IDENTIFIKASI KOMBINASI POLA ISYARAT TANGAN UNTUK MENJALANKAN PERINTAH PADA SEBUAH APLIKASI KOMPUTER

SKRIPSI

JURUSAN TEKNIK ELEKTRO

Diajukan Untuk Memenuhi Sebagian Persyaratan

Memperoleh Gelar Sarjana Teknik

Disusun Oleh:

QUDSI SETYAWAN

NIM. 0710633054-63

KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN

UNIVERSITAS BRAWIJAYA

FAKULTAS TEKNIK

MALANG

2012

LEMBAR PERSETUJUAN

IDENTIFIKASI KOMBINASI POLA ISYARAT TANGAN UNTUK MENJALANKAN PERINTAH PADA SEBUAH APLIKASI KOMPUTER

SKRIPSI

JURUSAN TEKNIK ELEKTRO

Diajukan Untuk Memenuhi Sebagian Persyaratan
Memperoleh Gelar Sarjana Teknik

Disusun oleh:

QUDSI SETYAWAN

NIM. 0710633054-63

Telah diperiksa dan disetujui oleh :

Dosen Pembimbing I	**Dosen Pembimbing II**
Ir. Muhammad Aswin, MT.	**Adharul Muttaqin, ST., MT.**
NIP. 19640626 199002 1 001	**NIP. 19760121 200501 1 001**

LEMBAR PENGESAHAN

IDENTIFIKASI KOMBINASI POLA ISYARAT TANGAN UNTUK MENJALANKAN PERINTAH PADA SEBUAH APLIKASI KOMPUTER

SKRIPSI

JURUSAN TEKNIK ELEKTRO

Diajukan Untuk Memenuhi Sebagian Persyaratan

Memperoleh Gelar Sarjana Teknik

Disusun oleh:

QUDSI SETYAWAN

NIM. 0710633054-63

Skripsi ini telah diuji dan dinyatakan lulus pada

tanggal 10 Agustus 2012

DOSEN PENGUJI

A Abdul Razak, ST., MT., M.Eng.	**R. Arief Setyawan, ST., MT.**
NIP. -	**NIP. 19760121 200501 1 001**

Waru Djuriatno, ST., MT.

NIP. 19690725 199702 1 001

Mengetahui

Ketua Jurusan Teknik Elektro

Dr. Ir. Sholeh Hadi Pramono, MS.

NIP. 19580728 198701 1 001

PENGANTAR

Alhamdulillah, segenap puji dan syukur penulis panjatkan kepada Allah SWT yang telah melimpahkan rahmat, hidayah, dan karunia-Nya sehingga penulis dapat menyelesaikan skripsi dengan judul "Identifikasi Kombinasi Pola Isyarat Tangan Untuk Menjalankan Perintah Sebuah Aplikasi Komputer" yang diajukan untuk memenuhi sebagian persyaratan memperoleh gelar Sarjana Teknik di Jurusan Teknik Elektro Universitas Brawijaya.

Penulis ingin mengucapkan terima kasih yang sebesar-besarnya kepada berbagai pihak yang telah membantu dan mendukung dalam penyelesaian skripsi ini, yaitu :

- Bapak Dr. Ir. Sholeh Hadi Pramono, MS. selaku Ketua Jurusan dan bapak Muhammad Aziz Muslim, ST., MT., PhD. selaku Sekretaris Jurusan Teknik Elektro Universitas Brawijaya,
- Ibu Rusmi Ambarwati, ST., MT., selaku dosen penasehat akademik yang telah memberikan nasehat, arahan, dan motivasi selama proses akademik penulis,
- Bapak Waru Djuriatno, ST., MT., selaku Ketua Kelompok Dosen Keahlian Rekayasa Komputer Jurusan Teknik Elektro Universitas Brawijaya,
- Bapak Ir. Muhammad Aswin., MT., dan Bapak Adharul Muttaqin, ST., MT., selaku Dosen Pembimbing atas segala bimbingan, nasehat, pengarahan, motivasi, saran dan masukan yang telah diberikan dalam pengerjaan skripsi,
- Keluarga di rumah, Dampit. Ibu Asri Indayani, Bapak Susilo Prasetyo, ketiga adikku Qalbi N.S., Qozein A.S., Qomarudin S. atas segala nasehat, kasih sayang, perhatian dan kesabarannya serta telah banyak mendoakan kelancaran penulis hingga terselesaikannya skripsi ini,
- Audylia Dhianti, SE. dan keluarga yang telah memberikan semangat, dukungan, doa dan bantuannya selama menyelesaikan skripsi ini.
- Tito, Wiro (Sableng), Fanni (Unyil), Wildan (Gitok), Gallant, Abdur, Boby, Ichwan (Gendheng), Khasbullah, Yoneth, Erez, Adam dan CORE 2007 yang telah memberikan bantuan dan motivasi yang banyak selama menyelesaikan skripsi ini.
- Agan, Bayu, yang merupakan sahabat kecil penulis dan temen – temen di Dampit. Terima kasih atas dukungan dan doanya.

Dalam penulisan skripsi ini, penulis menyadari bahwa skripsi ini masih belum sempurna. Oleh karena itu, penulis sangat mengharapkan kritik dan saran yang membangun untuk kelengkapan dan kesempurnaan skripsi ini. Penulis berharap semoga skripsi ini dapat bermanfaat khususnya bagi rekan-rekan mahasiswa.

Malang, Agustus 2012

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

ABSTRAK

QUDSI SETYAWAN, Jurusan Teknik Elektro, Fakultas Teknik, Universitas Brawijaya, 2012***, "Identifikasi Kombinasi Pola Isyarat Tangan Untuk Menjalankan Perintah Sebuah Aplikasi Komputer",*** Dosen Pembimbing: **Ir. Muhammad Aswin, MT.** dan **Adharul Muttaqin, ST., MT.**

Tangan adalah organ utama untuk memanipulasi lingkungan fisik, digunakan untuk keterampilan motorik kasar (seperti menggenggam sebuah objek besar) dan keterampilan motorik halus (seperti mengambil kerikil kecil). Beberapa hasil penelitian membuktikan bahwa pola tangan bisa dipakai sebagai sumber data. Pada suatu pola tangan dapat membentuk suatu tanda atau kode sehingga dapat dijadikan sebagai kode. Karena setiap jenis pola tangan memiliki karakteristik yang berbeda. Pada skripsi ini dibuat suatu program aplikasi yang dapat mengenali pola isyarat tangan sebagai kode perintah yang dapat digunakan untuk pengendali suatu alat menggunakan media webcam. Pada studi kasus ini digunakan pola tangan dengan pola lebih dari satu sebagai objek identifikasi. Dengan demikian dapat digunakan sebagai alternatif berupa kode kombinasi isyarat tangan untuk menjalankan perintah sebuah aplikasi. Pada penelitian ini menggunakan integral proyeksi sebagai nilai fitur yang dipakai oleh Jaringan Syaraf Tiruan *Learning Vector Quantization (LVQ)* yang memiliki kemampuan untuk melatih pola-pola sebagai nilai masukan untuk data yang dilatih dengan demikian dapat dikembangkan aplikasi untuk mengidentifikasi pola isyarat tangan. Hasil dari proses aplikasi ini adalah informasi objek berupa hasil dari kombinasi yang diinginkan. Dari semua citra objek yang diujikan diperoleh prosentase keberhasilan dalam mengidentifikasi yang berbeda-beda tergantung pada proses pelatihan yang dilakukan. Prosentase keberhasilan tertinggi sebesar 80% menggunakan pelatihan dengan *Epoh* 30, *Learning Rate* 0,6, dan *Decreasing Learning Rate* 0,1.

Kata kunci: *isyarat tangan, Epoh, Learning Rate, Decreasing Learning Rate*.

BAB I
PENDAHULUAN

1.1 Latar Belakang

Komputer merupakan alat yang dipakai untuk mengolah data menurut prosedur yang dirumuskan. Perkembangan teknologi komputer yang pesat serta kebutuhan akan komputer itu sendiri semakin tinggi. Hal ini memudahkan manusia untuk melakukan pekerjaannya menjadi lebih cepat. Perangkat pendukung komputer ini juga semakin banyak dikembangkan. Dengan perkembangan teknologi komputer yang sangat pesat, maka *Personal Computer (PC)* kini dapat meniru sistem kerja dari jaringan syaraf biologi atau sistem syaraf manusia untuk diaplikasikan pada suatu jaringan syaraf tiruan (JST). Sehingga dapat membuat program yang memggunakan JST memiliki akurasi yang tinggi yang sesuai dengan apa yang diinginkan penggunanya.

Berbagai ciri yang melekat pada manusia dan relatif mudah dilihat secara langsung seperti pola sidik jari, bentuk wajah, pola telapak tangan (palm print), bentuk telapak tangan hingga yang perlu peralatan khusus seperti iris mata, pola pembuluh darah, dan distribusi suhu tubuh, telah dimanfaatkan sebagai data primer yang mampu merepresentasikan identitas seseorang. Beberapa hasil penelitian membuktikan bahwa pola telapak tangan bisa dipakai sebagai sumber data.

Kamera atau *webcam* merupakan salah satu perangkat komputer yang dewasa ini sering dipakai dalam *image processing. Webcam* ini digunakan untuk merekam atau mencuplik citra sesuai apa yang kita inginkan, sehingga wabcam merupakan salah satu alat yang wajib untuk bisa menyelesaikan masalah pada *Image Processing.*

Dalam kehidupan sehari-hari kita sudah banyak melihat macam-macam alat yang digunakan untuk memproteksi suatu benda atau ruangan. Contohnya seperti menggunakan kunci, scanner fingerprint, atau tombol kode *password.*

Berdasarkan alasan itulah maka penulis mencoba merancang dan membuat sebuah sistem alternatif berupa alat identifikasi kombinasi pola isyarat tangan yang fungsinya untuk keamanan seperti tombol kode *password* dan fingerprint tetapi dengan pola isyarat tangan menggunakan *webcam*.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dipaparkan diatas, maka rumusan masalah ditekankan pada:

1. Bagaimana melakukan ekstrasi ciri dari citra referensi dan pengelompokan sesuai dengan kelompoknya menggunakan metode *Learning Vector Quantization.*
2. Bagaimana mengenali pola isyarat tangan manusia dibandingkan dengan citra referensi menggunakan *template matching* dengan algoritma *Euclidean distance*.

1.3 Batasan Masalah

Beberapa hal yang menjadi batasan-batasan dalam pembuatan program ini adalah sebagai berikut :

1. Tempat pengambilan gambar sudah disediakan.
2. Jarak antara kamera dengan tangan berjarak 30cm.
3. Dalam satu frame hanya terdapat satu isyarat tangan dan dilakukan dengan tangan kanan.
4. Isyarat tangan yang digunakan adalah isyarat satu tangan dengan pola tertentu yang telah ditentukan sebelumnya.
5. Isyarat tangan dibatasi tangan orang dewasa.
6. Isyarat tangan yang diambil oleh kamera berupa isyarat diam.
7. Membutuhkan pencahayaan yang konstan.
8. Latar belakang obyek dibatasi berwarna putih.
9. Aplikasi berupa software untuk membuka menggunakan kode kombinasi.
10. Aplikasi tidak membahas sistem mekaniknya.

1.4 Tujuan

Tujuan penyusunan skripsi ini adalah :

Merancang dan membangun suatu aplikasi software yang dapat mengenali pola isyarat tangan sebagai kode perintah yang dapat digunakan untuk pengendali suatu alat yang memberikan perintah membuka menggunakan media *webcam*.

1.5 Sistematika Penulisan

Penulisan skripsi dibagi dalam tujuh bab dengan sistematika penulisan sebagai berikut :

BAB 1 PENDAHULUAN

Memuat latar belakang, rumusan masalah, batasan masalah, tujuan penulisan, manfaat dan sistematika pembahasan.

BAB II TINJAUAN PUSTAKA

Membahas kajian pustaka dan dasar teori yang digunakan pada skripsi ini.

BAB III METODE PENELITIAN

Membahas metode yang digunakan dalam skripsi ini serta langkah-langkah yang diambil.

BAB IV PERANCANGAN DAN IMPLEMENTASI

Menjelaskan langkah langkah perancangan aplikasi dan pengidentifikasian objek beserta penjelasan algoritma yang digunakan.

BAB V PENGUJIAN DAN ANALISIS

Pengujian perangkat lunak dan analisis hasil pengujian.

BAB VI PENUTUP

Memuat kesimpulan yang diperoleh serta saran untuk pengembangan lebih lanjut.

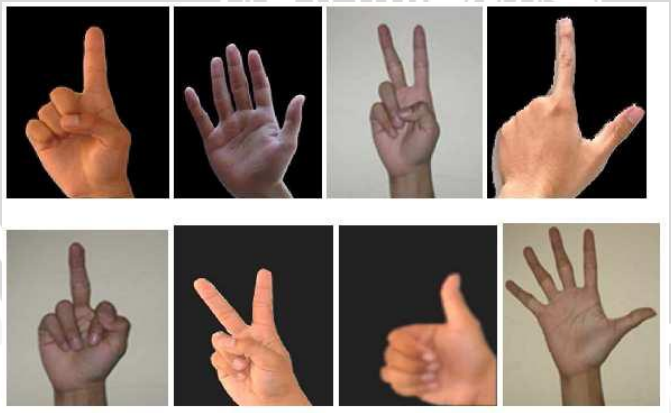
BAB II
TINJAUAN PUSTAKA

2.1 Tangan

Tangan adalah organ utama untuk memanipulasi lingkungan fisik, digunakan untuk keterampilan motorik kasar (seperti menggenggam sebuah objek besar) dan keterampilan motorik halus (seperti mengambil kerikil kecil). Ujung jari berisi beberapa daerah terpadat ujung saraf pada tubuh dan memiliki kemampuan posisi terbesar dari tubuh, sehingga rasa sentuh sangat erat berhubungan dengan tangan. Seperti halnya organ tubuh pasangan lainnya (mata, kaki, kaki), masing-masing tangan dominan dikendalikan oleh belahan otak yang berlawanan, sehingga orientasi, atau pilihan yang lebih disukai untuk tangan-tangan tunggal kegiatan seperti menulis dengan pena, mencerminkan fungsi otak individu.

Tangan adalah bagian tubuh di ujung suatu lengan. Sebagian besar manusia memiliki dua tangan, biasanya dengan empat jari dan satu ibu jari. Bagian dalam tangan adalah telapak tangan. Ibu jari disebut juga jari jempol. Selain itu keempat jari yakni jari telunjuk, jari tengah, jari manis dan jari kelingking.

Isyarat tangan adalah merupakan sistem unik komunikasi yang efektif menyampaikan informasi dasar yang diperlukan untuk melakukan suatu pola isyarat tangan dalam hal membentuk suatu tanda atau kode sehingga dapat dijadikan komunikasi.



Gambar 2.1 Isyarat tangan

2.2 Citra Digital

Citra digital terdiri dari pixels (picture element). Setiap piksel merepresentasikan warna atau tingkat kebuan pada satu titik di dalam citra. Citra digital dapat dilihat sebagai fungsi kontinyu f(x,y) yang berada pada bidang dua dimensi dimana (x,y) merupakan koordinat spasial dari suatu titik sedangkan nilai f(x,y) merupakan intensitas cahaya pada titik koordinat tersebut. Citra digital dapat diperoleh dari proses pencuplikan objek tiga dimensi dan membentuk suatu matriks dimana setiap elemennya menyatakan intensitas cahaya. Citra digital dapat dihasilkan dari penangkapan objek menggunakan kamera digital, sensor, scaner atau perekam lainya yang menghasilkan data format raster.

Citra digital tersusun dalam bentuk raster (grid atau kisi). Titik dalam suatu citra disebut piksel (picture element). Nilai x pada titik koordinat (x,y) merupakan sumbu mendatar (horisontal) yang menunjukkan kolom dari suatu piksel dalam citra sedangkan y (sumbu vertikal) menunjukkan baris dari suatu piksel. Setiap piksel memiliki nilai yang menunjukkan tingkat intensitas keabuan dari piksel itu sendiri.

Citra digital memilliki resolusi yang menunjukkan tingkat kerincian suatu citra. Resolusi dapat dinyatakan dengan banyaknya piksel per satuan panjang atau sering disebut dengan piksel per inci (dot per inci – dpi). Semakin besar nilai dpi yang dimiliki oleh suatu citra, maka resolusinya akan semakin tinggi. Nilai resolusi juga dapat dinyatakan dengan satuan panjang, misalnya 120 x 100 m.

Citra digital dianggap matrik dengan ukuran M x N dimana baris dan kolom menunjukkan titik-titiknya. Citra berwarna menggunakan metode RGB. Adapun masing-masing warna dalam tabel memiliki 3 buah kombinasi angka yaitu R, G, dan B yang menentukan proporsi warna merah, warna hijau, dan warna biru dari warna tersebut. RGB masing-masing memiliki range antara 0 hingga 63 sehingga jumlah warna yang dapat kita pilih untuk mengisi warna pada tiap cell ditabel adalah 63x63x63= 16 juta warna. Tetapi seluruh tabel hanya dapat diisi dengan 256 pilihan warna. Kita dapat mengubah intensitas warna dari sebuah warna pada tabel dengan cara menggunakan *interupt-interupt*.

Citra digital didefinisikan sebagai fungsi *f(x,y)* berukuran M baris dan N kolom, Citra digital dapat ditulis dalam bentuk matrik sebagai berikut.

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,N-1) \\ f(1,0) & f(1,1) & \cdots & f(1,N-1) \\ \vdots & \vdots & & \vdots \\ f(M-1,0) & f(M-1,1) & \cdots & f(M-1,N-1) \end{bmatrix}$$

dengan:

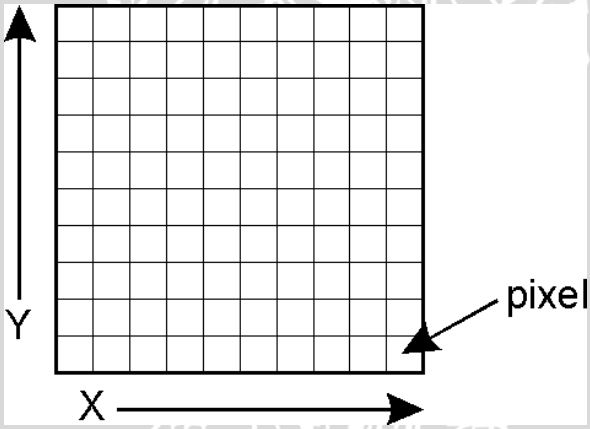
f(x, y) = intensitas pixel pada posisi x dan y

M = jumlah kolom

N = jumlah baris

Nilai pada suatu irisan antara baris dan kolom (pada posisi x,y) disebut dengan pixel

Pixel adalah unsur gambar atau representasi sebuah titik terkecil dalam sebuah gambar grafis yang dihitung per inci. Pixel sendiri berasal dari akronim bahasa Inggris (Picture Element) yang disingkat menjadi Pixel. Pada ujung tertinggi skala resolusi, mesin cetak gambar berwarna dapat menghasilkan hasil cetak yang memiliki lebih dari 2.500titik per inci denga pilihan 16 juta warna lebih untuk setiap inci, dalam istilah komputer berarti gambar seluas satu inci persegi yang bisa ditampilkan pada tingkat resolusi tersebut sepadan dengan 150 juta bit informasi.



Gambar 2.2 *Posisi pixel*

2.3 Jenis Citra Digital

2.3.1. Citra RGB

RGB adalah suatu model warna yang terdiri dari tiga warna primer yaitu merah, hijau, dan biru. Ketiga warna tersebut digabung dan membentuk suatu susunan warna yang luas. Setiap warna dasar memiliki rentang-nilai dari 0 hingga 255. Sehingga gabungan dari ketiga warna tersebut akan menghasilkan warna campuran sebanyak 256 x 256 x 256 = 1677726. Satu jenis warna dapat digambarkan sebagai sebuah vektor di ruang tiga dimensi yang memiliki komponen R, komponen G, dan komponen

B. Jadi, sebuah jenis warna dapat dituliskan sebagai berikut: warna =RGB(58, 38, 100). Nilai putih = RGB (255,255,255), sedangkan untuk hitam=RGB(0,0,0).

2.3.2 Citra Grayscale

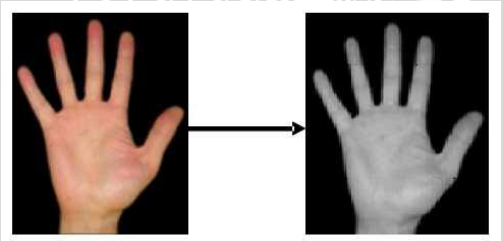
Citra beraras keabuan (gray-scale) adalah citra yang memiliki warna dalam rentang gradasi antara hitam dan putih. Rentang warna yang dimiliki antara 0 hingga 255. Proses awal yang banyak dilakukan dalam image processing adalah mengubah citra berwarna menjadi citra gray-scale, hal ini digunakan untuk menyederhanakan model citra. Pada awalnya citra terdiri dari 3 layer matrik yaitu R-layer, G-layer dan B-layer. Sehingga untuk melakukan proses-proses selanjutnya tetap diperhatikan tiga layer di atas. Bila setiap proses perhitungan dilakukan menggunakan tiga layer, berarti dilakukan tiga perhitungan yang sama. Sehingga konsep itu diubah dengan mengubah 3 layer di atas menjadi 1 layer matrik gray scale dan hasilnya adalah citra gray-scale. Dalam citra ini tidak ada lagi warna, yang ada adalah derajat keabuan.

Untuk mengubah citra berwarna yang mempunyai nilai matrik masing-masing r, g dan b menjadi citra gray scale dengan nilai s, maka konversi dapat dilakukan dengan mengambil rata-rata dari nilai r, g dan b sehingga dapat dituliskan menjadi:

$$S = 0{,}299R + 0{,}587G + 0{,}114B$$

Atau metode lainnya dengan menghitung nilai rata-rata dari RGB itu sendiri.

$$S = (R + G + B) / 3$$



Gambar 2.3 Grayscale image

2.3.3 Thresholding

Thresholding digunakan untuk mengubah citra dengan format skala keabuan ke citra biner, yang hanya memiliki nilai (0 atau 1). Dalam hal ini, titik dengan nilai rentang nilai keabuan tertentu diubah menjadi berwarna hitam dan sisanya menjadi putih, atau sebaliknya.

Salah satu dari dua operasi pengambangan yang banyak digunakan adalah pengambangan tunggal. Pengambangan tunggal memiliki sebuah nilai batas ambang. Untuk menetukan nilai ambang dengan persamaan:

$$T = \frac{f_{maks} + f_{\min}}{2}$$

Dengan ketentuan :

T = nilai ambang

f_{maks} = nilai intensitas maksimum pada citra

$f_{\min}$ = nilai intensitas minimum pada citra

Apabila nilai ambang sudah ditemukan maka warna *pixel* akan digantikan dengan warna hitam atau putih tergantung kondisi berikut :

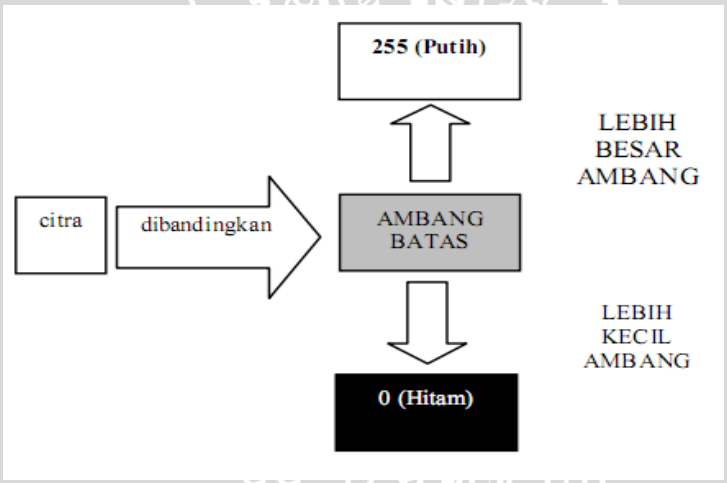
$f(x,y) = 255$, jika $f(x,y) \geq T$

$f(x,y) = 0$, jika $f(x,y) \leq T$

dengan :

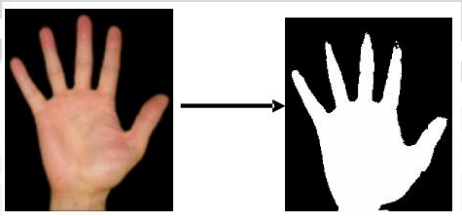
$f(x,y)$ = nilai intensitas *pixel* pada posisi (x,y)

T = nilai ambang



Gambar 2.4 Ilustrasi pengambilan keputusan pada proses *thresholding*

Tujuan utama dari proses *thresholding* ini adalah untuk memisahkan dan membedakan antara objek dengan *background* (latar belakang) suatu citra.

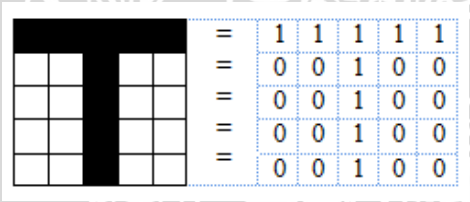


Gambar 2.5 Konversi citra Thresholding

2.3.4 Citra Biner

Pengolahan citra digital menangani data yang besar pada komputer, ini berarti membutuhkan memori yang jauh lebih besar untuk penyimpanan dan meletakkan bingkai citra dalam memori baik sebelum, selama dan sesudah proses pengolahan berlangsung. Kebutuhan memori dapat ditekan bila citra ditangani dengan citra *biner*. Citra *biner* ini dicapai untuk pembentukan citra adalah dua, yang berarti citra hanya mengandung informasi hitam dan putih sebagai penyusunnya. Dengan menggunakan citra biner dapat memudahkan otak dan mata manusia mengerti makna citra siluet (bayangan, yang juga hanya mengandung dua macam informasi yaitu objek dan latar belakang dengan tingkat intensitas yang berlawanan). Citra biner biasanya digunakan sebagai penyederhanaan dengan memperhatikan ada atau tidak.

Pada umumnya pengolahan citra biner digunakan sebagai penyederhanaan data saat pemrosesan. Saat pengolahan data menggunakan citra biner memudahkan proses pemisahan antara obyek dengan latar belakang yang umumnya memiliki perbedaan intensitas piksel yang besar. Hal ini biasanya digunakan dalam membedakan antara objek dan *background* yang dikehendaki.



Gambar 2.6 Contoh citra biner

2.4 Pengolahan Citra

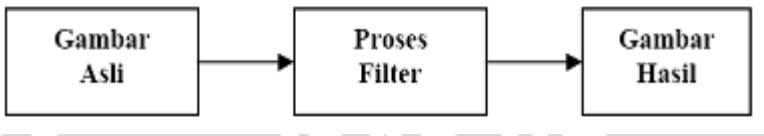
Kegiatan memperbaiki kualitas citra agar mudah diinterpretasi oleh manusia/mesin(komputer). Inputannya adalah citra dan keluarannya juga citra tapi dengan kualitas lebih baik daripada citra masukan → misal citra warnanya kurang tajam, kabur (blurring), mengandung noise (misal bintik-bintik putih), dll sehingga perlu ada pemrosesan untuk memperbaiki citra karena citra tersebut menjadi sulit diinterpretasikan karena informasi yang disampaikan menjadi berkurang.

Dalam pengolahan citra, dilakukan operasi terhadap citra asli menjadi citra baru berdasarkan citra asli. Operasi yang dilakukan pada citra dikategorikan sebagai berikut :

1. Point, yaitu operasi yang menghasilkan output dimana setiap pixel hanya dipengaruhi oleh pixel pada posisi yang sama dari citra asli.

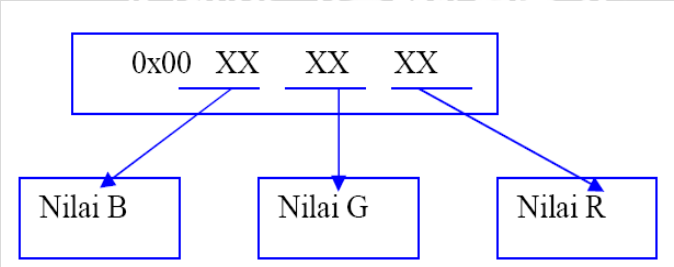
2. Local, yaitu operasi yang menghasilkan output dimana pixelnya dipengaruhi oleh pixel-pixel tetangga pada citra asli.

3. Global, yaitu opeasi yang menghasilkan output dimana pixelnya dipengaruhi oleh semua pixel yag ada dalam citra asli. Misalnya ada suatu gambar yang terlalu gelap maka dengan image processing gambar tersebut bisa diproses sehingga mendapatkan gambar yang jelas. Secara garis besar bisa gambarkan seperti blok diagram pada gambar dibawah ini:



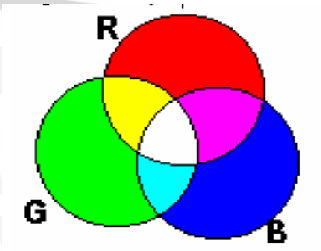
Gambar 2.7 Blok diagram pengolah citra

Prinsip dasar pengolahan citra adalah pengolahan warna RGB pada posisi tertentu. Dalam pengolahan citra warna direpresentasikan dengan nilai hexadesimal dari 0x00000000 sampai 0x00ffffff. Definisi nilai warna tersebut berisikan variabel 0x00 yang menyatakan angka dibelakangnya adalah hexadesimal. Seperti yang dijelaskan dalam gambar 2.8.



Gambar 2.8 Nilai warna RGB dalam hexadecimal

Terlihat bahwa setiap warna mempunyai range nilai 00 (angka desimal bernilai 0) sampai ff (angka desinal bernilai 255) atau mempunyai derajat keabu-abuan 256 = 2^8, dengan demikian range warna RGB adalah (2^8) (2^8) (2^8) = 2^{24} atau dikenal dalam windows dengan istilah True Colour. Nilai warna yang digunakan merupakan gabungan warna cahaya merah, hijau, biru. Jadi untuk menyajikan warna tertentu dapat dengan mudah dilakukan dengan cara mencampur ketiga warna dasar RGB.



Gambar 2.9 Komposisi warna RGB

2.5 Ekstraksi Fitur

Ekstraksi fitur merupakan pengambilan fitur atau ciri pada objek melalui proses tertentu. Ciri atau fitur merupakan karakteristik unik dari suatu objek. Karakteristik fitur yang baik sebisa mungkin memenuhi persyaratan beikut.

a. Dapat membedakan suatu objek dengan yang lainnya.
b. Memperhatikan kompleksitas komputasi dalam memperoleh fitur. Kompleksitas yang tinggi akan menjadi beban tersendiri dalam menemukan fitur.
c. Tidak terikat dalam arti bersifat invarian terhadap berbagai transformasi.
d. Jumlahnya sedikit.

Ekstraksi fitur dikatagorikan menjadi 3 level, yaitu level rendah (*low-level*), level medium (*middle-level*), dan level tinggi (*high-level*). *Low-level* fitur merupakan ekstraksi ciri berdasarkan fitur visual seperti warna dan tekstur. *Middle-level* fitur merupakan ekstraksi berdasarkan wilayah citra yang ditentukan dengan segmentasi. High-level fitur merupakan ekstraksi ciri berdasarkan informasi semantik yang terkandung dalam citra. Untuk memperoleh suatu fitur atau ciri dari citra diperlukan metode analisis citra.

2.5.1 Integral Proyeksi

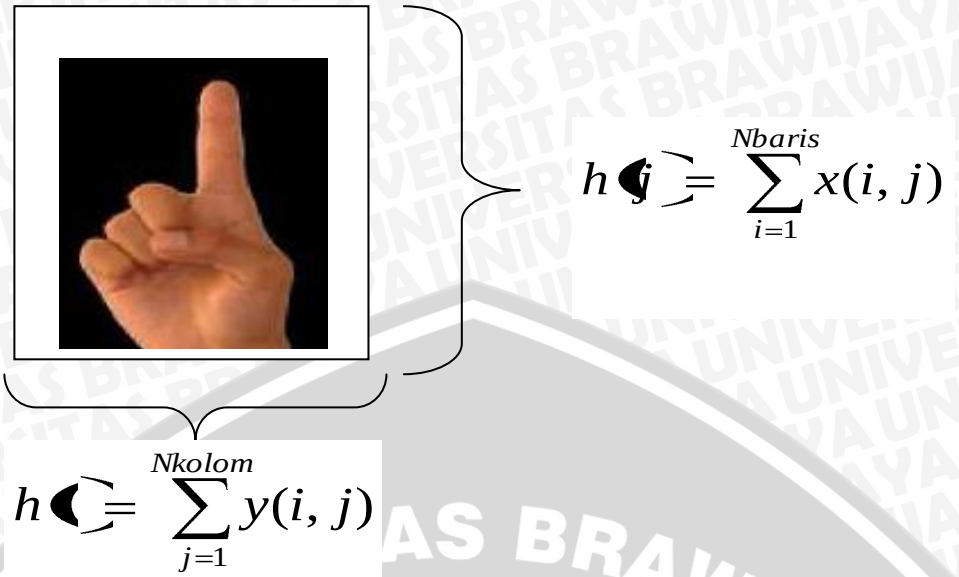
Integral Proyeksi adalah suatu teknik yang menjumlahkan nilai setiap kolom atau setiap baris. Secara detail,image merupakan sekumpulan titik-titik (*pixel*) yang membentuk deret matrix NxM,atau dapat difungsikan sebagai $f(x,y)$. Jadi pada dasarnya hal tersebut terbentuk dari sederetan informasi pada sumbu X dan Y. dengan metode Integral Proyeksi ini kita dapat mencari informasi dalam bentuk apa pun,dalam hal ini kita batasi jumlah kolom dan baris.

$$h(j) = \sum_{i=1}^{Nbaris} x(i, j)$$

$$h(i) = \sum_{j=1}^{Nkolom} x(i, j)$$

Gambar 2.10 Definisi Integral proyeksi

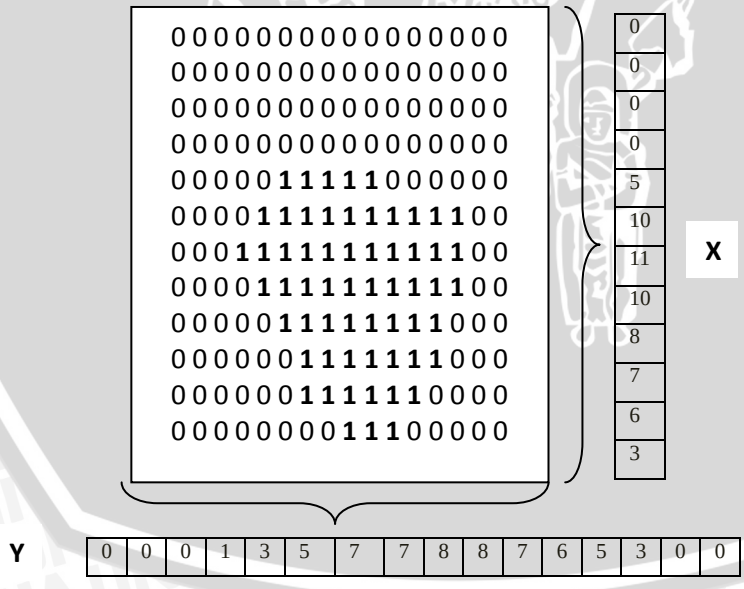
Sedangkan bagaimana implementasi Integral Proyeksi dalam image ini dapat diilustrasikan sebagai berikut:



Gambar 2.11 Ilustrasi pengambilan Informasi Fitur dengan Integral Proyeksi

Berdasarkan gambar tangan di atas. Gambar tersebut discan berdasarkan kolom-Y dan kolom-X. Kemudian kita dapat informasi berupa sederetan nilai yaitu h(j) dan h(i). Kedua matrix tersebut dijadikan satu dalam matrik Fitur{h(j) dan h(i). Hasil dari fitur itulah yang dijadikan ciri dari masing-masing citra.

Berikut pola citra yang dijadikan ciri menggunakan integral proyeksi.



Gambar 2.12 pola 0

Dari contoh pola di atas,maka didapatkan fitur citranya berupa:

1. Pola 0, x(baris) { 0, 0, 0, 0, 5, 10, 11, 10, 8, 7, 6, 3}

   Y(kolom) { 0, 0, 0, 1, 3, 10, 11, 8, 8, 8, 7, 6, 5, 3, 0, 0}

2.6 Pengenalan Pola

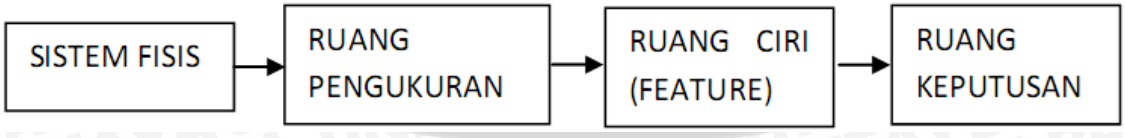
Sebuah pola adalah setiap antar hubungan data (analog atau digital), kejadian atau konsep yang dapat dibedakan. Pengenalan pola merupakan bidang dalam pembelajaran mesin dan dapat diartikan sebagai tindakan mengambil data mentah dan bertindak berdasarkan klasifikasi data. Dengan demikian, pengenalan pola merupakan himpunan kaidah bagi pembelajaran (supervised learning).

Ada beberapa definisi lain tentang pengenalan pola, di antaranya:

1. Penentuan suatu objek fisik atau kejadian ke dalam salah satu atau beberapa kategori.
2. Ilmu pengetahuan yang menitikberatkan pada deskripsi dan klasifikasi (pengenalan) dari suatu pengukuran.
3. Suatu pengenalan secara otomatis suatu bentuk, sifat, keadaan, kondisi, susunan tanpa keikutsertaan manusia secara aktif dalam proses pemutusan.

Berdasarkan beberapa definisi di atas, pengenalan pola dapat didefinisikan sebagai cabang kecerdasan yang menitik-beratkan pada metode pengklasifikasian objek ke dalam kelas-kelas tertentu untuk menyelesaikan masalah tertentu. Salah satu aplikasinya adalah pengenalan suara, klasifikasi teks dokumen dalam kategori (contoh. surat-E spam/bukan-spam), pengenalan tulisan tangan, pengenalan kode pos secara otomatis pada sampul surat, atau sistem pengenalan wajah manusia. Aplikasi ini kebanyakan menggunakan analisis citra bagi pengenalan pola yang berkenaan dengan citra digital sebagai input ke dalam sistem pengenalan pola.

Pengenalan pola komputer dapat dipandang sebagai tugas yang berisikan ajar (learning) perilaku-perilaku invarian dan lazim dari sekumpulan sampel yang mencirikan sebuah kelas, dan memutuskan sebuah sampel baru. Langkah pengoperasian yang perlu dalam mengembangkan serta melaksanakan aturan keputusan dalam sistem pengenalan pola praktis, ditunjuk dalam blok-blok sebagai berikut:



Gambar 2.13 Tahap Pengoperasian Suatu sistem pengenalan pola

Sebuah sistem fisis untuk tujuan pengenalan pola ditandai oleh beberapa perwujudan fisisnya yang dinyatakan secara numerik yang membentuk ruang pengukuran. Pemilihan dan ekstraksi feature dalam pengenalan pola merupakan proses pemilihan sebuah pemetaan bentuk X = f(Y) yang berasal dari sampel Y(y1, y2,...., yQ)

dalam ruang dimensi yang ditransformasi ke suatu titik X (x1, x2,...., xN). Fungsi f(Y) akan meminimumkan jarak dan memaksimumkan jarak dalam ruang feature. Proses penurunan sebuah aturan keputusan berdasarkan sekumpulan sampel untuk mengklasifikasi suatu titik dalam ruang feature terhadap sampel.

2.7 Jaringan Saraf Tiruan

Jaringan saraf tiruan didefinisikan sebagai suatu sistem pemrosesan informasi yang memiliki karakteristik menyerupai jaringan saraf manusia. Jaringan saraf tiruan merupakan paradigma pemrosesan informasi yang terinspirasi oleh sistem sel saraf biologi seperti otak yang memproses suatu informasi. Jaringan saraf buatan mencoba mensimulasikan proses pembelajaran pada otak manusia. JST diimplementasikan dengan menggunakan program komputer yang mampu menyelesaikan proses perhitungan pada saat pembelajaran.

Otak manusia terdiri dari kurang lebih 100.000.000.000 sel saraf yang bertugas memproses informasi. Setiap sel saraf bekerja seperti prosesor sederhana. Masing-masing sel tersebut saling berinteraksi sehingga mendukung kemampuan kerja otak. Setiap satu sel saraf (neuron) memiliki satu inti sel. Inti sel bertugas memproses informasi. Informasi yang datang akan diterima oleh dendrit sedangkan hasil pemrosesan akan diteruskan oleh axon untuk menjadi inputan bagi sel saraf yang lain. Informasi yang dikirimkan antar neuron berupa rangsangan yang dilewatkan melalui dendrit. Dendrit kedua sel tersebut dipertemukan dengan synapsis. Informasi yang datang akan dijumlahkan dan dikirim melalui axon ke dendrit akhir yang bersentuhan dengan dendrit dari neuron yang lain. Informasi ini akan diterima oleh neuron yang lain jika mencapai batas tertentu(threshold). Pada kasus seperti ini neuron tersebut dikatakan teraktivasi. Hubungan antar neuron terjadi secara adaptif.

Jaringan saraf tiruan tercipta sebagai suatu generalisasi model matematis dari pemahaman manusia yang didasarkan atas asumsi berikut:

1. Pemrosesan informasi terjadi pada elemen sederhana yang disebut dengan neuron.
2. Sinyal mengalir di antara sel saraf/neuron melalui suatu penghantar/penghubung.
3. Setiap penghubung memiliki bobot yang bersesuaian. Bobot ini akan digunakan untuk mengalikan/menggandakan sinyal yang dikirim melaluinya.

4. Setiap sel menerapkan fungsi aktivasi terhadap sinyal hasil penjumlahan berbobot yang masuk kepadanya untuk menentukan sinyal keluarannya.

Jaringan saraf tiruan dapat belajar dari pengalaman, melakukan generalisasi atas contoh-contoh yang diperoleh dan mengabstraksi karakteristik esensial input bahkan untuk data yang tidak relevan. Algoritma jaringan saraf tiruan beroperasi langsung dengan angka sehingga data yang berbetuk tidak numerik harus diubah kedalam bentuk numerik. JST tidak diprogram untuk menghasilkan keluaran tertentu, tetapi semua keluaran atau kesimpulan yang ditarik oleh jaringan didasarkan pada pengalaman selama proses pembelajaran.

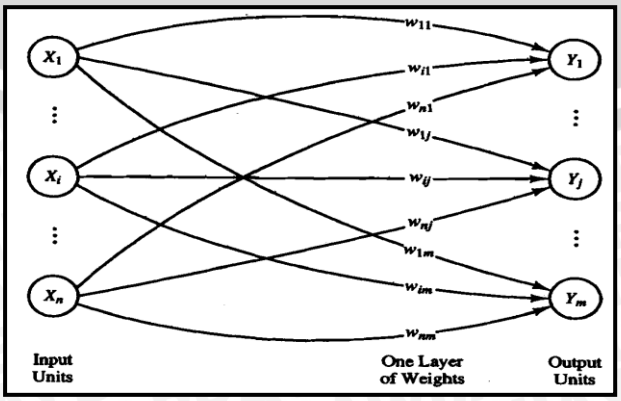
Krakteristik jaringan saraf tiruan ditentukan oleh pola hubungan antar neuron yang disebut dengan arsitektur jaringan, metode penentuan bobot-bobot sambungan yang disebut dengan pelatihan atau proses pambelajaran, dan fungsi aktifasi.

2.7.1 Arsitektur Jaringan Saraf Tiruan

Neuron-neuron pada JST akan dikumpulkan kedalam lapisan-lapisan (layers) yang disebut dengan lapisan neuron (neuron layers). Setiap neuron pada satu lapisan akan dihubungkan dengan lapisan sebelumnya dan sesudahnya. Informasi yang diberikan pada jaringan akan dirambatkan dari lapisan ke lapisan, mulai dari lapisan input hingga ke lapisan output melalui lapisan tersembunyi (hidden layers).

1. Jaringan dengan lapisan tunggal (single layer net)

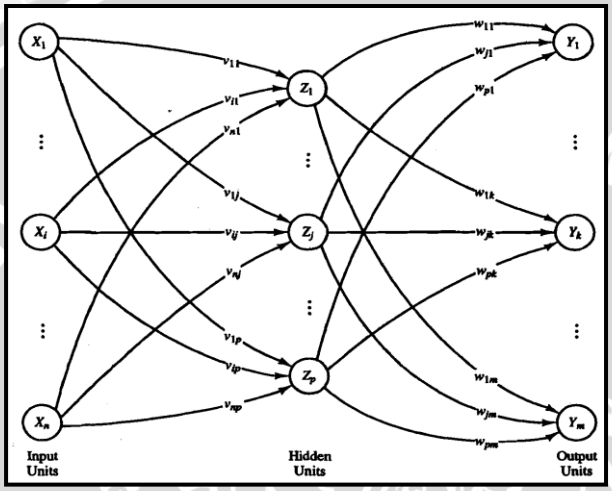
Jaringan lapisan tunggal hanya memiliki satu lapisan dengan bobot-bobot terhubung. Jaringan ini menerima input kemudian langsung mengolahnya manjadi output tanpa harus melewati lapisan tersembunyi. Besarnya hubungan antar dua neuron ditentukan berdasarkan bobot yang bersesuaian. Setiap unit input akan dihubungkan dengan unit output.



Gambar 2.14 Arsitektur lapisan tunggal

2. Jaringan dengan lapisan banyak (multilayer net)

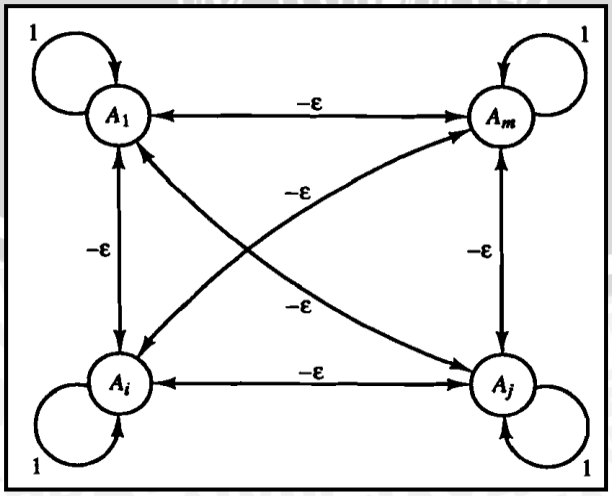
Jaringan multilayer memiliki satu atau lebih lapisan yang terletak di antara lapisan lapisan input dan lapisan output. Umumnya ada lapisan bobot-bobot yang terletak diantara dua lapisan yang bersebelahan. Multilayer net dapat menyelesaikan permasalahan yang rumit dibanding dengan jaringan dengan satu layer, tapi proses pembelajarannya juga lebih rumit. Pada banyak kasus, pembelajaran pada jaringan dengan banyak kasus memiliki tingkat keberhasilan yang lebih tinggi dalam menyelesaikan masalah.



Gambar 2.15 Arsitektur lapisan jamak

3. Jaringan dengan lapisan kompetitif (competitive layer net)

Pada jaringan kompetitif, sekumpulan neuron akan saling bersaing untuk menjadi neuron yang aktif. Umumnya hubungan antar neuron tidak diperlihatkan pada diagram arsitektur jaringan.



Gambar 2.16 Arsitektur lapisan kompetitif

2.7.2 Pembelajaran JST

Informasi yang sudah diketahui hasil keluarannya dimasukan ke dalam JST melalui node-node atau unit-unit input. Bobot-bobot antar koneksi di dalam arsitektur diberi nilai awal, kemudian proses pembelajaran dijalankan. Jaringan menggunakan bobot-bobot untuk belajar dan mengingat informasi. Pengaturan bobot dilakukan secara terus menerus dengan menggunakan criteria tertentu hingga diperoleh nilai yang diharapkan.

1. Pembelajaran terawasi (supervised learning)

Pada pembelajaran ini, kumpulan input yang digunakan sudah diketahui output-nya. Perbedaan antara keluaran aktual dengan keluaran yang diinginkan digunakan untuk mengoreksi bobot JST agar JST dapat menghasilkan jawaban yang mirip/sedekat mungkin dengan jawaban yang benar.

2. Pembelajaran tak terawasi (Unsupervised learning)

Pada pembelajaran ini, JST mengorganisasi dirinya sendiri untuk membentuk vektor input yang serupa tanpa menggunakan data atau contoh-contoh pelatihan. Struktur menggunakan dasar data atau korelasi antar pola data yang dieksplorasi.

3. Gabungan pembelajaran terawasi dan tidak terawasi (hybrid)

Menggunakan kombinasi dari kedua pembelajaran di atas. Sebagian bobot-bobotnya ditentukan dari pembelajaran terawasi dan sebagian bobot-bobot yang lain ditentukan dari pembelajaran tak terawasi.

2.7.3 Fungsi Aktivasi

Fungsi aktivasi digunakan untuk menentukan keluaran dari neuron. Merupakan fungsi yang menggambarkan hubungan antara tingkat aktivasi internal (summation function) yang mungkin terbentuk linier atau non linier. Beberapa fungsi aktivasi JST di antaranya hard limit, purelin, dan sigmoid. Yang populer digunakan adalah fungsi sigmoid yang memiliki beberapa varian : sigmoid logaritma, sigmoid biner, sigmoid bipolar, sigmoid tangent. Hard limit memberikan batasan tegas 0 atau 1, purelin memisahkan secara linier, sigmoid berupa fungsi smooth bernilai antara 0 sampai dengan 1 (bila biner) atau antara -1 sampai 1 (bila bipolar).

Jaringan saraf tiruan merupakan representasi buatan yang mencoba mensimulasikan proses pembelajaran pada otak manusia. Jaringan saraf tiruan diimplementasikan dengan menggunakan program komputer sehingga mampu menyelesaikan proses perhitungan ketika proses pembelajaran berlangsung. Jaringan

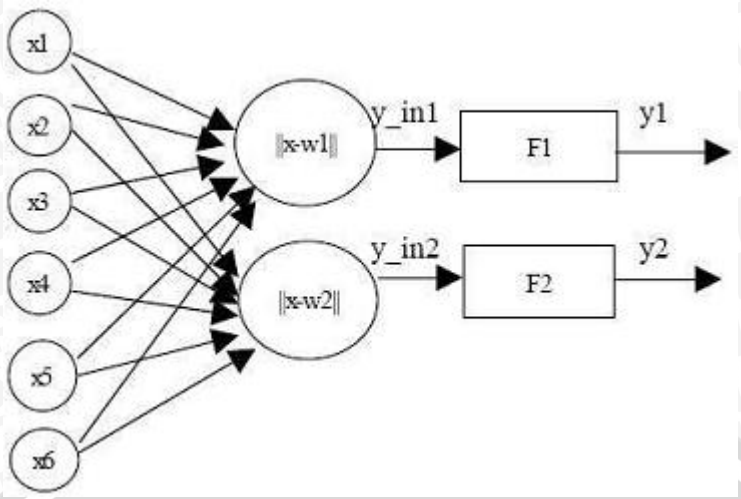
saraf tiruan tersusun atas neuron-neuron dan dendrit. Sebuah neuron berhubungan dengan neuron yang lain melalui dendrit. Jika suatu neuron menerima rangsangan berupa informasi, maka neuron tersebut akan membangkitkan output ke semua neuron yang berhubungan dengannya hingga informasi sampai ke tujuan. Pada saat proses pembelajaran akan terjadi perubahan nilai bobot yang menghubungkan antar neuron. Nilai bobot akan bertambah jika informasi yang dikirim oleh neuron tersampaikan, tapi jika informasi yang dikirim oleh sebuah neuron ke neuron yang lain tidak tersampaikan, maka nilai bobot yang menghubungkan kedua neuron akan dikurangi.

2.7.4 *Learning Rate*

Dalam pemilihan learning rate terdapat beberapa pertimbangan. Pelatihan yang lambat membutuhkan waktu yang lebih lama untuk menghasilkan sistem yang terlatih. Pelatihan yang cepat memungkan waktu yang singkat akan tetapi jaringan belum tentu menghasilkan sistem yang terlatih seperti pada pelatihan yang lambat. Fungsi pelatihan memiliki beberapa ketentuan untuk nilai learning rate. Secara umum bernilai antara 0 sampai positif 1. Jika lebih dari 1 algoritma pelatihan akan mengalami *overshoot* dalam perubahan bobot. Nilai learning rate yang kecil tidak akan memperbaiki kesalahan dengan cepat, akan tetapi memiliki kesempatan untuk mencapai konvergensi minimum terbaik.

2.7.5 Learning Vector Quantization

Learning Vector Quantization (LVQ) adalah metode untuk melakukan pembelajaran pada lapisan kompetitif yang terawasi. Suatu lapisan kompetitif akan secara otomatis belajar untuk mengklasifikasikan vektor-vektor input. Kelas yang dihasilkan berdasarkan jarak vektor tersebut. Jika ada dua vektor memiliki jarak yang cukup dekat atau mendekati sama maka kedua vektor tersebut dikelompokkan ke dalam kelas yang sama.



Gambar 2.17 Arsitektur Jaringan LVQ

Arsitektur jaringan LVQ pada dasarnya sama dengan Kohonen Self Organizing Map (tanpa struktur topologis yang diasumsikan sebagai output). Arsitekturnya terdiri dari lapisan input, lapisan kompetitif (lapisan tersembunyi/hiden layer) dan lapisan output. Masing-masing output memiliki kelas yang sebelumnya telah dibentuk dan dikenali dari hasil pelatihan jaringan LVQ ini.

2.7.6 Algoritma LVQ

Algoritma LVQ bertujuan akhir untuk mencari bobot yang sesuai untuk mengelompokkan vektor masukan kedalam kelas tujuan yang telah diinisialisasi pada saat pembentukan jaringan LVQ. Sedangkan algoritma pengujiannya adalah menghitung nilai output (kelas vektor) yang terdekat dengan vektor input atau bisa disebut dengan proses klasifikasi (pengelompokkan).

Algoritma LVQ adalah sebagai berikut:

1. Tetapkan :
   bobot(w), maksimum epoh(maxEpoh), error minimum yang diharapkan(eps), learning rate(α).
2. Masukan:
   Input : x(m,n)
   Target : T(1,n)
3. Tetapkan kondisi awal:
   epoch = 0;
   err = 1
4. Kerjakan jika : (epoh < maxEpoh) atau (α > eps)

a. epoh = epoh + 1;

b. Kerjakan untuk i = 1 sampai n

i. Tentukan j sedemikian hingga || x – wj || minimum (sebut sebagai Cj)

ii. Perbaiki wj dengan ketentuan:

jika T = Cj maka wj(baru)=wj(lama) + α (x-wj(lama))

jika T ≠ Cj maka wj(baru)=wj(lama) - α (x- wj(lama))

c. Kurangi nilai α (*learning rate*)

Keterangan:

x : vektor pelatihan(x_1,..., x_i).

T : kategori atau kelas yang tepat untuk vektor pelatihan.

w_J : bobot vektor untuk unit output J(w_{1J},..., w_{iJ},..., w_{nJ}).

C_J : kategori atau kelas yang diwakili oleh unit output J.

|| x-w_J ||: Euclidean distance antara vektor input dan bobot vektor untuk unit output ke-J

Terdapat beberapa metode dalam penentuan bobot awal dalam LVQ. Metode pertama adalah mengambil data vektor pelatihan pertama tiap kelas kemudian menggunakannya sebagai bobot vektor. Metode kedua adalah menggunakan K-means clustering dan ketiga adalah menggunakan *self-organizing map*.

2.8 Euclidean Distance

Euclidean Distance digunakan untuk mencari jarak antara dua nilai untuk mengetahui nilai kedekatan antara nilai-nilai tersebut, pengunaan distance ini dapat dilakukan jika nilai-nilai tersebut mempunyai besaran yang sama. Euclidean distance menghitung akar dari kuadrat perbedaan dua vektor. Rumus dari euclidean distance sebagai berikut:

$$d_g = \sqrt{\sum_{k=1}^{n} (x_k - y_k)^2}$$

Keterangan:

1. d_g = jarak *euclidean distance*
2. n = jumlah fitur
3. x_k = fitur data pelatihan ke k
4. y_k = fitur data pengujian ke k

2.9 Basis Data

2.9.1 Pengertian Basis Data

Basis data adalah sebuah cara mendokumentasikan berbagai macam data yang kemudian dimanajemen dengan sebuah sistem untuk kemudian disimpan dalam sebuah media penyimpanan. Dengan demikian data-data tersebut dapat diakses dengan mudah dan cepat. Media penyimpanan tersebut dapat kita ibaratkan sebuah *storage* penyimpanan, misalnya hardisk. Dalam basisdata, data yang ada tidak hanya diletakkan dan disimpan begitu saja dalam sebuah media penyimpanan, akan tetapi dikelola dengan sebuah sistem pengaturan basisdata yang seiring disebut dengan *Database Management System (DBMS)*. Dengan begitu, suatu data dengan jumlah besar dan kompleks dapat tersusun sangat baik sehingga memungkinkan pengaksesan data dengan mudah dan cepatoleh pengguna. Basisdata merupakan sekumpulan informasi yang sangat kompleks yang berguna untuk mengatur semua data yang ada di dalamnya sehingga dapat diakses oleh pengguna dengan mudah dan cepat.

2.9.2 Elemen Basis Data

1) **Entitas**

Adalah sekumpulan objek yang terdefinisikan yang mempunyai karakteristik sama dan bisa dibedakan satu dengan lainnya. Objek dapat berupa barang, orang, tempat atau suatu kejadian.

2) **Atribut**

Adalah deskripsi data yang bisa mengidentifikasi entitas yang membedakan entitas tersebut dengan entitas yang lain. Seluruh atribut harus cukup untuk menyatakan identitas obyek, atau dengan kata lain, kumpulan atribut dari setiap entitas dapat mengidentifikasi keunikan suatu individu.

3) **Data Value (*Nilai Data*)**

Data Value adalah data aktual atau informasi yang disimpan pada tiap data, elemen, atau atribut. Atribut nama pegawai menunjukan tempat dimana informasi nama karyawan disimpan, nilai datanya misalnya adalah Anjang, Arif, Suryo, dan lain-lain yang merupakan isi data nama pegawai tersebut.

4) **File/Tabel**

Kumpulan record sejenis yang mempunyai panjang elemen yang sama, atribut yang sama, namun berbeda nilai datanya.

5) **Record/Tuple**

Kumpulan elemen-elemen yang saling berkaitan menginformasikan tentang suatu entitas secara lengkap. Satu record mewakili satu data atau informasi.

2.9.3 Relasional Database Model

Relational Database Model merupakan salah satu jenis basis data yang paling banyak digunakan. Pada model ini, tabel-tabel dapat dihubungkan satu sama lain sehingga timbul suatu relasi antar tabel. Ada berbagai macam relasi antar tabel, diantaranya:

1). *One-to-One*

Relasi *One-to-One* biasanya digunakan untuk menghilangkan nilai NULL pada baris kolom tabel. Satu baris tabel mengacu pada satu baris pada tabel lainnya.

2). *One-to-Many*

Relasi *One-to-Many* merupakan relasi yang paling sering dijumpai pada *relational database model.* Satu baris tabel mengacu pada satu baris atau lebih pada tabel lainnya.

3) *Many-to-Many*

Relasi *many-to-many* berarti bahwa setiap baris tabel memiliki kemungkinan memiliki relasi dengan banyak baris pada tabel lainnya, dan sebaliknya. Biasanya relasi *many-to-many* membutuhkan tabel tambahan sebagai penghubung.

BAB III
METODE PENELITIAN

Pada tahap ini dijelaskan mengenai langkah-langkah yang akan dilakukan untuk merancang dan mengimplementasikan perangkat lunak yang akan dibuat. Adapun langkah-langkah yang akan dilakukan adalah sebagai berikut:

3.1 Studi Literatur

Dalam penyusunan skripsi ini, pengumpulan data dilakukan dengan melakukan studi literatur dengan sasaran tinjauan antara lain :

1) Pustaka Referensi
2) Informasi Internet
3) Pustaka Penunjang

Studi literatur yang dilakukan bertujuan untuk mengkaji hal-hal yang berhubungan dengan teori-teori yang mendukung dalam perencanaan dan perealisasian aplikasi. Adapun teori-teori yang dikaji adalah sebagai berikut:

1) Tangan
2) Pengolahan citra digital
3) Ekstraksi ciri
4) Algoritma *Learning Vector Quantization*
5) Pemrograman visual *C#*

3.2 Penentuan Spesifikasi Aplikasi

Menentukan perangkat yang digunakan untuk menunjang pembuatan aplikasi:

1. Perangkat Keras

a. Laptop

Intel Core™ 2 i3 M350 2.27GHz, DDR II 880MB RAM, harddisk 250 GB

b. *Webcam*

Internal laptop, Resolusi: 2 megapiksel, *frame rate*: up to 30fps.

M-tech, Resolusi: VGA, *frame rate*: up to 30fps.

c. Lightent

2. **Perangkat Lunak**

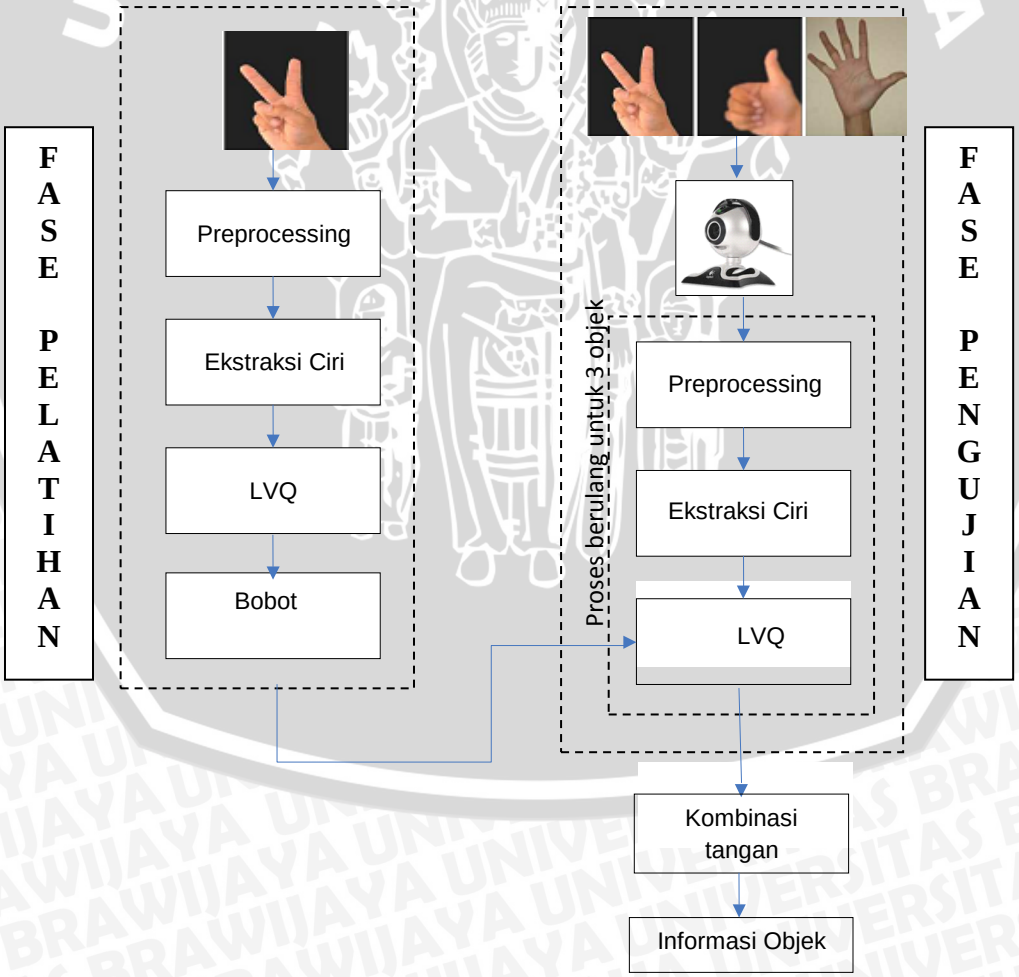
a. Windows XP Service Pack 3

b. Microsoft Visual Studio 2010 .NET

3.3 Perancangan dan Implementasi Sistem

Pada aplikasi identifikasi kombinasi pola isyarat tangan menggunakan webcam ini dibagi menjadi beberapa tahap perancangan, yaitu perancangan basis data, inisialisasi *webcam*, pengambilan gambar, *preprocessing*, ekstraksi ciri, pembelajaran *Learning Vector Quantization(LVQ)*, kombinasi tangan, identifikasi objek.

Komputer akan memproses citra objek menggunakan pengolahan citra digital untuk mendapatkan ciri yang terkandung pada objek. Hasil dari pengolahan ini digunakan untuk mengidentifikasi informasi yang terdapat pada objek dan menampilkannya.



Gambar 3.1 Diagram Sistem

(**Sumber:** Perancangan)

Gambar 3.1 menunjukkan gambaran sistem secara keseluruhan. Aplikasi berjalan dalam 2 fase yaitu fase pelatihan dan fase pengujian. Pada fase pelatihan citra diambil dengan cara di *upload* sedangkan fase pengujian pengambilan citra menggunakan *webcam*.

Pada fase pelatihan, citra objek masukan merupakan citra objek tangan kemudian citra objek dilakukan *resize* citra untuk memperkecil ukuran citra asli. Lalu dilakukan *preprocessing* yang mulai dari citra grayscale, thresholding, dan citra biner. Setelah diperoleh citra binernya dilakukan ekstraksi ciri berupa integral proyeksi untuk menentukan ciri dari pola yang diambil.

Pada fase pelatihan, ciri akan disimpan pada *database* untuk dilakukan proses pembelajaran LVQ. Pembelajaran LVQ merupakan salah satu metode pengklasifikasian pada jaringan syaraf tiruan. Dalam membuat LVQ terlebih dahulu ditentukan jumlah masukan dan keluaran pada jaringan serta beberapa parameter seperti nilai *learning rate*, nilai *maximum epoh*, dan nilai bobot.

Pada pembelajaran LVQ, data masukan berupa citra tangan dengan jumlah 20 buah untuk setiap pola isyarat tangannya, sehingga berjumlah 60 citra pola isyarat tangan dengan 3 pola keluaran. Data masukan ini tentunya telah melalui tahap ekstraksi ciri. Pola keluaran mencerminkan jenis pola isyarat tangan yang akan diidentifikasi.

Ekstraksi ciri yang didapat pada fase pengujian tidak akan tersimpan pada *database*. Hasil ekstraksi ciri ini digunakan untuk menghitung nilai *euclidean distance* terhadap nilai bobot hasil pembelajaran LVQ pada fase pelatihan. Tahap ini merupakan tahap kombinasi tangan dengan mencocokkan kombinasi tangan yang diinginkan dengan kombinasi tangan dari fase pengujian yang kemudian diperoleh informasi keluaran.

3.4 Pengujian dan Analisis

Pengujian dilakukan agar dapat menunjukkan bahwa aplikasi mampu bekerja sesuai dengan perancangan yang dilakukan dengan tingkat kesalahan yang sedikit. Pengujian yang dilakukan meliputi pengujian ekstraksi ciri, pengujian pengenalan isyarat, dan pengujian kombinasi serta analisis faktor kegagalan.

3.5 Pengambilan Kesimpulan dan Saran

Pada tahap ini, diambil dari hasil pengujian dan analisa terhadap aplikasi identifikasi isyarat tangan. Tahap Selanjutnya adalah membuat saran untuk perbaikan terhadap kesalahan-kesalahan yang terjadi pada penelitian sehingga dapat menyempurnakan penulisan serta untuk memberikan pertimbangan atas pengembangan aplikasi selanjutnya.

BAB IV
PERANCANGAN DAN IMPLEMENTASI

Pada tahap ini dijelaskan mengenai langkah-langkah yang akan dilakukan untuk merancang dan mengimplementasikan aplikasi identifikasi pola isyarat tangan berdasarkan ciri dari hasil integral proyeksi. Perancangan dan implementasi dikerjakan dengan beberapa tahap meliputi basis data, inisialisasi *webcam*, pengambilan gambar, *preprocessing*, ekstraksi ciri, pembelajaran LVQ, dan identifikasi kombinasi tangan. Pembuatan yang dilakukan secara bertahap bertujuan untuk memudahkan saat menganalisis setiap bagian aplikasi maupun secara keseluruhan.

4.1 Perancangan Secara Umum

Perancangan aplikasi secara umum merupakan tahap awal dalam melakukan perancangan aplikasi identifikasi objek yang akan dibuat. Perancangan diawali dengan penggambaran blok diagram kerja sistem yang menunjukkan cara kerja aplikasi secara umum.

4.1.1 Blok Diagram Sistem

Pembuatan diagram sistem merupakan dasar dari perancangan sistem agar perancangan dan perealisasian aplikasi berjalan secara sistematis. Citra masukan adalah citra objek yang ditangkap menggunakan kamera. Blok diagram dapat ditunjukkan seperti pada gambar 4.1.

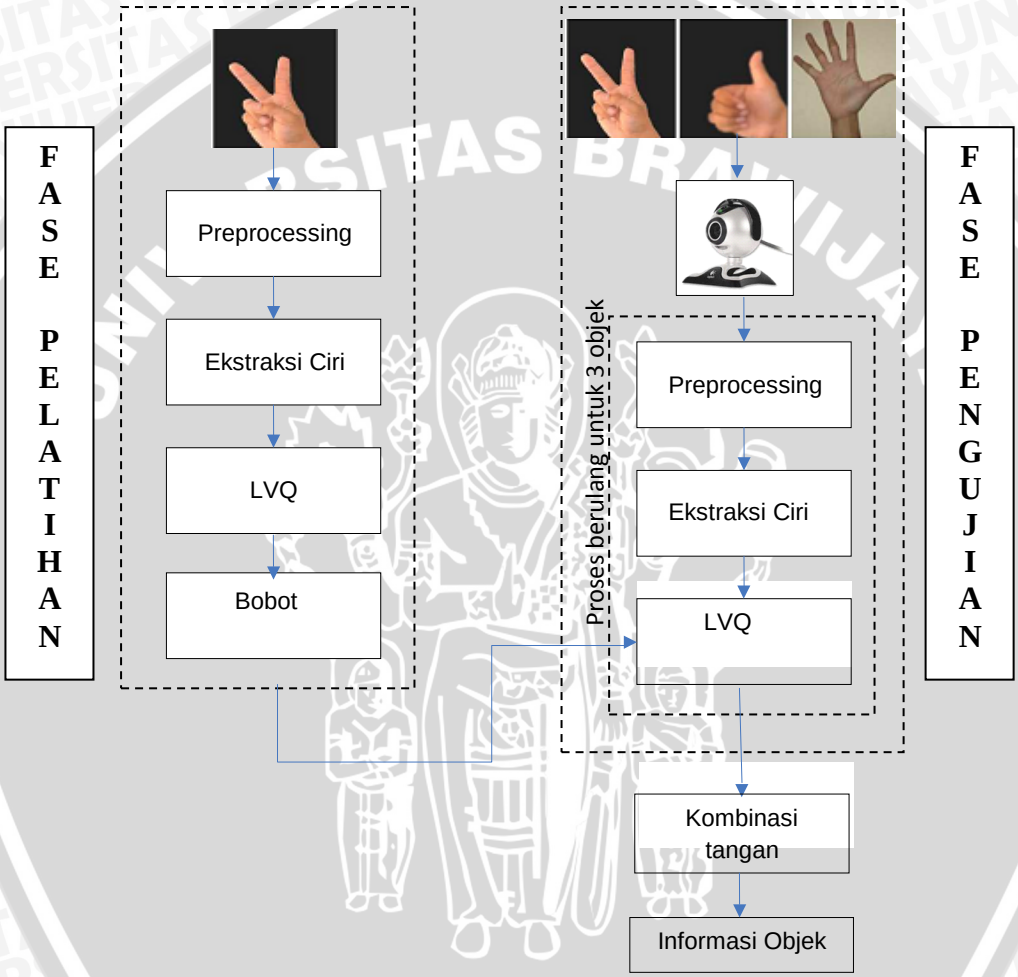
Gambar 4.1 Blok Diagram

(**Sumber:** Perancangan)

Fungsi dari masing-masing bagian pada diagram blok ini akan dijelaskan sebagai berikut:

1. Citra objek merupakan gambar yang merepresentasikan objek yang akan digunakan sebagai input sistem.
2. Dalam sistem ini piranti masukan yang dipakai adalah *webcam* yang digunakan untuk meng-*capture* citra

3. Komputer digunakan sebagai sarana untuk proses pengolahan citra mulai dari penskalaan citra, *image processing*, pelatihan dan untuk mendeteksi objek.
4. Mencocokkan kombinasi dari hasil pengujian dengan kombinasi yang sudah ditentukan.
5. Dari proses tahap sebelumnya maka akan didapatkan informasi yang menampilkan hasil identifikasi yang sesuai dengan tujuan perancangan dan pembuatan aplikasi ini.



Gambar 4.2 Diagram Sistem

(**Sumber:** Perancangan)

Gambar 4.2 menunjukkan gambaran diagram proses secara keseluruhan. Aplikasi berjalan dalam 2 fase yaitu fase pelatihan dan fase pengujian. Pada fase pelatihan citra diambil dengan cara di *upload* sedangkan fase pengujian pengambilan citra menggunakan *webcam*.

Pada fase pelatihan, citra objek masukan merupakan citra objek tangan kemudian citra objek dilakukan *resize* citra untuk memperkecil ukuran citra asli. Lalu dilakukan *preprocessing* yang mulai dari citra grayscale, thresholding, dan citra biner. Setelah diperoleh citra binernya dilakukan ekstraksi ciri berupa integral proyeksi untuk menentukan ciri dari pola yang diambil.

Pada fase pelatihan, ciri akan disimpan pada *database* untuk dilakukan proses pembelajaran LVQ. Pembelajaran LVQ merupakan salah satu metode pengklasifikasian pada jaringan syaraf tiruan. Dalam membuat LVQ terlebih dahulu ditentukan jumlah masukan dan keluaran pada jaringan serta beberapa parameter seperti nilai *learning rate*, nilai *maximum epoh*, dan nilai bobot.

Pada pembelajaran LVQ, data masukan berupa citra tangan dengan jumlah 20 buah untuk setiap pola isyarat tangannya, sehingga berjumlah 60 citra pola isyarat tangan dengan 3 pola keluaran. Data masukan ini tentunya telah melalui tahap ekstraksi ciri. Pola keluaran mencerminkan jenis pola isyarat tangan yang akan diidentifikasi.

Ekstraksi ciri yang didapat pada fase pengujian tidak akan tersimpan pada *database*. Hasil ekstraksi ciri ini digunakan untuk menghitung nilai *euclidean distance* terhadap nilai bobot hasil pembelajaran LVQ pada fase pelatihan. Tahap ini merupakan tahap kombinasi tangan dengan mencocokkan kombinasi tangan yang diinginkan dengan kombinasi tangan dari fase pengujian yang kemudian diperoleh informasi keluaran.

4.1.2 Cara Kerja Aplikasi

Cara kerja aplikasi identifikasi kombinasi pola isyarat tangan dengan pengolahan citra digital yaitu dimulai dengan proses *capture* atau proses pengambilan citra melalui kamera atau webcam. Untuk mendapatkan citra yang sesuai dengan keinginan, diperlukan peletakan *webcam* dan objek yang tepat pada tempat sistem diimplementasikan. Tangan akan diletakkan pada suatu tempat dengan latar belakang berwarna putih.

Setelah didapatkan hasil citra dari webcam kemudian dilanjutkan dengan proses *resize* dan merubah citra menjadi citra *bitmap* agar dapat diproses dengan baik menggunakan teknik pengolahan citra.

Setelah melalui tahap *preprocessing* yang bertujuan agar objek tangan terpisahkan dengan latar belakangnya, maka dilakukan proses greyscale dan tresholding yang kemudian didapatkan hasil berupa citra biner yang digunakan oleh integral

proyeksi untuk mendapatkan hasil ekstraksi ciri. Yang kemudian akan disimpan dalam *database* untuk proses pelatihan LVQ.

Dalam proses pelatihan LVQ dibutuhkan suatu masukan vektor yaitu vektor ciri hasil proses ektraksi. Selanjutnya *user* dapat mengatur nilai parameter *learning rate*, dan *maximum epoch*. Dari pelatihan tersebut akan didapatkan nilai bobot yang kemudian disimpan dalam database dan akan digunakan sebagai acuan identifikasi.

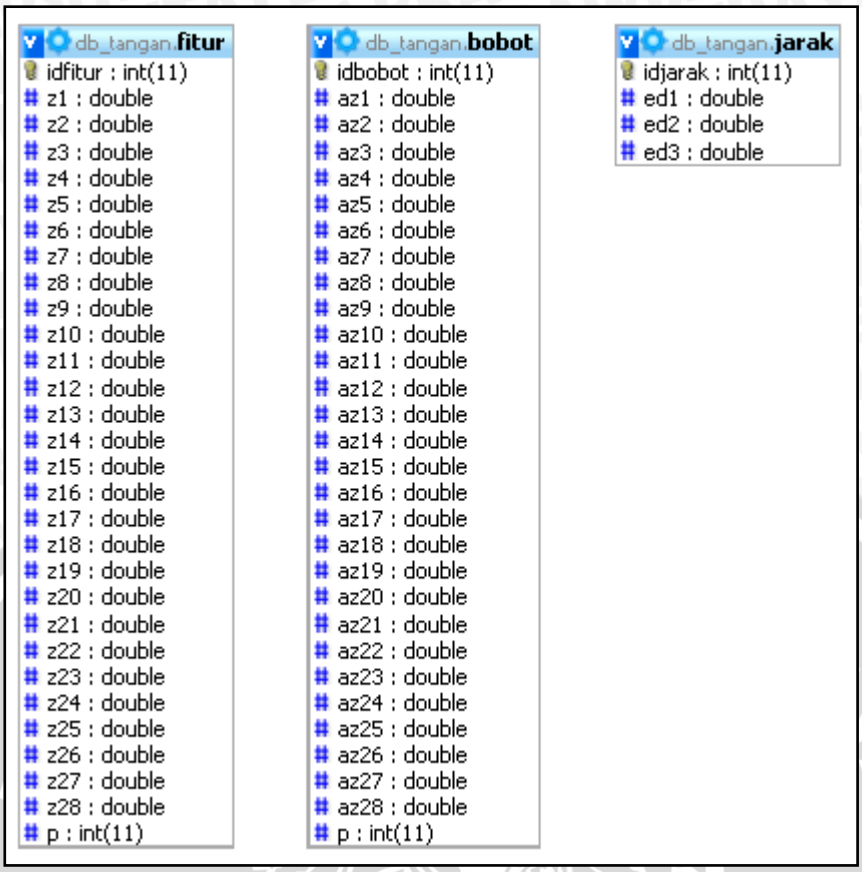
Pada fase pengujian sendiri diawali dengan proses *preprocessing* yang memiliki tujuan sama yaitu memisahkan objek dengan latar belakang, kemudian tiap–tiap objek tersebut akan diidentifikasi dengan cara diekstraksi ciri, setelah didapat hasil ekstrasi ciri kemudian dari tiap objek tersebut dilakukan penghitungan jarak dengan menghitung *euclidean distance* antara bobot tiap jenis objek latih dengan objek pengujian. Jika didapatkan jarak terdekat terhadap salah satu jenis objek tertentu maka objek uji tersebut akan masuk ke dalam jenis objek yang sama dengan objek yang telah ditentukan tersebut, proses ini dilakukan hingga seluruh objek dalam citra telah diidentifikasi. Apabila sudah didapatkan hasil dari indentifikasi kemudian mencocokkan kombinasi hasik pengujian dengan hasil dari kombinasi yang diinginkan sehingga diperoleh keluaran berupa suatu perintah untuk membuka.

4.2 Perancangan Perangkat Lunak

Perancangan perangkat lunak dibangun dengan menggunakan bahasa pemrograman Microsoft Visual Studio C# 2010. Disini akan membahas secara detail tentang setiap tahapan proses pada aplikasi identifikasi kombinasi pola isyarat tangan berdasarkan nilai fitur. Beberapa tahap perancangan tersebut meliputi basis data, *load* gambar, *pre-pocessing*, ekstraksi ciri, pembelajaran *Learning Vector Quantization(LVQ)*, identifikasi objek.

4.2.1 Basis data

Perancangan basis data dibangun menggunakan MySQL. Basis data ini akan digunakan sebagai tempat penyimpanan hasil ekstraksi ciri dan juga bobot. Beberapa tabel yang dibutuhkan adalah tabel ciri, bobot, dan jarak seperti yang ditunjukkan pada gambar 4.3.



Gambar 4.3 Tabel basis data

(**Sumber:** Perancangan)

Gambar 4.3 menunjukkan bahwa tabel ciri akan menyimpan nilai hasil ekstraksi ciri berupa z1 sampai dengan z28 dimana z tersebut berisi nilai ciri hasil integral proyeksi. Selain itu tabel ciri juga akan menyimpan beberapa informasi penting seperti jenis objek(p). Tabel bobot menyimpan bobot hasil pelatihan LVQ untuk setiap jenis objek yaitu nilai az1 sampai dengan az28. Tabel jarak menyimpan nilai jarak setiap pelatihan LVQ dimana terdapat jarak terhadap kelas 1 sampai kelas 3 yang diwakili dengan ed1, ed2, dan ed3.

Menghubungkan basis data dengan perangkat lunak pada bahasa C# menggunakan MySQL *Connector*. Berikut ini adalah bagian kode bahasa C# agar dapat terhubung dengan basis data:

```
//inisialisasidatabase
string connectionSQL =
"server=localhost;user=root;database=db_tangan;port=3306;password='';";
MySqlConnection con = new MySqlConnection(connectionSQL);
```

4.2.2 Inisialisasi webcam

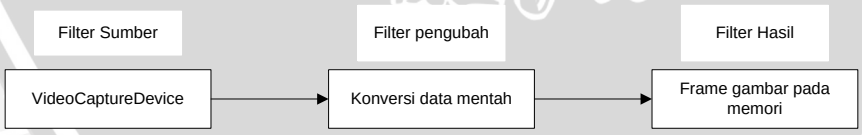
Perancangan inisialisasi *webcam* dilakukan dengan memanfaatkan beberapa fitur komponen *Direct Show* yang dibuat oleh Andrew Kirillov (*developer* Aforge *framework*) sebagai antar muka perangkat keras. *Direct Show* adalah pustaka *Component Object Model* (COM) untuk dapat mengakses perangkat keras berbasis *Windows Driver Model* (WDM) seperti *webcam*. Berikut ini adalah pengelompokan class yang ada di dalam *Direct Show* dapat dilihat pada Tabel 4.1.

Tabel 4.1 Daftar class interface yang ada di dalam *Direct Show*

No	Class	Keterangan
1	FileVideoSource	Sumber video untuk file video
2	FilterCategory	Kategori *Direct Show* filter
3	FilterInfo	Informasi Filter *Direct Show*
4	FilterInfoColection	Koleksi objek informasi filter
5	VideoCapabilities	Kemampuan perangkat video seperti ukuran frame dan frame rate
6	Video*Capture*Device	Sumber video untuk pengambilan gambar video pada perangkat local (*webcam*)

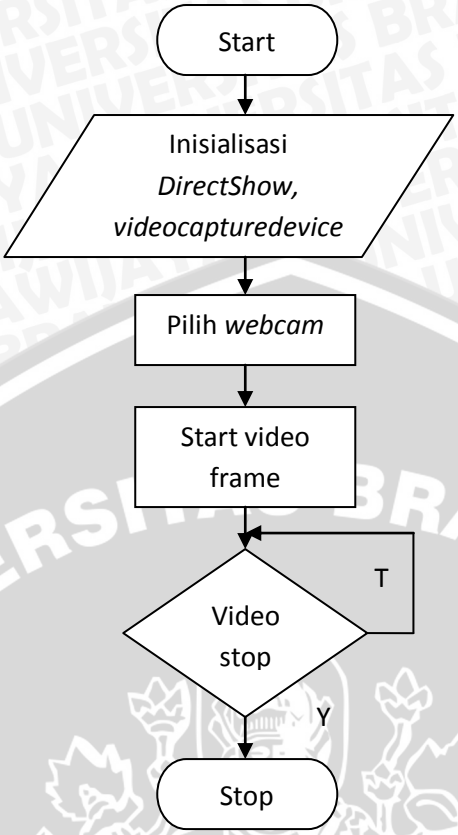
(**Sumber:** AForge .NET Framework)

Pada perancangan inisialisasi *webcam* ini, *Direct Show* mengerjakan tugas dengan cara menghubungkan serangkaian filter bersama-sama, sehingga keluaran dari satu filter menjadi masukan untuk filter lainnya. Kumpulan filter-filter yang terhubung membentuk graf disebut graf filter, seperti ditunjukkan pada Gambar 4.4 .



Gambar 4.4 Graf filter mengakses *webcam*

Hasil tangkapan yang diletakkan pada obyek berupa aliran data *real–time* pada memori. Agar dapat diolah maka harus diubah menjadi file gambar. Data dalam obyek pada satu waktu dicuplik ke dalam sebuah array kosong dan diberi format gambar *bitmap* (BMP). Flowchart inisialisasi *webcam* seperti yang ditunjukkan pada Gambar 4.5.



Gambar 4. 1 Flowchart inisialisasi *webcam*

Implementasi inisialisasi *webcam* dan pengambilan gambar pada bahasa C# adalah sebagai berikut:

```
using AForge.Video.DirectShow;
videoSource = new
VideoCaptureDevice(videoDevices[comboBox1.SelectedIndex].MonikerString);
```

4.2.3 Load Image

Sebelum memulai proses pengolahan citra aplikasi ini membutuhkan *input* berupa citra yang merepresentasikan citra objek, sebelumnya citra objek ini didapatkan dengan cara pengambilan citra melalui webcam yang sudah dilakukan diluar proses aplikasi. Proses load image hanya digunakan pada fase pelatihan saja.

Citra yang didapat dari pengambilan gambar memiliki ukuran 320x240 piksel kemudian dilakukan tahap *resize* dengan ukuran 16x12 piksel dan diberi format bitmap (BMP).

Berikut ini adalah bagian kode bahasa C# yang berfungsi mengaktifkan menu dialog untuk memilih sumber gambar :

```
OpenFileDialog menu = new OpenFileDialog();
DialogResult result = menu.ShowDialog();
```

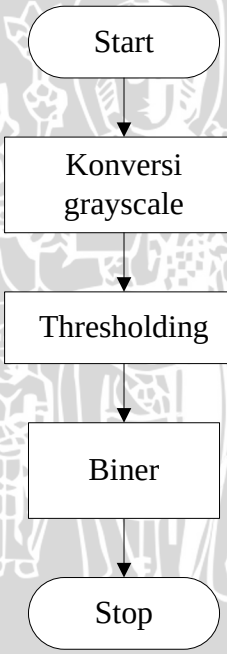
Untuk resize dan konversi format sebagai berikut :

```
public static Bitmap ResizeBitmap(Bitmap sourceBMP, int width, int height)
{
  Bitmap result = new Bitmap(width, height);
  using (Graphics g = Graphics.FromImage(result))g.DrawImage(sourceBMP, 0, 0,
width, height);
  return result;
}

 Bitmap resize1 = ResizeBitmap(gambar_ubah, 16, 12);
```

4.2.4 Preprocessing

Setelah melalui tahap *load image* dan citra objek telah siap untuk digunakan maka sebelum proses berjalan lebih jauh citra objek akan melalui tahap *preprocessing*. *Preprocessing* yang dilakukan bertujuan memisahkan citra objek dengan latar belakangnya. Proses *preprocessing* yang berlangsung yaitu konversi *grayscale*, dan *thresholding* dan hasilnya berupa citra biner seperti pada gambar 4.6.



Gambar 4.6 Flowchart *preprocessing*

(**Sumber:** perancangan)

Gambar 4.6 menjelaskan bahwa citra yang menjadi masukan akan dilakukan konversi ke dalam bentuk *grayscale* karena hasil pada tahap *load image* berupa citra warna atau RGB. Tujuan konversi *grayscale* ini adalah supaya kombinasi warna menjadi lebih sederhana sehingga memudahkan untuk proses selanjutnya. Perhitungan

grayscale yaitu 0.2125(R), 0.7154(G), 0.0721(B). *Grayscale* menyebabkan citra warna menjadi citra abu-abu yang memiliki rentang nilai piksel mulai 0 – 255.

Selanjutnya adalah melakukan *thresholding* atau pengambangan. Proses *thresholding* bertujuan untuk menghilangkan latar belakang citra. Nilai *threshold* ideal pada proses ini adalah 210, yang didapatkan dengan melakukan *trial* dan *error*. Perhitungan *threshold* adalah dengan mengubah piksel bernilai >210 menjadi berwarna putih (255, 255, 255). Sedangkan piksel bernilai ≤210 akan dikembalikan menjadi warna hitam.

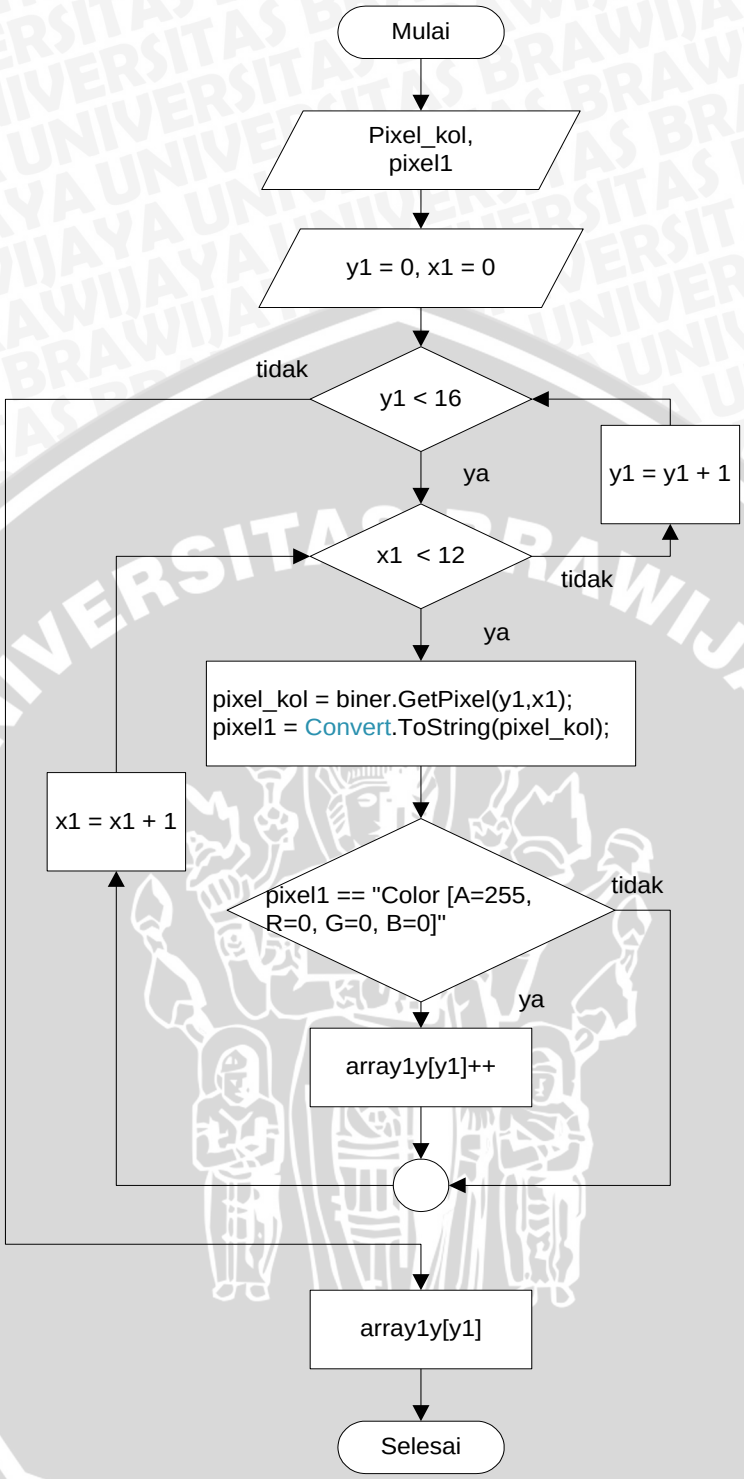
Implementasi proses *preprocessing* pada bahasa C# adalah sebagai berikut:

```
//proses grayscale
Grayscale filter = new Grayscale(0.2125, 0.7154, 0.0721);
Bitmap abuabu = filter.Apply(resize1);


//proses threshold
Threshold filter2 = new Threshold(210);
Bitmap biner = filter2.Apply(abuabu);
```

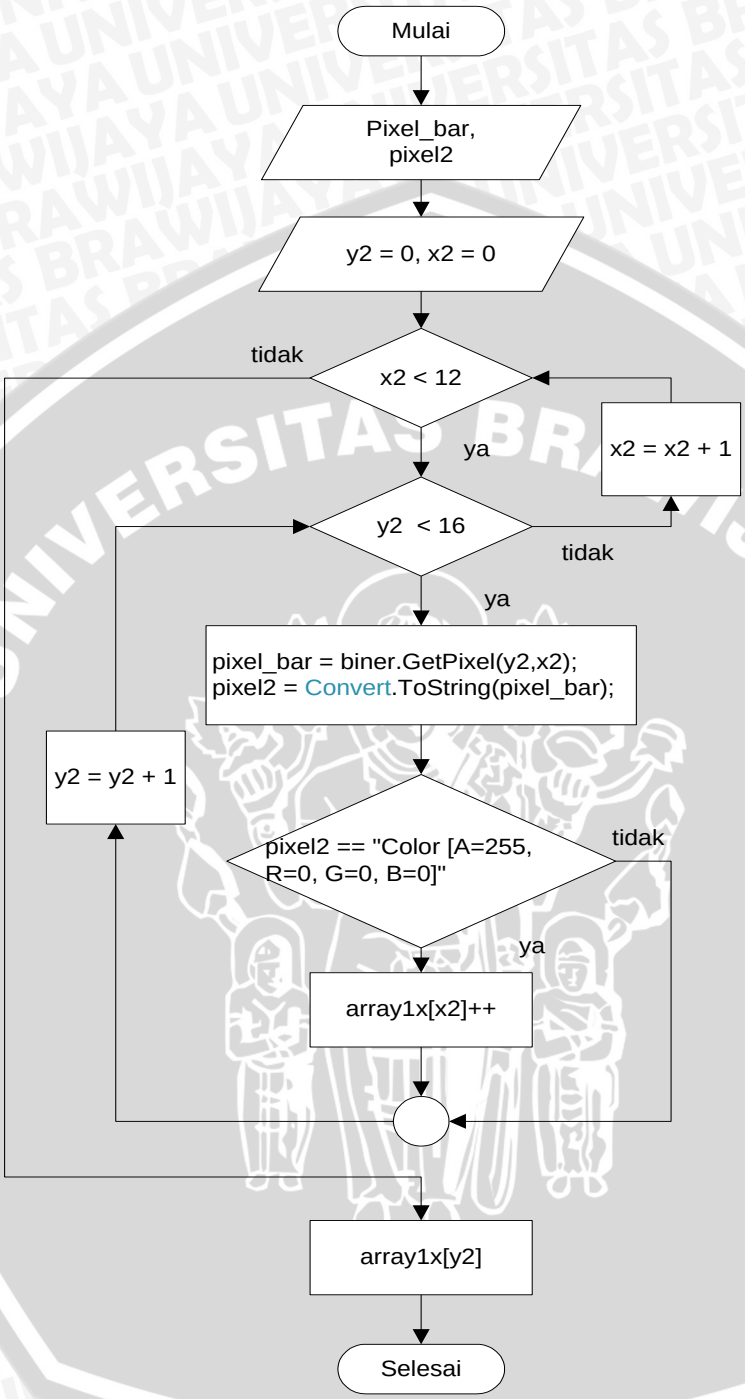
4.2.5 Perancangan ekstraksi ciri

Pada tahap ekstraksi ciri, citra masukan berupa citra biner hasil *pre-pocessing*. Ekstraksi ciri bertujuan untuk mendapatkan karakteristik dari suatu objek tertentu yang dapat membedakan dengan jenis objek yang lain. Ciri yang digunakan adalah nilai fitur hasil integral proyeksi berupa nilai kolom dan baris. Untuk mendapatkan nilai fitur tersebut maka digunakan metode seperti yang dijelaskan pada gambar 4.7.



Gambar 4.7 Flowchart integral proyeksi kolom

(**Sumber:** Perancangan)



Gambar 4.8 Flowchart integral proyeksi baris

(**Sumber:** Perancangan)

Gambar 4.7 dan gambar 4.8 menunjukkan bahwa citra hasil *pre-processing* merupakan masukan dalam tahap ekstraksi ciri. Setelah didapatkan nilai hasil integral

proyeksi dari baris dan kolom, kemudian digabungkan menjadi satu array dengan inisial array[z].

Implementasi proses ekstraksi ciri integral proyeksi kolom pada bahasa C# adalah sebagai berikut:

```
//looping pengambilan data/informasi (INTEGRAL PROYEKSI)
for (int y1 = 0; y1 < 16; y1++)
{   //scan kebawah
    for (int x1 = 0; x1 < 12; x1++)
        {
            Color pixel_kol = biner.GetPixel(y1, x1);
            string pixel1 = Convert.ToString(pixel_kol);

            if (pixel1 == "Color [A=255, R=0, G=0, B=0]")
               {
                   array1y[y1]++;
               }
        }
 }
```

Implementasi proses ekstraksi ciri integral proyeksi baris pada bahasa C# adalah sebagai berikut:

```
//looping pengambilan data/informasi (INTEGRAL PROYEKSI)
for (int x2 = 0; x2 < 12; x2++)
{   //scan kebawah
    for (int y2 = 0; y2 < 16; y2++)
        {
             Color pixel_bar = biner.GetPixel(y2, x2);
             string pixel2 = Convert.ToString(pixel_bar);

             if (pixel2 == "Color [A=255, R=0, G=0, B=0]")
                {
                    array1x[x2]++;
                }
         }
 }
```

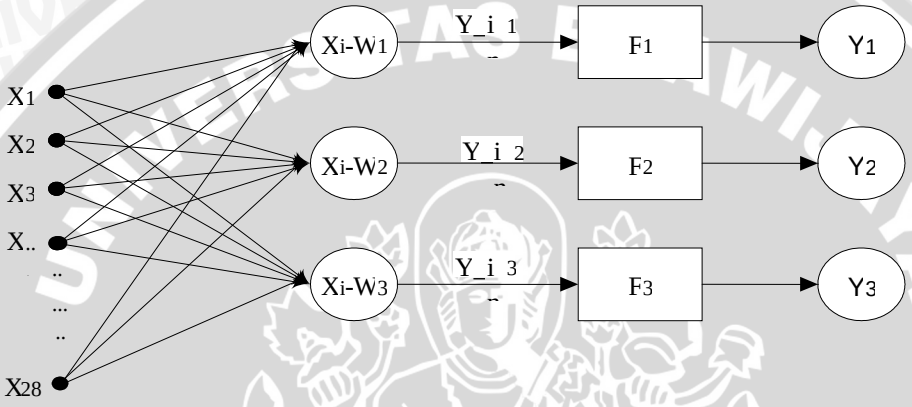
Implementasi proses menggabungkan integral proyeksi kolom dan baris pada bahasa C# adalah dengan nenasukkan nilainya satu per satu, seperti berikut ini:

```
//menggabungkan array y dan array x
            z1 = array1y[0];                    z17 = array1x[0];
            z2 = array1y[1];                    z18 = array1x[1];
            z3 = array1y[2];                    z19 = array1x[2];
            z4 = array1y[3];                    z20 = array1x[3];
            z5 = array1y[4];                    z21 = array1x[4];
            z6 = array1y[5];                    z22 = array1x[5];
            z7 = array1y[6];                    z23 = array1x[6];
            z8 = array1y[7];                    z24 = array1x[7];
            z9 = array1y[8];                    z25 = array1x[8];
            z10 = array1y[9];                   z26 = array1x[9];
            z11 = array1y[10];                  z27 = array1x[10];
            z12 = array1y[11];                  z28 = array1x[11];
            z13 = array1y[12];
            z14 = array1y[13];
            z15 = array1y[14];
            z16 = array1y[15];
```

Sebanyak 60 objek pelatihan mengalami ekstraksi ciri secara bergantian, dan secara acak. Hasil dari ekstraksi ciri ini nilai-nilainya akan disimpan ke dalam basis data.

4.2.6 Perancangan pembelajaran *Learning Vector Quantization*

Proses pembelajaran LVQ diawali dengan pengambilan data training. Data training ini diambil dari basis data yang berisi nilai-nilai ekstraksi ciri yang telah disimpan. Data training ini merupakan input *layer* pada jaringan saraf tiruan seperti yang ditunjukkan pada gambar 4.11.



Gambar 4.9 Struktur jaringan LVQ

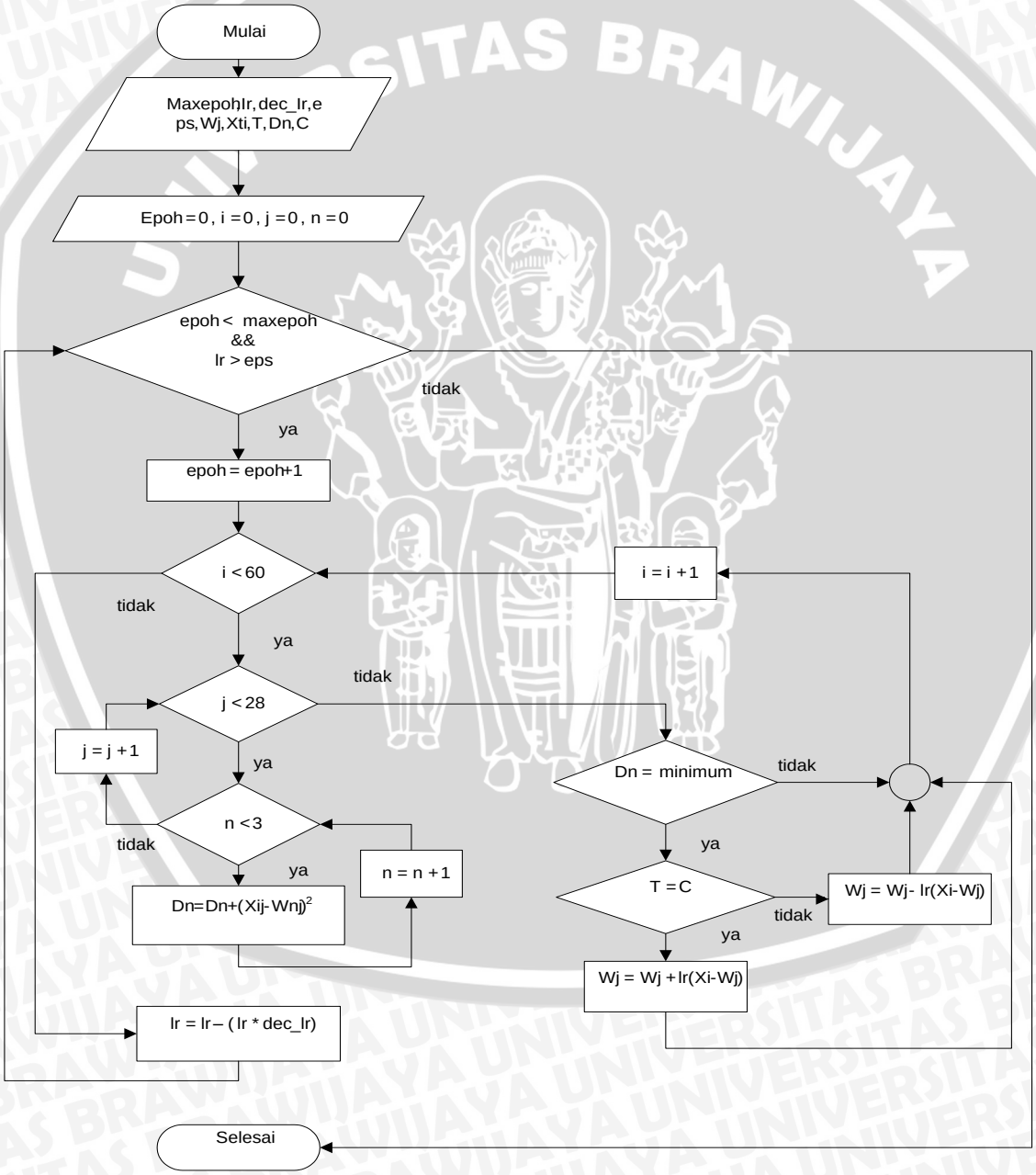
(**Sumber:** Perancangan)

Gambar 4.9 adalah arsitektur jaringan LVQ (*Learning Vector Quantization*) pada aplikasi yang terdiri dari 28 *node* lapisan masukan (*input layer*), 3 *node* lapisan tersembunyi (*hidden layer*) serta 3 *node* lapisan keluaran (*output layer*). Lapisan input memiliki 28 node yang disimbolkan dengan nilai x_1, x_2, x_3, sampai x_{28}. Pada tiap lapisan masukan terlebih dahulu diberikan 3 nilai bobot yang berbeda yaitu W_1, W_2, dan W_3. Pemberian bobot bertujuan agar nilai tiap masukan memiliki penimbang (bobot) yang kemudian akan dihitung nilainya dan diselesaikan dengan persamaan yang dimiliki oleh metode LVQ (*Learning Vector Quantization*), sehingga jaringan memiliki 3 kelas yang berbeda, yaitu kelas Y_1, Y_2, Y_3. Kelas Y_1 memiliki bobot W_1, kelas Y_2 memiliki bobot W_2 dan seterusnya. Keluaran dari lapisan ini akan menjadi masukan bagi lapisan tersembunyi (*hidden layer*) sebanyak 3 *node* yaitu Y_in_1, Y_in_2, dan Y_in_3.

Bobot awal pada aplikasi ini diberikan dengan mengambil nilai data pelatihan pertama untuk setiap kelasnya. Dengan menggunakan metode *euclidean distance* bahwa

nilai paling kecil yang dihasilkan adalah pemenang dan merupakan kelas dari input tersebut maka pada lapisan keluaran (*output layer*) digunakan sebuah fungsi pembanding yang berguna membandingkan dua nilai tersebut untuk dicari nilai terkecilnya. Dalam gambar 4.11 fungsi pembanding tersebut dituliskan dengan simbol F_1, F_2, dan F_3. Pada aplikasi ini nilai *learning rate* ditentukan oleh user begitu juga nilai pengurangan *learning rate*.

Nilai input yang diberikan ke lapisan masukan (input layer) berupa vektor. Tujuannya adalah untuk mempermudah penentuan jumlah node pada lapisan masukan serta perhitungannya dengan metode LVQ (Learning Vector Quantization).



Gambar 4.10 Flowchart algoritma LVQ

(**Sumber:** perancangan)

Gambar 4.10 menjelaskan algoritma jaringan saraf tiruan metode LVQ. Langkah pertama adalah menentukan masing-masing kelas output (C), bobot (W), *learning rate* (lr), maksimum epoh (maxepoh), dan eror minimum yang diharapkan (eps). Kemudian memasukkan data input (X_i) dengan i=1, 2, 3, ..., 60 serta target kelas (T).

Masing-masing input dibandingkan dengan bobot yang telah ditetapkan dengan melakukan pengukuran jarak antara masing-masing bobot W dan input X. Nilai minimum dari hasil perbandingan akan menentukan kelas dari vektor input dan perubahan bobot dari kelas tersebut. Untuk input dan bobot yang memiliki kelas yang sama maka bobot diubah menjadi $W = W + Lr(X - W)$ sedangkan untuk input dan bobot dengan kelas yang berbeda diubah menjadi $W = W - Lr(X - W)$. Proses ini dilakukan secara terus-menerus sampai kondisi maxepoh dan learning rate terpenuhi.

Berikut ini adalah salah satu contoh pengukuran dan pembandingan jarak pada aplikasi untuk 3 input pertama.

Vektor input objek:

2. Pola tangan ke 1 = [0, 0, 2, 2, 2, 3, 9, 10, 10, 8, 7, 4, 0, 1, 2, 4, 6, 6, 6, 5, 6, 6, 6, 6, 3, 0, 0, 0]

Vektor bobot masing-masing jenis objek:

1. [0, 0, 0, 0, 5, 10, 11, 10, 8, 7, 6, 3, 0, 0, 0, 1, 3, 10, 11, 8, 8, 8, 7, 6, 5, 3, 0, 0]

2. [2, 2, 2, 3, 5, 10, 11, 10, 8, 7, 6, 3, 0, 0, 0, 1, 3, 10, 11, 8, 8, 8, 7, 6, 5, 3, 0, 0]

3. [4, 4, 4, 5, 6, 10, 11, 10, 8, 7, 6, 3, 0, 0, 0, 1, 6, 9, 9, 8, 8, 9, 12, 10, 5, 3, 0, 0]

Proses pelatihan yang terjadi adalah sebagai berikut:

Iterasi 1:

Data ke-1 0, 0, 2, 2, 2, 3, 9, 10, 10, 8, 7, 4, 0, 1, 2, 4, 6, 6, 6, 5, 6, 6, 6, 6, 3, 0, 0, 0] mewakili pola tangan ke 1:

- Jarak ke bobot 1

$=\sqrt{(0-0)^2+(0-0)^2+(2-0)^2+(2-0)^2+(2-5)^2+(3-10)^2+(9-11)^2+(10-8)^2+(8-7)^2+(7-6)^2+(4-3)^2+(0-0)^2+(1-0)^2+(2-0)^2+(4-1)^2+(6-3)^2+(6-10)^2+(6-11)^2+(5-8)^2+(6-8)^2+(6-8)^2+(6-7)^2+(6-6)^2+(3-5)^2+(0-3)^2+(0-0)^2+(0-0)^2}$

$=13{,}1149$

- Jarak ke bobot 2

$=\sqrt{(0-2)^2+(0-2)^2+(2-2)^2+(2-3)^2+(2-5)^2+(3-10)^2+(9-11)^2+(10-8)^2+(8-7)^2+(7-6)^2+(4-3)^2+(0-0)^2+(1-0)^2+(2-0)^2+(4-1)^2+(6-3)^2+(6-10)^2+(6-11)^2+(5-8)^2+(6-8)^2+(6-8)^2+(6-7)^2+(6-6)^2+(3-5)^2+(0-3)^2+(0-0)^2+(0-0)^2}$

$=13,1529$

- Jarak ke bobot 3

$=\sqrt{(0-4)^2+(0-4)^2+(2-4)^2+(2-5)^2+(2-6)^2+(3-10)^2+(9-11)^2+(10-8)^2+(8-7)^2+(7-6)^2+(4-3)^2+(0-0)^2+(1-0)^2+(2-0)^2+(4-1)^2+(6-6)^2+(6-9)^2+(6-9)^2+(5-8)^2+(6-8)^2+(6-9)^2+(6-12)^2+(6-10)^2+(3-5)^2+(0-3)^2+(0-0)^2+(0-0)^2}$

$=15,4919$

Jarak terkecil pada bobot ke-1

Target data ke-1 = 1

Bobot ke-1 baru :

$W11 = W11 + \alpha*(Z11-W11) = 0 + 0,05*(0-0) = 0$

$W12 = W12 + \alpha*(Z12-W12) = 0 + 0,05*(0-0) = 0$

$W13 = W13 + \alpha*(Z13-W13) = 0 + 0,05*(2-0) = 0,1$

$W14 = W14 + \alpha*(Z14-W14) = 0 + 0,05*(2-0) = 0,1$

$W15 = W15 + \alpha*(Z15-W15) = 5 + 0,05*(2-5) = 4,85$

$W16 = W16 + \alpha*(Z16-W16) = 10 + 0,05*(3-10) = 9,65$

$W17 = W17 + \alpha*(Z17-W17) = 11 + 0,05*(9-11) = 10,9$

$W18 = W18 + \alpha*(Z18-W18) = 10 + 0,05*(10-10) = 10$

$W19 = W19 + \alpha*(Z19-W19) = 8 + 0,05*(10-8) = 8,1$

$W110 = W110 + \alpha*(Z110-W110) = 7 + 0,05*(8-7) = 7,05$

$W111 = W111 + \alpha*(Z111-W111) = 6 + 0,05*(7-6) = 6,05$

$W112 = W112 + \alpha*(Z112-W112) = 3 + 0,05*(4-3) = 3,05$

$W113 = W113 + \alpha*(Z113-W113) = 0 + 0,05*(0-0) = 0$

$W114 = W114 + \alpha*(Z114-W114) = 0 + 0,05*(1-0) = 0,05$

$W115 = W115 + \alpha*(Z115-W115) = 0 + 0,05*(2-0) = 0,1$

$W116 = W116 + \alpha*(Z116-W116) = 1 + 0,05*(4-1) = 1,15$

$W117 = W117 + \alpha*(Z117-W117) = 3 + 0,05*(6-3) = 3,15$

$W118 = W118 + \alpha*(Z118-W118) = 10 + 0,05*(6-10) = 9,8$

$W119 = W119 + \alpha*(Z119-W119) = 11 + 0,05*(6-11) = 10,75$

$W120 = W120 + \alpha*(Z120-W120) = 8 + 0,05*(5-8) = 7,85$

$W121 = W121 + \alpha*(Z121-W121) = 8 + 0,05*(6-8) = 7,9$

$W122 = W122 + \alpha*(Z122-W122) = 8 + 0,05*(6-8) = 7,9$

$W123 = W123 + \alpha*(Z123-W123) = 7 + 0,05*(6-7) = 6,95$

W124 = W124 + α*(Z124-W124) = 6 + 0,05*(6-6) = 6

W125 = W125 + α*(Z125-W125) = 5 + 0,05*(3-5) = 4,9

W126 = W126 + α*(Z126-W126) = 3 + 0,05*(0-3) = 2,85

W127 = W127 + α*(Z127-W127) = 0 + 0,05*(0-0) = 0

W128 = W128 + α*(Z128-W128) = 0 + 0,05*(0-0) = 0

Jadi :

W_1=(0 0 0,1 0,1 4,85 9,65 10,9 10 8,1 7,05 6,05 3,05 0 0,05 0,1 1,15 3,15 9,8 10,75 7,85 7,9 7,9 6,95 6 4,9 2,85 0 0)

Implementasi proses pelatihan LVQ dimulai dengan melakukan inisialisasi input, kelas target, dan parameter-parameter seperti maksimal epoh, *learning rate*, eror minimum yang diharapkan, serta nilai pengurangan *learning rate*. Inisialisasi yang dilakukan pada bahasa C# adalah sebagai berikut:

```
//inisialisi input training
double[,] x = new double[60, 29];

//inisialisi input testing
double[,] tes = new double[1, 28];

//inisialisasi bobot
double[,] w = new double[3, 28];

//jenis target
int[] C = { 1, 2, 3 };
int pola;

//max epoh
int maxepoh;
int epoh;

//learning rate
int pilih_lr;
double lr;

//pengurangan learning rate
int pilih_dec_lr;
double dec_lr;

//eror minimum yg diharapkan
double eps = 0.01;

//euclidean distance
double ed1;
double ed2;
double ed3;
```

Untuk penentuan nilai awal bobot adalah 3 data pertama untuk setiap jenis objek. Pada pengambilan nilai bobot yang dilakukan pada bahasa C# adalah sebagai berikut:

```
int id = 0;

MySqlConnection con = new MySqlConnection(connectionSQL);

    string strSQL = "SELECT * FROM bobot";
    MySqlCommand mysqlCmd = new MySqlCommand(strSQL, con);
    con.Open();
    MySqlDataReader reader;
    reader = mysqlCmd.ExecuteReader();
    while (reader.Read())
    {
        w[id, 0] = (double)reader["az1"];
        w[id, 1] = (double)reader["az2"];
        w[id, 2] = (double)reader["az3"];
        w[id, 3] = (double)reader["az4"];
        w[id, 4] = (double)reader["az5"];
        w[id, 5] = (double)reader["az6"];
        w[id, 6] = (double)reader["az7"];
        w[id, 7] = (double)reader["az8"];
        w[id, 8] = (double)reader["az9"];
        w[id, 9] = (double)reader["az10"];
        w[id, 10] = (double)reader["az11"];
        w[id, 11] = (double)reader["az12"];
        w[id, 12] = (double)reader["az13"];
        w[id, 13] = (double)reader["az14"];
        w[id, 14] = (double)reader["az15"];
        w[id, 15] = (double)reader["az16"];
        w[id, 16] = (double)reader["az17"];
        w[id, 17] = (double)reader["az18"];
        w[id, 18] = (double)reader["az19"];
        w[id, 19] = (double)reader["az20"];
        w[id, 20] = (double)reader["az21"];
        w[id, 21] = (double)reader["az22"];
        w[id, 22] = (double)reader["az23"];
        w[id, 23] = (double)reader["az24"];
        w[id, 24] = (double)reader["az25"];
        w[id, 25] = (double)reader["az26"];
        w[id, 26] = (double)reader["az27"];
        w[id, 27] = (double)reader["az28"];
        id = id + 1;
    }
    con.Close();
```

Setelah itu dilakukan pengambilan data *training.* Pengambilan data *training* dari basis data adalah membaca setiap *record* kemudian menyimpannya dalam suatu array. Implementasinya pada bahasa C# dilakukan sebagai berikut:

```
int id = 0;

MySqlConnection con = new MySqlConnection(connectionSQL);

    string strSQL = "SELECT * FROM fitur";
    MySqlCommand mysqlCmd = new MySqlCommand(strSQL, con);
    con.Open();
```

```
MySqlDataReader reader;
reader = mysqlCmd.ExecuteReader();
while (reader.Read())
{
    x[id, 0] = (double)reader["z1"];
    x[id, 1] = (double)reader["z2"];
    x[id, 2] = (double)reader["z3"];
    x[id, 3] = (double)reader["z4"];
    x[id, 4] = (double)reader["z5"];
    x[id, 5] = (double)reader["z6"];
    x[id, 6] = (double)reader["z7"];
    x[id, 7] = (double)reader["z8"];
    x[id, 8] = (double)reader["z9"];
    x[id, 9] = (double)reader["z10"];
    x[id, 10] = (double)reader["z11"];
    x[id, 11] = (double)reader["z12"];
    x[id, 12] = (double)reader["z13"];
    x[id, 13] = (double)reader["z14"];
    x[id, 14] = (double)reader["z15"];
    x[id, 15] = (double)reader["z16"];
    x[id, 16] = (double)reader["z17"];
    x[id, 17] = (double)reader["z18"];
    x[id, 18] = (double)reader["z19"];
    x[id, 19] = (double)reader["z20"];
    x[id, 20] = (double)reader["z21"];
    x[id, 21] = (double)reader["z22"];
    x[id, 22] = (double)reader["z23"];
    x[id, 23] = (double)reader["z24"];
    x[id, 24] = (double)reader["z25"];
    x[id, 25] = (double)reader["z26"];
    x[id, 26] = (double)reader["z27"];
    x[id, 27] = (double)reader["z28"];

    //load category
    x[id, 28] = (int)reader["p"];

    id = id + 1;
}
con.Close();
```

Dengan telah dilakukan inisialisasi dan pengambilan data *training* dari *database*, maka pelatihan LVQ dapat dilakukan. Pelatihan berlangsung sesuai jumlah maksimal epoh dan *learning rate* seperti berikut ini:

Int I = 60;

```
while (epoh < maxepoh && lr > eps)
{
  epoh = epoh + 1;
  for (int i = 0; i <= I; i++)
  {
    train(i);
  }
  lr = lr - (lr * dec_lr);
}
```

Pada fungsi *train()* proses yang terjadi berupa penghitungan jarak *euclidean*, kemudian melakukan perubahan nilai bobot sesuai target kelasnya. Berikut ini adalah proses penghitungan jarak *euclidean* pada bahasa C#:

```
for (int j = 0; j < 28; j++)
{
  ed1 = ed1 + ((x[a, j] - w[0, j]) * (x[a, j] - w[0, j]));
  ed2 = ed2 + ((x[a, j] - w[1, j]) * (x[a, j] - w[1, j]));
  ed3 = ed3 + ((x[a, j] - w[2, j]) * (x[a, j] - w[2, j]));
}

//akar
ed1 = Math.Sqrt(ed1);
ed2 = Math.Sqrt(ed2);
ed3 = Math.Sqrt(ed3);
```

Proses perubahan salah satu nilai bobot jika sesuai dengan kelasnya pada bahasa C# adalah sebagai berikut:

```
//ubah bobot w1
for (int i = 0; i <= 27; i++)
    {
      w[0, i] = w[0, i] + lr * (x[a, i] - w[0, i]);
     }
con.Open();
string strSQL = "UPDATE bobot SET az1='" + w[0, 0] + "',az2='" + w[0, 1] +
"',az3='" + w[0, 2] + "',az4='" + w[0, 3] + "',az5='" + w[0, 4] + "',az6='" +
w[0, 5] + "',az7='" + w[0, 6] + "',az8='" + w[0, 7] + "',az9='" + w[0, 8] +
"',az10='" + w[0, 9] + "',az11='" + w[0, 10] + "',az12='" + w[0, 11] + "',az13='"
+ w[0, 12] + "',az14='" + w[0, 13] + "',az15='" + w[0, 14] + "',az16='" + w[0,
15] + "',az17='" + w[0, 16] + "',az18='" + w[0, 17] + "',az19='" + w[0, 18] +
"',az20='" + w[0, 19] + "',az21='" + w[0, 20] + "',az22='" + w[0, 21] +
"',az23='" + w[0, 22] + "',az24='" + w[0, 23] + "',az5='" + w[0, 24] + "',az26='"
+ w[0, 25] + "',az27='" + w[0, 26] + "',az28='" + w[0, 27] + "' WHERE
idbobot='1'";
MySqlCommand mysqlCmd = new MySqlCommand(strSQL, con);
mysqlCmd.ExecuteNonQuery();
con.Close();
```

Proses perubahan salah satu nilai bobot jika tidak sesuai dengan kelasnya pada bahasa C# adalah sebagai berikut:

```
//ubah bobot w1
for (int i = 0; i <= 27; i++)
    {
      w[0, i] = w[0, i] - lr * (x[a, i] - w[0, i]);
     }
con.Open();
string strSQL = "UPDATE bobot SET az1='" + w[0, 0] + "',az2='" + w[0, 1] +
"',az3='" + w[0, 2] + "',az4='" + w[0, 3] + "',az5='" + w[0, 4] + "',az6='" +
w[0, 5] + "',az7='" + w[0, 6] + "',az8='" + w[0, 7] + "',az9='" + w[0, 8] +
"',az10='" + w[0, 9] + "',az11='" + w[0, 10] + "',az12='" + w[0, 11] + "',az13='"
+ w[0, 12] + "',az14='" + w[0, 13] + "',az15='" + w[0, 14] + "',az16='" + w[0,
15] + "',az17='" + w[0, 16] + "',az18='" + w[0, 17] + "',az19='" + w[0, 18] +
"',az20='" + w[0, 19] + "',az21='" + w[0, 20] + "',az22='" + w[0, 21] +
"',az23='" + w[0, 22] + "',az24='" + w[0, 23] + "',az5='" + w[0, 24] + "',az26='"
+ w[0, 25] + "',az27='" + w[0, 26] + "',az28='" + w[0, 27] + "' WHERE
idbobot='1'";
```

```
MySqlCommand mysqlCmd = new MySqlCommand(strSQL, con);
mysqlCmd.ExecuteNonQuery();
con.Close();
```

4.2.7 Identifikasi objek

Pada tahap perancangan identifikasi objek berdasarkan nilai ciri dilakukan dengan cara mengukur dan membandingkan jarak suatu vektor input objek yang belum diketahui jenisnya dengan vektor bobot setiap jenis objek yang telah didapatkan melalui proses pelatihan.

$$d_{ij} = \sqrt{(x_1 - w_1)^2 + (x_2 - w_2)^2 + \cdots + \left(x_i - w_j\right)^2}$$

dengan:

d_{ij} = euclidean distance input(X_i) dengan bobot(W_j)

x_1 = nilai vektor input ke-1

x_2 = nilai vektor input ke-2

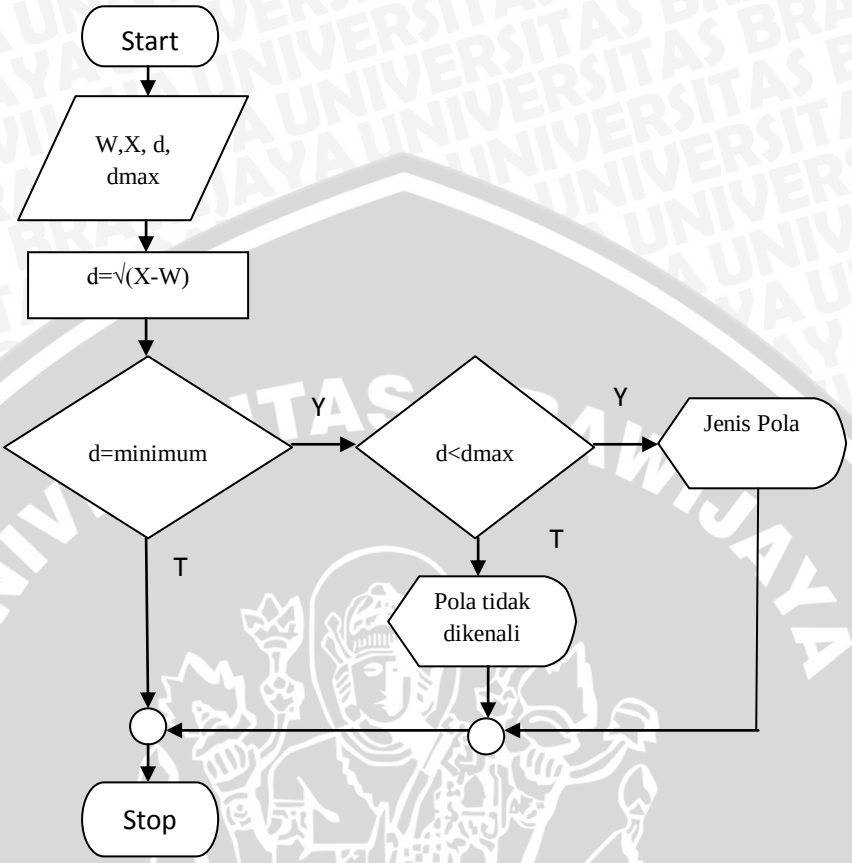
x_i = nilai vektor input ke-i

w_1 = nilai vektor bobot ke-1

w_2 = nilai vektor bobot ke-2

w_j = nilai vektor bobot ke-j

Pengukuran jarak dilakukan dengan menggunakan metode euclidean distance. Jarak terkecil vektor objek input yang belum dikenali dengan vektor bobot jenis objek Jika jarak terkecil tersebut melewati nilai maksimum maka objek tidak akan dikenali, hal ini seperti yang dijelaskan pada gambar 4.11.



Gambar 4.11 Flowchart identifikasi Objek

(**Sumber:** Perancangan)

Implementasi proses identifikasi objek diawali dengan melakukan deklarasi variabel yang menyimpan perhitungan toleransi jarak. Deklarasi yang dilakukan pada bahasa C# adalah sebagai berikut:

```
double ed1max;
double ed2max;
double ed3max;
```

Selanjutnya adalah memanggil data *testing* yang berupa hasil ekstraksi ciri pada fase pengujian yang telah disimpan dalam array. Pemanggilan data *testing* yang terjadi pada bahasa C# adalah sebagai berikut:

```
tes[0, 0] = z1;
tes[0, 1] = z2;
tes[0, 2] = z3;
tes[0, 3] = z4;
tes[0, 4] = z5;
tes[0, 5] = z6;
tes[0, 6] = z7;
tes[0, 7] = z8;
```

```
        tes[0, 8] = z9;
        tes[0, 9] = z10;
        tes[0, 10] = z11;
        tes[0, 11] = z12;
        tes[0, 12] = z13;
        tes[0, 13] = z14;
        tes[0, 14] = z15;
        tes[0, 15] = z16;
        tes[0, 16] = z17;
        tes[0, 17] = z18;
        tes[0, 18] = z19;
        tes[0, 19] = z20;
        tes[0, 20] = z21;
        tes[0, 21] = z22;
        tes[0, 22] = z23;
        tes[0, 23] = z24;
        tes[0, 24] = z25;
        tes[0, 25] = z26;
        tes[0, 26] = z27;
        tes[0, 27] = z28;
```

Setelah itu melakukan proses perhitungan jarak *euclidean* terhadap nilai bobot yang akan menentukan jenis daripada objek uji. Berikut ini adalah implementasi perhitungan jarak *euclidean* pada bahasa C#:

```
for (int j = 0; j < 28; j++)
{
  ed1 = ed1 + ((tes[a, j] - w[0,j]) * (tes[a, j] - w[0,j]));
  ed2 = ed2 + ((tes[a, j] - w[1,j]) * (tes[a, j] - w[1,j]));
  ed3 = ed3 + ((tes[a, j] - w[2,j]) * (tes[a, j] - w[2,j]));
}
//akar
ed1 = Math.Sqrt(ed1);
ed2 = Math.Sqrt(ed2);
ed3 = Math.Sqrt(ed3);
```

Untuk menentukan jenis objek harus dilakukan suatu pemeriksaan jarak terkecil terhadap salah satu nilai bobot. Selain itu juga masih dilakukan pemeriksaan toleransi terhadap nilai jarak tersebut. Berikut ini adalah implementasi salah satu penentuan jenis objek pada bahasa C#:

```
if (ed1 < ed2 && ed1 < ed3)
{
  con.Open();
  string strSQL = "SELECT MAX(ed1) AS ed1max FROM ed1";
  MySqlCommand mysqlCmd = new MySqlCommand(strSQL, con);
  mysqlCmd.ExecuteNonQuery();
  MySqlDataReader reader;
  reader = mysqlCmd.ExecuteReader();

  while (reader.Read())
  {
    ed1max=(double)reader["ed1max"];
  }
  connect.Close();


  if (ed1 <= ed1max)
```

```
{
  kun1 = 1;
  textBox2.Text = "1";
}
else
{
  textBox2.Text = "X";
}
```

4.3 Implementasi Sistem

Setelah tahap perancangan tahap selanjutnya adalah tahap implementasi. Implementasi ini merupakan proses transformasi hasil perancangan perangkat lunak yang telah dibuat ke dalam kode *(coding)* sesuai dengan sintaks dari bahasa pemrograman yang digunakan.

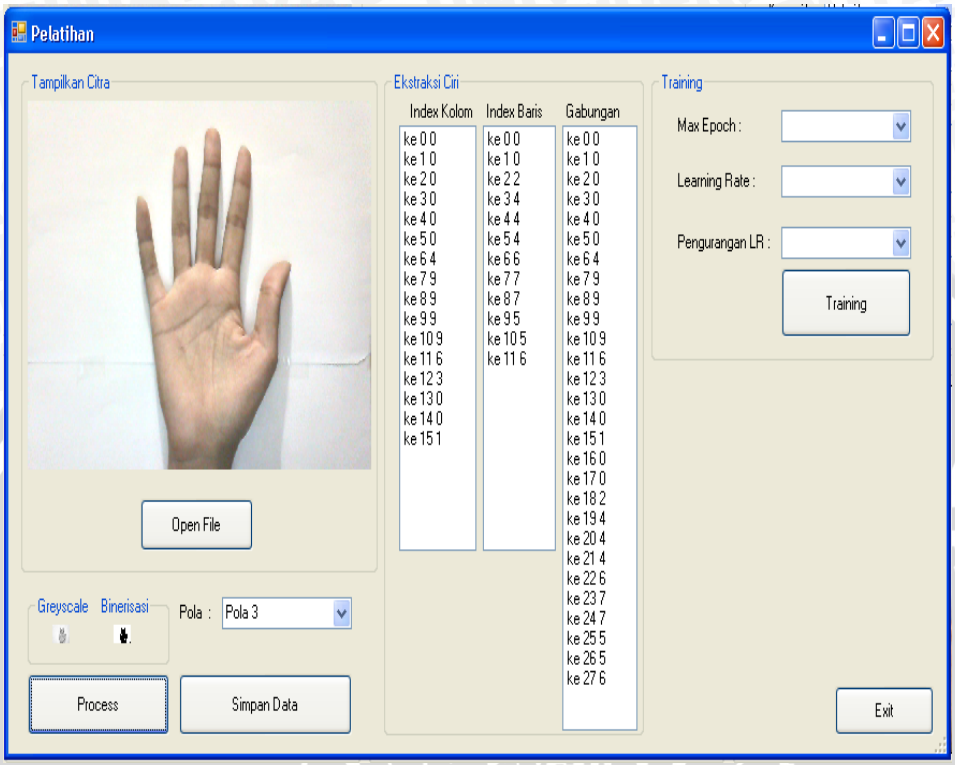
4.3.1 Lingkungan Implementasi

Aplikasi dibuat dengan menggunakan Microsoft Visual Studio C# dengan Aforge .NET framework. Sistem diimplementasikan dengan menggunakan spesifikasi sebagai berikut:

1. Perangkat keras :
   A. Laptop
      Spesifikasi :
      - Processor : Intel Core™ i3 M 350 2.27GHz
      - Memory : 880 MB RAM
      - OS : Windows XP Service Pack 3
      - VGA : Mobile Intel 965 Express Chipset Family
2. Perangkat Lunak :
   - Sistem operasi : Microsoft Windows XP Service Pack
   - Bahasa pemrograman: Microsoft Visual Studio C# .NET 2010
   - Framework : Aforge .NET Framework
   - Basis data : MySQL
   - Tools : XAMPP 1.7.2

4.4 Implementasi Antarmuka

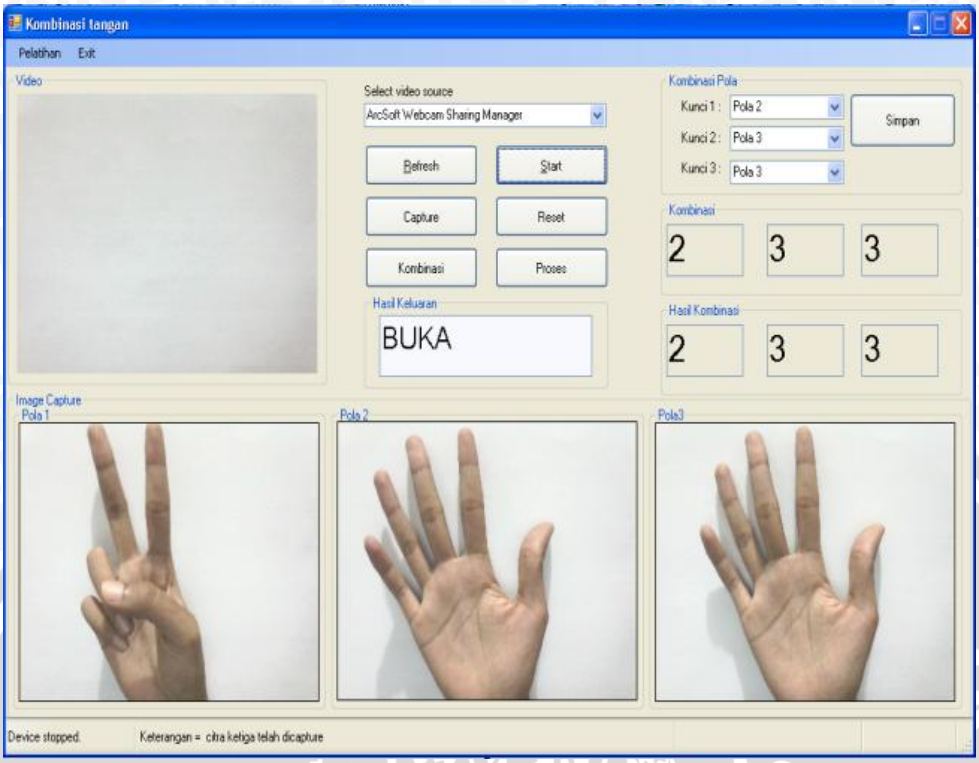
Aplikasi identifikasi kombinasi pola isyarat tangan memiliki tampilan seperti pada gambar dibawah. Aplikasi ini memiliki 2 tampilan yang menampilkan informasi berbeda sesuai fungsinya.



Gambar 4.12 Tampilan aplikasi pada tab Pelatihan

(**Sumber:** perancangan)

Pada gambar 4.12 adalah tampilan pada tab Pelatihan dimana terdapat 1 buah picturebox yang berfungsi untuk menampilkan citra objek input yang kita pilih guna dilakukan pelatihan. Lalu terdapat tombol "*Process"* untuk mengekstraksi ciri dari citra yang dtampilkan. Untuk dapat melakukan penyimpanan data suatu objek, maka pengguna dapat menggunakan fasilitas tombol "Simpan Data". Bagian selanjutnya adalah pelatihan, dimana pengguna dapat melakukan proses pelatihan LVQ dengan mengatur terlebih dahulu nilai *maxepoh* dan *learning rate* dan *decreasing learning rate* kemudian menggunakan fasilitas tombol "Training".



Gambar 4.13 Tampilan aplikasi pada tab Identifikasi

(**Sumber:** perancangan)

Pada gambar 4.13 adalah tampilan pada tab utama. Pada tab ini terdapat 4 buah picturebox, yang salah satu picturebox digunakan sebagai penampil video. Kemudian terdapat 7 tombol yang berfungsi sebagai berikut :

1. Refresh = Untuk menampilkan device camera yang akan dipakai.
2. Start =Untuk menampilkan video pada picturebox,setelah device camera telah dipilih.
3. Capture =Untuk mengambil citra menggunakan timer untuk ditampilkan di 3 picturebox yang lain.
4. Reset =Untuk membersihkan tampilan pada 3 picturebox yang berisi citra yang telah di*capture.*
5. Kombinasi =Untuk menampilkan kombinasi yang telah dimasukkan.
6. Proses = ntuk memproses citra yang telah di*capture*.
7. Simpan =Untuk menyimpan kombinasi yang telah dipilih sebelumnya.

BAB V
PENGUJIAN

Rancangan yang telah diimplementasikan dalam bentuk nyata memerlukan suatu pengujian. Uji coba ini adalah cara untuk mengetahui hasil dari penelitian yang dilakukan sekaligus sebagai sarana pemunculan ide-ide bagi proses pengembangan selanjutnya. Pengujian yang dilakukan dalam bab ini adalah sebagai berikut:

1. Pengujian ekstraksi ciri
2. Pengujian Pengenalan Isyarat
3. Pengujian Kombinasi
4. Analisis faktor Kegagalan

5.1 Pengujian Ekstraksi Ciri

Pengujian ini bertujuan untuk mengetahui hasil pengambilan gambar objek melalui kamera, dan ekstraksi ciri telah bekerja sesuai yang diharapkan atau tidak.

Setelah dilakukan pengujian diketahui bahwa pengambilan citra yang bagus adalah saat penerangan yang konstan, karena jika penerangannya kurang atau berlebihan maka akan menyebabkan eror pada proses selanjutnya.

Hasil ekstraksi ciri pada 60 objek pelatihan dapat dilihat pada tabel 5.1. Tabel 5.1 menunjukkan range nilai *ecludien distance* min dan max masing – masing pola.

Tabel 5.1 Nilai *min dan max*

No.	Jenis Objek	Minimum	Maximum
1.	Pola tangan ke 1	5.124206373	36.42151826
2.	Pola tangan ke 2	6.288200214	34.75036549
3.	Pola tangan ke 3	7.034801162	31.83761053

(**Sumber:** Pengujian**)**

Dari tabel 5.1 tampak bahwa terdapat selisih antara nilai maksimum dan nilai minimum untuk setiap ciri objek. Hal ini dikarenakan pada satu jenis objek yang diambil memiliki pola yang berbeda namun tetap pada ciri dari jenis objek tersebut.

5.2 Pengujian Pengenalan Isyarat

Pengujian identifikasi ini dilakukan dengan tujuan mengetahui tingkat keberhasilan aplikasi dalam mengenali objek pada 3 pola yang diujikan. Sebanyak 20 citra untuk masing-masing citra yang berisi kumpulan objek yang akan diuji merupakan objek yang jenisnya masuk dalam pelatihan yaitu citra dengan komposisi parameter maksimal epoh=30, *learning rate*=0.6, dan pengurangan *learning rate*=0. 1.

Untuk pola pertama didapatkan bahwa aplikasi dapat mengenali citra objek dengan cukup baik atau memiliki tingkat keberhasilan pengenalan objek sebesar 80%. Tabel dibawah menunjukkan beberapa contoh hasil pengujian identifikasi terhadap objek.

Gambar 5.1 Citra uji no.1
(**Sumber:** Pengujian)

Tabel 5.2 Contoh hasil pengujian identifikasi citra no.1
(keberhasilan identifikasi 80%)

No.	Jenis Objek	Hasil Identifikasi	Keterangan
1.	Pola 1	Pola 1	Sesuai
2		Pola 1	Sesuai
3		Pola 1	Sesuai
4		Pola 1	Sesuai
5		Pola 1	Sesuai
6		Pola 2	Tidak Sesuai
7		Pola 1	Sesuai
8		Pola 1	Sesuai
9		Pola 3	Tidak Sesuai
10		Pola 1	Sesuai

(**Sumber:** Pengujian)

Untuk pola kedua didapatkan bahwa aplikasi dapat mengenali citra objek dengan baik atau memiliki tingkat keberhasilan pengenalan objek sebesar 80%. Tabel dibawah menunjukkan beberapa contoh hasil pengujian identifikasi terhadap objek.

Gambar 5.2 Citra uji no.2

(**Sumber:** Pengujian)

Tabel 5.3 Contoh hasil pengujian identifikasi citra no.2

(keberhasilan identifikasi 80%)

No.	Jenis Objek	Hasil Identifikasi	Keterangan
1.	Pola 2	Pola 2	Sesuai
2		Pola 2	Sesuai
3		Pola 3	Tidak Sesuai
4		Pola 2	Sesuai
5		Pola 2	Sesuai
6		Pola 2	Sesuai
7		Pola 2	Sesuai
8		Pola 2	Sesuai
9		Pola 2	Sesuai
10		Pola 3	Tidak Sesuai

(**Sumber:** Pengujian)

Untuk pola ketiga didapatkan bahwa aplikasi dapat mengenali citra objek dengan baik atau memiliki tingkat keberhasilan pengenalan objek sebesar 100%. Tabel dibawah menunjukkan beberapa contoh hasil pengujian identifikasi terhadap objek.

Gambar 5.3 Citra uji no.3

(**Sumber:** Pengujian)

Tabel 5.4 Contoh hasil pengujian identifikasi citra no.3
(keberhasilan identifikasi 100%)

No.	Jenis Objek	Hasil Identifikasi	Keterangan
1.	Pola 3	Pola 3	Sesuai
2		Pola 3	Sesuai
3		Pola 3	Sesuai
4		Pola 3	Sesuai
5		Pola 3	Sesuai
6		Pola 3	Sesuai
7		Pola 3	Sesuai
8		Pola 3	Sesuai
9		Pola 3	Sesuai
10		Pola 3	Sesuai

(**Sumber:** Pengujian)

5.3 Pengujian Kombinasi

Pengujian identifikasi ini dilakukan dengan tujuan mengetahui tingkat keberhasilan aplikasi dalam mengenali kombinasi pola isyarat tangan apa sudah sesuai dengan kombinasi yang diinginkan, sehingga dapat mengeluarkan perintah yang dikehendaki. Terdapat 10 Kombinasi yang diujikan.

Setelah dilakukan pengujian didapatkan bahwa aplikasi dapat sesuai dengan yang diinginkan yang memiliki tingkat keberhasilan pengenalan objek sebesar 90%. dengan nilai maksimal epoh=30, *learning rate*=0.6, dan pengurangan *learning rate*=0.

Gambar 5.4 Citra uji kombinasi no.3, no.3, no.2
(**Sumber:** Pengujian)

Tabel 5.5 Contoh hasil pengujian identifikasi kombinasi
(keberhasilan identifikasi 90%)

No.	Kombinasi	Hasil Identifikasi	Keterangan
1.	1 3 1	1 3 1	Buka
2	2 1 2	2 1 2	Buka
3	3 2 3	3 2 3	Buka
4	3 1 3	3 1 3	Buka
5	3 2 2	3 2 2	Buka
6	1 3 2	3 3 2	Gagal
7	2 3 2	2 3 2	Buka
8	2 1 1	2 1 1	Buka
9	2 3 3	2 3 3	Buka
10	3 3 3	3 3 3	Buka

5.4 Analisis Faktor Kegagalan

Pada kenyataannya aplikasi identifikasi pola isyarat tangan ini tidak selalu berjalan secara optimal. Ada beberapa faktor yang dapat menyebabkan proses yang dilakukan menjadi gagal. Berikut adalah beberapa faktor-faktor yang menyebabkan kegagalan tersebut.

1. Pencahayaan yang berbeda antara proses pengambilan data training dengan data online akan mempengaruhi tingkat kecerahan warna yang dihasilkan pada citra yang diambil. Tingkat pencahayaan yang berbeda ini disebabkan oleh penggunaan kualitas *webcam* yang berbeda.
2. Jumlah data sample akan mempengaruhi tingkat keragaman data yang akan dijadikan data training. Semakin banyak jumlah data training yang digunakan, semakin baik titik vektor bobot referensi yang dihasilkan. Dengan catatan bahwa data dalam satu kelompok yang diambil tidak mempunyai perbedaan yang mencolok. Karena akan mempengaruhi pengelompokan data.

BAB VI
PENUTUP

6.1 Kesimpulan

Berdasarkan hasil perancangan, implementasi, pengujian dan analisis sistem maka dapat diambil kesimpulan sebagai berikut:

1. Ektraksi ciri dilakukan dengan menggunakan integral proyeksi dan pengelompokan menggunakan LVQ telah memperoleh tingkat keberhasilan pengenalan pola sebesar 80% untuk pola 1, 80% untuk pola 2, 100% untuk pola 3, dan 90% untuk kombinasi pola.
2. Pola isyarat tangan manusia dapat dikenali menggunakan *euclidean distance* dengan membandingkan jarak nilai masukan dengan nilai bobot.

6.2 Saran

Dalam perancangan dan pembuatan aplikasi identifikasi kombinasi pola isyarat tangan untuk menjalankan perintah sebuah aplikasi komputer masih terdapat kekurangan dan kelemahan, oleh karena itu diperlukan penyempurnaan untuk pengembangan selanjutnya. Berikut ini adalah beberapa hal yang perlu diperhatikan untuk pengembangan:

1. Diperlukan suatu metode ekstraksi ciri yang lebih baik daripada integral proyeksi untuk memperbesar persentase keberhasilan untuk membedakan ciri suatu objek.
2. Penambahan citra objek dapat menghasilkan identifikasi yang lebih akurat.
3. Penggunaan metode lain selain *LVQ* dalam pengambilan keputusan.
4. Diharapkan sistem ini dalam pengembangan lebih lanjut mampu mengenali isyarat tangan dengan latar belakang yang lebih kompleks.

DAFTAR PUSTAKA

[1] Anonim. http://id.wikipedia.org/wiki/Tangan (Akses tanggal 10 Januari 2012).

[2] Anonim. http://id.wikipedia.org/wiki/Isyarat_Tangan (Akses tanggal 13 Januari 2012).

[3] Gunawan, Budi," Deteksi Isyarat Tangan Oleh Komputer Dengan Image Processing", Universitas Muria Kudus, 2009.

[4] Wahyono, Eko Sri, *"Makalah Implementasi Metode Jaringan Syaraf Buatan Learning Vector Quantization Pada Identifikasi Citra"* Universitas Gunadarma, 2009

[5] Novrandy, Bherly, "*APLIKASI TRANSFORMASI CITRA MENGGUNAKAN MATLAB 6.5."* Universitas Gunadarma, 2008

[6] Hestiningsih, Idhawati," PENGOLAHAN CITRA " 2009

[7] Difla Yustisia Qur'ani, "*JARINGAN SYARAF TIRUAN LEARNING VECTOR QUANTIZATION UNTUK APLIKASI PENGENALAN TANDA TANGAN"* Universitas Islam Indonesia, 2010

[8] Setiabudi, Agus, "PENGENALAN ISYARAT TANGAN UNTUK INTERAKSI MESIN PENJUAL OTOMATIS DENGAN MENGGUNAKAN METODE JARINGAN SYARAF TIRUAN" PENS Surabaya, 2007

[9] Talibo, Lukman. *"Sistem Pengenalan Obyek Real-Time Dengan Jaringan Syaraf Tiruan Backpropagation"*. Teknik Informatika UNIKOM, 2004.

[10] Firdausy, Kartika dan Kholis, M. Hana Mirza, " *PURWARUPA SISTEM DETEKSI WAKTU – NYATA BERBASIS LAYANAN PESAN SINGKAT*" Universitas Ahmad Dahlan, 2010

LAMPIRAN

Lampiran 1. Citra objek untuk pelatihan