangunan aplikasi ini mencakup :

1. Windows 7 *or higher,* dibutuhkan sebagai sistem operasi yang digunakan untuk menjalankan aplikasi.
2. *Scripting* dan *Coding* yang digunakan untuk membuat *interface* dan *function* menggunakan aplikasi MATLAB R2013B dengan bahasa pemrograman C dan Pascal.

4.2 Implementasi Modul Program

Modul program yang diimplementasi mencakup seluruh aplikasi yang dibuat. Tahapan mengunggah citra huruf ke dalam aplikasi, *Preprocessing*, mendapatkan nilai *Chain Code* serta tahapan pencocokan dengan algoritma LVQ diimplementasikan dengan memanggil beberapa fitur yang disediakan oleh MATLAB.

4.2.1 Implementasi *Form Load Image*

Form Load Image adalah *form* yang melakukan pengambilan terhadap citra huruf. Pada *form* ini menggunakan beberapa fitur dari MATLAB yaitu `guidata`, `uigetfile`, `imread` serta `imshow`. Fungsi dan kegunaan fitur diuraikan pada `Tabel Lampiran 1` Lampiran B Fungsi Fitur.

4.2.2 Implementasi *Form Preprocessing*

Form Preprocessing adalah *form* yang melakukan tahapan *Preprocessing* pada citra yang meliputi proses *resizing* dan *thinning*. Pada *form* ini menggunakan beberapa fitur dari MATLAB yaitu `guidata`, `imresize`, `rgb2gray`, `im2bw`, `bwmorph`, `imshow` serta `assignin`. Fungsi dan kegunaan fitur diuraikan pada `Tabel Lampiran 2` Lampiran B Fungsi Fitur.

4.2.3 Implementasi *Form Chain Code Extraction*

Form Chain Code adalah *form* yang melakukan proses ekstraksi citra huruf untuk mendapatkan nilai *Chain Code* serta menyimpan nilai *Chain Code* yang sudah didapat. Pada form ini menggunakan *function* utama yaitu `rantaikode()` serta beberapa fitur seperti `imread`, `guidata`, `length`, `assignin`, `num2str`, `uiputfile`, `evalin` serta `function` Fungsi dan kegunaan fitur diuraikan pada `Tabel Lampiran 3` Lampiran B Fungsi Fitur.

4.2.4 Implementasi *Form Matching*

Form Matching adalah *form* yang digunakan untuk proses pencocokan citra huruf yang diuji dengan citra huruf yang ada pada data latih. Pada *form* ini menggunakan beberapa fitur dari MATLAB yaitu `guidata`, `uigetfile`, `imread`, `imshow` serta `vec2ind`. Fungsi dan kegunaan fitur diuraikan pada `Tabel Lampiran 4` Lampiran B Fungsi Fitur.

4.3 Pengujian Hasil Implementasi Perangkat Lunak

Pada aplikasi pengenalan huruf Hijaiyah dengan algoritma *Chain Code* ini, dilakukan pengujian *black box testing* yang berfungsi untuk menguji setiap fungsionalitas aplikasi yang telah dibuat. Tujuan dari proses pengujian ini yaitu untuk melihat apakah aplikasi yang dibuat sudah sesuai dengan fungsionalitas yang sudah dibuat pada bagian perancangan. Proses pengujian dilakukan berdasarkan tabel skenario pengujian yang dijelaskan pada sub bab 4.3.1 sampai dengan sub bab 4.3.6.

4.3.1 Pengujian Fungsionalitas *Load Image*

Proses pengujian untuk fungsionalitas *Load Image* dijelaskan pada `Tabel 16`.

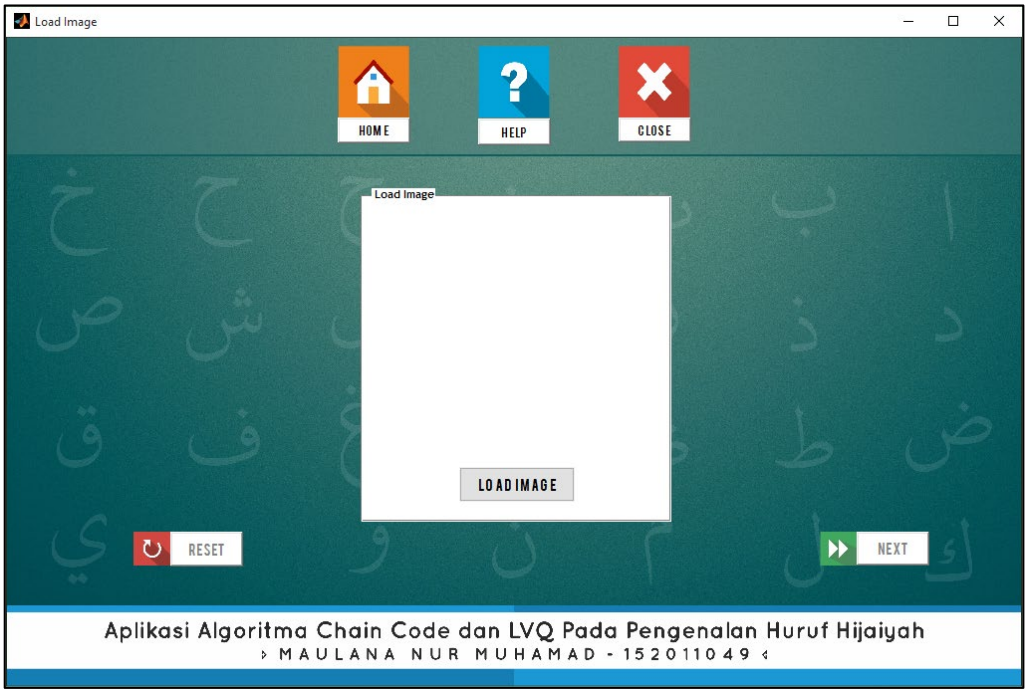
`Tabel 16. Pengujian Fungsi Load Image`

IDENTIFIKASI	
Nomor	TIH-01
Nama Butir Uji	*Load Image*
Tujuan	Memilih citra yang dimasukkan ke data latih dan data uji
Deskripsi	*User* menekan tombol *Load Image* untuk memilih citra yang dimasukkan pada data latih atau data uji.
Kondisi Awal	- Aktor berada pada form *Load Image* - *Axes* untuk citra yang dipilih, masih dalam keadaan kosong
PENGUJIAN	
Skenario Uji	

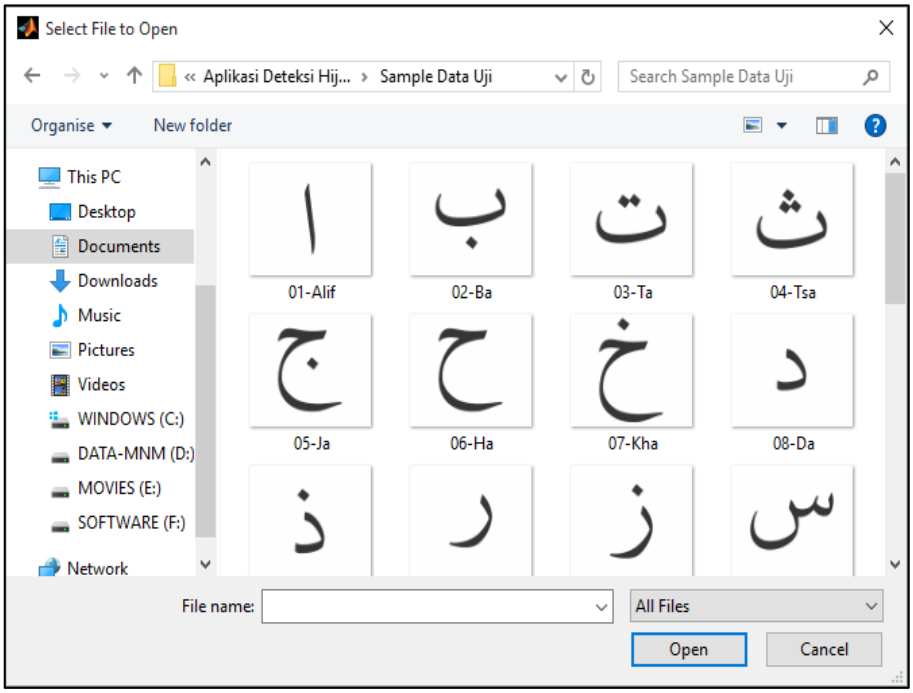
Tabel 16. Pengujian Fungsi Load Image (Lanjutan)

1. Menekan tombol *Load Image* 2. Memilih citra yang dimasukkan ke data latih atau citra yang diuji 3. Menekan tombol *Next* untuk masuk ke *form* berikutnya			
Kriteria Evaluasi Uji			
Sistem menampilkan citra yang dipilih pada *Axes*.			
Kasus dan Hasil Uji			
Masukan	Harapan	Pengamatan	Kesimpulan
Citra huruf Hijaiyah	1. Apabila *User* menekan tombol *Load Image*, sistem menampilkan *dialog box* lokasi *path* untuk memilih citra yang dimasukkan. 2. Sistem menampilkan citra yang dipilih pada *Axes.* 3. Apabila *User* menekan tombol *Next*, maka tampil *form* berikutnya (*Form Preprocessing*).	1. Apabila *User* menekan tombol *Load Image*, sistem menampilkan *dialog box* lokasi *path* untuk memilih citra yang dimasukkan 2. Sistem menampilkan citra yang dipilih pada *Axes.* 3. Apabila *User* menekan tombol *Next*, maka tampil *form* berikutnya (*Form Preprocessing*).	**[X] Terima** **[ ] Tolak**

Berdasarkan hasil pengujian yang dilakukan pengguna terhadap butir uji *Load Image* dengan mengikuti skenario yang dinyatakan pada Tabel 16 diperoleh hasil pengujian seperti yang ditunjukkan pada Gambar 34, Gambar 35 dan Gambar 36.



Gambar 34. Hasil Pengujian Load Image (1)



Gambar 35. Hasil Pengujian Load Image (2)

Gambar 36. Hasil Pengujian Load Image (3)

4.3.2 Pengujian Fungsionalitas *Preprocessing*

Proses pengujian untuk fungsionalitas *Preprocessing* dijelaskan pada Tabel 17.

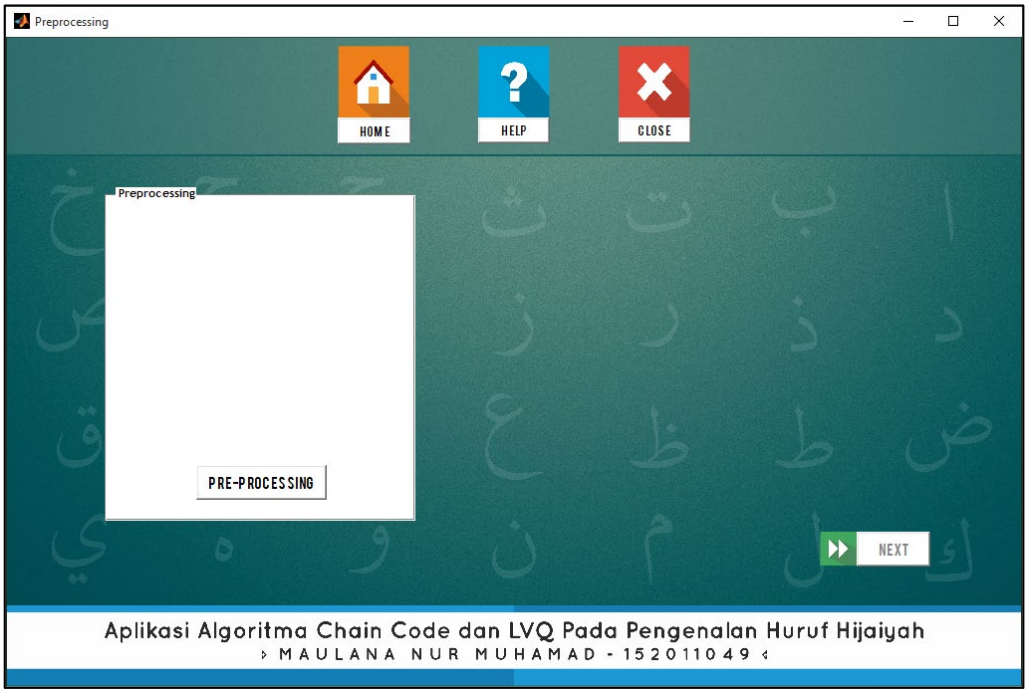
Tabel 17. Pengujian Fungsi Preprocessing

IDENTIFIKASI	
Nomor	TIH-02
Nama Butir Uji	*Preprocessing Image*
Tujuan	Menampilkan hasil dari tahap *Preprocessing* yang terdiri dari proses *resizing* dan *thinning*.
Deskripsi	*User* menekan tombol *Preprocessing* untuk mendapatkan hasil citra yang telah dilakukan proses *resizing* dan *thinning*. Sistem juga menampilkan nilai biner dari citra yang sudah melalui proses *resizing* dan *thinning*.
Kondisi Awal	- Aktor berada pada *form Preprocessing* - *Axes* masih dalam keadaan kosong, serta tabel hasil binerisasi belum ditampilkan
PENGUJIAN	
Skenario Uji	

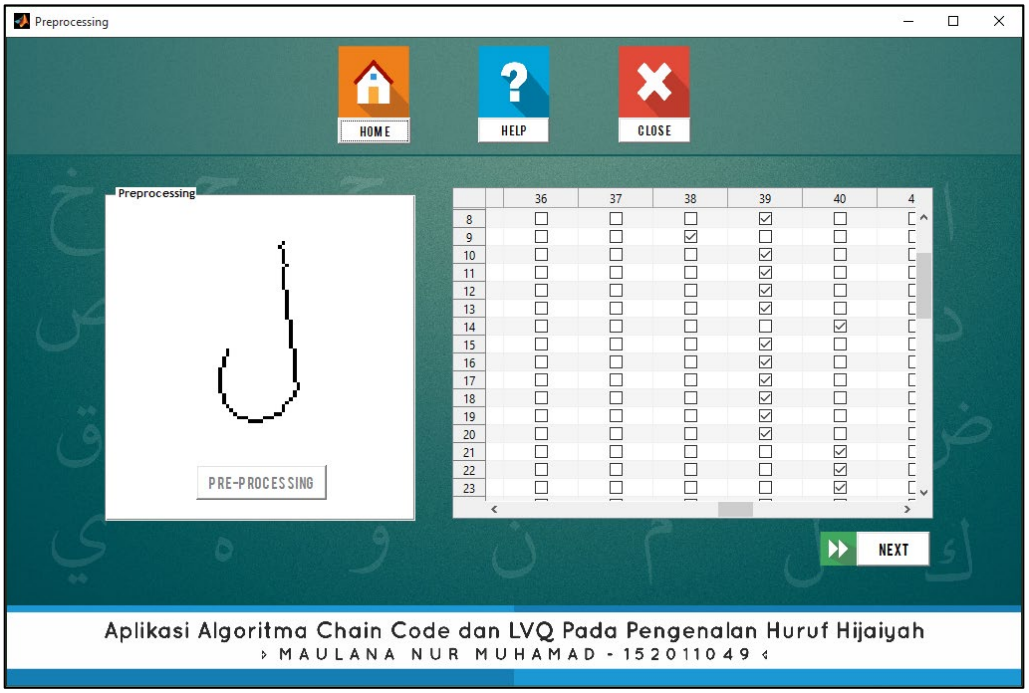
Tabel 17. Pengujian Fungsi Preprocessing (Lanjutan)

1. Menekan tombol *Preprocessing* 2. Menekan tombol *Next* untuk masuk ke *form* berikutnya			
Kriteria Evaluasi Uji			
Sistem menampilkan hasil *resizing* dan *thinning* pada *Axes* serta nilai binerisasi vektor citra pada tabel.			
Kasus dan Hasil Uji			
Masukan	Harapan	Pengamatan	Kesimpulan
Citra huruf Hijaiyah yang sudah dipilih	1. Apabila *User* menekan tombol *Preprocessing*, maka sistem menampilkan citra yang telah di *resizing* dan *thinning* pada *Axes* serta sistem menampilkan nilai biner dari citra tersebut. 2. Apabila *User* menekan tombol *Next*, maka tampil *form* berikutnya (*Form Chain Code Extraction*)	1. Apabila *User* menekan tombol *Preprocessing*, maka sistem menampilkan citra yang telah di *resizing* dan *thinning* pada *Axes* serta sistem menampilkan nilai biner dari citra tersebut. 4. Apabila *User* menekan tombol *Next*, maka tampil *form* berikutnya (*Form Chain Code Extraction*)	**[X] Terima** **[ ] Tolak**

Berdasarkan hasil pengujian yang dilakukan pengguna terhadap butir uji *Preprocessing* dengan mengikuti skenario yang dinyatakan pada Tabel 17 diperoleh hasil pengujian seperti yang ditunjukkan pada Gambar 37 dan Gambar 38.



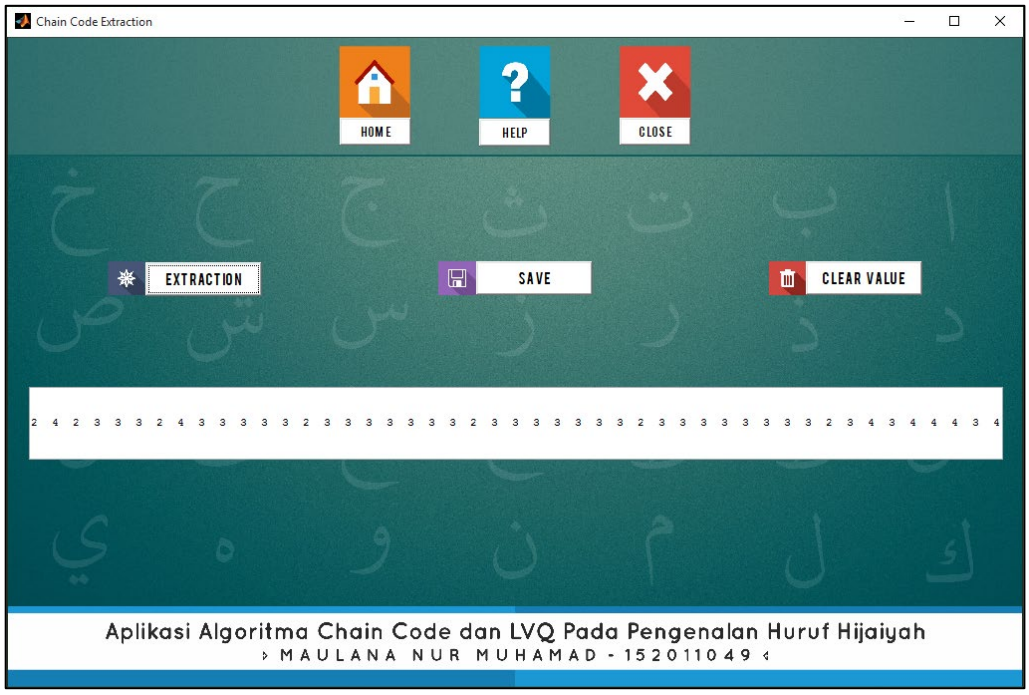
Gambar 37. Hasil Pengujian Preprocessing (1)



Gambar 38. Hasil Pengujian Preprocessing (2)

Berdasarkan hasil pengujian yang dilakukan pengguna terhadap butir uji *Chain Code Extraction* dengan mengikuti skenario yang dinyatakan pada Tabel 18 diperoleh hasil pengujian seperti yang ditunjukkan pada Gambar 39 dan Gambar 40.

Gambar 39. Hasil Pengujian Chain Code Extraction (1)



Gambar 40. Hasil Pengujian Chain Code Extraction (2)

4.3.4 Pengujian Fungsionalitas *Save Chain Code Value*

Proses pengujian untuk fungsionalitas *Save Chain Code Value* dijelaskan pada Tabel 19.

Tabel 19. Pengujian Fungsi Save Chain Code Value

IDENTIFIKASI			
Nomor	TIH-04		
Nama Butir Uji	*Save Chain Code Value*		
Tujuan	Aktor dapat menyimpan nilai *Chain Code* dari citra yang dipilih ke basis pengetahuan.		
Deskripsi	*User* menekan tombol *Save* untuk menyimpan nilai vektor *Chain Code* dari citra huruf		
Kondisi Awal	- Aktor berada pada form *Chain Code Extraction* - Nilai *Chain Code* sudah berhasil di tampilkan oleh sistem		
PENGUJIAN			
Skenario Uji			
1. Menekan tombol *Save* 2. Memasukkan nama file *Chain Code*			
Kriteria Evaluasi Uji			
Sistem menyimpan nilai *Chain Code* citra huruf ke dalam basis pengetahuan.			
Kasus dan Hasil Uji			
Masukan	Harapan	Pengamatan	Kesimpulan
Nilai *Chain Code* dari citra yang dipilih	1. Apabila *User* menekan *Save*, sistem menampilkan *dialog box* lokasi *path* untuk menyimpan nilai *Chain Code*. 2. *User* memasuk-kan nama *file* nilai *Chain Code* tersebut. 3. Sistem menyimpan nilai *Chain Code*.	1. Apabila *User* menekan *Save*, sistem menampilkan *dialog box* lokasi *path* untuk menyimpan nilai *Chain Code*. 2. *User* memasuk-kan nama *file* nilai *Chain Code* tersebut. 3. Sistem menyimpan nilai *Chain Code*.	**[X] Terima** **[ ] Tolak**

4.3.5 Pengujian Fungsionalitas *Matching*

Proses pengujian untuk fungsionalitas *Matching* dijelaskan pada Tabel 20.

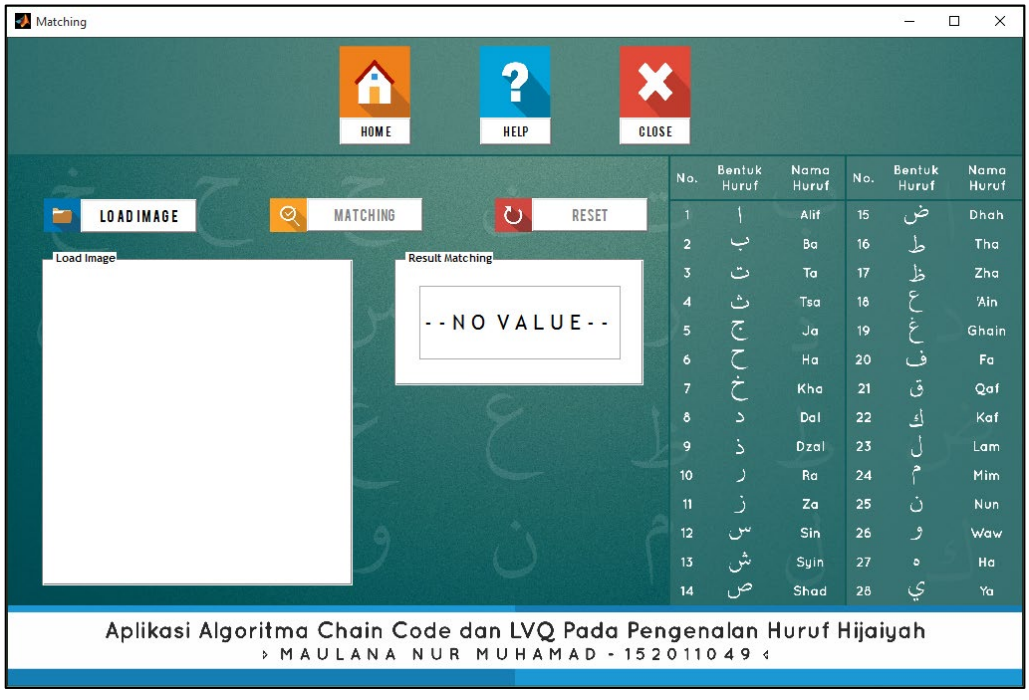
Tabel 20. Pengujian Fungsi Matching

IDENTIFIKASI			
Nomor	TIH-05		
Nama Butir Uji	*Matching*		
Tujuan	Mencari pola huruf yang sesuai antara citra huruf yang ada pada data latih dengan citra huruf yang ada pada data uji.		
Deskripsi	*User* memilih citra huruf yang diuji. Lalu *User* menekan tombol *Matching* untuk mendapatkan pola huruf yang sesuai.		
Kondisi Awal	- Aktor berada pada form *Matching* - *Axes* serta keterangan kecocokan huruf dalam keadaan kosong		
PENGUJIAN			
Skenario Uji			
1. Menekan tombol *Load Image* 2. Menekan tombol *Matching* 3. Menekan tombol *Reset*			
Kriteria Evaluasi Uji			
Sistem menampilkan pola huruf yang sesuai dengan yang ada pada data latih.			
Kasus dan Hasil Uji			
Masukan	Harapan	Pengamatan	Kesimpulan

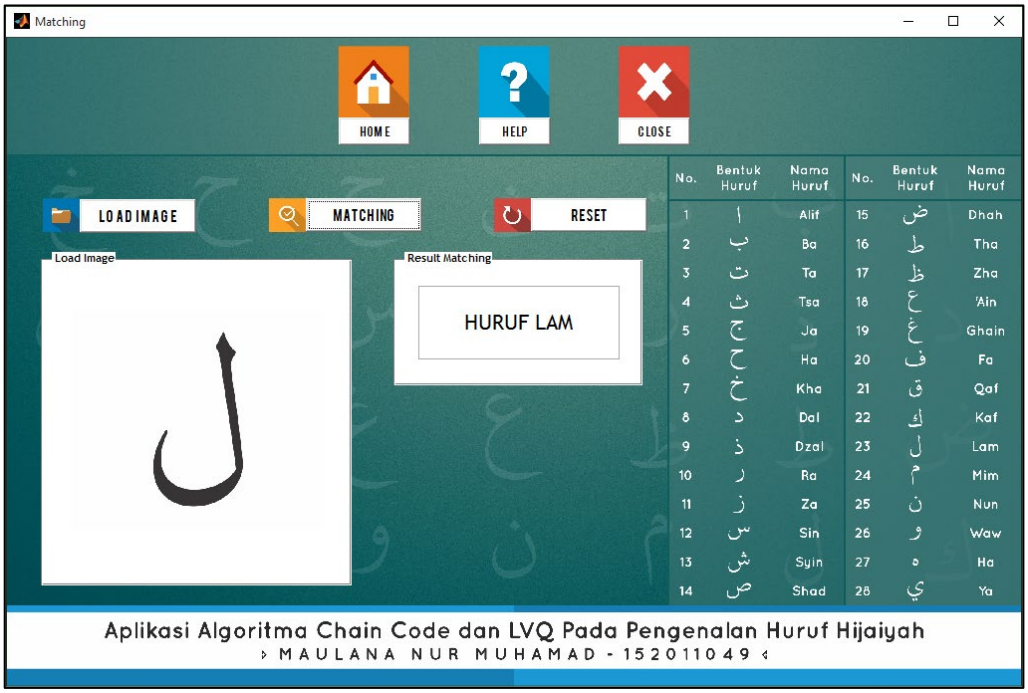
Tabel 20. Pengujian Fungsi Matching (Lanjutan)

Citra huruf Hijaiyah yang sudah dipilih	1. Apabila *User* menekan tombol *Load Image*, sistem menampilkan *dialog box* lokasi *path* untuk memilih citra yang dimasukkan. 2. Sistem menampilkan citra yang dipilih pada *Axes*. 3. Apabila *User* menekan tombol *Matching*, sistem menampilkan pola huruf yang sesuai.	1. Apabila *User* menekan tombol *Load Image*, sistem menampilkan *dialog box* lokasi *path* untuk memilih citra yang dimasukkan. 2. Sistem menampilkan citra yang dipilih pada *Axes*. 3. Apabila *User* menekan tombol *Matching*, sistem menampilkan pola huruf yang sesuai.	**[X] Terima** **[ ] Tolak**

Berdasarkan hasil pengujian yang dilakukan pengguna terhadap butir uji *Matching* dengan mengikuti skenario yang dinyatakan pada Tabel 20 diperoleh hasil pengujian seperti yang ditunjukkan pada Gambar 41 dan Gambar 42.



Gambar 41. Hasil Pengujian Matching (1)



Gambar 42. Hasil Pengujian Matching (2)

4.3.6 Pengujian Fungsionalitas *Help*

Proses pengujian untuk fungsionalitas *Help* dijelaskan pada Tabel 21.

Tabel 21. Pengujian Fungsi Help

IDENTIFIKASI	
Nomor	TIH-06
Nama Butir Uji	*Help*
Tujuan	Aktor dapat mengetahui cara kerja aplikasi serta informasi mengenai aplikasi.
Deskripsi	Sistem menampilkan *form* bantuan ketika *User* menekan tombol *Help*.
Kondisi Awal	Aktor berada pada form aplikasi
PENGUJIAN	
Skenario Uji	
Menekan tombol *Help*	
Kriteria Evaluasi Uji	
Sistem menampilkan form *Help*.	
Kasus dan Hasil Uji	

Tabel 21. Pengujian Fungsi Help(Lanjutan)

Masukan	Harapan	Pengamatan	Kesimpulan
	Apabila *User* menekan tombol *Help*, sistem menampilkan *form Help* yang berisi mengenai cara kerja aplikasi.	Apabila *User* menekan tombol *Help*, sistem menampilkan *form Help* yang berisi mengenai cara kerja aplikasi.	**[X] Terima** **[ ] Tolak**

Berdasarkan hasil pengujian yang dilakukan pengguna terhadap butir uji *Help* dengan mengikuti skenario yang dinyatakan pada Tabel 21 diperoleh hasil pengujian seperti yang ditunjukkan pada Gambar 43.

Gambar 43. Hasil Pengujian Help

4.4 Hasil Pengujian

Pengujian dilakukan dengan mencocokkan 28 citra huruf Hijaiyah pada data latih dengan jenis huruf KFFQPC Uthman Taha Naskh dan ukuran huruf 120 pt dengan 28 citra huruf Hijaiyah pada data uji.

Berdasarkan hasil pengujian pada fungsionalitas *Matching* dengan mengikuti skenario yang dinyatakan pada Tabel 20, diperoleh hasil pencocokan seperti yang ditunjukkan pada Tabel 22.

Tabel 22. Tabel Hasil Pengujian

No.	Nama Huruf Uji	Bentuk Huruf Uji	Hasil Kecocokan
1	Alif	ا	Tidak Cocok
2	Ba	ب	Cocok
3	Ta	ت	Tidak Cocok
4	Tsa	ث	Cocok
5	Jim	ج	Cocok
6	Ha	ح	Tidak Cocok
7	Kha	خ	Cocok
8	Dal	د	Cocok
9	Dzal	ذ	Cocok
10	Ra	ر	Tidak Cocok
11	Za	ز	Cocok
12	Sin	س	Cocok
13	Syin	ش	Cocok
14	Shad	ص	Cocok
15	Dha	ض	Cocok
16	Tha	ط	Cocok
17	Zha	ظ	Tidak Cocok
18	‘Ain	ع	Cocok
19	Gha	غ	Cocok
20	Fa	ف	Cocok
21	Qaf	ق	Cocok
22	Kaf	ك	Cocok
23	Lam	ل	Cocok
24	Mim	م	Tidak Cocok
25	Nun	ن	Cocok
26	Waw	و	Cocok
27	Haa	ه	Cocok
28	Ya	ي	Cocok

Dari Tabel 22 diatas, terdapat sebanyak enam citra huruf uji yang ‘Tidak Cocok’ dengan citra huruf yang terdapat pada data latih. Hal tersebut disebabkan karena kedekatan nilai bobot dari citra huruf uji tidak mendekati citra huruf latih. Nilai bobot citra huruf uji yang tidak cocok tersebut lebih dekat dengan nilai bobot citra huruf latih lainnya.

Untuk menghitung tingkat kesuksesan citra uji yaitu dengan menggunakan Rumus 4.

$$Hasil = \frac{hasil\ yang\ sesuai}{total\ citra\ yang\ diuji} \times 100 \quad \text{..... (4)}$$

Maka didapat tingkat kesuksesan (%) untuk 28 citra huruf Hijaiyah yang diuji yaitu :

$$Hasil = \frac{22}{28} \times 100 = 78{,}5\ \%$$

Maka metode LVQ memasukkan citra huruf uji yang tidak cocok tersebut ke dalam kelas citra huruf yang memiliki kedekatan nilai bobotnya. Dari hasil pengujian, didapat kedekatan nilai bobot citra uji dengan citra latih yang direpresentasikan ke dalam kelas seperti pada Tabel 23.

Tabel 23. Tabel Kedekatan Nilai Bobot Citra Uji dan Citra Latih

No.	Nama Huruf Uji	Bentuk Huruf Uji	Jarak terdekat dengan bobot kelas ke -	Hasil Identifikasi	Hasil Kecocokan
1	Alif	ا	8	Dal (د)	Tidak Cocok
2	Ba	ب	2	Ba (ب)	Cocok
3	Ta	ت	2	Ba (ب)	Tidak Cocok
4	Tsa	ث	4	Tsa (ث)	Cocok
5	Jim	ج	5	Jim (ج)	Cocok
6	Ha	ح	5	Jim (ج)	Tidak Cocok
7	Kha	خ	7	Kha (خ)	Cocok
8	Dal	د	8	Dal (د)	Cocok
9	Dzal	ذ	9	Dzal (ذ)	Cocok
10	Ra	ر	8	Dal (د)	Tidak Cocok
11	Za	ز	11	Za (ز)	Cocok
12	Sin	س	12	Sin (س)	Cocok
13	Syin	ش	13	Syin (ش)	Cocok
14	Shad	ص	14	Shad (ص)	Cocok
15	Dha	ض	15	Dha (ض)	Cocok
16	Tha	ط	16	Tha (ط)	Cocok
17	Zha	ظ	16	Tha (ط)	Tidak Cocok
18	‘Ain	ع	18	‘Ain (ع)	Cocok
19	Gha	غ	19	Gha (غ)	Cocok
20	Fa	ف	20	Fa (ف)	Cocok
21	Qaf	ق	21	Qaf (ق)	Cocok

Tabel 23. Tabel Kedekatan Nilai Bobot Citra Uji dan Citra Latih

No.	Nama Huruf Uji	Bentuk Huruf Uji	Jarak terdekat dengan bobot kelas ke -	Hasil Identifikasi	Hasil Kecocokan
22	Kaf	ك	22	Kaf (ك)	Cocok
23	Lam	ل	23	Lam (ل)	Cocok
24	Mim	م	18	‘Ain (ع)	Tidak Cocok
25	Nun	ن	25	Nun (ن)	Cocok
26	Waw	و	26	Waw (و)	Cocok
27	Haa	ه	27	Haa (ه)	Cocok
28	Ya	ي	28	Ya (ي)	Cocok

BAB V
PENUTUP

Pada bagian ini menjelaskan mengenai kesimpulan dari penelitian yang dilakukan serta dari sistem yang telah dibuat.

5.1 Kesimpulan

Berdasarkan hasil penelitian serta pengujian sistem yang telah dilakukan, maka dapat disimpulkan bahwa tingkat kesuksesan pengenalan huruf Hijaiyah untuk jenis *font* KFGQPC Uthman Taha Naskh dengan ukuran *font* 120 pt. mencapai 78,5%. Algoritma *Chain Code* yang diterapkan dalam ekstraksi ciri mampu mengenali struktur pembentuk dari citra yang dipilih. Namun dalam tahap pencocokan huruf, algoritma LVQ hanya mampu mengenali 22 huruf dari 28 huruf yang masuk ke dalam data latih.

DAFTAR PUSTAKA

[1] Abdurohim, Acep Iim. (2003). *Pedoman Ilmu Tajwid Lengkap*. Bandung: CV Penerbit Dipenogoro.

[2] Acharya Tinku, Ray Ajoy K. (2005). *Image Processing, Principle and Application*. New Jersey: John Wiley & Sons, Inc.

[3] Budi Wirayuda, Tjokorda, Vaulin Syilvia, Novi Dayawati, R. (2009) *Pengenalan Huruf Komputer Menggunakan Algortima Berbasis Chain Code dan Sequence Alignment.* Fakultas Teknik Informatika, IT Telkom Bandung.

[4] H. Izakian, S. A. Monadjemi, B. Tork Ladani, and K. Zamanifar,. (2008). *Multi-Font Farsi/Arabic Isolated Character Recognition Using Chain Codes.* Proceeding of World Academy of Science, Engineering and Technology Volume 33 September 2008, ISSN 2070-3740

[5] Kadir, Abdul., Adhi Susanto. (2013). *Teori dan Aplikasi Pengolahan Citra*. Yogyakarta: Andi Publisher.

[6] Kusumadewi, Sri. (2003). *Artificial Intelligence (Teknik dan Aplikasinya).* Yogyakarta: Graha Ilmu.

[7] Maula Akhmad Robit, dkk. *Optical Character Recognition Dengan Metode Naïve Bayes*.

[8] Munir, R. (2004). *Pengolahan Citra Digital dengan Pendekatan Algoritmik*. Bandung: Informatika.

[9] Shalahuddin, M., Rosa A.S. (2013). *Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Objek*. Bandung: Informatika.

[10] Sofian Bahri, Irfan Maliki. (2012). *Perbandingan Algoritma Template Matching dan Feature Extraction Pada Optical Character Recognition*. Fakultas Teknik dan Ilmu Komputer Universitas Komputer Indonesia Bandung.

[11] _____. *Mengenal dan Memahami Arti Huruf Hijaiyah.* Online. (http://visiuniversal.blogspot.com/2014/07/mengenal-dan-memahami-arti-huruf.html, diakses pada 18 April 2015)

[12] _____. *Angka Penting dan Pengolahan Data.* Online. (http://sitrampil.ui.ac.id/elaboratory/file.php/1/Angka_Penting_dan_Pengolahan_Data.pdf, diakses pada 27 April 2015)

[13] _____. *Bahasa Arab, Bahasa Dunia.* Online. (http://bit.ly/1P73ug2, diakses pada 7 Mei 2015)

LAMPIRAN

LAMPIRAN

A

DAFTAR HURUF HIJAIYAH

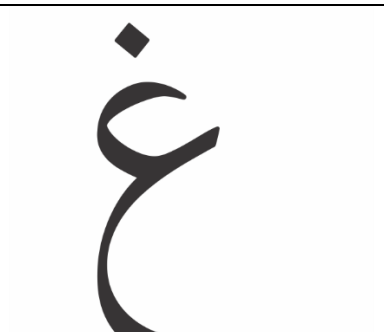
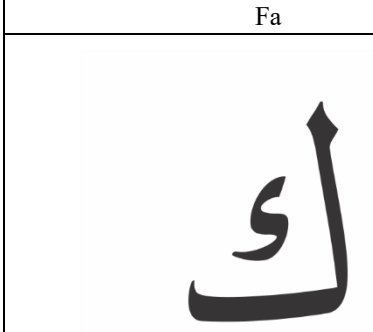
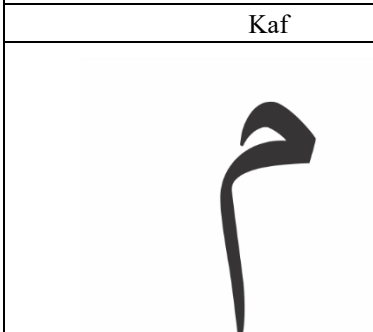
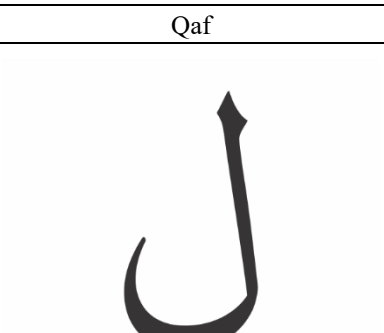
Tabel 1 Lampiran A. Daftar Huruf Hijaiyah

Daftar Huruf Hijaiyah (Ukuran Font 120pt)	
	
Ba	Alif
	
Tsa	Ta
	
Ha	Jim
	
Dal	Kha

Tabel 1 Lampiran A. Daftar Huruf Hijaiyah

Daftar Huruf Hijaiyah (Ukuran Font 120pt)	
	
Ra	Dzal
	
Sin	Zal
	
Shad	Syin
	
Tha	Dha

Tabel 1 Lampiran A. Daftar Huruf Hijaiyah

Daftar Huruf Hijaiyah (Ukuran Font 120pt)	
	
‘Ain	Zha
	
Fa	Gha
	
Kaf	Qaf
	
Mim	Lam

Tabel 1 Lampiran A. Daftar Huruf Hijaiyah

Daftar Huruf Hijaiyah (Ukuran Font 120pt)	
	
Waw	Nun
	
Ya	Haa

LAMPIRAN
B
FUNGSI FITUR

Tabel Lampiran 1. Fitur Form Load Image

No.	Nama Fitur	Keterangan
1	guidata	Meyimpan atau mengambil UI data
2	uigetfile	Membuka kotak dialog untuk memilih data
3	imread	Membaca gambar dari file grafis
4	imshow	Menampilkan gambar dari file grafis

Tabel Lampiran 2. Fitur Form Preprocessing

No.	Nama Fitur	Keterangan
1	guidata	Meyimpan atau mengambil UI data
2	imresize	Mengubah ukuran gambar
3	rgb2gray	Mengkonversi gambar RGB atau colormap ke *grayscale*
4	im2bw	Mengubah gambar ke gambar biner, berdasarkan *threshold*
5	bwmorph	Operasi morfologi pada gambar biner
6	imshow	Menampilkan gambar
7	assignin	Menetapkan nilai ke variabel dalam *workspace* yang ditentukan

Tabel Lampiran 3. Fitur Form Chain Code

No.	Nama Fitur	Keterangan
1	imread	Membaca gambar dari file grafis
2	guidata	Meyimpan atau mengambil UI data

Tabel Lampiran 3. Fitur Form Chain Code

No.	Nama Fitur	Keterangan
3	`length`	Panjang dari dimensi array
4	`assignin`	Menetapkan nilai ke variabel dalam *workspace* yang ditentukan
5	`num2str`	Mengkonversi nilai ke dalam bentuk string
6	`uiputfile`	Membuka kotak dialog untuk menyimpan data
7	`evalin`	Menjalankan ekspresi MATLAB yang terdapat pada workspace
8	`rantaikode()`	Fungsi untuk mendapatkan nilai *chain code*

Tabel Lampiran 4. Fitur Form Matching

No.	Nama Fitur	Keterangan
1	`guidata`	Meyimpan atau mengambil UI data
2	`uigetfile`	Membuka kotak dialog untuk memilih data
3	`imread`	Membaca gambar dari file grafis
4	`imshow`	Menampilkan gambar dari file grafis
5	`vec2ind`	Mengkonversi vektor ke dalam indeks