IMPLEMENTASI JARINGAN SYARAF TIRUAN UNTUK MENGENALI JALUR JALAN KENDARAAN DARAT BERBASIS KAMERA

SKRIPSI

Disusun Oleh :

MARTHALIA DEWI ANGGRAINI
06.12.629

JURUSAN TEKNIK ELEKTRO S-1
KONSENTRASI TEKNIK KOMPUTER & INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG
2011

LEMBAR PERSETUJUAN

IMPLEMENTASI JARINGAN SYARAF TIRUAN UNTUK MENGENALI JALUR JALAN KENDARAAN DARAT BARBASIS KAMERA

SKRIPSI

Disusun dan diajukan untuk melengkapi dan memenuhi persyaratan guna mencapai gelar Sarjana Teknik

Disusun Oleh :

Marthalia Dewi Anggraini

NIM. 0612629

Mengetahui,

Ketua Jurusan Teknik Elektro S-1

Ir. Yusuf Ismail Nakhoda, MT

NIP.Y.1018800189

Diperiksa dan Disetujui,

Dosen Pembimbing I	Dosen Pembimbing II
Ir. Yusuf Ismail Nakhoda, MT	Dr. Eng. Aryuanto Soetedjo, ST, MT
NIP.Y.1018800189	NIP.P.1030800417

JURUSAN TEKNIK ELEKTRO S-1

KONSENTRASI TEKNIK KOMPUTER DAN INFORMATIKA

FAKULTAS TEKNOLOGI INDUSTRI

INSTITUT TEKNOLOGI NASIONAL MALANG

ABSTRAK

IMPLEMENTASI JARINGAN SYARAF TIRUAN UNTUK MENGENALI JALUR JALAN KENDARAAN DARAT BERBASIS KAMERA

Marthalia Dewi Anggraini,NIM 0612629
Dosen Pembimbing : Ir. Yusuf Ismail Nakhoda, MT dan Dr. Eng. Aryuanto Soetedjo,ST, MT.

Jaringan saraf tiruan adalah teknik pembelajaran yang meniru cara kerja sistem saraf manusia. Jaringan saraf tiruan dapat menyelesaikan dengan performansi cukup baik permasalahan – permasalahan kompleks yang tidak dapat ditangani algoritma konvensional. Salah satu contohnya adalah persoalan sistem kemudi otomatis bagaimana mobile robot dapat berjalan di lingkungan tertentu dengan baik. Dalam mengemudi kendaraan, seorang pengemudi akan mengikuti garis jalan yang ada dan perbedaan kualitas hasil sensor dipengaruhi oleh kondisi lingkungan intensitas cahaya yang rendah pada saat cuaca berawan dan adanya bayangan yang mempengaruhi kualitas hasil sensor yang akhirnya juga mempengaruhi keputusan dari mobile robot. Tujuan dirancang sistem ini untuk mengenali jalur jalan kendaraan darat yang dibangun dari pengambilan gambar kondisi jalan darat dan diolah melalui proses pengolahan citra dijital yang kemudian dilakukan pengambilan presentase kebenaran melalui jaringan saraf tiruan dengan metode backpropagation. Aplikasi ini dijalankan menggunakan program Matlab 7.04.

Kata Kunci: Jaringan Syaraf Tiruan, backpropagation, citra

KATA PENGANTAR

Puji syukur kehadirat Tuhan Yang Maha Esa atas segala Kasih dan Anugerah–Nya, yang telah memberikan kekuatan, kesabaran, bimbingan dan perlindungan sehingga penulis dapat menyelesaikan laporan skripsi dengan judul :

"IMPLEMENTASI JARINGAN SYARAF TIRUAN UNTUK MENGENALI JALUR JALAN KENDARAAN DARAT BERBASIS KAMERA "

Pembuatan skripsi ini disusun guna memenuhi syarat akhir kelulusan pendidikan jenjang Strata-1 di Institut Teknologi Nasional Malang. Dalam penyusunan skripsi ini penulis banyak mendapat bantuan baik moril maupun materiil, saran dan dorongan semangat dari berbagai pihak, untuk itu penulis mengucapkan terima kasih kepada :

1. Bapak Prof. Dr. Ir. Abraham Lomi, MSEE selaku rektor ITN Malang
2. Bapak Ir. H. Sidik Noertjahjono, MT selaku Dekan Fakultas Teknologi Industri.
3. Bapak Ir. Yusuf Ismail Nakhoda, MT selaku Ketua Jurusan Teknik Elektro S-1 ITN Malang.
4. Bapak Ir. Yusuf Ismail Nakhoda, MT selaku Dosen Pembimbing I.
5. Bapak Dr. Eng. Aryuanto Soetedjo, MT selaku Dosen Pembimbing II.
6. Kedua orang tua penulis yang selalu memberikan semangat, dorongan dan doa serta seluruh keluarga yang telah banyak memberikan bantuan baik moril maupun materi.
7. Rekan-rekan dan semua pihak yang tidak dapat penulis sebutkan satu per satu, yang telah membantu baik secara lansung maupun tidak dalam penyelesaian skripsi ini.

Penulis menyadari bahwa laporan skripsi ini masih banyak yang perlu disempurnakan. Oleh sebab itu kritik dan saran yang membangun sangat diharapkan.

Akhir kata, penulis mohon maaf kepada semua pihak bilamana selama penyusunan skripsi ini penyusun membuat kesalahan secara tidak sengaja dan semoga skripsi ini dapat bermanfaat bagi kita semua.

Malang, Februari 2011

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1 Latar Belakang

Keberadaan komputer semakin tidak dapat dipisahkan dari kehidupan manusia. Berbagai macam pekerjaan manusia dapat diselesaikan dengan bantuan komputer. Namun, ada persoalan – persoalan tertentu yang tidak dapat diselesaikan dengan kemampuan komputer biasa sehingga dibutuhkan lebih dari sekedar algoritma konvensional untuk menyelesaikan persoalan – persoalan tersebut. Salah satu contohnya adalah persoalan sistem kemudi otomatis bagaimana *mobile robot* dapat berjalan di lingkungan tertentu dengan baik.

Sebuah *mobile robot* harus memiliki kemampuan untuk menentukan arah gerakan yang akan dilakukan selanjutnya pada suatu kondisi. Hasil masukan sensor diolah sehingga menghasilkan aksi, misalnya arah gerakan dari *mobile robot.* Tingkat kesulitan untuk membuat *mobile robot* yang dapat menentukan arah gerakan, tergantung pada lingkungan dimana *robot* tersebut dijalankan. Salah satu contoh lingkungan yang cukup kompleks adalah jalan darat untuk kendaraan bermotor. *Mobile robot* pada lingkungan jalan darat sudah mulai diimplementasikan dalam bentuk mobil tanpa pengemudi.

Kesulitan – kesulitan pada sistem kemudi otomatis di lingkungan jalan darat antara lain :

1. Perbedaan jenis jalur jalan

   Dalam mengemudi kendaraan, seorang pengemudi akan mengikuti garis jalan yang ada. Terdapat berbagai jenis garis jalan, antara lain : garis jalan tengah sebagai pemisah jalur, garis jalan putus – putus, garis jalan solid, garis jalan pinggir sebagai batas jalan, dsb.
2. Perbedaan kualitas hasil sensor yang dipengaruhi kondisi lingkungan intensitas cahaya yang rendah pada saat cuaca berawan dan adanya bayangan yang mempengaruhi kualitas hasil sensor yang akhirnya juga mempengaruhi keputusan dari *mobile robot.*

   Berdasarkan kesulitan – kesulitan di atas maka dirancang sistem untuk mengenali jalur jalan kendaraan darat yang dibangun dari pengambilan gambar kondisi jalan darat dan diolah melalui proses pengolahan citra dijital yang kemudian dilakukan pengambilan presentase kebenaran melalui jaringan saraf tiruan.

Jaringan syaraf tiruan adalah teknik pembelajaran yang meniru cara kerja sistem syaraf manusia. Sistem manusia terdiri dari berjuta – juta neuron yang masing – masing memproses informasi secara pararel. Jaringan syaraf tiruan dapat menerima input yang bervariasi berdasarkan situasi dari permasalahan yang diambil. Jaringan syaraf tiruan dapat

menyelesaikan dengan performansi cukup baik permasalahan – permasalahan kompleks yang tidak dapat ditangani algoritma konvensional.

Dengan adanya Skripsi ini diharapkan dapat menjadi acuan sebagai pengembangan proses *study* selanjutnya seperti dibuatnya *mobile robot* pada lingkungan jalan darat, sehingga Skripsi ini merupakan dasar untuk mengenali jalur jalan kendaraan darat yang berbasiskan kamera.

1.2 Rumusan Masalah

Berdasarkan latar belakang di atas, maka permasalahan yang diangkat dari Skripsi ini adalah bagaimana proses pengolahan dan pengenalan pola pada citra serta pembelajaran dalam mengenali jalur jalan dengan menggunakan jaringan syaraf tiruan.

1.3 Tujuan

Tujuan yang ingin dicapai dalam Skripsi ini adalah untuk mengimplementasikan konsep penyelesaian masalah dengan menggunakan jaringan syaraf tiruan dalam mengenali jalur jalan kendaraan darat.

1.4 Batasan Masalah

Batasan masalah pada Skripsi ini adalah :

1. Lingkungan pengambilan gambar dibatasi pada jalan beraspal, kering, dan tidak berlubang.
2. Pengambilan gambar dilakukan pada siang hari dan tidak mendung.
3. Resolusi citra *input* 320x240 *pixels*

1.5 Metodologi

Metodologi yang diterapkan dalam pengerjaan Skripsi ini adalah sebagai berikut :

1. Studi literatur

   Dilakukan dengan cara mempelajari literatur – literatur baik yang berupa buku (*textbook*), jurnal, dan artikel maupun *website* yang berhubungan dengan jaringan syaraf tiruan dan citra dijital.
2. Analisa Kebutuhan Sistem

   Dalam tahap ini dilakukan analisis mengenai bagaimana rancangan jaringan syaraf tiruan yang digunakan untuk permasalahan mengenali jalur jalan kendaraan darat yang

meliputi jumlah *layer*, spesifikasi *input*, spesifikasi *output*, serta pemrosesan lainnya yang dibutuhkan.

3. Perancangan dan Implementasi Perangkat Lunak

   Dalam tahap ini perancangan perangkat lunak meliputi perancangan kelas dan antarmuka dari perangkat lunak yang dibuat yang berupa data jalan.

4. Pengujian

   Pada tahapan pengujian ini menganalisa performansi dari jaringan syaraf tiruan dalam menyelesaikan persoalan dalam mengenali jalur jalan kendaraan darat berbasis citra dijital

1.6 Sistematika Penulisan

Penyusunan laporan Skripsi ini menggunakan kerangkan pembahasan yang terbentuk dalam susunan bab, yang dapat dijelaskan sebagai berikut :

BAB I PENDAHULUAN

Bab ini merupakan dasar penyusunan laporan Skripsi yang di dalamnya berisi tentang latar belakang masalah, rumusan masalah, tujuan Skripsi, batasan masalah, metodologi pengembangan sistem, dan sistematika pembahasan Skripsi.

BAB II LANDASAN TEORI

Bab ini berisi tentang dasar teori yang digunakan dalam analisis, perancangan, dan implementasi Skripsi, diantaranya penjabaran mengenai citra dijital dan jaringan syaraf tiruan.

BAB III PERANCANGAN SISTEM

Bab ini berisi tentang analisis dalam mengolah citra yang berupa gambar jalan dan analisis terhadap jaringan syaraf tiruan dalam proses pembelajaran dalam mengenali jalur jalan.

BAB IV IMPLEMENTASI DAN PENGUJIAN SISTEM

Bab ini berisi tentang perancangan perangkat lunak dengan menggunakan perangkat lunak Matlab, hasil pengujian dan pengimplementasi program dalam mengenali jalur jalan dengan metode pembelajaran jaringan syaraf tiruan menggunakan Matlab.

BAB V PENUTUP

Bab ini merupakan bab terakhir dari laporan skripsi ini yang berisi tentang kesimpulan dari hasil pembahasan pada perancangan dan pengujian akhir sistem yang dibuat serta saran-saran untuk penyempurnaan dalam pengembangannya.

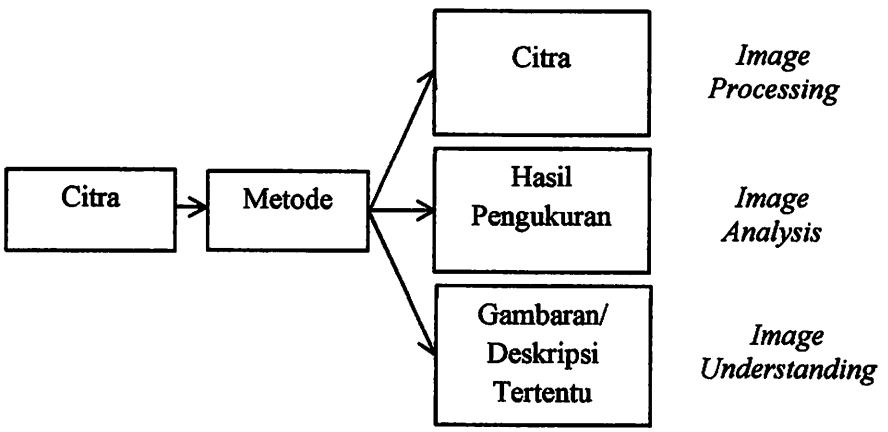
BAB II
LANDASAN TEORI

2.1 Pengolahan Citra Dijital

Secara umum, terminologi pengolahan citra dijital mengacu pada proses pengolahan / manipulasi dari citra dua dimensi dengan menggunakan komputer . Jika ditinjau dari sudut pandang yang lain, pengertian tersebut menyatakan pemrosesan secara dijital dari berbagai data dua dimensi.

Pengolahan citra dijital memiliki bidang aplikasi yang cukup luas, seperti halnya *remore sensing* melalui satelit, transmisi dan penyimpanan citra untuk berbagai aplikasi bisnis dan bidang lainnya, pengolahan citra medis, dan bidang – bidang terkait lainnya.

Teknologi dijital modern sekarang ini telah memungkinkan proses manipulasi sinyal multi-dimensi menggunakan sistem yang sederhana sampai sistem yang canggih sekalipun. Tujuan dari proses manipulasi ini dapat dibagi menjadi tiga buah kategori dan ditunjukkan dalam Gambar 2.1.



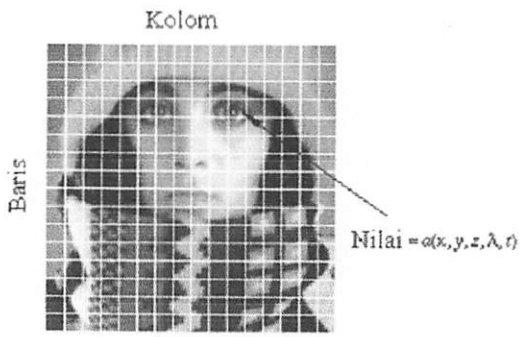
Gambar 2.1. Diagram dasar sistem pengolahan citra

Dalam proses manipulasi citra, diawali dengan proses *image processing* yaitu dengan memperbaiki kulitas suatu gambar, sehingga dapat lebih mudah diinterpretasi oleh mata manusia, yang dilanjutkn dengan *image analysis* yaitu mengolah informasi yang terdapat dalam suatu gambar untuk keperluan pengenalan objek secara otomatis. Pengenalan pola tidak hanya dengan bertujuan untuk mendapatkan citra dengan suatu kualitas tertentu, tetapi juga untuk mengklasifikasikan bermacam-macam citra. Dari sejumlah citra diolah sehingga citra dengan ciri yang sama akan dikelompokkan pada suatu kelompok tertentu. Interpretasi meliputi penekanan dalam mengartikan objek yang dikenali.

2.1.1 Definisi Citra Dijital

Sebuah citra didefinisikan sebagai fungsi dua dimensi *f (x,y)*, dimana *x* dan *y* merupakan koordinat spasial (ruang) dan amplitudo fungsi *f* pada pasangan koordinat *(x,y)*adalah intensitas atau nilai derajat keabuan *(gray level)* citra pada titik tersebut. Jika komponen – komponen *x,y* dan nilai amplitudo *f* merupakan bilangan yang diskrit dan terbatas, maka citra tersebut adalah citra dijital. Nilai amplitudo citra selalu berupa bilangan riil atau bilangan bulat (karena biasanya merupakan hasil proses kuantitasi).

Bidang pengolahan citra dijital mengacu pada pemrosesan citra- citra dijital menggunakan komputer. Suatu citra dijital terdiri dari jumlah elemen yang terbatas, dimana setiap elemen tersebut memiliki lokasi dan nilai tertentu. Elemen – elemen tersebut dikenal dengan elemen citra (piksel). Suatu citra dijital *a [m,n]* dideskripsikan dalam ruang diskrit dua dimensi yang diturunkan dari citra analog *a (x,y)* dalam ruang kontinu dua dimensi melalui proses sampling yang sering dikenal dengan proses dijitalisasi. Citra kontinu dua dimensi *a (x,y)* dibagi menjadi *N* baris dan *M* kolom. Nilai yang diberikan pada koordinat integer *[m.n]* dengan *{m=0,1,2,....,M-1}* dan *{n=0,1,2,...N-1}* adalah nilai dari *a[m,n].* Biasanya komponen – komponen *a(x,y)* merupakan fungsi dari beberapa variabel, termasuk kedalaman (*depth/z*), warna (λ), dan waktu (*t*).



Gambar 2.2. Proses dijitalisasi dari suatu citra kontinu

Citra yang ditampilkan pada gambar 2.2 telah dibagi menjadi $N = 16$ dan $M = 16$ kolom. Proses untuk merepresentasikan amplitudo sinyal dua dimensi pada koordinat yang diberikan sebagai nilai integer dengan tingkat derajat keabu – abuan L sering dianggap sebagai proses kuantisasi. Nilai dari derajat keabu- abuan yang jelas terlihat merupakan bilangan kelipatan pangkat dua, sehingga $L = 2^B$, dimana B merupakan jumlah bit dalam representasi bilangan biner dari tingkat *brightness.* Jika $B > 1$, maka citra tersebut dinamakan citra derajat abu – abu (*grayscale*), jika $B = 1$ maka citra tersebut adalah biner. Pada citra

biner, hanya ada dua nilai derajat keabu – abuan, yaitu "hitam" dan "putih" direpresentasikan dengan "0" dan "1".

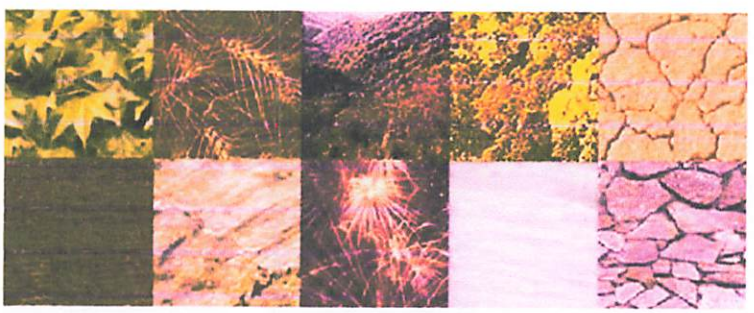
Gambar 2.3. Contoh citra dijital; citra berwarna (RGB); citra biner;citra keabuan (*grayscale*)

2.2 Tekstur

2.2.1 Teori Tekstur

Secara umum, tekstur seringkali menyediakan berbagai sumber informasi visual yang alamiah. Tekstur merupakan sesuatu yang sangat menarik, tidak hanya karena merupakan komponen penting dalam analisis citra untuk proses pengenalan (*recognition*), segmentasi dan sintesis, akan tetapi dapat berperan sebagai alat bantu untuk memahami mekanisme dasar dari persepsi visual manusia.

Tekstur merupakan karakteristik intrinsik suatu jenis citra yang berhubungan dengan tingkat kekasaran (*roughness*) dan keteraturan (*regularity*) susunan struktural dari piksel citra. Aspek-aspek tersebut dapat dimanfaatkan untuk proses segmentasi, klasifikasi, maupun interpretasi citra.



Gambar 2.4. Contoh tekstur alami (*natural*)

Gambar 2.5. Contoh tekstur reguler buatan (*artificial*)

Obyek-obyek yang memiliki karakteristik tekstural biasanya diamati sebagai obyek buatan (*artificial*) maupun alami (*natural*). Contohnya adalah tekstur - tekstur pada kayu, tumbuh-tumbuhan, material, dan kulit.

Walaupun tekstur merupakan suatu bidang penelitian yang cukup penting, tetapi belum ada definisi yang benar-benar pasti untuk merepresentasikan tekstur. Alasan utamanya adalah tekstur-tekstur alami sering menampilkan sifat-sifat yang saling bertentangan, seperti misalnya *regularity* dengan *randomness uniformity* dengan *distortion* , yang agak sulit untuk dideskripsikan dalam aturan yang seragam. Walaupun begitu, beberapa peneliti memiliki definisi sendiri mengenai tekstur sesuai dengan aplikasi yang sedang dikerjakannya.

2.2.2 Analisis Tekstur

Ada empat buah kategori besar dalam bidang analisis tekstur, yaitu:

1. Ekstraksi fitur: menghitung suatu karakteristik dari citra dijital yang dapat mendeskripsikan sifat-sifat teksturalnya secara numerik.
2. Segmentasi tekstur: memilah-milah suatu citra bertekstur menjadi beberapa daerah, dimana setiap daerah tersebut berhubungan dengan tekstur – tekstur yang homogen.
3. Klasifikasi tekstur: untuk menentukan kelompok dari tekstur-tekstur homogen menuju sejumlah kelas yang sudah didefinisikan.
4. Pembentukan obyek dari tekstur: untuk merekonstruksi geometri permukaan tiga dimensi (atau obyek dengan dimensi yang lebih tinggi) dari berbagai informasi tekstural.

Di dalam proses yang umum dilakukan, biasanya tahap ekstraksi fitur merupakan tahapan pertama dari proses analisis citra tekstural dan hasilnya akan digunakan untuk proses selanjutnya.

Analisis tekstur memiliki peran yang cukup penting pada banyak aplikasi pengolahan citra, mulai dari metode penginderaan jauh sampai pencitraan medis. Tujuan utama dari penelitian

tentang tekstur adalah untuk memahami, memodelkan dan memproses tekstur, serta untuk mensimulasikan proses pembelajaran sistem visual manusia menggunakan komputer. Sistem yang biasa digunakan adalah sebagai berikut:

Gambar 2. 6. Sistem yang biasa digunakan dalam bidang *computer vision*

Analisis tesktur berusaha untuk mencari suatu deskripsi kuantitatif umum, efisien, dan sederhana dari tekstur sehingga berbagai operasi matematis dapat digunakan untuk mengubah, membandingkan, dan mentransformasikan tekstur. Sebagian besar algoritma analisis tekstur cenderung melakukan proses ekstraksi fitur dan menghasilkan suatu skema pengkodean citra untuk merepresentasikan fitur-fitur yang dipilih. Beberapa aplikasi yang berhubungan dengan analisis tekstur adalah klasifikasi tekstur, segmentasi tekstur, bentuk dari tekstur, dan sintesis tekstur .

Sistem visual manusia mampu untuk mengenali dan membedakan tekstur dengan mudah. Akan tetapi, proses tersebut menjadi sesuatu yang lebih sulit untuk dilakukan perhitungan oleh komputer jika didasarkan pada parameter- parameter tertentu. Oleh karena itu, masalah-masalah yang dihadapi dalam analisis tekstur akan dibatasi pada proses membedakan antara beberapa nilai derajat keabuan (*gray-level values*). Hal tersebut dimaksudkan untuk mempermudah proses komputasi yang akan dilakukan.

Berbagai pendekatan untuk melakukan proses analisis tekstur dapat dikategorikan menjadi:

1. Metode struktural

   Metode pendekatan struktural akan mendefinisikan tekstur melalui komponen-komponen mikro-tekstur yang sudah didefinisikan dan kaidah- kaidah penyusunan spasial dari komponen-komponen mikro tersebut (membentuk makro-tekstur). Keuntungan utamanya adalah pendekatan memberikan suatu deskripsi simbolik yang baik dari citra. Bagaimanapun, fitur-fitur yang didapatkan akan lebih bermanfaat untuk proses sintesis daripada keperluan analisis.

2. Metode statistik

   Berbeda dengan pendekatan struktural, pendekatan statistik tidak akan menentukan struktur-struktur hirarki dari citra tekstural. Pendekatan statistik akan merepresentasikan tekstur secara tidak langsung melalui sifat-sifat non-deterministiknya yang mengatur distribusi dan hubungan antar nilai derajat keabu-abuan dari citra tersebut. Metode statistik

orde kedua yang paling populer untuk melakukan analisis tekstur berasal dari pembentukkan matriks kookurensi.

3. Metode *model –based*

   Pendekatan *model –based* dalam analisis tekstur biasanya menggunakan model *fractal* dan stokastik. Model *fractal* sangat berguna untuk memodelkan beberapa tekstur-tekstur natural, analisis tekstur maupun segmentasi tekstur.

4. Metode Tranformasi

   Metode transformasi dalam analisis tekstur, seperti Fourier, Gabor, dan Wavelet, mencoba merepresentasikan suatu citra dalam domain yang memiliki sebuah interpretasi tertentu yang berhubungan erat dengan karateristik tekstural tersebut (seperti frekuensi dan sebagainya).

Analisis tekstur seringkali dilakukan dengan mengamati pola ketetanggaan antar piksel dalam domain spasial dan dua macam persoalan yang berkaitan dengan analisis tekstur adalah:

- Ekstraksi ciri

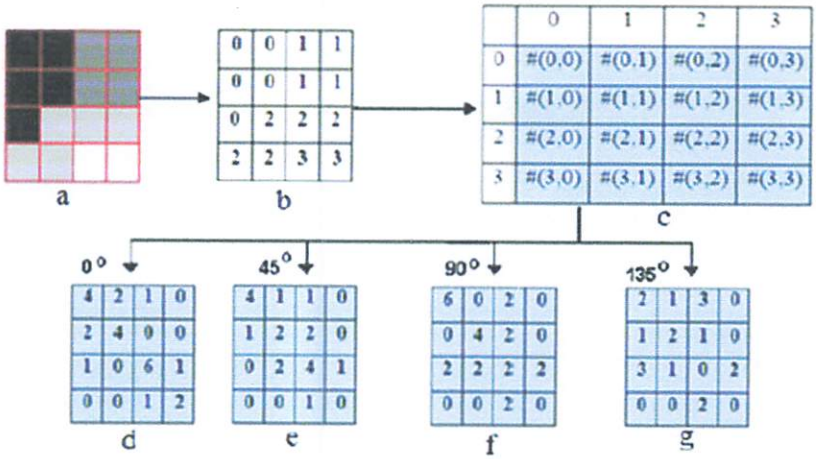
  Ekstraksi ciri merupakan suatu langkah awal dalam melakukan klasifikasi dan interpretasi citra. Adapun metode-metode yang sering digunakan adalah *Grey Level Co-occurrence Matrix* (GLCM)

- Segmentasi citra

  Segmentasi citra merupakan suatu proses untuk memisahkan suatu daerah pada citra dengan daerah lainnya. Segmentasi citra bertekstur tidak didasarkan pada intensitas per piksel, tetapi perlu mempertimbangkan perulangan pola dalam suatu wilayah ketetanggaan lokal.

2.2.3 GLCM

GLCM merupakan metode analisis tekstur yang menggambarkan hubungan ketetanggaan antara dua piksel pada suatu citra digital *grayscale* 2D. Hubungan semacam ini dinamakan kookurensi. Variasi distribusi nilai derajat keabuan antar satu piksel dengan piksel tetangganya menunjukkan suatu objek yang terdapat pada citra jalan . Karena variasi distribusi nilai derajat keabuan ini merupakan representasi dari pola struktural objek – objek pada citra jalan. Suatu tekstur dikatakan lembut jika distribusi nilai derajat keabuannya konstan dan periodik. Sedangkan tekstur kasar mempunyai distribusi nilai derajat keabuan yang cenderung tidak teratur. Pada gambar terdapat contoh singkat mengenai proses pembentukan matriks kookurensi simetrik dari suatu citra.



Gambar 2.7. Pembentukan GLCM dari Sebuah Citra

Diagram alir di atas merupakan proses pembentukan matriks kookurensi. Proses tersebut dimulai dengan mempresentasikan citra keabuan (a) menjadi matriks (b) dengan dimensi 4x4 piksel dengan empat nilai derajat keabuan yaitu 0,1,2, dan 3. Matriks kookurensi dari citra a dapat dibentuk dengan mengisi elemen- elemen kosong pada matriks kookurensi dasar (c). Pada matriks kookurensi dasar tersebut terdapat angka 0-3 yang menunjukkan jangkauan nilai derajat keabuan dari matriks b, dimana komponen baris (horizontal) merupakan nilai derajat keabuan yang dijadikan sebagai acuan sedangkan komponen kolom (vertikal) merupakan nilai derajat keabuan piksel tetangga. Nilai elemen pada matriks c, merupakan jumlah ketetanggaan antara piksel acuan dengan piksel tetangga pada jarak *d* piksel dari piksel acuan dengan arah orientasi tertentu. Pada umumnya digunakan jarak antara piksel sebesar satu piksel dengan orientasi arah 0°, 45°, 90°, 135°, 180°, 225°, 270°, dan 315° untuk matriks kookurensi asimetris atau 0°, 45°, 90°, 135° untuk matriks kookurensi simetris. Pada matriks kookurensi simetris dilakukan suatu pengamatan pada orientasi sudut θ dan $(180+\theta)$ dari piksel acuan citra, sedangkan pada matriks kookurensi asimetris dilakukan pengamatan pada orientasi sudut θ. Dengan kata lain (matriks kookurensi 0^0 + matriks kookurensi 180^0) asimetris sama dengan matriks kookurensi 0^0 simetris. Pada gambar 2.7 matriks kookurensi yang dibentuk bersifat simetris , sehingga nilai elemen #(2,1) pada matriks d merupakan jumlah piksel tetangga dengan nilai piksel 1 yang berada di sekeliling piksel dengan nilai 2 pada arah 0^0 dan 180^0. Untuk memperoleh GLCM, seluruh matriks kookurensi harus dijumlahkan .

2.2.4 Ekstraksi Fitur Harralick

Untuk proses perhitungan lebih lanjut, seperti ekstraksi fitur Harralick, diperlukan nilai probabilitas dari GLCM yang telah dibentuk. Matriks probabilitas kookurensi ini membagi GLCM dengan jumlah total kemungkinan kejadian pada matriks tersebut. Persamaanya ditunjukkan dengan Persamaan 2.1 dan Persamaan 2.2.

$$R = \sum_{i,j=1}^{Ng} P[i,j] \quad \text{(2-1)}$$

$$p[i,j] = \frac{P[i,j]}{R} \quad \text{(2-2)}$$

Dimana *Ng* merupakan jumlah tingkatan nilai derajat keabuan (*gray level*), *P [i,j]* adalah matriks kookurensi, R adalah jumlah total kemungkinan kejadian ketetanggaan di dalam matriks, dan *p[i.j]* merupakan probabilitas matriks kookurensi yang nantinya akan digunakan untuk proses analisis citra.

Fitur – fitur yang akan digunakan pada penelitian ini adalah fitur- fitur yang didefinisikan oleh Harralick. Fitur Haralick tersebut merupakan salah satu fitur tekstural yang telah banyak dugunakan untuk melakukan proses analisis tekstur. Untuk memudahkan proses ekstraksi fitur Harallick , maka sebaiknya komponen – komponen pendukung proses perhitungan dapat ditentukan terlebih dahulu.

Fitur – fitur tekstur Haralick tersebut antara lain:

1. Kontras

   Kontras menunjukan ukuran dari variasi lokal antar derajat keabuan dalam sebuah citra . Nilai kontras dari sebuah citra dapat dinyatakan dengan Persamaan 2.3.

   $$\text{Kontras} = \sum_{i,j=1}^{Ng-1} p_{i,j} \ (i-j)^2 \quad \text{(2-3)}$$

   Nilai ini merupakan perbedaan nilai intensitas antar piksel dalam sebuah citra.

2. Homogenitas

   Homogenitas menunjukkan ukuran keseragaman dari matriks kookurensi. Nilai dari parameter ini akan meningkat jika sebagian besar elemen matriks terkonsentrasi pada area diagonal umum . Nilai homogenitas dari sebuah citra dapat dinyatakan melalui Persamaan 2.4.

   $$\text{Homogenitas} = \sum_{i,j=1}^{Ng} \frac{p_{i,j}}{1+(i-j)^2} \quad \text{(2-4)}$$

   Nilai homogenitas berbanding terbalik dengan kontras.

3. Energi

Energi menunjukkan ukuran keteraturan dari nilai piksel pada sebuah citra. Nilai energi dari sebuah citra dapat dinyatakan dengan Persamaan 2.5.

$$\text{Energi} = \sqrt{\sum_{i,j=1}^{N_g-1} p_{i,j^2}} \quad \text{.....(2-5)}$$

Nilai dari parameter ini akan besar jika distribusi nilai derajat keabuan kostan atau periodik. Suatu citra yang homogen akan memiliki nilai transisi derajat keabuan yang kecil.

4. Korelasi

Korelasi menunjukkan ukuran ketergantungan linier antar nilai derajat keabuan pada piksel – piksel yang bertetangga. Nilai korelasi dari sebuah citra dapat dinyatakan dengan Persamaan 2.6.

$$\text{Korelasi} = \sum_{i,j=1}^{N_g-1} p_{i,j} \frac{(i-\mu_i)(i-\mu_j)}{\sqrt{(\sigma_i^2)(\sigma_j^2)}} \quad \text{.....(2-6)}$$

Nilai korelasi ini didefinisikan antara 0 sampai 1. Nilai 0 menunjukkan tidak mempunyai korelasi dan nilai 1 menunjukkan korelasi sempurna yang dinyatakan dengan Persamaan 2.7 dan Persamaaan 2.8.

Dimana μ adalah mean :

$$\mu_j = \sum_{i,j=1}^{N_g-1} j\,[p_{i,j}] \quad \text{.....(2-7)}$$

$$\mu_j = \sum_{i,j=1}^{N_g-1} i\,[p_{i,j}] \quad \text{.....(2-8)}$$

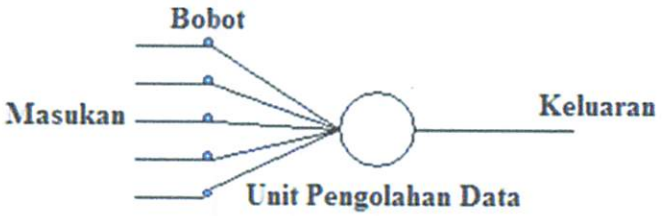
σ^2 adalah variansi yang dinyatakan dengan Persamaan 2.9 dan Persamaan 2.10.

$$\sigma_i^2 = \sum_{i,j=1}^{Ng-1} p_{i,j}\ (i-\mu_i)^2 \quad \text{.....(2-9)}$$

$$\sigma_i^2 = \sum_{i,j=1}^{Ng-1} p_{i,j}\ (j-\mu_i)^2 \quad \text{.....(2-10)}$$

2.3 Jaringan Syaraf Tiruan

Jaringan saraf tiruan (JST) merupakan suatu sistem pemrosesan informasi dengan karakteristik menyerupai jaringan saraf biologis dan dibentuk sebagai generalisasi model matematika dari jaringan saraf biologis tersebut. Secara sederhana, JST meliputi elemen-elemen pengolahan sederhana (neuron) yang dapat menunjukkan karakteristik data yang kompleks, yang ditentukan dari hubungan-hubungan antara elemen-elemen pengolahan dan elemen-elemen parameter.



Gambar 2.8. Gambaran jaringan saraf tiruan secara umum

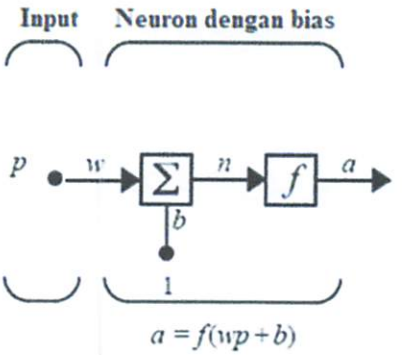
JST merupakan sistem yang bersifat adaptif yang dapat mengubah strukturnya sendiri berdasarkan informasi internal maupun eksternal yang memasuki jaringan tersebut selama masa pembelajaran. Salah satu keuntungan utama dari JST adalah JST dapat digunakan untuk menentukan hubungan kompleks antara suatu kelompok input dengan kelompok output untuk menemukan kecenderungan pola-pola datanya.

JST merupakan suatu metode pemodelan data yang dapat membawa dan merepresentasikan hubungan antara komponen masukan dan keluaran yang bersifat kompleks. Perilaku sistem JST yang menyerupai otak manusia bekerja sebagai berikut:

1. Sistem JST mendapatkan pengetahuan melalui proses pembelajaran (*training*).
2. Pengetahuan sistem JST disimpan dalam hubungan-hubungan antar neuron yang biasa disebut dengan bobot sinaptik, atau bobot saja.

2.3.1 Model Neuron

Gambar 2.9 menunjukkan masukan skalar tunggal dengan komponen bias di setiap *summing junction.*



Gambar 2.9. Contoh sistem JST dengan masukan tunggal dan bisa di summing junction.

Masukan skalar p dikirimkan melalui suatu hubungan dengan mengalikan nilainya dengan suatu bobot skalar w, yang hasilnya juga berupa skalar. Pada bagian *summing junction* , hasil perkalian wp akan ditambahkan dengan komponen bias b. Hasil tersebut akan berperan sebagai masukan dari komponen fungsi aktivasi f.

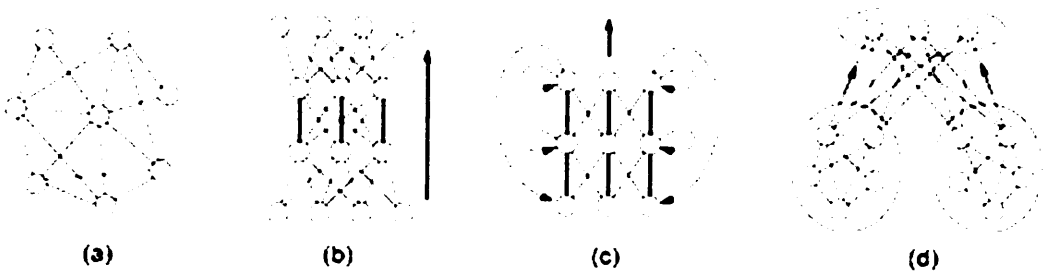
sebagai masukan dari komponen fungsi aktivasi *f*.

Persamaan akhir yang merepresentasikan sistem tersebut adalah $a = f(wp+b)$.

Ada beberapa buah jenis sistem JST, akan tetapi mereka semua memiliki empat komponen dasar yang sama, yaitu :

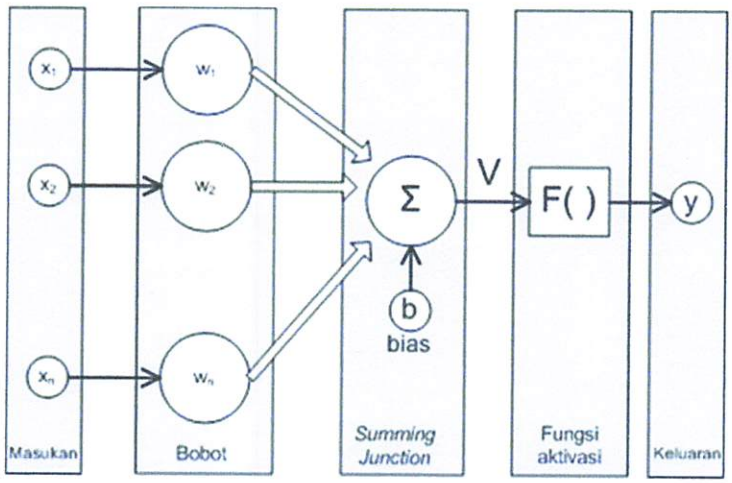
1. Sekelompok unit pengolahan
2. Sekelompok hubungan-hubungan antar neuron
3. Prosedur penghitungan
4. Prosedur pembelajaran/pelatihan (*training*)

JST terdiri dari banyak sekali unit-unit pengolahan data yang sederhana, yang dapat dianalogikan sebagai neuron di dalam otak manusia. Unit-unit tersebut bekerja sekaligus untuk mendukung keselarasan antara satu unit dengan unit yang lain. Unit-unit di dalam JST biasanya dibagi menjadi unit masukan, yang menerima data mentah dari lingkungan luar; unit tersembunyi (*hidden unit*), yang dapat mengubah karakteristik suatu data; dan unit keluaran, yang menghasilkan keputusan atau hasil numerik tertentu.



Gambar 2.10 Macam topologi JST; (a) *unstructured*; (b) *layered*; (c) *recurrent*; (d) *modular*

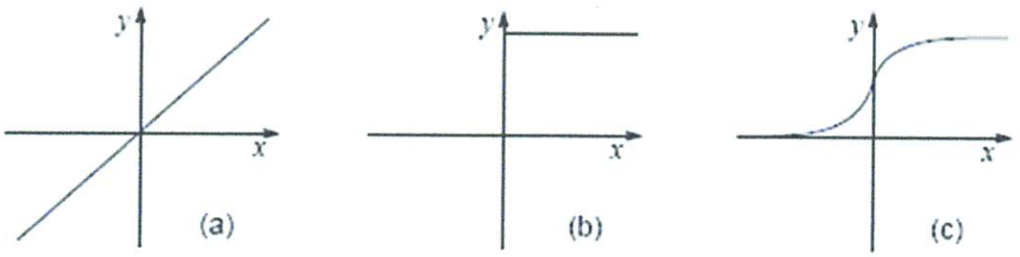
Unit-unit dalam JST diatur sedemikian rupa ke dalam suatu topologi dengan sekelompok hubungan atau bobot (ditunjukkan dengan garis pada Gambar 2.10). Setiap bobot memiliki nilai riil, dengan jangkauan antara $-\infty$ $^{s}/_{d}$ $+\infty$. Bobot-bobot tersebut dapat berubah-ubah sebagai akibat dari proses pelatihan.



Gambar 2.11. Aktivasi unit komputasi; x = masukan, w = bobot, b = bias, F = unit aktivasi, y = keluaran

Proses perhitungan selalu dimulai dengan memberikan pola komponen masukan ke dalam JST. Secara sederhana, prosesnya dimulai dengan penghitungan komponen masukan terlebih dahulu dan kemudian dilanjutkan dengan penghitungan nilai fungsi aktivasi keluaran dari masukan tersebut.

Persamaan yang umum digunakan adalah $V_j = \sum x_i \; w_i + b_j$, dimana x merupakan komponen masukan, V merupakan hasil dari bagian *summing junction*, y merupakan nilai fungsi aktivasi F, dan b merupakan nilai *bias* dari setiap bagian *summing junction*. Fungsi aktivasi (atau *transfer function*) dapat bersifat deterministik ataupun stokastik. Fungsi aktivasi yang bersifat deterministik biasanya merupakan salah satu dari ketiga bentuk ini, yaitu linier, *threshold* , dan sigmoidal.



Gambar 2.12. Fungsi aktivasi deterministik; (a) linier; (b) threshold; (c) sigmoidal

Di dalam bentuk linier, yang dinyatakan dengan Persamaan 2.11.

$y = F(V)$ ..... (2-11)

dimana komponen keluaran bersifat proporsional dengan masukannya. Bentuk linier macam ini jarang digunakan karena kinerjanya tidak terlalu baik. Bentuk fungsi aktivasi yang lain, yaitu *threshold* memiliki persamaan yang lain yang dinyatakan dengan Persamaan 2.12.

$y = F(V) = 0$ jika $V < 0$; 1 jika $V > 0$ .....(2-12)

Fungsi *threshold* ini merupakan fungsi yang memberikan nilai keluaran pada suatu batas nilai tertentu dengan bergantung apakah nilai masukan yang diberikan melebihi nilai batas *threshold* atau tidak. Dalam beberapa kasus, diperlukan suatu proses pencarian bobot yang bersifat eksponensial, dengan jangkauan nilai antara 0 $^{s}/_{d}1$. Fungsi yang biasa digunakan sebagai representasi fungsi eksponensial adalah fungsi sigmoidal. Fungsi sigmoidal direpresentasikan dengan persamaan 2.13.

$$y = F(V) = \frac{1}{1+\exp(-V)} \quad \text{.....(2-13)}$$

Proses-proses di atas perlu dilakukan untuk memodelkan JST sehingga sistem tersebut dapat melaksanakan tugas yang diharapkan. Hubungan-hubungan pada model JST akan menentukan pengaruh antara satu unit dengan unit yang lainnya dan nilai bobot menunjukkan kekuatan pengaruh dari suatu nilai unit tertentu. Secara umum, proses pembelajaran/pelatihan (*training*) dilakukan sebagai berikut:

1. Memberikan contoh-contoh kombinasi pada JST, yang menunjukkan pola- pola yang diinginkan serta hasil yang diinginkan.
2. Mengubah nilai-nilai bobot dan bias pada setiap hubungan untuk memperoleh nilai keluaran yang diinginkan.

2.3.2 *Backpropagation* (Propagasi balik)

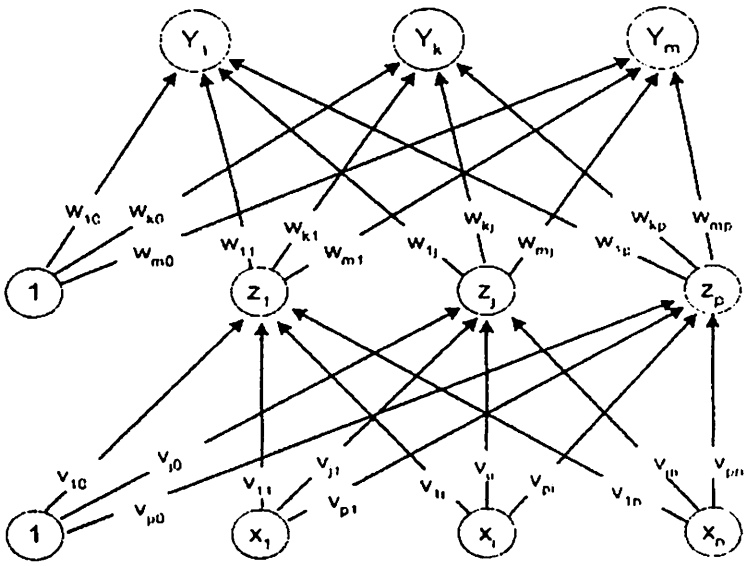
Seperti halnya model JST lain, *Backpropagation* melatih jaringan untuk mendapatkan keseimbangan antara kemampuan jaringan untuk mengenali pola yang digunakan selama pelatihan serta kemampuan jaringan untuk memberikan respon yang benar terhadap pola masukkan yang serupa (tapi tidak sama) dengan pola yang dipakai selama pelatihan.

Pelatihan jaringan syaraf tiruan dikatakan berhasil jika pelatihan *konvergen*, dan gagal jika pelatihan *divergen*. Suatu pelatihan dikatakan konvergen jika galat pada setiap iterasi pelatihan selalu mengecil, sampai pada titik dimana nilai bobot pada setiap *neuron* telah mencapai nilai yang paling baik untuk data pelatihan yang diberikan. Sebaliknya, pelatihan dikatakan *divergen* jika galat pada pelatihan tidak cenderung mengecil menuju sebuah titik tertentu.

2.3.3 Arsitektur *Backpropagation*

Jaringan ini memiliki satu lapis masukan ditambah dengan sebuah bias, satu atau lebih lapisan tersembunyi (*hidden layer*) ditambah dengan sebuah bias, dan satu lapisan keluaran.

Setiap lapis memiliki *neuron-neuron* (unit-unit). Di antara *neuron* pada satu lapis dengan *neuron* pada lapis berikutnya dihubungkan dengan model koneksi yang memiliki bobot-bobot (*weights*).



Gambar 2.13 Arsitektur *Backpropagation*

Berdasarkan Gambar 2.13, v_{ij} merupakan bobot dari neuron *input layer* x_i ke neuron *hidden layer* z_j (v_{j0} merupakan bobot yang menghubungkan bias dari neuron *input layer* ke neuron *hidden layer*). Sedangkan w_{kj} merupakan bobot dari neuron *hidden layer* z_j ke neuron *output layer* y_k (w_{k0} merupakan bobot dari bias di neuron *hidden layer* ke neuron *output layer* y_k).

2.3.4 Algoritma *Backpropagation*

Pelatihan sebuah jaringan *backpropagation* terdiri dari tiga langkah yaitu pelatihan pola *input* secara *feedforward*, *backpropagation* dari kumpulan kesalahan dan penyesuaian bobot [8]. Pelatihan dilakukan berulang-ulang dan berhenti jika telah mencapai batas iterasi maksimum yang ditentukan dan nilai *error* kurang dari *Mean Square Error* (MSE). Ketepatan algoritma *backpropagation* ditentukan dengan *Mean Square Error* (MSE). Semakin kecil nilai MSE maka dapat dianggap bahwa arsitektur jaringan semakin baik. MSE dihitung dengan Persamaan 2.14.

$$MSE = \frac{1}{2}\sum_{p}(t_p - y_p)^2 \quad \text{..... (2-14)}$$

dimana p adalah jumlah *neuron* (unit) *output*, t adalah target dan y adalah output.

Algoritma pelatihan *backpropagation* adalah sebagai berikut :

1. Inisialisasi bobot.

   Bobot awal ditentukan secara acak dengan nilai sekecil mungkin, misalkan [-0.05, 0.05].

2. Inisialisasi *error* target, jumlah *hidden layer*, *learning rate* dan jumlah maksimum epoh.

3. Selama (Epoh < Maksimum epoh) dan (MSE < Target Error), maka:

   a. Lakukan *feedforward* untuk setiap pasangan pelatihan:

   1.Tiap-tiap unit *input* (x_i, i=1,2,..n) menerima sinyal dan menyalurkan sinyal ini ke semua unit lapisan tersembunyi.

   2.Tiap-tiap unit tersembunyi (z_j, j=1,...,p) menjumlahkan perkalian nilai input dan bobot sinyal input dengan Persamaan 2.15.

$$z\_in_j = v_{0j} + \sum_{i=1}^{n} x_i v_{ij} \quad \text{..... (2-15)}$$

   Gunakan fungsi aktivasi untuk menghitung *output*nya yaitu:

$$z_j = f(z\_in_j) \quad \text{..... (2-16)}$$

   Dan kirimkan sinyal tersebut ke semua unit di lapisan atasnya (lapisan *output*).

   3. Tiap-tiap unit *output* (y_k ; k=1,...,m) akan menjumlahkan bobot sinyal output dengan Persamaan 2.17.

$$y\_in_k = v_{0k} + \sum_i z_j w_{jk} \quad \text{..... (2-17)}$$

   Gunakan fungsi aktivasi untuk menghitung sinyal *output*, ditunjukkan oleh Persamaan 2.18.

$$y_k = f(y\_in_k) \quad \text{.....(2-18)}$$

b. Lakukan *Backpropagation* untuk masing-masing pasangan pelatihan :

   1. Tiap-tiap unit *output* (y_k;k=1,...,m) menerima target yang bersesuaian dengan pola input pelatihan, hitung informasi kesalahan dengan Persamaan 2.19.

$$\delta_k = (t_k - y_k) f'(y\_in_k) \quad \text{.....(2-19)}$$

   kemudian hitung koreksi bobot, digunakan untuk memperbarui w_{jk} dengan Persamaan 2.20.

$$\Delta w_{jk} = \alpha \delta_k z_j \quad \text{.....(2-20)}$$

   hitung koreksi bias untuk memperbaiki bobot bias (w_{0k}) dengan Persamaan 2.21.

$$\Delta w_{0k} = \alpha \delta_k \quad \text{.....(2-21)}$$

kemudian kirimkan nilai δ_k ke seluruh unit yang berada pada lapisan dibawahnya (*backward*)

2. Tiap-tiap unit tersembunyi (Z_j; j =1,..p) menjumlahkan *input delta* dari unit lapisan atasnya.yang dinyatakan dengan Persamaan 2.22.

$$\delta\_in_j = \sum_{k=1}^{m} \delta_k w_{jk} \quad \text{....(2-22)}$$

kalikan fungsi ini dengan turunan fungsi aktivasi untuk menghitung informasi kesalahan, yang dinyatakan dengan Persamaan 2.23.

$$\delta_j = \delta\_in_j f'(y\_in_j) \quad \text{....(2-23)}$$

hitung koreksi obobt untuk memperbaiki v_{ij} ,dengan persamaan 2.24.

$$\Delta v_{ij} = \alpha \delta_j x_i \quad \text{....(2-24)}$$

hitung koreksi bobot bias untuk memperbaiki v_{0j} dengan Persamaan 2.25.

$$\Delta v_{0j} = \alpha \delta_j \quad \text{....(2-25)}$$

c. Perbaiki Bobot dan bias

Tiap-tiap unit *output* (y_k,k=1,..m) memperbaiki bobot dan bias (j = 0,... p) , yang dinyatakan dengan Persamaan 2.26.

$$w_{jk}(\text{baru}) = w_{jk}(\text{lama}) + \Delta w_{jk} \quad \text{....(2-26)}$$

Tiap-tiap unit tersembunyi (Z; 1,..,p) memperbaiki bobot dan bias (i; 0,..,n), yang dinyatakan dengan Persamaan 2.27.

$$v_{ij}(\text{baru}) = v_{ij}(\text{lama}) + \Delta v_{ij} \quad \text{....(2-27)}$$

Jika iterasi mencapai maksimum maka berhenti.

Setelah proses pelatihan, tahap pengujian jaringan syaraf *backpropagation* diaplikasikan dengan hanya menggunakan tahap perambatan maju (*feed fordward*) dari algoritma pelatihan Prosedur aplikasinya adalah sebagai berikut :

1. Inisialisasi bobot yang diperoleh dari proses pelatihan jaringan.
2. Tiap-tiap unit *input* (x_i, i=1,2,..n) menerima sinyal dan menyalurkan ke semua unit lapisan tersembunyi.
3. Tiap-tiap unit tersembunyi (z_j, j=1,...,p) menjumlahkan perkalian nilai input dan bobot sinyal input dengan persamaan 2.28.

$$z\_in_j = v_{0j} + \sum_{i=1}^{n} x_i v_{ij} \quad \text{.....(2-28)}$$

Gunakan fungsi aktivasi untuk menghitung *output*nya yaitu dengan Persamaan 2.29.

$$z_j = f(z\_in_j) \quad \text{.....(2-29)}$$

4. Tiap-tiap unit *output* (y_k ; k=1,...,m) akan menjumlahkan bobot sinyal output dengan Persamaan 2.30.

$$y\_in_k = v_{0k} + \sum_i z_j w_{jk} \quad \text{.....(2-30)}$$

Gunakan fungsi aktivasi untuk menghitung nilai *output*, ditunjukkan oleh persamaan 2.31.

$$y_k = f(y\_in_k) \quad \text{.....(2-31)}$$

Jika $y_k \geq 0{,}5$ maka yk = 1, *else* yk = 0.

Keterangan dari simbol :

t = *Output* vektor target, t = (t1, ..., xk, ..., xm)

δk = Informasi tentang kesalahan pada unit yk yang disebarkan kembali ke unit tersembunyi

δj = Informasi tentang kesalahan dari lapisan *output* ke unit tersembunyi zj

α = *learning rate*

Xi = unit *input* i

Voj = Bobot awal bias pada lapisan tersembunyi j

Vij = Bobot menuju hidden

Zj = Unit tersembunyi j

Zinj = *Input* jaringan ke zj

Wok = Bobot awal bias pada menuju *output*

Wjk = Bobot menuju output

Yk = Unit *output* i

Y-ink = *Input* jaringan ke yk

E = Error tiap pasangan

2.3.5 Jumlah Unit Tersembunyi

Menentukan jumlah lapis pada *hidden layer* dan menentukan jumlah unit per layernya sangat sulit. Hasil teoritis yang didapat menunjukkan bahwa jaringan dengan sebuah *hidden layer* sudah cukup bagi *backpropagation* untuk mengenali sembarang perkawanan antara masukan dan target dengan tingkat ketelitian yang ditentukan. Jumlah unit tersembunyi dapat ditentukan dengan melakukan beberapa percobaan (*trial and error*). Tetapi penambahan jumlah *hidden* layer kadangkala membuat pelatihan menjadi lebih mudah.

2.3.6 Lama Iterasi

Tujuan utama penggunaan *Backpropagation* adalah mendapatkan keseimbangan antata pengenalan pola pelatihan secara benar dan respon yang baik untuk pola lain yang sejenis (disebut pengujian). Jaringan dapat dilatih terus menerus hingga semua pola pelatihan dikenalai dengan benar. Akan tetapi hal itu tidak menjamin jaringan akan mampu mengenali pola pengujian dengan tepat.

Umumnya data dibagi menjadi 2 bagian saling asing, yaitu pola data yang dipakai sebagai pelatihan dan data yang dipakai untuk pengujian. Perubahan bobot dilakukan berdasarkan pola pelatihan. Akan tetapi selama pelatihan (misal setiap 10 epoch), kesalahan yang terjadi dihitung berdasarkan semua data (pelatihan dan pengujian). Selama kesalahan ini menurun, pelatihan terus dijalankan. Akan tetapi jika kesalahannya sudah meningkat, pelatihan tidak ada gunanya untuk diteruskan lagi.

2.4. Matlab 7.04

Matlab(*Matrix Laboratory*) merupakan salah satu bahasa pemrograman yang dikembangkan oleh MathWorks. Matlab tidak hanya berfungsi sebagai bahasa pemrograman . tetapi seklaigus sebagai alat visualisasi, yang berhubungan langsung dengan ilmu matematika. (Erick, 2007 :1) Oleh karena itu , Matlab semakin banyak digunakan oleh para programmer yang menghendaki kepraktisan dalam membuat program.

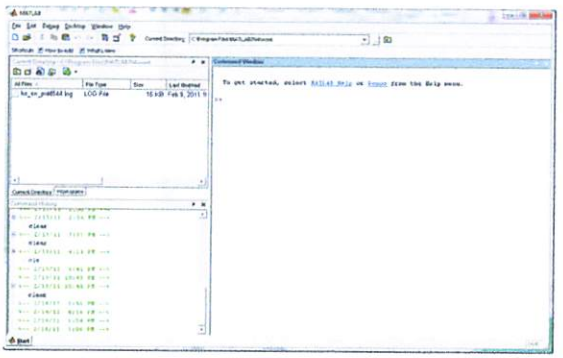
Matlab adalah bahasa pemrograman level tinggi yang dikhususkan untuk komputasi teknis. Bahasa ini mengintegrasikan kemampuan komputasi, visualisasi dan pemrograman dalam sebuah lingkungan yang tunggal dan mudah digunakan. Matlab memberikan sistem interaktif yang menggunakan konsep *array*/matrik sebagai standar variabel elemennya tanpa membutuhkan pen-deklarasi-an array seperti pada bahasa lainnya.

2.4.1 Pengenalan Matlab 7.0.4

Sebagaimana bahasa pemrograman lainnya, Matlab juga menyediakan lingkungan kerja terpadu yang sangat mendukung dalam pembangunan aplikasi. Tampilan jendela Matlab dapat dibagi menjadi beberapa bagian, yaitu:

1. Jendela Utama

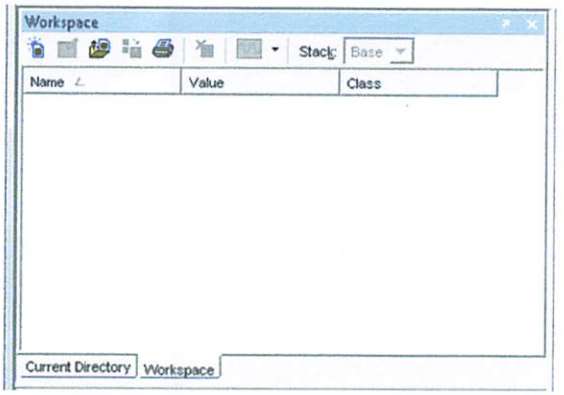
Tampilan jendela utama dalam Matlab ditunjukkan dalam Gambar 2.14.



Gambar 2.14.Tampilan jendela utama Matlab

2. Workspace

Tampilan *workspace* dalam Matlab ditunjukkan dalam Gambar 2.15.

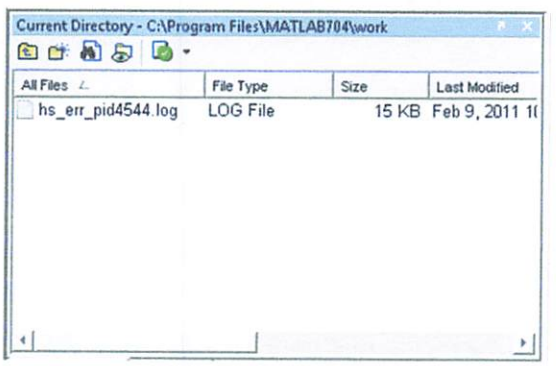


Gambar 2.15. Tampilan *Workspace*

Fungsi *workspace* adalah menginformasikan kepada pengguna tentang variabel – variabel yang dipakai selama pengguna Matlab berlangsung.

3. Current directory

Tampilan *current directory* dalam Matlab ditunjukkan dalam Gambar 2.16.

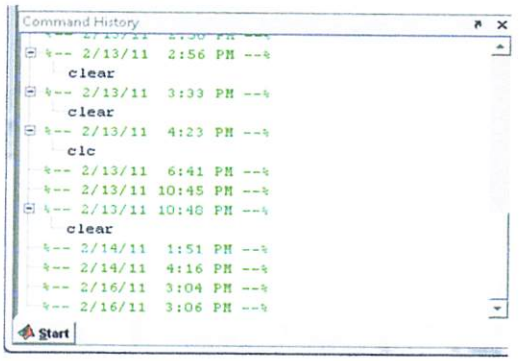


Gambar 2.16 Tampilan *Current Directory*

Fungsi *current directory* adalah memilih direktori yang aktif dan akan digunakan selama penggunaan Matlab berlangsung.

4. Command History

Tampilan *command history* dalam Matlab ditunjukkan dalam Gambar 2.17.

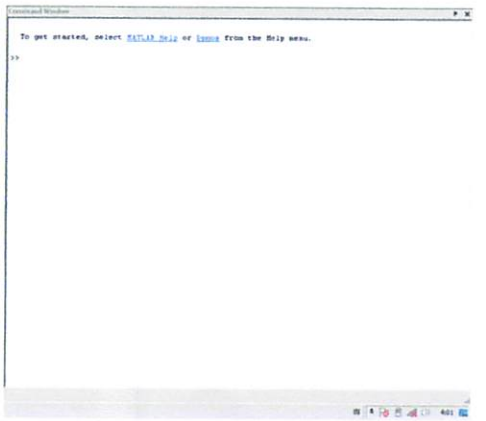


Gambar 2.17 Tampilan *Command History*

Fungsi *command history* adalah menyimpan perintah-perintah yang pernah ditulis pada *command window*.

5. Command window

Tampilan *command window* dalam Matlab ditunjukkan dalam Gambar 2.18.

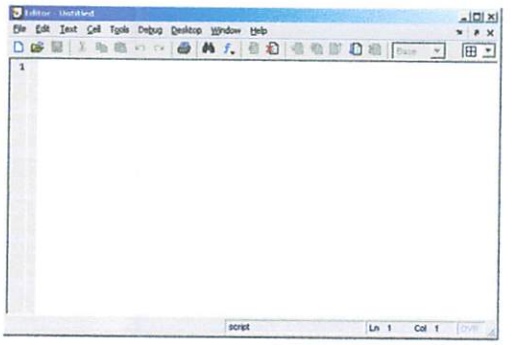


Gambar 2.18 Tampilan *Command Window*

Jendela ini berfungsi sebagai penerima perintah dari pemakai untuk menjalankan seluruh fungsi yang disediakan oleh Matlab. Pada dasarnya jendela ini adalah inti dari pemrograman Matlab yang menjadi media pengguna berinteraksi dengan Matlab.

6. Matlab Editor

Tampilan Matlab *Editor* ditunjukkan dalam Gambar 2.19.

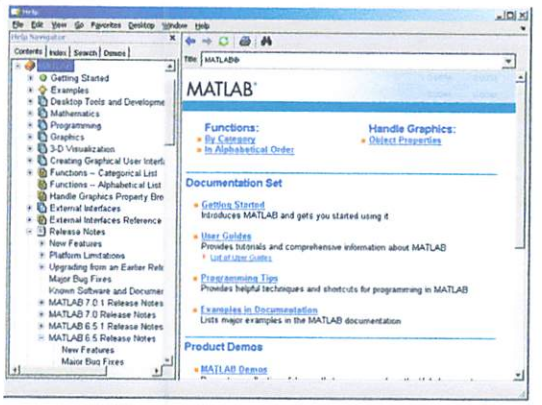


Gambar 2.19 Tampilan Matlab *Editor*

Fungsi Matlab *editor* adalah tempat membuat *script* program pada Matlab. Untuk memunculkan Matlab *Editor*, kita menggunakan perintah File – New – M-file atau dengan mengetikkan >>edit pada command window.

7. Help

Tampilan *Help* dalam Matlab ditunjukkan dalam Gambar 2.20.



Gambar 2.20 Tampilan *Help*

2.4.2 Membuat Aplikasi Matlab 7

Dalam melakukan pekerjaan pemrograman menggunakan bahasa Matlab, ada beberapa cara yang digunakan yaitu sebagai berikut :

1. Pada Command Window

Untuk membuat program, hanya diperlukan untuk mengetikkan program pada prompt Matlab dalam Command Window, misalnya :

>> pjg = 5;

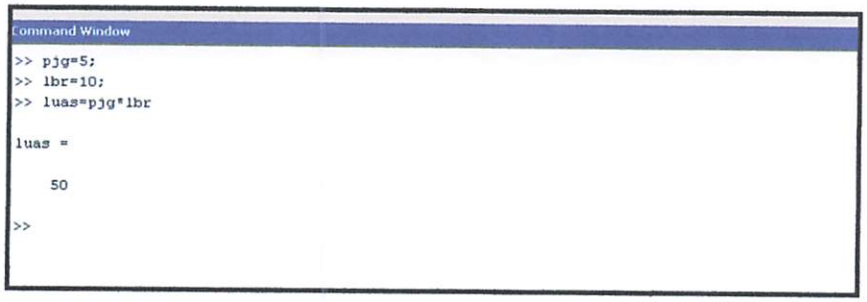
Tekan tombol enter, lalu ketikkan :

>> lbr = 10;

Tekan enter, lalu ketikkan :

>> luas = pjg * lbr

Untuk skrip terakhir tidak diberikan tanda titik koma (;) sehingga hasil dari perhitungan diatas dapat langsung dilihat yang ditunjukkan dalam Gambar 2.21.



Gambar 2.21 Hasil Perhitungan pada Command Window

2. Menggunakan Matlab Editor

Pada command window ketikkan:

>> edit

Tekan enter, selanjutnya muncul Matlab Editor dan ketikkan program dibawah ini:

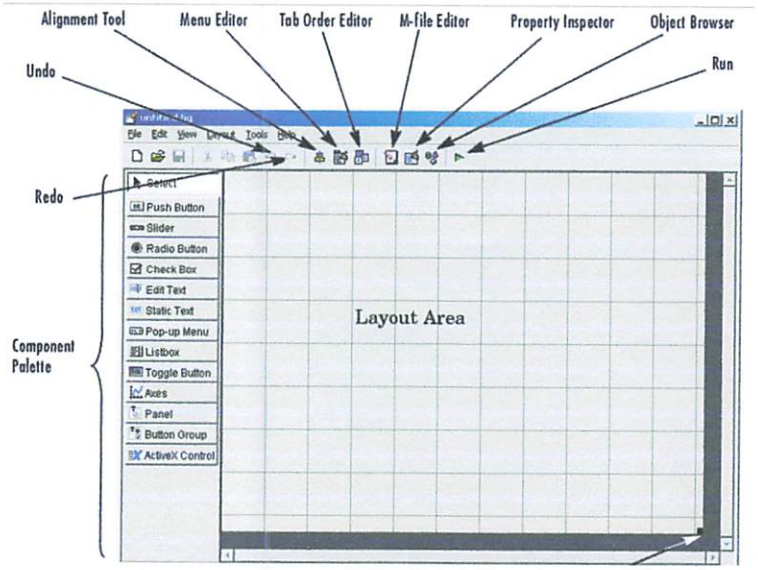
```
% ------------------------------------------------
% Matlab Programming
% oleh : Ngie Kozemake
% ------------------------------------------------
pjg=10;
lbr=5;
luas=pjg*lbr;
disp(['Luas->' num2str(luas)]);
%------------------------------------------------
```

2.4.3 Bekerja dengan GUI

GUI merupakan tampilan grafis yang memudahkan user berinteraksi dengan perintah teks. Dengan GUI, program yang dibuat menjadi lebih *user friendly*, sehingga user mudah menjalankan suatu aplikasi program.

Untuk membuka lembar kerja GUI dalam Matlab, kita menggunakan perintah File – New – GUI atau dengan mengetikkan >>guide pada command window.

Tampilan lembar kerja GUI dalam Matlab ditunjukkan dalam Gambar 2.22



Gambar 2.22 Tampilan Lembar Kerja GUI Matlab

Pada lembar kerja GUI terdapat component pallete dimana komponen-komponen tersebut yang akan digunakan untuk membuat tampilan Matlab yang lebih user friendly, yaitu :

1. Push Button

   Push button merupakan tombol yang jika diklik akan menghasilkan suatu tindakan.

2. Slider

   Slider menerima masukan berupa angka pada suatu range tertentu dimana pengguna menggeser kontrol pada slider.

3. Radio Button

   Radio Button merupakan kontrol yang digunakan untuk memilih satu pilihan dari beberapa pilihan yang ditampilkan.

4. Check Box

   Check box merupakan kontrol yang digunakan untuk memilih antara satu atau lebih pilihan dari beberapa pilihan yang ditampilkan.

5. Edit Text

   Edit text merupakan kontrol untuk meng-inputkan atau memodifikasi teks.

6. Static Text

   Static text merupakan kontrol untuk membuat teks label.

7. Pop Up Menu

   Pop up menu merupakan kontrol yang digunakan untuk membuka tampilan daftar pilihan yang telah didefinisikan dengan mengklik tanda panah yang terdapat pada pop up menu.

8. List Box

   List Box merupakan kontrol yang digunakan untuk menampilkan semua daftar item. Kemudian, pengguna memilih satu diantara item-item yang ada.

9. Toggle Button

   Toggle button hampir sama dengan push button, hanya push button diklik, tombol akan kembali ke posisi semula. Sebaliknya jika toggle button diklik, tombol tidak kembali ke posisi semula kecuali diklik kembali.

10. Axes

    Axes digunakan untuk menampilkan grafik atau gambar.

11. Panel

    Panel merupakan kotak yang digunakan untuk menandai atau mengelompokkan daerah tertentu pada figure.

12. Button group

    Button group hampir sama dengan panel, tetapi button group lebih digunakan untuk mengelompokkan radio button dan toggle button.

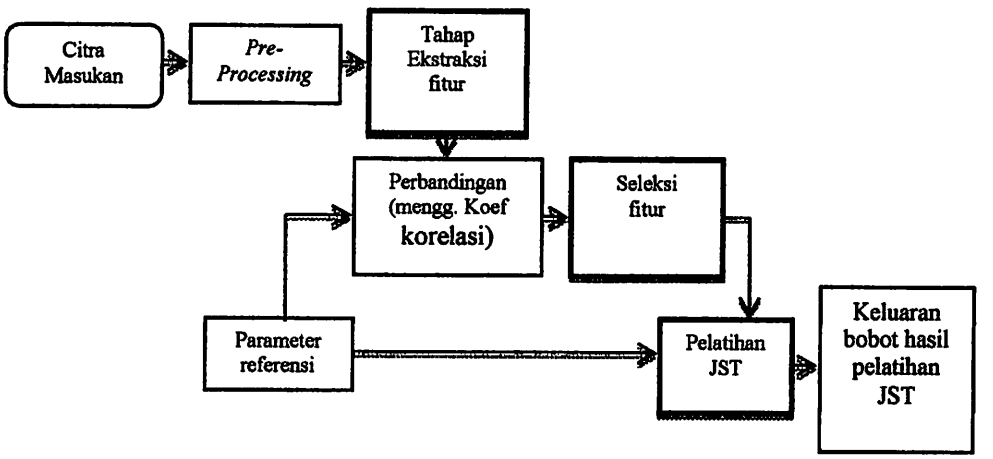
13. Active X Control

    Digunakan jika Matlab berjalan pada Sistem Operasi Microsoft Windows, dengan menggunakan komponen ini Matlab dapat terhubung dengan komponen Microsoft Windows.

BAB III
PERANCANGAN SISTEM

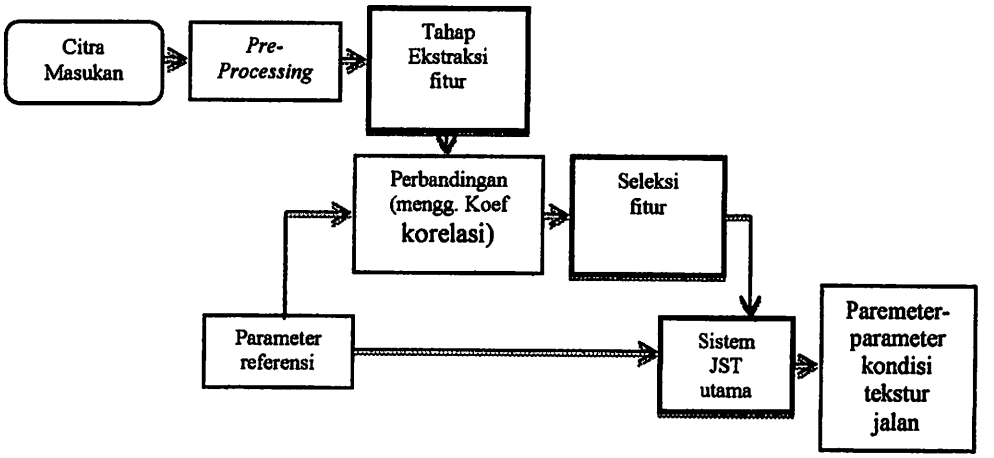
3.1 Perancangan Sistem

Proses perancangan sistem ini menggunakan perangkat lunak MATLAB 7.0 sebagai komponen pengolah data, untuk menjalankan algoritma – algoritma yang akan digunakan pada proses analisis.Sistem analisis yang dikembangkan terbagi menjadi beberapa bagian, yaitu sistem ekstraksi fitur yang dilanjutkan dengan tahap seleksi fitur, sistem pelatihan jaringan saraf tiruan (JST), dan sistem analisis citra menggunakan JST.



Gambar 3.1 . Proses ekstraksi fitur, seleksi fitur, dan pelatihan sistem JST

Sebelum melakukan proses pelatihan JST, proses ekstraksi fitur tekstur dilakukan terlebih dahulu dari suatu citra jalan. Setelah proses ekstraksi fitur, maka tahap selanjutnya adalah membandingkan nilai-nilai fitur tersebut dengan nilai parameter referensi citra jalan berupa hasil segmentasi yang diolah secara manual. Proses seleksi fitur dilakukan dengan memilih beberapa fitur yang memiliki koefisien korelasi terbesar terhadap parameter referensi. Fitur-fitur hasil seleksi tersebut menjadi komponen masukan dari sistem pelatihan JST. Pada gambar 3.1 di atas, terlihat bahwa proses pelatihan JST melibatkan dua buah komponen, yaitu komponen masukan yang berupa fitur-fitur hasil seleksi dan komponen data latih yang berupa parameter referensi untuk setiap sampel citra jalan. Hasil dari proses pelatihan JST ini adalah bobot-bobot untuk setiap hubungan neuron sistem JST dan bias di setiap *summing junction.*



Gambar 3. 2. Sistem utama analisis citra menggunakan JST

Pada proses analisis citra menggunakan JST ini, tahap akuisisi citra dijital sampai tahap seleksi fitur merupakan proses yang sama dengan tahap pelatihan sistem JST. Sistem JST yang digunakan pada proses analisis utama ini telah memiliki bobot dan bias masing-masing, hasil dari tahap pelatihan JST.

Komponen masukan dari sistem JST ini adalah fitur-fitur tekstur hasil seleksi menggunakan koefisien korelasi. Sedangkan komponen keluaran sistem adalah suatu nilai parameter yang dapat merepresentasikan mana *object* jalan dan mana bukan *object* jalan.

3.2 Implementasi Sistem

Pada sub-bab ini akan dijelaskan secara terperinci mengenai komponen- komponen perancangan sistem dan penjelasan mengenai algoritma-algoritma pengolahan citra yang digunakan dalam penelitian skripsi ini. Sistem ini dirancang dan dijalankan menggunakan alat bantu perangkat lunak MATLAB 7.0.

3.2.1. Pengambilan sampel

Untuk pengambilan sampel dilakukan dengan pengambilan langsung di lapangan di berbagai tempat yang berbeda. Kemudian dari gambar – gambar photo tersebut diolah agar diperoleh objek tertentu, yaitu dengan cara mengambil bagian tertentu dari gambar (misal jalan saja, pohon saja dengan ukuran 10 x 10 pixel), kemudian mengkopinya, lalu disimpan di tempat lain, dalam hal ini adalah direktori work yang terdapat dalam path :\MatlabRoot\work. Semua pengolahan citra ini dilakukan dengan Adobe photoshop CS 4.

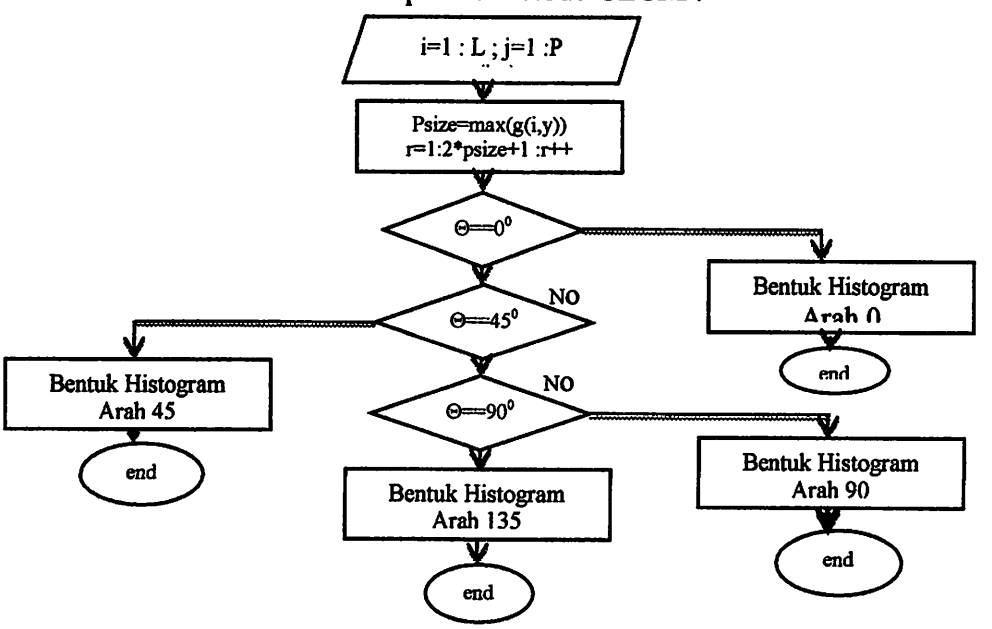
3.2.2. Tahap *pre-processing* (Normalisasi Citra)

Proses ini dilakukan setelah mendapatkan citra – citra dijital jalan. Komponen input sistem adalah citra dijital RGB tekstur jalan hasil dari proses cropping dengan resolusi 320 x 240 pixel dalam format JPG. Proses *pre-processing* ini dilakukan karena citra – citra tersebut masih merupakan data mentah. Sebagai langkah awal, citra RGB jalan tersebut harus melalui proses normalisasi terlebih dahulu, dimana dengan mengubah citra berwarna yang akan digunakan sebagai masukan */input* terlebih dahulu diubah kebentuk *grayscale* , dengan menggunakan fungsi matlab "RGB2GRAY", Setelah itu, maka citra siap melakukan proses ekstraksi fitur.

3.2.3. Proses Ekstraksi Fitur

Penelitian Skripsi ini lebih menitikberatkan pada pencarian fitur-fitur yang dapat merepresentasikan perbedaan kondisi tekstur jalan menggunakan pendekatan GLCM. Hal tersebut didasarkan pada kenyataan bahwa citra tekstural dengan ciri statistik yang berbeda, akan dengan mudah dibedakan satu sama lain. Untuk melakukan proses analisis menggunakan pendekatan statistik GLCM, maka diperlukan suatu hubungan ketetanggaan dari piksel-piksel dalam citra *grayscale.*

Hubungan semacam itu sering dinamakan dengan sifat kookurensi. Penjelasan lebih lanjut mengenai pendekatan statistik GLCM telah dipaparkan secara detail pada bab dasar teori. Berikut adalah flowchart dari proses metode GLCM :



Gambar 3. 3. Flowchart pembentukan matriks kookuransi secara umum

Sebelum citra diubah ke bentuk antara histogram jumlah dan selisih, maka citra berwarna (RGB) yang akan digunakan sebagai masukan/input, terlebih dahulu diubah ke bentuk *gray scale*, dengan menggunakan fungsi matlab "RGB2GRAY". Setelah itu, maka citra siap diubah ke bentuk histogram jumlah dan selisih. Misal citra grayscale dengan nilai keabuan pada suatu titik (piksel) adalah g(i,j) (0 < g(i,j) <G), lebar citra "L", panjang "P" dan histogram jumlah yang akan dihasilkan adalah S(x) dan histogram selisih D(x)

3.2.4. Tahap Perbandingan

Komponen masukan terdiri dari 100 citra tekstur jalan dan masing-masing citra tersebut akan dihitung nilai-nilai fitur tekstur Haralick, sebanyak 13 fitur. Nilai-nilai fitur tekstur akan disimpan dalam suatu matriks. Kemudian, fitur-fitur tersebut akan dibandingkan dengan parameter referensi yang didapatkan dari segmentasi citra dijital jalan yang dilakukan secara manual, yaitu segmantasi citra jalan dan segmentasi citra bukan jalan.

Proses perbandingan yang dilakukan menggunakan metode koefisien korelasi. Penghitungan koefisien korelasi tersebut dilakukan menggunakan fungsi internal MATLAB. Dalam proses perhitungan ini, setiap parameter referensi segmentasi citra jalan dan segmentasi citra bukan jalan digabungkan dengan matriks fitur tekstur Haralick. Perhitungan koefisien korelasi ini akan menghasilkan matriks dengan dimensi 10 x 10 elemen, yang merepresentasikan hubungan antara setiap fitur-fitur tekstur tersebut dengan parameter referensi. Korelasi merupakan suatu ukuran statistik yang menunjukkan seberapa dekat hubungan antara dua kelas yang berbeda atau lebih. Nilai korelasi yang dihasilkan akan memiliki jangkauan nilai -1 s/d +1 , dimana nilai -1 menunjukkan hasil korelasi negatif, nilai +1 menunjukkan korelasi positif, dan nilai 0 menunjukkan tidak ada korelasi apapun (perubahan nilai dari suatu kelas tidak akan memberikan dampak apapun terhadap kelas lainnya). Matriks korelasi ini akan menjadi keluaran dari blok sistem ini, yang selanjutnya akan digunakan sebagai masukan bagi proses seleksi fitur.

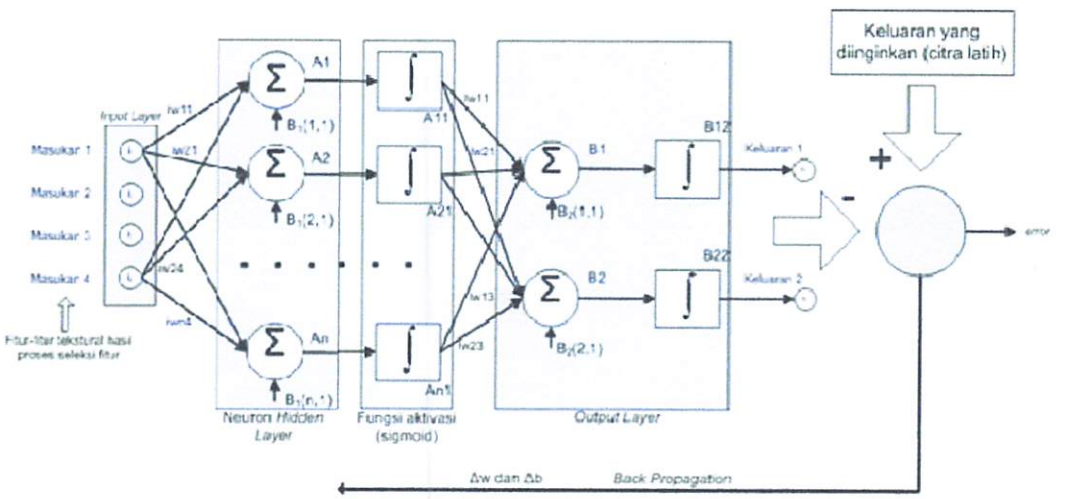
3.2.5. Proses Seleksi Fitur

Setelah melakukan proses perhitungan koefisien korelasi, maka hasil yang didapat adalah sebuah matriks korelasi berukuran 10 x 10 elemen. Tujuan dari penelitian skripsi ini adalah untuk menentukan suatu parameter yang dapat mengenali *object* jalan dan *object* bukan jalan. Dari 13 fitur tekstur yang diproses pada tahap sebelumnya untuk setiap citra dijital jalan,

maka akan ditentukan beberapa fitur yang memiliki koefisien korelasi terbesar. Komponen masukan dari blok sistem ini adalah matriks korelasi dengan dimensi 10 x 10 elemen. Di dalam proses seleksi fitur ini, maka dipilih sebanyak 3 buah fitur tekstur yang akan menjadi komponen masukan untuk sistem jaringan saraf tiruan (JST), baik dalam proses pelatihan bobot-bobot neuron maupun dalam proses analisis utama dengan menggunakan bobot yang telah didapatkan dari tahap pelatihan JST.

3.2.6. Proses Pelatihan Jaringan Saraf Tiruan (JST)

Pada bagian ini akan dijelaskan mengenai proses pelatihan dari sistem sebanyak 210 citra dijital yang akan digunakan untuk proses pelatihan sistem JST , yang terdiri dari 80 citra jalan dan 150 citra bukan jalan. Fitur-fitur yang merupakan hasil seleksi fitur pada bagian sebelumnya akan menjadi komponen input dari sistem JST. Tujuan penggunaan sistem JST ini adalah untuk membuat suatu model yang dapat menghasilkan nilai parameter yang dapat merepresentasikan *object* jalan dan *object* bukan jalan setiap fitur hasil proses ekstraksi fitur citra tekstural. Dalam kasus ini, sistem JST digunakan untuk menghasilkan suatu nilai yang dianggap mewakili parameter *object* jalan dan parameter *object* bukan jalan dengan menggunakan fitur-fitur tekstural. Sistem pelatihan JST yang dirancang akan menggunakan algoritma propagasi balik (*back propagation)*



Gambar 3. 4. Diagram sistem pelatihan JST metode propagasi balik

Pada gambar diatas terlihat bahwa sistem JST yang dirancang (baik dalam proses pelatihan maupun dalam sistem utama nanti) membutuhkan empat buah masukan, yang merupakan fitur-fitur hasil seleksi. Khusus untuk proses pelatihan JST ini, komponen data latih berasal dari kombinasi nilai dari segmentasi *object* jalan dan segmentasi *object* bukan jalan.

Untuk model JST utama, data latih dibentuk dari sekelompok pasangan nilai segmentasi *object* jalan dan segmentasi *object* bukan jalan

Berikut ini akan dijelaskan mengenai pembentukkan kelompok-kelompok data latih secara umum. Pertama-tama, komponen citra latih yang menjadi acuan dalam proses pelatihan ini dipilih berdasarkan nilai parameter segmentasi *object* jalan dan segmentasi *object* bukan jalan dari masing-masing sampel. Proses tersebut akan menghasilkan sebuah matriks berukuran 100 x 2 yang berisi nilai parameter referensi yang sudah diurutkan untuk dijadikan sebagai data latih. Agar matriks tersebut dapat digunakan sebagai komponen data latih dalam proses pelatihan JST, maka matriks tersebut harus di *transpose* terlebih dahulu, dimana bagian baris matriks akan menjadi kolom matriks dan sebaliknya.

Untuk komponene masukan, dimana sebelumnya telah didapatkan suatu matriks yang berisi fitur tekstur Haralick unutk 100 citra dijital dengan dimensi 100x13, maka proses seleksi fitur akan menyebabkan berkurangnya dimensi matriks tersebut menjadi 100x4, dimana lebar empat kolom menunjukkan jumlah fitur tekstur hasil proses seleksi fitur. Dalam proses pelatihan ini, jumlah citra yang dibutuhkan adalah sebanyak 100 buah citra dijital. Pemilihan citra latih tersebut ditentukan menurut konfigurasi dari masing-masing model JST utama, yaitu berdasarkan pengurutan atas nilai parameter segmentasi *object* jalan dan segmentasi *object* bukan jalan.

Dari urutan citra-citra tersebut akan diperoleh suatu kelompok nilai fitur tekstur, dalam posisi horisontal (bagian baris dari matriks fitur tekstur). Setelah proses penghitungan nilai fitur tekstur dari citra-citra latih yang telah ditentukan, maka akan dihasilkan sebuah matriks fitur tekstur dengan dimensi 100 x 4 elemen. Untuk menjadikan matriks tersebut sebagai masukan dalam sistem pelatihan JST, maka matriks tersebut perlu di *transpose* terlebih dahulu, sehingga pada akhirnya dimensi matriks menjadi 4 x 100 elemen.

Dari perlakuan yang diberikan terhadap matriks fitur tekstur dan juga matriks parameter data latih, setiap kolom dari matriks masukan akan berkorespondensi dengan kolom matriks data latih. Hal tersebut berarti untuk sebuah citra yang sama, maka data fitur tekstur dan data parameter referensi, yang disimpan pada arah vertikal (bagian kolom dari masing-masing matriks, setelah di *tranpose*), akan memasuki sistem pelatihan secara bersamaan. Proses tersebut akan menyebabkan data fitur tekstur akan dibentuk sedemikian rupa sehingga menyerupai komponen data latih, yaitu nilai-nilai data yang berasal dari parameter referensi.

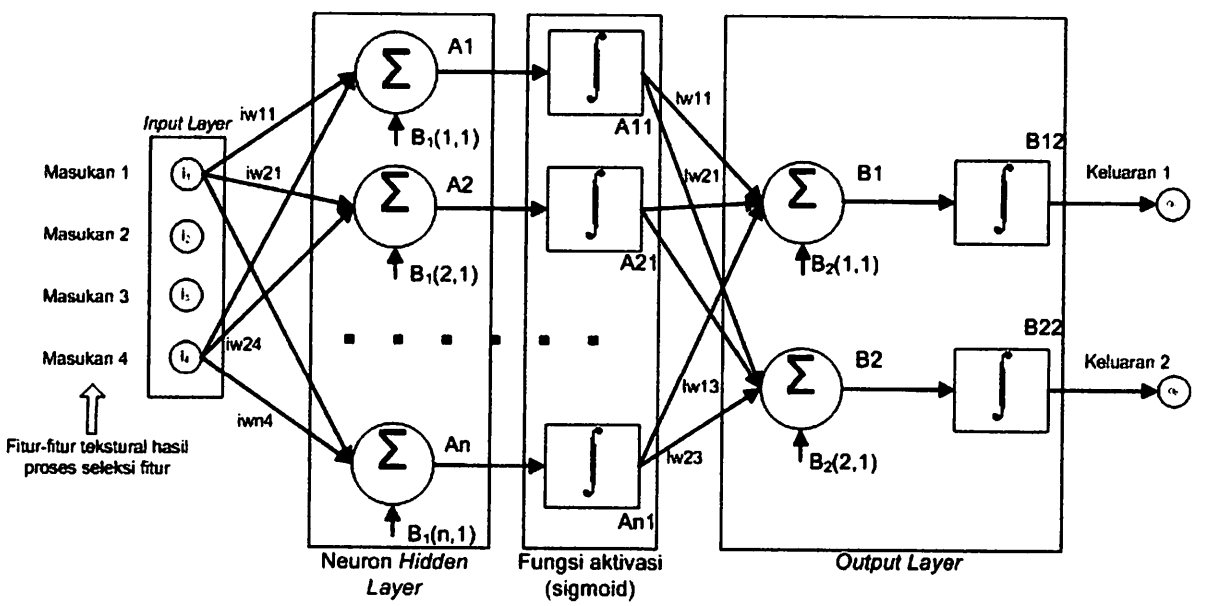
Sistem JST yang akan dirancang terdiri dari beberapa lapisan, yaitu *layer* masukan,

bagian *hidden layer* sebanyak satu *layer* , dan keluaran. Proses untuk menentukan konfigurasi/model JST yang sesuai dalam penelitian ini akan dilakukan dengan membuat suatu variasi dari jumlah neuron yang berada di bagian *hidden layer.*

3.2.7. Proses Analisis Citra Menggunakan Jaringan Saraf Tiruan

Proses analisis citra tekstur jalan akan dilakukan menggunakan sistem JST dengan algoritma *feed-forward back propagation* . Sistem JST yang dirancang memiliki konfigurasi yang sama dengan sistem JST di dalam proses pelatihan, yaitu memiliki *layer* masukan dengan tiga buah masukan, *layer* keluaran dengan dua keluaran, dan *hidden layer* sebanyak satu lapis.

Tujuan dari penggunaan sistem JST ini adalah untuk mendapatkan suatu nilai keluaran dari citra jalan kulit yang memiliki kesalahan/perbedaan terkecil terhadap nilai parameter referensi untuk setiap sampel. Kesalahan model ditentukan dengan mencari nilai MSE (*Mean Squared Error*). Diagram blok dari sistem JST yang digunakan dalam penelitian ini ditunjukkan dengan gambar berikut :



Gambar 3. 5. Diagram sistem JST algoritma *feed-forward back propagation*

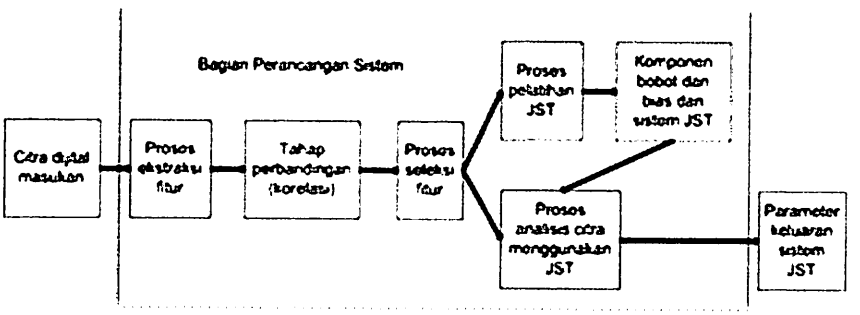
Seperti telah dijelaskan sebelumnya bahwa sistem JST ini memiliki konfigurasi yang hampir sama dengan sistem JST pada proses pelatihan, dimana komponen masukan merupakan fitur-fitur tekstur hasil seleksi dan komponen keluaran merupakan parameter tekstur jalan yang diharapkan dapat memberikan suatu deskripsi mengenai karakteristik tekstur jalan. Komponen masukan dan komponen keluaran dari sistem JST ini merupakan suatu

bilangan numerik. Diagram pada gambar 3.5 di atas hanya merupakan salah satu bagian saja dari keseluruhan sistem analisis citra yang digunakan dalam penelitian skripsi ini.

3.3. Ringkasan Sistem Analisis Citra

Pada sub-bab ini akan diberikan suatu ringkasan secara menyeluruh darisistem analisis citra yang dijelaskan pada bab-bab sebelumnya. Tahap pertama yang dilakukan adalah proses akuisisi citra dijital yang berasal dari citra jalan. Proses akuisisi citra ini akan menghasilkan citra dijital tekstur jalan yang akan digunakan dalam tahap selanjutnya.

Setelah proses akuisisi citra (dilanjutkan dengan tahap *pre-processing*), maka citra dijital tersebut akan digunakan dalam proses ekstraksi fitur tekstur Haralic. Dengan menggunakan perangkat lunak MATLAB, beberapa fitur tekstur Haralick akan dihitung dari setiap citra tersebut. Setelah tahap ekstraksi fitur, maka langkah selanjutnya adalah menghitung korelasi dari setiap fitur tekstur tersebut terhadap parameter referensi yang didapatkan dari segmentasi citra jalan



Gambar 3. 6. Diagram blok proses analisis citra secara sederhana

Proses perbandingan menggunakan korelasi bertujuan untuk mengetahui bagaimana hubungan antara fitur tekstur yang dihitung dengan parameter referensi . Setelah menghitung koefisien korelasi dari fitur-fitur tersebut, maka langkah selanjutnya melakukan proses seleksi fitur berdasarkan nilai koefisien korelasi terbesar. Tahap ini akan mengambil empat buah fitur untuk masing-masing parameter referensi. Proses seleksi fitur bertujuan untuk menentukan fitur-fitur mana yang dapat merepresentasikan kedua parameter referensi tersebut, karena ada kemungkinan semua fitur tekstur Haralick belum tentu dapat merepresentasikan karakteristik tekstur jalan dengan baik.

Empat buah fitur hasil seleksi akan menjadi komponen masukan dalam sistem JST, baik untuk proses pelatihan JST maupun sistem JST yang akan digunakan dalam proses analisis citra tekstur jalan. Perbedaan antara proses pelatihan dan proses analisis adalah pada proses pelatihan citra jalan diurutkan menurut konfigurasi nilai segmentasi citra jalan nya ,

sedangkan pada proses analisis citra jalan tidak perlu diurutkan, tetapi hanya menggunakan susunan citra jalan yang telah ada sebelumnya. Pelatihan JST bertujuan untuk menentukan bobot-bobot hubungan neuron beserta komponen biasnya, berdasarkan nilai-nilai parameter referensi. Beberapa model JST akan diujicoba dengan melakukan variasi jumlah neuron pada *hidden layer* . Setelah menentukan bobot dan bias dari sistem JST, maka kelompok nilai bobot dan bias tersebut akan digunakan dalam sistem JST untuk analisis citra tekstur jalan. Pemilihan model JST yang terbaik didasarkan pada nilai MSE terkecil terhadap parameter referensi. Diharapkan model JST tersebut akan dapat menghitung nilai-nilai parameter yang diperlukan untuk mengetahui karakteristik perubahan tekstur jalan.

BAB IV

IMPLEMENTASI DAN PEMBAHASAN

4.1 Lingkungan Implementasi

Lingkungan implementasi yang akan dijelaskan pada sub bab ini adalah lingkungan perangkat keras dan lingkungan perangkat lunak yang digunakan untuk pengembangan sistem aplikasi JST dengan metode *Backpropagation* untuk mengenali jalur jalan kendaraan darat.

4.1.1 Lingkungan Perangkat Keras

Perangkat keras yang digunakan dalam pengembangan sistem aplikasi JST dengan metode *Backpropagation* untuk mengenali jalur jalan kendaraan darat adalah

1. Dual Core 2.0 GHz
2. 1,00 GB DDR2
3. Harddisk dengan kapasitas 320 GB
4. Monitor 15"
5. Keyboard
6. Mouse

4.1.2 Lingkungan Perangkat Lunak

Perangkat keras yang digunakan dalam pengembangan sistem aplikasi JST dengan metode *Backpropagation* untuk mengenali jalur jalan kendaraan darat ini adalah :

1. Sistem Operasi Windows 7 Ultimate
2. Matlab 7.04
3. Adobe Photoshop CS4

4.2 Implementasi Program

Pada sub bab ini akan dibahas mengenai implementasi dari sistem aplikasi JST dengan metode *Backpropagation* untuk mengenali jalur jalan kendaraan darat.

4.2.1. Tampilan Program.

Pada program aplikasi JST dengan menggunakan metode *backpropagation* untuk mengenali jalur jalan kendaraan darat, akan terdapat beberapa komponen yang digunakan untuk mempermudah pengguna menggunakan program deteksi jalan ini.

Gambar 4.1 Tampilan GUI Program

Pada program Alplikasi JST dengan menggunakan metode *backpropagation* unuk mengenali jalur jalan kendaraan darat terdiri dari aplikasi.m yang berisi callback dari GUI dan sebagai interaksi komponen-komponen yang digunakan pada GUI program deteksi jalan.

- listing program untuk training jst dalam Gambar 4.2.

```
%training jst
data=data';
clear net
clear net2
net=newff(minmax(data),[10 2],{'tansig','purelin'});
net.trainParam.epochs = 500;

init(net);
net2 = train(net,data,t'); % training jaringannya
```

Gambar 4.2 *Source code training* JST

- listing program untuk data latih dalam Gambar 4.3.

```
n1=str2num(get(handles.ed_JmlDataLatih,'string'));

dir=cd;
filename=sprintf('%s\\%s',dir,'jalan');
disp(filename);
cd(filename);

for i=1 :n1
   s = sprintf('%s\\%d.jpg',filename,i) % untk menggabungkan antara string dan angka
```

```
    gb=(imread(s));
    gb1 = gb(:,:,1);
    gb2 = gb(:,:,2);
    gb3 = gb(:,:,3);

    xoffset = [0 1; -1 1; -1 0; -1 -1];

    glcm = graycomatrix (gb1,'offset',xoffset);
    stats = graycoprops (glcm,{'all'});
    data1 (i,1)=mean (stats.Contrast);
    data1 (i,3)=mean (stats.Energy);
    data1 (i,4)=mean (stats.Homogeneity);

    glcm = graycomatrix (gb2,'offset',xoffset);
    stats = graycoprops (glcm,{'all'});
    data1 (i,5)=mean (stats.Contrast);
    data1 (i,7)=mean (stats.Energy);
    data1 (i,8)=mean (stats.Homogeneity);

    glcm = graycomatrix (gb3,'offset',xoffset);
    stats = graycoprops (glcm,{'all'});
    data1 (i,9)=mean (stats.Contrast);
    data1 (i,11)=mean (stats.Energy);
    data1 (i,12)=mean (stats.Homogeneity);
end

data1
n2=str2num(get(handles.ed_JmlDataLatihBknJalan,'string'));

dir=cd;
filename=sprintf('%s\\%s',dir,'bukan jalan');
disp(filename);
cd(filename);

for i=1 :n2
   s = sprintf('%s\\%d.jpg',filename,i) % untk menggabungkan antara string dan angka
   gb=(imread(s));
   gb1 = gb(:,:,1);
   gb2 = gb(:,:,2);
   gb3 = gb(:,:,3);

   xoffset = [0 1; -1 1; -1 0; -1 -1];

   glcm = graycomatrix (gb1,'offset',xoffset);
   stats = graycoprops (glcm,{'all'});
   data2 (i,1)=mean (stats.Contrast);
   data2 (i,3)=mean (stats.Energy);
   data2 (i,4)=mean (stats.Homogeneity);

   glcm = graycomatrix (gb2,'offset',xoffset);
   stats = graycoprops (glcm,{'all'});
   data2 (i,5)=mean (stats.Contrast);
   data2 (i,7)=mean (stats.Energy);
```

```
        data2 (i,8)=mean (stats.Homogeneity);

        glcm = graycomatrix (gb3,'offset',xoffset);
        stats = graycoprops (glcm,{'all'});
        data2 (i,9)=mean (stats.Contrast);
        data2 (i,11)=mean (stats.Energy);
        data2 (i,12)=mean (stats.Homogeneity);
    end
    cd ..
    data=[data1;data2];
    t1= [ones(n1,1);zeros(n2,1)];
    t2= [zeros(n1,1);ones(n2,1)];
    t=[t1 t2];
```

Gambar 4.3 *Source code* data latih

- Listing program test segmentasi dalam Gambar 4.4.

```
gambar =a;
 gambar2=rgb2gray(a); % convert to gray mode
 p=10;
 l=10;
 maxp=size (gambar,1)  %320
 maxl=size (gambar,2) %240

 np=maxp/p %4
 nl=maxl/l %3

 seg_gambar = zeros (p,l,3,np*nl);
 size (seg_gambar)

 %proses awal segmentasi
 for i = 1:np
    for j  = 1:nl
       xawal = (i-1)*p+1;
       xakhir= (i)*p;
       yawal= (j-1)*l+1;
       yakhir=(j)*l;
       nourut= (i-1)*nl+j;
       seg_gambar (:,:,:,nourut) = gambar ( xawal:xakhir, yawal:yakhir, :);
       gb_r = gambar ( xawal:xakhir, yawal:yakhir, 1);
       gb_g = gambar ( xawal:xakhir, yawal:yakhir, 2);
       gb_b = gambar ( xawal:xakhir, yawal:yakhir, 3);

       xoffset = [0 1; -1 1; -1 0; -1 -1];
       glcm = graycomatrix (gb_r,'offset',xoffset);
       stats = graycoprops (glcm,{'all'});
       % hasil glcm disimpan di variabel datacek
       datacek (i,j,1)=mean (stats.Contrast);
       datacek (i,j,2)=mean (stats.Correlation);
       datacek (i,j,3)=mean (stats.Energy);
```

```
        data2 (i,8)=mean (stats.Homogeneity);

        glcm = graycomatrix (gb3,'offset',xoffset);
        stats = graycoprops (glcm,{'all'});
        data2 (i,9)=mean (stats.Contrast);
        data2 (i,11)=mean (stats.Energy);
        data2 (i,12)=mean (stats.Homogeneity);
    end
    cd ..
    data=[data1;data2];
    t1= [ones(n1,1);zeros(n2,1)];
    t2= [zeros(n1,1);ones(n2,1)];
    t=[t1 t2];
```

Gambar 4.3 *Source code* data latih

- Listing program test segmentasi dalam Gambar 4.4.

```
gambar =a;
 gambar2=rgb2gray(a); % convert to gray mode
 p=10;
 l=10;
 maxp=size (gambar,1)  %320
 maxl=size (gambar,2) %240

 np=maxp/p %4
 nl=maxl/l %3

 seg_gambar = zeros (p,l,3,np*nl);
 size (seg_gambar)

 %proses awal segmentasi
 for i = 1:np
    for j  = 1:nl
       xawal = (i-1)*p+1;
       xakhir= (i)*p;
       yawal= (j-1)*l+1;
       yakhir=(j)*l;
       nourut= (i-1)*nl+j;
       seg_gambar (:,:,:,nourut) = gambar ( xawal:xakhir, yawal:yakhir, :);
       gb_r = gambar ( xawal:xakhir, yawal:yakhir, 1);
       gb_g = gambar ( xawal:xakhir, yawal:yakhir, 2);
       gb_b = gambar ( xawal:xakhir, yawal:yakhir, 3);

       xoffset = [0 1; -1 1; -1 0; -1 -1];
       glcm = graycomatrix (gb_r,'offset',xoffset);
       stats = graycoprops (glcm,{'all'});
       % hasil glcm disimpan di variabel datacek
       datacek (i,j,1)=mean (stats.Contrast);
       datacek (i,j,2)=mean (stats.Correlation);
       datacek (i,j,3)=mean (stats.Energy);
```

```
        datacek (i,j,4)=mean (stats.Homogeneity);

        glcm = graycomatrix (gb_g,'offset',xoffset);
        stats2 = graycoprops (glcm,{'all'});
        % hasil glcm disimpan di variabel datacek
        datacek (i,j,1)=mean (stats.Contrast);
        datacek (i,j,2)=mean (stats.Correlation);
        datacek (i,j,3)=mean (stats.Energy);
        datacek (i,j,4)=mean (stats.Homogeneity);

        glcm = graycomatrix (gb_b,'offset',xoffset);
        stats3 = graycoprops (glcm,{'all'});
        % hasil glcm disimpan di variabel datacek
        datacek (i,j,1)=mean (stats.Contrast);
        datacek (i,j,2)=mean (stats.Correlation);
        datacek (i,j,3)=mean (stats.Energy);
        datacek (i,j,4)=mean (stats.Homogeneity);
        try
          hasil = [mean(stats.Contrast); 0; mean(stats.Energy); mean(stats.Homogeneity)];
          hasil = [hasil; mean(stats2.Contrast); 0; mean(stats2.Energy);
mean(stats2.Homogeneity)];
          hasil = [hasil; mean(stats3.Contrast); 0; mean(stats3.Energy);
mean(stats3.Homogeneity)];

          y=sim(net2,hasil);
          if (y(1)>=0)
            gambar2(xawal:xakhir, yawal:yakhir)=0;
          else
            gambar2(xawal:xakhir, yawal:yakhir)=255;
          end

        catch
        end
      end;
end;
figure(1);
imshow(a);
figure(2);
imshow(gambar2);
```

Gambar 4.4 *Source code* tes segmentasi

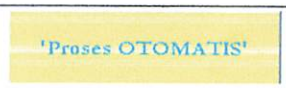
Cara kerja program deteksi jalan ini, pertama-tama adalah proses ekstraksi fitur tekstur dari suatu citra jalan yang di jalankan pada data latih. Setelah proses ekstraksi fitur, maka tahap selanjutnya adalah membandingkan nilai-nilai fitur tersebut dengan nilai parameter referensi citra jalan berupa hasil segmentasi yang diolah secara manual. Selanjutnya citra jalan dengan menggunakan algoritma GLCM akan di uji hasil dari segmentasi yang telah di proses

sebelumnya pada data latih, yang kemudian hasil seleksi fitur akan menjadi *inputan* pada *training* JST dan di proses selannjutnta pada test segmentasi.

4.2.2 Komponen yang Digunakan pada Program Deteksi Jalan

Dalam program deteksi jalan ini terdapat beberapa komponen yang telah disediakan oleh Matlab 7.0.4. Dalam Tabel 4.1.adalah komponen-komponen yang ada pada program deteksi Jalan

Tabel 4.1 Komponen pada Program Deteksi Jalan

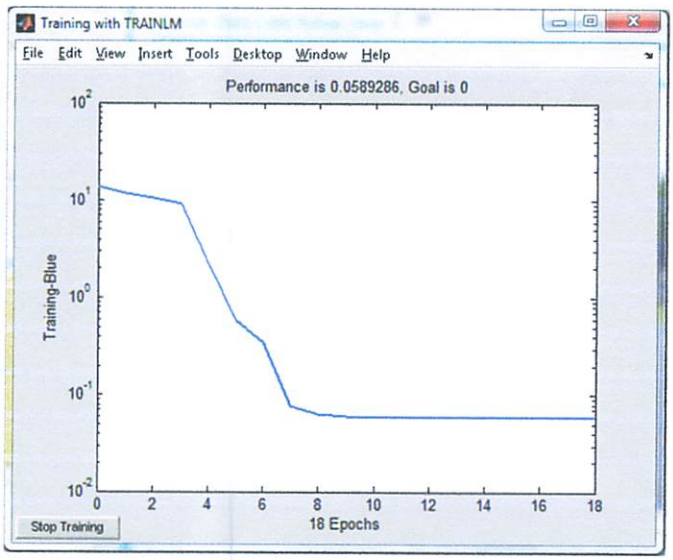
No	Komponen	Gambar
1	Push button data latih	
2	Push button *training* JST	
3	Push button ekstraksi fitur	
4	Push button Seleksi Fitur	
5	Push button proses otomatis	
6	Push button buka gambar	

4.3 Pengujian sistem

Gambar 4.5. adalah hasil implementasi program aplikasi JST dengan menggunakan metode *backpropagation* unutk megenali jalur jalan kendaraan darat.

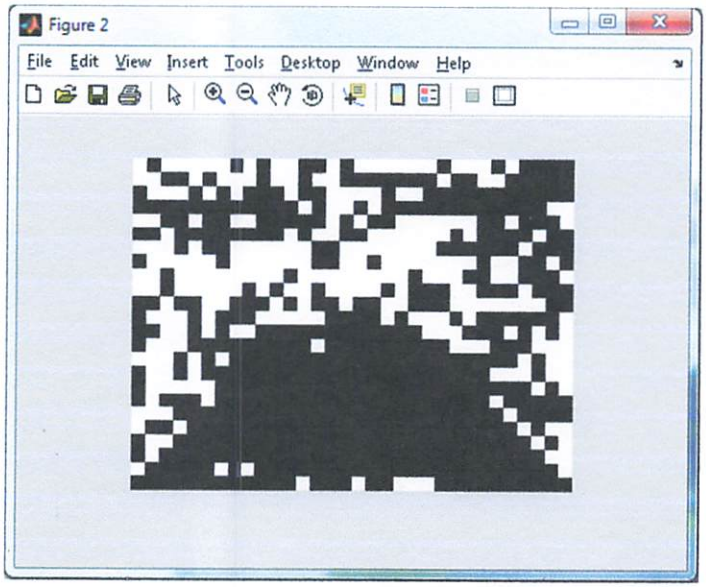
Gambar 4.5 Tampilan pada Saat Program Dijalankan

Setelah menentukan jumlah data latih yang akan di *inputkan* , selanjutnya memulai proses ekstraksi citra dengan menekan tombol data latih yangh dilanjutkan dengan proses *training* JST dengan menekan tombol 'Training JST' berikut tampilan setelah tombol 'Training JST ' di tekan.



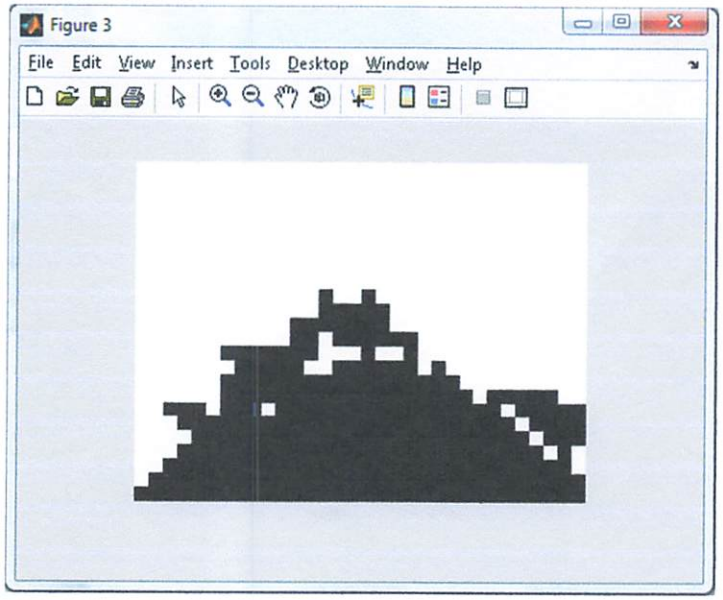
Gambar 4.6 Tampilan proses *training* JST

Setelah proses *training* JST selesai dilakukan , dilanjutkan dengan menekan tombol 'Ekstraksi Fitur' , pada tombol ini bobot yang di peroleh dari proses *training* JST akan di olajh untuk proses sistem utam JST sehingga akan didapatkan *Outputan* tahap awal, yang ditunjukkan dalam Gambar 4.7.



Gambar 4.7 Tampilan hasil test segmentasi

Proses selanjutnya dengan menekan tombol ' seleksi fitur' yang merupakan hasil *final* dari program ini, yang ditunjukkan dalam Gambar 4.8.



Gambar 4.8 Tampilan *outputan*

4.3.1 Hasil Percobaan

Untuk memperoleh struktur jaringan syaraf tiruan yang terbaik yang digunakan untuk deteksi jalan, maka dilakukan pengujian terhadap sistem. Pengujian dilakukan dengan melatih jaringan syaraf tiruan dengan parameter-pamameter yang berbeda, yang nantinya akan diambil satu struktur jaringan yang terbaik yang digunakan untuk melakukan deteksi jalan

4.3.1.2 Pengaruh Jumlah *Neuron* pada *Hidden Neuron*

Untuk menentukan jumlah *hidden neuron* pada sistem dilakukan melalui proses *trial and error*. Kemudian jumlah neuron yang diujikan yaitu 10, 15, 20, dan 25. Data hasil percobaan untuk mengetahui pengaruh jumlah unit neuron pada *hidden layer* dapat dilihat dalam tabel 4.2.

Tabel 4.2 Hasil percobaan pengaruh *hidden layer*

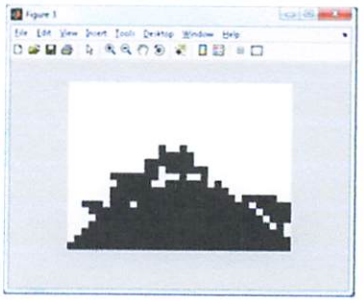
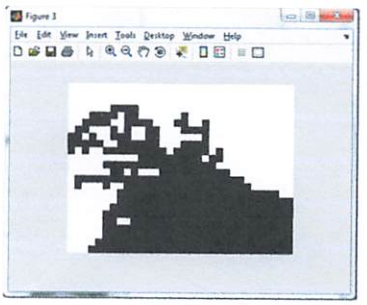
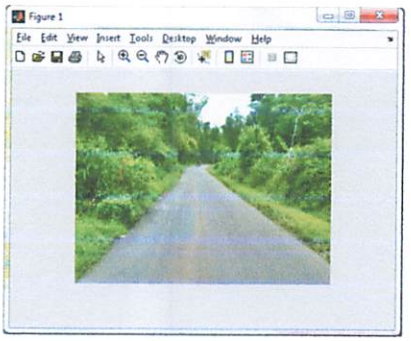
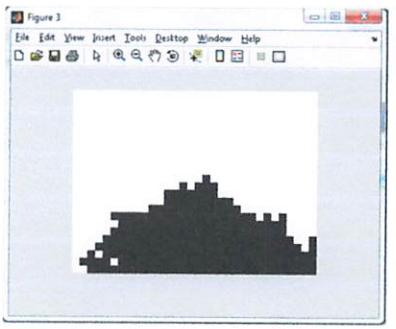
Percobaan ke	Maks. epoch	Hidden Neuron	MSE
1	500	10	0.0770541
2	500	15	0.0589325
3	500	20	0.0589325
4	500	25	0.0589286

dapat dilihat bahwa makin banyak jumlah neuron pada hidden layer, maka eror (*mean squared error*) yang dihasilkan akan semakin kecil, namun jumlah neuron yang terlalu besar tidak akan efesien lagi untuk menurunkan mse. Dari hasil penelitian diperoleh jumlah neuron yang optimal adalah 15.

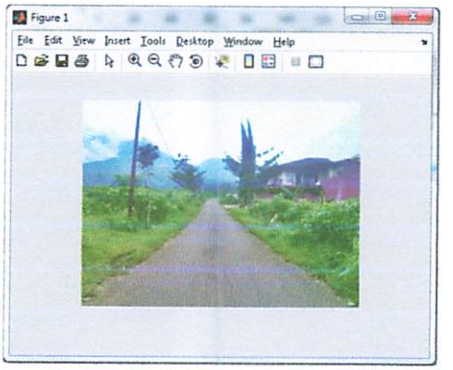
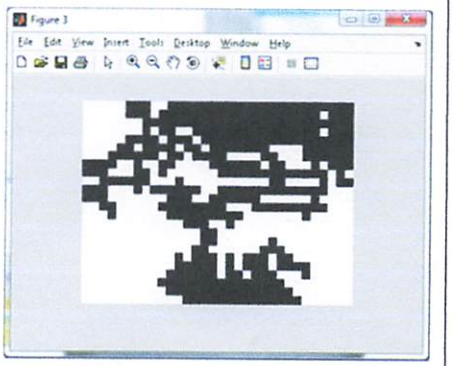
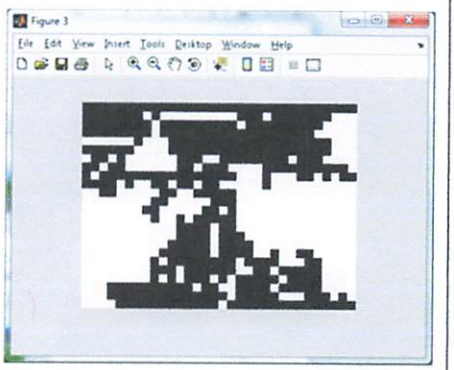
4.4 Analisa Hasil

Setelah mendapatkan arsitektur jaringan yang optimal, kemudian dilanjutkan dengan pengujian terhadap data yang sudah pernah dilakukan pembelajaran dan data yang belum pernah dilakukan pembelajaran sebelumnya. Kumpulan data uji terdiri 3 gambar yang telah dilakukan pembelajaran dan 2 gambar yang belum pernah melakukan pembelajaran. Bobot yang dipakai untuk pengujian adalah bobot dari hasil pembelajaran dengan jumlah unit pada *hidden layer* sebesar 15 unit, dan *max epoch* sebesar 500. Hasil pengujian terhadap data yang sudah pernah dilakukan pembelajaran dapat dilihat pada table berikut :

Tabel 4.3 Hasil gambar jalan yang sudah pernah dilakukan pembelajaran

No	Citra *input*	Citra *output*
1.		
2.		
3.		

Tabel 4.4 Hasil gambar jalan yang belum pernah dilakukan pembelajaran

No	Citra *input*	Citra *output*
1.		
2.		

BAB V

PENUTUP

Dari beberapa uraian yang telah dikemukakan pada bab-bab sebelumnya, dapat diambil kesimpulan dan saran sebagai berikut :

5.1 Kesimpulan

Berdasarkan hasil pengujian yang dilakukan pada proyek akhir ini, maka disimpulkan bahwa :

1. Pencahayaan dan posisi letak pengambilan citra gambar dengan kamera mempengaruhi dalam proses deteksi.
2. Dari hasil penelitian, arsitektur Jaringan Saraf Tiruan terbaik yang berhasil disusun adalah jaringan saraf *backpropagation* yang terdiri atas lapisan *input* dengan 15 *neuron* dengan maksimal epoch 500 .
3. Setelah dilakukan pelatihan denga n500 iterasi, didapatkan nilai MSE yang berbeda-beda. Semakin banyak jumlah *neuron hidden*, maka semakin akurat hasil yang didapatkan (Tabel 4.2)

5.2 Saran

Perencanaan dan pembuatan aplikasi ini terdapat beberapa hal yang perlu diperhatikan dalam pengembangan lebih lanjut yaitu :

1. Penggunaan data pelatihan yang semakin banyak akan memberikan keakuratan jaringan terhadap data uji. Pada penelitian ini masih membutuhkan data training dari masing-masing citra jalan yang berbeda.
2. Penelitian ini masih perlu dikembangkan lebih lanjut untuk dapat mengidentifikasi objek jalan dan objek bukan jalan.

DAFTAR PUSTAKA

Analisis Efikasi *Moisturizer* Terhadap Kulit Manusia Dengan Metode Haralick dan Jaringan Saraf Tiruan. ITB.

Kristanto, Andri. 2004. *Jaringan Syaraf Tiruan (Konsep dasar,Algoritma dan Aplikasi*). Gaya Media: Yogyakarta.

Purnomo ,Mauridhi Hery; Kurniawan, Agus. 2006.*Supervised Neural Networks dan aplikasinya*. Graha Ilmu: Yogyakarta.

Kusumadewi, S. 2003. *Artificial Intelligence (Teknik & Aplikasinya)*. Graha Ilmu: Yogyakarta.

Kusumadewi, S. 2003. *Analisis Jaringan Saraf Tiruan dengan Metode Backpropagation Untuk Mendeteksi Gangguan Psikologi.* Jurnal Teknik Informatika, Universitas Islam Indonesia: Yogyakarta.

Fadlisyah., 2007, *Computer Vision dan Pengolahan Citra*, Penerbit Andi, Yogyakarta.

http://ussie.staff.gunadarma.ac.id/Downloads/files/13436/PENGOLAHAN+CITRA.pdf

http://www.mathworks.com/access/helpdesk/help/techdoc/creating_plots/f2-1667.html

http://www.ittelkom.ac.id/library/index.php?view=article&catid=15:pemrosesan-sinyal&id=383:pengolahan-citra-digital&option=com_content&Itemid=15

L A M P I R A N

PERKUMPULAN PENGELOLA PENDIDIKAN UMUM DAM TEKNOLOGI NASIONAL MALANG
INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
FAKULTAS TEKNIK SIPIL DAN PERENCANAAN
PROGRAM PASCASARJANA MAGISTER TEKNIK
PT. BNI (PERSERO) MALANG
BANK NIAGA MALANG
Kampus I : Jl. Bendungan Sigura-gura No. 2 Telp. (0341) 551431 (Hunting), Fax. (0341) 553015 Malang 65145
Kampus II : Jl. Raya Karanglo, Km 2 Telp. (0341) 417636 Fax. (0341) 417634 Malang

BERITA ACARA UJIAN SKRIPSI

FAKULTAS TEKNOLOGI INDUSTRI

Nama : MARTHALIA DEWI ANGGRAINI
NIM : 06.12.629
Jurusan : Teknik Elektro S-1
Konsentrasi : Teknik Komputer & Informatika
Judul Skripsi : Implementasi Jaringan Syaraf Tiruan Untuk Mengenali Jalur Jalan Kendaraan Darat Berbasis Kamera

Dipertahankan di hadapan Tim Penguji Ujian Skripsi Jenjang Program Strata Satu (S-1) pada :
Hari : Jumat
Tanggal : 18 Februari 2011
Dengan Nilai : 84,05 (A)

Panitia Ujian Skripsi :

Ketua Majelis Penguji

Ir. Yusuf Ismail Nakhoda, MT
NIP.Y.1018800189

Sekretaris Majelis Penguji

Dr. Eng. Aryuanto S. ST, MT
NIP.Y.1030800417

Anggota Penguji :

Dosen Penguji I

Joseph Dedy Irawan, ST.MT
NIP.Y.197404162005011002

Dosen Penguji II

Sonny Prasetio, ST, MT
NIP.P.1031000433

PERKUMPULAN PENGELOLA PENDIDIKAN UMUM DAM TEKNOLOGI NASIONAL MALANG
INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
FAKULTAS TEKNIK SIPIL DAN PERENCANAAN
PROGRAM PASCASARJANA MAGISTER TEKNIK
T. BNI (PERSERO) MALANG
BANK NIAGA MALANG
Kampus I : Jl. Bendungan Sigura-gura No. 2 Telp. (0341) 551431 (Hunting), Fax. (0341) 553015 Malang 65145
Kampus II : Jl. Raya Karanglo, Km 2 Telp. (0341) 417636 Fax. (0341) 417634 Malang

FORMULIR PERBAIKAN SKRIPSI

Dalam pelaksanaan Ujian Skripsi Jenjang Program Strata Satu (S-1) Jurusan Teknik Elektro Konsentrasi Teknik Komputer & Informatika, maka perlu adanya perbaikan Skripsi untuk mahasiswa :

Nama	:	MARTHALIA DEWI ANGGRAINI
NIM	:	06. 12. 629
Jurusan	:	Teknik Elektro S-1
Konsentrasi	:	Teknik Komputer & Informatika
Masa Bimbingan	:	21 Oktober s/d 21 April 2011
Judul Skripsi	:	Implementasi Jaringan Syaraf Tiruan Untuk Mengenali Jalur Jalan Kendaraan Darat Berbasis Kamera

Penguji&Tanggal	Uraian	Paraf
Penguji I 18 Februari 2011	-	
Penguji II 18 Februari 2011	Keterangan gambar dan tabel.	

Dosen Penguji :

Dosen Penguji I

Joseph Dedy Irawan, ST,MT
NIP.Y.197404162005011002

Dosen Penguji II

Sonny Prasetio, ST, MT
NIP.P.1031000433

Mengetahui :

Dosen Pembimbing I

Yusuf Ismail Nakhoda, ST, MT
NIP.Y.1018800189

Dosen Pembimbing II

Dr. Eng. Aryuanto S. ST, MT
NIP.Y.1030800417

INSTITUT TEKNOLOGI NASIONAL
Jl. Bendungan Sigura – gura No.2
MALANG

FORMULIR BIMBINGAN SKRIPSI

Nama : Marthalia Dewi Anggraini
Nim : 06.12.629
Masa Bimbingan : 21 Oktober 2011 s/d 21 April 2011
Judul Skripsi : Implementasi Jaringan Syaraf Tiruan Untuk Mengenali Jalur Jalan Kendaraan Darat Berbasis Kamera

No	Tanggal	Uraian	Paraf Pembimbing
1	**12-08-2010**	**Konsultasi Program**	
2	**14-08-2010**	**Konsultasi Program**	
3	**13-10-2010**	**Konsultasi Program**	
4	**10-02-2011**	**Konsultasi Prograpm**	
5	**11-02-2011**	**Konsultasi Refisian Makalah Hasil**	
6	**12-02-2011**	**Konsultasi Makalah Hasil (fix)**	
7	**14-02-2011**	**Konsultasi Laporan (refisi gambar dan letak halaman)**	
8	**17-02-2011**	**Konsultasi Laporan dan Acc Laporan Dosen Pembimbing**	
9			
10			

Malang,

Dosen pembimbing

Ir. Yusuf Ismail Nakhoda, MT
NIP.Y.1018800189

Form S-4b

INSTITUT TEKNOLOGI NASIONAL
Jl. Bendungan Sigura – gura No.2
MALANG

FORMULIR BIMBINGAN SKRIPSI

Nama : Marthalia Dewi Anggraini
Nim : 06.12.629
Masa Bimbingan : 21 Oktober 2011 s/d 21 April 2011
Judul Skripsi : Implementasi Jaringan Syaraf Tiruan Untuk Mengenali Jalur Jalan Kendaraan Darat Berbasis Kamera

No	Tanggal	Uraian	Paraf Pembimbing
1	12-08-2010	Konsultasi Program	
2	14-08-2010	Konsultasi Program	
3	13-10-2010	Konsultasi Program	
4	01-02-2011	Konsultasi Program	
5	07-02-2011	Konsultasi Program	
6	11-02-2011	Konsultasi Program	
7	14-02-2011	Konsultasi Refisian Makalah Hasil	
8	17-02-2011	Konsultasi Laporan dan Acc Laporan Dosen Pembimbing	
9			
10			

Malang,

Dosen pembimbing

Dr. Eng. Aryuanto S. ST, MT
NIP.P.1030800417

Form S-4b

INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
JURUSAN TEKNIK ELEKTRO

Formulir Perbaikan Ujian Skripsi

Dalam pelaksanaan Ujian Skripsi Janjang Strata 1 Jurusan Teknik Elektro Konsentrasi T. Energi Listrik / T. Elektronika / T.Infokom, maka perlu adanya perbaikan skripsi untuk mahasiswa :

NAMA : Marthalia Dewi Anggraini
NIM : 06.12.629
Perbaikan meliputi :

- keterangan gambar & tabel diperbaiki lihat hal 5

Malang, 18/2/2014

(SONNY PRASETIO ST,MT)

LAMPIRAN A

Listing Program

```
function varargout = Applikasi(varargin)
% APPLIKASI M-file for Applikasi.fig
%      APPLIKASI, by itself, creates a new APPLIKASI or raises the existing
%      singleton*.
%
%      H = APPLIKASI returns the handle to a new APPLIKASI or the handle to
%      the existing singleton*.
%
%      APPLIKASI('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in APPLIKASI.M with the given input arguments.
%
%      APPLIKASI('Property','Value',...) creates a new APPLIKASI or raises the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before Applikasi_OpeningFunction gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to Applikasi_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help Applikasi

% Last Modified by GUIDE v2.5 03-Feb-2011 10:40:23

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @Applikasi_OpeningFcn, ...
                   'gui_OutputFcn',  @Applikasi_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
```

```
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before Applikasi is made visible.
function Applikasi_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to Applikasi (see VARARGIN)

%++++++++++++Menampilkan Background++++++++++++++++++++
background = importdata('background.jpg');
axes(handles.ax_background);
image(background);
axis off

%++++++++++++Menampilkan Gambar++++++++++++++++++
backgroundImage = importdata(get(handles.ed_lokasiimg,'string'));
axes(handles.ax_logo);
image(backgroundImage);
axis off

% Choose default command line output for Applikasi
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes Applikasi wait for user response (see UIRESUME)
```

```
% uiwait(handles.frm_Applikasi);


% --- Outputs from this function are returned to the command line.
function varargout = Applikasi_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


%++++++++++++++++++++++++++Tambah Data Latih++++++++++++++++++++++++++++++++
clear data1
clear data2
clear data
clc

n1=str2num(get(handles.ed_JmlDataLatih,'string'));

dir=cd;
filename=sprintf('%s\\%s',dir,'jalan');
disp(filename);
cd(filename);

for i=1 :n1
   s = sprintf('%s\\%d.jpg',filename,i) % untk menggabungkan antara string dan angka
   gb=(imread(s));
   gb1 = gb(:,:,1);
   gb2 = gb(:,:,2);
   gb3 = gb(:,:,3);

   xoffset = [0 1; -1 1; -1 0; -1 -1];

   glcm = graycomatrix (gb1,'offset',xoffset);
   stats = graycoprops (glcm,{'all'});
   data1 (i,1)=mean (stats.Contrast);
```

```
    data1 (i,3)=mean (stats.Energy);
    data1 (i,4)=mean (stats.Homogeneity);

    glcm = graycomatrix (gb2,'offset',xoffset);
    stats = graycoprops (glcm,{'all'});
    data1 (i,5)=mean (stats.Contrast);
    data1 (i,7)=mean (stats.Energy);
    data1 (i,8)=mean (stats.Homogeneity);

    glcm = graycomatrix (gb3,'offset',xoffset);
    stats = graycoprops (glcm,{'all'});
    data1 (i,9)=mean (stats.Contrast);
    data1 (i,11)=mean (stats.Energy);
    data1 (i,12)=mean (stats.Homogeneity);
end

data1
cd .. %direktori naik satu step
%n2=20; %jumlah data latih yang di masukan bkn jln
n2=str2num(get(handles.ed_JmlDataLatihBknJalan,'string'));

dir=cd;
filename=sprintf('%s\\%s',dir,'bukan jalan');
disp(filename);
cd(filename);

for i=1 :n2
    s = sprintf('%s\\%d.jpg',filename,i) % untk menggabungkan antara string dan angka
    gb=(imread(s));
    gb1 = gb(:,:,1);
    gb2 = gb(:,:,2);
    gb3 = gb(:,:,3);

    xoffset = [0 1; -1 1; -1 0; -1 -1];

    glcm = graycomatrix (gb1,'offset',xoffset);
    stats = graycoprops (glcm,{'all'});
    data2 (i,1)=mean (stats.Contrast);
    data2 (i,3)=mean (stats.Energy);
```

```
    data2 (i,4)=mean (stats.Homogeneity);

    glcm = graycomatrix (gb2,'offset',xoffset);
    stats = graycoprops (glcm,{'all'});
    data2 (i,5)=mean (stats.Contrast);
    data2 (i,7)=mean (stats.Energy);
    data2 (i,8)=mean (stats.Homogeneity);

    glcm = graycomatrix (gb3,'offset',xoffset);
    stats = graycoprops (glcm,{'all'});
    data2 (i,9)=mean (stats.Contrast);
    data2 (i,11)=mean (stats.Energy);
    data2 (i,12)=mean (stats.Homogeneity);
end
cd ..
data=[data1;data2];
t1= [ones(n1,1);zeros(n2,1)];
t2= [zeros(n1,1);ones(n2,1)];
t=[t1 t2];
%++++++++++++++++++++Tambah Data Latih++++++++++++++++++++++++++++++


% --- Executes on button press in btn_TrainingJST.
function btn_TrainingJST_Callback(hObject, eventdata, handles)
% hObject    handle to btn_TrainingJST (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
%training

%++++++++++++++++++++Traning JST++++++++++++++++++++++++++++++
data=data';

clear net
clear net2
net=newff(minmax(data),[15 2],{'tansig','purelin'});
net.trainParam.epochs = 250;

init(net);
net2 = train(net,data,t'); % training jaringannya
```

```
%++++++++++++++++++++++Traning JST++++++++++++++++++++++++++++++

% --- Executes on button press in btn_TestSegmentasi.
function btn_TestSegmentasi_Callback(hObject, eventdata, handles)
% hObject    handle to btn_TestSegmentasi (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% ukuran (px)
%  gambar =vout (:,:,:,1);

%+++++++++++++++Test Segmentasi+++++++++++++++

 %% ambil contoh gambar yang akan ditest
 a=imread(get(handles.ed_lokasiimg,'string'));
 gambar =a;
 gambar2=rgb2gray(a); % convert to gray mode
 %p=10; % ukuran panjang = 20px
 %l=10; % ukuran lebar = 20px
 p=str2num(get(handles.ed_Panjang,'string'));
 l=str2num(get(handles.ed_Lebar,'string'));



 maxp=size (gambar,1) %320
 maxl=size (gambar,2)%240

 np=maxp/p %4
 nl=maxl/l %3

 seg_gambar = zeros (p,l,3,np*nl);
 size (seg_gambar) % cek segmentasinya

 gbr = zeros(np,nl);
 %proses awal segmentasi
 for i = 1:np
    for j  = 1:nl
        xawal = (i-1)*p+1;
        xakhir= (i)*p;
        yawal= (j-1)*l+1;
```

```
        yakhir=(j)*l;
        nourut= (i-1)*nl+j;
        %% Setiap hasil segment langsung dihitung nilai GLCM nya (stats)
        seg_gambar (:,:,:,nourut) = gambar ( xawal:xakhir, yawal:yakhir, :);
%         gb2 = rgb2gray(gambar ( xawal:xakhir, yawal:yakhir, :));
        gb_r = gambar ( xawal:xakhir, yawal:yakhir, 1);
        gb_g = gambar ( xawal:xakhir, yawal:yakhir, 2);
        gb_b = gambar ( xawal:xakhir, yawal:yakhir, 3);

        xoffset = [0 1; -1 1; -1 0; -1 -1];
        glcm = graycomatrix (gb_r,'offset',xoffset);
        stats = graycoprops (glcm,{'all'});
        % hasil glcm disimpan di variabel datacek
        datacek (i,j,1)=mean (stats.Contrast);
        datacek (i,j,2)=mean (stats.Correlation);
        datacek (i,j,3)=mean (stats.Energy);
        datacek (i,j,4)=mean (stats.Homogeneity);

        glcm = graycomatrix (gb_g,'offset',xoffset);
        stats2 = graycoprops (glcm,{'all'});
        % hasil glcm disimpan di variabel datacek
        datacek (i,j,1)=mean (stats.Contrast);
        datacek (i,j,2)=mean (stats.Correlation);
        datacek (i,j,3)=mean (stats.Energy);
        datacek (i,j,4)=mean (stats.Homogeneity);

        glcm = graycomatrix (gb_b,'offset',xoffset);
        stats3 = graycoprops (glcm,{'all'});
        % hasil glcm disimpan di variabel datacek
        datacek (i,j,1)=mean (stats.Contrast);
        datacek (i,j,2)=mean (stats.Correlation);
        datacek (i,j,3)=mean (stats.Energy);
        datacek (i,j,4)=mean (stats.Homogeneity);
        try
            %% dicoba dimasukkan ke dlam jaringan syaraf tiruan
%             hasil = [mean(stats.Contrast); 0mean(stats.Correlation); mean(stats.Energy);
mean(stats.Homogeneity)];
            hasil = [mean(stats.Contrast); 0; mean(stats.Energy); mean(stats.Homogeneity)];
            hasil = [hasil; mean(stats2.Contrast); 0; mean(stats2.Energy); mean(stats2.Homogeneity)];
            hasil = [hasil; mean(stats3.Contrast); 0; mean(stats3.Energy); mean(stats3.Homogeneity)];
```

```
            y=sim(net2,hasil);
            if (y(1)>=0) % kalau hasilnya positif, maka dianggap jalan (0)
               gambar2(xawal:xakhir, yawal:yakhir)=0;
               gbr(i,j)=0;
            else
               gambar2(xawal:xakhir, yawal:yakhir)=255;
               gbr(i,j)=255;
            end

        catch
        end
      end;
  end;
%++++++++++++++++++++++++++++ Test Segmentasi++++++++++++++++++++++

x_flood = np;
 y_flood = nl/2;

 head=1;
 tail=0;
 clear q;
 q(1).x=x_flood;
 q(1).y=y_flood;

 gambar3=gbr;
 gambar_final=ones(size(gambar2))*255;
 jarak=1;

 while (head>tail)
    tail=tail+1;
    x_flood=q(tail).x;
    y_flood=q(tail).y;

%     disp (q)
%     gambar3 (q(tail).x,q(tail).y)
    %4 ARAH MATA ANGIN
    nc = 0; % pixel yang di cek.
    if (x_flood>=1) && (x_flood<=np) && (y_flood>=1) && (y_flood<=nl)
```

```
    if (gambar3 (x_flood,y_flood) == nc)
        gambar3 (x_flood,y_flood) = 255;

        xawal = (x_flood-1)*p+1;
        xakhir= (x_flood)*p;
        yawal= (y_flood-1)*l+1;
        yakhir=(y_flood)*l;

        gambar_final (xawal:xakhir,yawal:yakhir) = 0;
        head=head+1;
        q(head).x=x_flood+jarak;
        q(head).y=y_flood;

        head=head+1;
        q(head).x=x_flood-jarak;
        q(head).y=y_flood;

        head=head+1;
        q(head).x=x_flood;
        q(head).y=y_flood+jarak;

        head=head+1;
        q(head).x=x_flood;
        q(head).y=y_flood-jarak;
    end
  end
end

figure (3);
imshow (gambar_final)
```

LAMPIRAN B

- Tabel hasil percobaan dengan epoch 250

Percobaan ke	Maks. epoch	Hidden Layer	MSE
1	250	10	0.0589361
2	250	15	0.0589289
3	250	20	0.0589286
4	250	25	0.0589286

- Tabel hasil percobaan dengan epoch 300

Percobaan ke	Maks. epoch	Hidden Layer	MSE
1	300	10	0.0593970
2	300	15	0.0589433
3	300	20	0.0589318
4	300	25	0.0589286

- Tabel hasil percobaan dengan epoch 350

Percobaan ke	Maks. epoch	Hidden Layer	MSE
1	350	10	0.0599490
2	350	15	0.0589421
3	350	20	0.0591605
4	350	25	0.0642792

- Tabel hasil percobaan dengan epoch 400

Percobaan ke	Maks. epoch	Hidden Layer	MSE
1	400	10	0.0589287
2	400	15	0.0593230
3	400	20	0.0592470
4	400	25	0.0632683

- Tabel hasil percobaan dengan epoch 450

Percobaan ke	Maks. epoch	Hidden Layer	MSE
1	450	10	0.0702863
2	450	15	0.0589286
3	450	20	0.0645880
4	450	25	0.0591911

- Tabel hasil percobaan dengan epoch 500

Percobaan ke	Maks. epoch	Hidden Layer	MSE
1	500	10	0.0770541
2	500	15	0.0589325
3	500	20	0.0589325
4	500	25	0.0589286