21.662/TI/H/05

MILIK PERPUSTAKAAN
INSTITUT TEKNOLOGI
SEPULUH NOPEMBER

PERANCANGAN DAN PEMBUATAN PERANGKAT LUNAK KLASIFIKASI TEKSTUR DENGAN NEURAL NETWORK

RSIF
005.1
Eff
p-1
1997

PERPUSTAKAAN
ITS
Tgl. Terima 15-7-2003
Terima Dari H
No. Agenda Prp. 218101

OLEH :

IRFAN EFFENDI
NRP. 2690 100 022

JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
1997

PERANCANGAN DAN PEMBUATAN PERANGKAT LUNAK KLASIFIKASI TEKSTUR DENGAN NEURAL NETWORK

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan
Untuk Memperoleh Gelar Sarjana Teknik Informatika
Pada
Jurusan Teknik Informatika
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember
Surabaya

Mengetahui / Menyetujui

Dosen Pembimbing

Ir. HANDAYANI TJANDRASA, Msc.,PhD.
NIP. 130 532 048

SURABAYA
MARET , 1997

ABSTRAKSI

Dalam tugas akhir ini akan dijelaskan penggunaan jaringan saraf buatan (*Artificial Neural Network*) untuk aplikasi klasifikasi tekstur. Dalam bidang citra digital banyak aplikasi seperti analisa suatu gambar, pemetaan suatu daerah serta pengenalan pola dari suatu gambar memerlukan data tekstur untuk mengenali bentuk karakteristik dari gambar tersebut. Dari hasil analisa karakteristik tekstur pada gambar tersebut dapat dilakukan proses selanjutnya untuk berbagai keperluan. Dalam kasus sederhana dapat kita lihat pada proses citra digital untuk melakukan proses segmentasi dan klasifikasi terhadap objek-objek yang terdapat pada suatu daerah atau gambar.

Metode jaringan saraf yang digunakan untuk menyelesaikan persoalan klasifikasi tekstur adalah **Fungsi Energi Hopfield**, yang merupakan suatu algoritma yang terdapat dalam jaringan saraf Hopfield. Metode ini akan melakukan proses klasifikasi setiap piksel ke dalam objek-objek tertentu dalam beberapa iterasi serta melakukan perhitungan untuk menentukan saat kapan iterasi dinyatakan berhenti. Sebelum dilakukan proses klasifikasi terlebih dahulu dilakukan proses pemodelan tekstur untuk menganalisa karakteristik tekstur dari citra input.

Input dari sistem perangkat lunak ini berupa citra tekstur. Citra tekstur tersebut kemudian dimodelkan dengan menggunakan metode Markov Random Field (MRF). Dari pemodelan citra tekstur tersebut akan didapatkan parameter-parameter citra, dimana parameter-parameter ini nantinya akan digunakan sebagai parameter dalam jaringan saraf.

Algoritma yang digunakan untuk proses klasifikasi setiap piksel ke dalam kelas-kelas tertentu yaitu dengan cara ***Deterministic Relaxation*** yang diimplementasikan ke dalam fungsi energi Hopfield. Proses klasifikasi akan dilakukan terhadap setiap piksel dari citra dalam beberapa kali iterasi sampai dicapai sistem jaringan yang konvergen. Penentuan kondisi konvergen disini yaitu apabila nilai energi pada setiap *stage* tidak bertambah atau tidak mengalami perubahan.

KATA PENGANTAR

Segala puji syukur kami panjatkan kepada Tuhan Yang Maha Esa, karena hanya berkat rahmat dan pertolonganNya akhirnya penulis dapat menyelesaikan Tugas Akhir yang berjudul "Perancangan dan Pembuatan Perangkat Lunak Klasifikasi Tekstur dengan Neural Network".

Tugas Akhir ini disusun untuk memenuhi salah satu persyaratan guna meraih gelar sarjana pada Jurusan Teknik Informatika Fakultas Teknologi Industri Institut Teknologi Sepuluh Nopember Surabaya.

Penulis berharap semoga Tugas Akhir ini dapat memberikan manfaat dan tambahan pengetahuan kepada pembaca khususnya kepada mereka yang tertarik pada masalah Neural Network dan penerapannya untuk menyelesaikan suatu masalah berkaitan dengan klasifikasi dan segmentasi suatu objek.

Pada kesempatan ini pula penulis menyampaikan terima kasih dan penghargaan yang setinggi-tingginya atas bantuan, bimbingan dan dorongan moril kepada :

1. Dr. Ir. Arief Djunaidy selaku Ketua Jurusan Teknik Informatika FTI ITS.
2. Dr. Ir. Drs. Ec. Riyanarto Sarno selaku dosen wali di Teknik Informatika FTI ITS.
3. Ir. Handayani Tjandrasa, Msc., PhD. selaku dosen pembimbing pada tugas akhir ini.

4. Seluruh staf dosen Jurusan Teknik Informatika FTI ITS atas bimbingannya selama masa perkuliahan.
5. Seluruh staf dan karyawan Jurusan Teknik Informatika FTI ITS atas bantuannya dalam bidang administrasi kemahasiswaan.
6. Ayah dan Ibu beserta kakak-kakak tercinta mbak Din, mas Teguh, mas Ma'ruf dan adik-adik tersayang Pepi dan I'in yang semuanya tak pernah berhenti memberikan dukungan moril dan materiil serta bimbingan kepada penulis selama ini.
7. Keluarga Ir. Gandung Wiyadi yang begitu tulus dan penuh perhatian ikut memberikan dukungan moril dan materiil.
8. "Claudyna" semoga engkau tidak akan pernah berhenti memberikan dorongan semangat serta kasih sayang.
9. Rekan Nico Artanto, S.Kom dan Argo Among N, S.Kom yang telah berjasa meminjamkan buku serta masukan-masukannya.
9. Rekan Joseph Judistira, S.Kom yang telah banyak memberikan solusi pada saat-saat sulit serta kesediannya mengirimkan E-Mail secara gratis.
10. Bapak Dwi S, S.Kom yang membantu mengeluarkan buku di RBC serta informasi-informasinya yang 'off the record'.
11. Rekan-rekan angkatan C-06 'seperjuangan' Syaiful, Irwan, Dedy Rahmanto, Lukie, Andi, Ari, Mustakim, Nanik, Mutia, Bahtiar, Dedy Rudianto, Anang, Pramu, Cheppy serta rekan-rekan C-06 nun jauh disana yang telah bersama-sama ikut merasakan manis dan pahitnya selama masa perkuliahan di Informatika ITS.

12. Rekan-rekan sesama TA-wan dan TA-wati yang senasib menunggu hari-hari terakhir di belantara Teknik Informatika ITS.
13. Rekan-rekan mahasiswa Teknik Informatika FTI ITS.
14. Rekan-rekan 'Manyar Indah' yang telah memberikan sumbangan hiburan-hiburan segar meski terkadang mereka menjadi partner yang menjengkelkan, ayo kapan kalian mau menyusul 'kakakmu'.
15. Semua pihak yang telah membantu selama masa perkuliahan dan pengerjaan Tugas Akhir ini yang tidak dapat kami sebut satu persatu.

Penulis sadar bahwa apa yang tertuang dalam buku ini masih jauh dari sempurna, karena bagaimanapun penulis mempunyai keterbatasan-keterbatasan yang layaknya dipunyai manusia. Akhirnya semoga Allah SWT memberikan berkah yang berlimpah kepada semua pihak yang tersebut diatas.

Surabaya, Maret 1997

Penulis

INSTITUT TEKNI
SEPULUH - NOPE

DAFTAR ISI

HALAMAN JUDUL i

LEMBAR PENGESAHAN ii

ABSTRAKSI iii

KATA PENGANTAR iv

DAFTAR ISI vii

DAFTAR GAMBAR xi

DAFTAR TABEL xiii

BAB I PENDAHULUAN 1

1.1 LATAR BELAKANG 1

1.2 TUJUAN 2

1.3 PERMASALAHAN 3

1.4 PEMBATASAN MASALAH 4

1.5 METODOLOGI 4

1.6 SISTEMATIKA PEMBAHASAN 6

BAB II CITRA DIGITAL 8

2.1 MODEL CITRA 8

2.2 SISTEM PENGOLAHAN CITRA DIGITAL 10

2.3 DASAR-DASAR MODEL HUBUNGAN ANTAR PIKSEL 12

2.4 DOMAIN SPASIAL - FREKUENSI 15

2.5 FILTER FREKUENSI RENDAH (LOW-PASS FILTER) 16

2.6 REGION-ORIENTED SEGMENTATION.....17

2.6.1 REGION GROWING BY PIXEL AGGREGATION 18

2.7 TEKSTUR 20

2.7.1 METODE STATISTIK..... 21

2.7.2 METODE SPEKTRAL..... 21

2.7.3 METODE STRUKTURAL..... 22

BAB III JARINGAN SARAF BUATAN..... 24

3.1 DEFINSI DAN KONSEP JARINGAN SARAF BUATAN..... 25

3.2 ARSITEKTUR JARINGAN SARAF 28

3.3 ELEMEN PEMROSES..... 31

3.4 NILAI BOBOT..... 33

3.5 METODE BELAJAR..... 35

3.5.1 METODE SUPERVISED..... 35

3.5.2 METODE UNSUPERVISED..... 36

BAB IV JARINGAN SARAF HOPFIELD..... 38

4.1 JARINGAN HOPFIELD DISKRIT.....39

4.1.1 FUNGSI ENERGI JARINGAN HOPFIELD DISKRIT..... 44

4.2 JARINGAN HOPFIELD KONTINYU..... 45

4.2.1 FUNGSI ENERGI JARINGAN HOPFIELD KONTINYU 48

BAB V PERANCANGAN DAN PEMBUATAN PERANGKAT LUNAK 49

5.1 KEBUTUHAN SISTEM 49

5.2 SISTEM KLASIFIKASI TEKSTUR 50

5.3 GAMBARAN UMUM PERANGKAT LUNAK 52

5.4 PEMODELAN TEKSTUR DENGAN MARKOV RANDOM FIELD (MRF) 54

5.4.1 ESTIMASI PARAMETER GMRF 57

5.4.2 PROSES INTENSITAS DAN LABEL 64

5.5 KLASIFIKASI TEKSTUR 67

5.5.1 FUNGSI ENERGI 68

5.5.2 ALGORITMA DETERMINISTIC RELAXATION 70

5.6 STRUKTUR FILE DATABASE 72

BAB VI UJI COBA DAN EVALUASI PERANGKAT LUNAK 75

6.1 UJI COBA DAN EVALUASI ALGORITMA PEMODELAN TEKSTUR 77

6.2 UJI COBA DAN EVALUASI PROSES INTENSITY 80

6.3 UJI COBA DAN EVALUASI PROSES KLASIFIKASI 82

BAB VII PENUTUP 88

7.1 KESIMPULAN 88

7.2 SARAN 89

DAFTAR PUSTAKA 91

DAFTAR GAMBAR

Hal

Gambar 2.1 Tahapan-tahapan pada sistem pengolahan citra 12

Gambar 2.2 (a) Susunan piksel-piksel pada bidang spasial dua dimensi 14

(b) 8-tetangga dari piksel '2' 14

(c) m-tetangga dari piksel '2' 14

Gambar 2.3 (a) Susunan piksel-piksel dalam array pada citra asli 19

(b) Citra hasil segmentasi dengan T=3 20

(c) Citra hasil segmentasi dengan T=6 20

Gambar 2.4 (a) Bntuk asli dari pola tekstur 23

(b) Bentuk tekstur yang dihasilkan dari aturan $S \rightarrow aS$ 23

(c) Bentuk tekstur yang dihasilkan dari semua aturan 23

Gambar 3.1 Bentuk elemen pemroses (PE) 27

Gambar 3.2 (a) Bentuk arsitektur jaringan single-layer 30

(b) Bentuk arsitektur jaringan multi-layer 30

Gambar 3.3 Struktur elemen pemroses pada jaringan saraf 30

Gambar 4.1 Arsitektur jaringan Hopfield diskrit 40

Gambar 4.2 Grafik dari fungsi $g^{-1}(v)$ dan fungsi integral dari $g^{-1}(v)$ 48

Gambar 5.1 Modul-modul yang terdapat pada perangkat lunak 54

Gambar 5.2 Struktur hirarki dari model GMRF 56

MILIK PERPUS INSTITUT TEK SEPULUH - NOPE

Gambar 5.3 Diagaram aliran data proses estimasi parameter GMRF-1 59

Gambar 5.4 Susunan elemen-elemen matrik Q_s 60

Gambar 5.5 Matrik Q_s pada posisi s(50,80) 61

Gambar 5.6 Diagaram aliran data proses estimasi parameter GMRF-2 63

Gambar 5.7 Window k x k berpusat di posisi s pada Ω_l dari citra 65

Gambar 6.1 Model citra tekstur "tidak terstruktur" 78

Gambar 6.2 Model citra tekstur "terstruktur" 78

Gambar 6.3 Citra output hasil proses klasifikasi dengan 3 kelas 83

Gambar 6.4 Citra output hasil proses klasifikasi dengan 2 kelas 84

Gambar 6.5 Citra output hasil proses klasifikasi dengan 4 kelas 85

DAFTAR TABEL

Hal

Tabel 6.1 Nilai-nilai parameter GMRF dari citra tekstur seperti pada gambar 6.1 79

Tabel 6.2 Nilai-nilai parameter GMRF dari citra tekstur seperti pada gambar 6.2 80

BAB I

PENDAHULUAN

BAB I

PENDAHULUAN

1.1 LATAR BELAKANG

Diantara beberapa aplikasi dalam bidang pengolahan citra digital yang banyak dikembangkan saat ini adalah klasifikasi dan segmentasi dari suatu citra khususnya citra tekstur. Hal ini dikarenakan kebanyakan dari aplikasi-aplikasi yang berkaitan dengan klasifikasi suatu objek, baik objek yang bersifat alami atau objek buatan manusia, mempunyai sifat-sifat yang dapat dikategorikan sebagai citra tekstur. Dengan adanya metode pengklasifikasian tekstur ini paling tidak akan dapat membantu memecahkan beberapa problem yang berkaitan dengan *computer vision*, *remote sensing* dan *image analysis*.

Beberapa macam metode untuk aplikasi klasifikasi tekstur yang belum banyak dipakai saat ini adalah dengan menggunakan metode jaringan saraf buatan. Metode jaringan saraf buatan ini mempunyai beberapa keunggulan dibandingkan dengan metode-metode lainnya, seperti dalam hal kecepatan proses, ketepatan hasil, kompleksitas data citra dan lain sebagainya. Sehingga diharapkan dengan adanya penyelesaian masalah klasifikasi tekstur dengan metode jaringan saraf buatan ini, nantinya akan dapat dikembangkan untuk berbagai macam

aplikasi yang lain, yang ada hubungannya dengan persoalan klasifikasi tekstur ini.

Dalam tugas akhir ini metode jaringan saraf yang dipakai adalah jaringan Hopfield, dimana jaringan saraf Hopfield ini merupakan implementasi dari teknik *Deterministic Relaxation*[1].

Untuk mendapatkan input citra tekstur dapat dilakukan dengan bantuan scanner, mula-mula kita cari suatu gambar suatu tekstur yang kita kehendaki kemudian kita scan sehingga didapatkan hasil berupa file gambar. Selain itu pengambilan input file gambar tekstur juga bisa kita dapatkan dari sotware-sofware yang sudah banyak beredar di pasaran.

Dari file gambar itu kemudian kita proses karakteristik teksturnya pada tiap-tiap gambar, sehingga didapatkan nilai parameter-parameter yang kita butuhkan. Nilai-nilai parameter inilah yang selanjutnya dipakai oleh sistem jaringan saraf buatan untuk proses pengklasifikasiannya.

1.2 TUJUAN

Pada tugas akhir ini dibuat suatu perangkat lunak berdasarkan teknik *Deterministic Relaxation* dengan menggunakan jaringan *Hopfield* yang merupakan salah satu bentuk dari model jaringan saraf buatan, untuk menyelesaikan masalah klasifikasi citra tekstur. Selain itu juga ditambah algoritma-algoritma lainnya untuk mendapatkan nilai parameter-parameter

---

[1] Bart Kosko, **NEURAL NETWORK FOR SIGNAL PROCESSING**, Prentice-Hall, International Inc., New Jersey, 1992, hal. 37.

yang dipakai sebagai variabel di dalam jaringan Hopfield. Diharapkan dari perangkat lunak ini akan dapat memberikan solusi masalah klasifikasi tekstur dengan menggunakan metode jaringan saraf buatan.

1.3 PERMASALAHAN

Permasalahan dalam pembuatan perangkat lunak klasifikasi tekstur ini yang utama adalah untuk mendapatkan solusi yang optimum adalah sangat sulit. Hal ini karena besarnya ukuran dimensi dari file citra tekstur tersebut. Sebagai gambaran, untuk proses klasifikasi tekstur dari suatu citra yang berukuran 256 x 256 ke dalam 4 kelas maka ada $2^{4^{16}}$ kemungkinan konfigurasi label. Agar perangkat lunak dapat mengklasifikasikan citra kedalam tekstur-tekstur tertentu maka sebelumnya harus diketahui terlebih dahulu jumlah tekstur yang terdapat di dalam citra tersebut, serta tipe kelas teksturnya. Dan tiap-tiap tekstur juga harus diproses terlebih dahulu untuk mendapatkan nilai-nilai parameternya. Adanya nilai-nilai parameter tertentu yang hanya tepat dilakukan dengan cara *trial and error* juga merupakan salah satu masalah dalam pembuatan perangkat lunak ini. Hambatan lainnya adalah untuk mencapai kondisi tertentu yang dikehendaki, sebagai syarat bahwa proses pengklasifikasian sudah selesai diperlukan beberapa kali iterasi, sehingga memerlukan waktu komputasi yang agak lama.

1.4 PEMBATASAN MASALAH

Dalam rancangan dan pembuatan perangkat lunak untuk klasifikasi tekstur dengan metode jaringan saraf buatan ini, batasan-batasan permasalahannya yaitu :

- File gambar yang akan diklasifikasikan, maksimum berukuran 256 x 256 piksel, dalam bentuk *gray-level* 8 bit.
- Maksimum jumlah tipe tekstur dari citra input adalah 10 macam.
- Input dari file gambar hanya terbatas pada file yang berbentuk .BMP, yang merupakan bentuk standar dari file citra windows.
- File gambar yang akan diklasifikasikan hanya terbatas pada file-file gambar yang tipe teksturnya mempunyai nilai parameter yang tidak jauh menyimpang dari nilai standard.
- Citra tekstur yang akan diproses hanya terbatas pada citra tekstur yang sifat teksturnya periodik (berulang) baik yang terstruktur maupun yang tidak terstruktur.

1.5 METODOLOGI

Metodologi yang digunakan dalam pembuatan Tugas Akhir ini adalah sebagai berikut :

- Pemilihan suatu masalah, yaitu pengembangan perangkat lunak klasifikasi tekstur dengan menggunakan metode jaringan saraf buatan.

- Studi literatur dari berbagai buku dan jurnal tentang jaringan saraf buatan.
- Merumuskan masalah agar menjadi jelas bagian-bagian mana yang harus diselesaikan serta batasan-batasan masalahnya.
- Merumuskan hipotesa untuk menyelesaikan masalahnya serta memilih metode atau algoritma dari jaringan saraf yang akan digunakan untuk solusinya.
- Perancangan dan pembuatan sistem, termasuk juga rancangan struktur data dan algoritma program.
- Pembuatan perangkat lunak klasifikasi tekstur.
- Pengujian perangkat lunak dengan melakukan percobaan pada berbagai macam bentuk tekstur dan nilai-nilai variabel yang berbeda dan kemudian membandingkan hasilnya.
- Melakukan perbaikan-perbaikan untuk mendapatkan hasil yang paling optimal sesuai dengan hasil uji coba yang telah dilakukan sebelumnya.
- Mengambil kesimpulan berdasarkan analisa.
- Membuat laporan.

1.6 SISTEMATIKA PEMBAHASAN

Dalam Tugas Akhir ini sistematika untuk pembahasannya adalah sebagai berikut :

Bab I, **Pendahuluan**. Memuat pendahuluan yang menjelaskan latar belakang masalah, tujuan, batasan masalah serta metodologi dalam pembuatan perangkat lunak ini.

Bab II, **Citra Digital**. Memuat tentang konsep-konsep dan teori-teori dasar mengenai pengolahan citra baik secara langsung maupun tidak langsung menunjang perancangan perangkat lunak ini.

Bab III, **Jaringan Saraf Buatan**. Membahas secara garis besar sistem jaringan saraf buatan termasuk komponen-komponen penyusun dan cara kerja jaringan saraf buatan.

Bab IV, **Jaringan Saraf Hopfield**. Membahas tentang jaringan saraf Hopfield, yang dipakai dalam pembuatan perangkat lunak ini.

Bab V, **Perancangan dan Pembuatan Perangkat Lunak**. Membahas tentang rancangan sistem perangkat lunak yang dibuat dengan berdasarkan pendekatan-pendekatan metode MRF untuk model citranya serta fungsi energi Hopfield untuk proses klasifikasi.

MILIK PERPUSTAKAAN
INSTITUT TEKNOLOGI
SEPULUH – NOPEMBER

Bab VI, **Uji Coba dan Analisa**. Melaporkan hasil uji coba sistem perangkat lunak yang telah dibuat serta mengevaluasi kehandalan sistem.

Bab VII, **Kesimpulan dan Saran**. Merupakan uraian kesimpulan dari bab-bab sebelumnya, serta beberapa saran untuk pengembangan program selanjutnya.

BAB II

CITRA DIGITAL

BAB II

CITRA DIGITAL

Pada bab II ini akan kami jelaskan beberapa hal penting yang berhubungan dengan citra digital, tentunya dasar-dasar mengenai citra digital yang berkaitan dengan tugas akhir ini. Diantaranya adalah mengenai teknik domain, dasar-dasar hubungan piksel, teknik dasar penggabungan region-region pada suatu citra, pemodelan suatu citra serta proses pengolahannya, dan deskripsi dari salah model citra yaitu tekstur.

2.1 MODEL CITRA

Citra merupakan hasil output dari suatu sistem perekaman data yang dapat bersifat optik berupa foto, bersifat analog berupa sinyal-sinyal video seperti pada gambar monitor televisi atau bersifat digital yang merupakan hasil proses pengolahan mesin komputer. Pada citra digital dapat langsung disimpan ke dalam pita magnetik.

Citra (image) dirumuskan sebagai suatu fungsi intensitas cahaya dua dimensi, yakni $f(x,y)$, dimana f menyatakan nilai intensitas (kecerahan) dari citra pada kooordinat (x,y) dalam ruang dua dimensi. Nilai fungsi $f(x,y)$ adalah terbatas dan tidak sama dengan 0 atau dapat ditulis :

$$0 < f(x,y) < \infty \qquad (2.1\text{-}1)$$

Pada citra digital, fungsi f(x,y) akan didiskritkan baik koordinat spasialnya maupun nilai intensitasnya. Sehingga dapat digambarkan, citra digital sebagai matriks yang posisi baris dan kolomnya, (x,y) menunjukkan sebuah titik dalam suatu citra dan nilai elemen matriksnya menunjukkan nilai intensitas pada titik yang bersangkutan.

Dalam kehidupan sehari-hari kita dapat merasakan, adanya sinar-sinar yang dipancarkan dari suatu objek. Berkaitan dengan nilai intensitas, komponen mendasar yang terdapat pada fungsi $f(x,y)$ dapat dikelompokkan sebagai 2 (dua) bagian. Komponen pertama yaitu sejumlah sinar yang tampak dan berasal dari peristiwa-peristiwa alam sedangkan komponen kedua adalah sinar yang dipancarkan oleh suatu objek. Kedua komponen itu disebut *illumination* dan *reflectance* komponen, yang dinyatakan sebagai $i(x,y)$ dan $r(x,y)$. Kombinasi dari fungsi $i(x,y)$ dan $r(x,y)$ akan menghasilkan fungsi $f(x,y)$, yaitu :

$$f(x,y) = i(x,y)r(x,y) \qquad (2.1\text{-}2)$$

dimana,

$$0 < i(x,y) < \infty \text{ dan,} \qquad (2.1\text{-}3)$$

$$0 < r(x,y) < 1 \qquad (2.1\text{-}4)$$

Persamaan (2.1-4) menandakan bahwa nilai kerefleksian dibatasi oleh nilai 0 (*total absorption*) dan nilai 1 (*total reflectance*). Fungsi $i(x,y)$ ditentukan oleh sumber/asal sinar sedangkan fungsi $r(x,y)$ ditentukan oleh karakteristik dari objek.

Pada intensitas dari suatu citra *monochrome* atau hitam putih, fungsi f pada koordinat (x,y) menyatakan tingkat keabuan dari hitam ke putih atau disebut *gray level* (ℓ), dengan nilai parameter ℓ terletak pada range,

$$L_{min} \leq \ell \leq L_{max} \qquad (2.1\text{-}5)$$

dengan syarat bahwa nilai L_{min} adalah positif dan L_{max}, terbatas (*finite*). Nilai L_{min} dan L_{max} dapat dirumuskan, yakni,

$$L_{min} = i_{min}\, r_{min} \quad \text{dan} \quad L_{max} = i_{max}\, r_{max} \qquad (2.1\text{-}6)$$

Interval (L_{min}, L_{max}) disebut *gray scale*[2].

2.2 SISTEM PENGOLAHAN CITRA DIGITAL

Proses pengolahan suatu citra kedalam bentuk citra digital dari spatial koordinat (x,y) disebut *image sampling*, sedang proses pengolahan citra digital kedalam nilai intensitasnya disebut *gray-level quantization*.

Persamaan (2.2-1) menunjukkan fungsi f(x,y) pada suatu citra berukuran N x N arary, dimana setiap elemen di dalam array mempunyai bentuk nilai diskrit.

$$f(x,y) \approx \begin{bmatrix} f(0,0) & f(0,1) & \cdots\cdots & f(0,N-1) \\ f(1,0) & f(1,1) & \cdots\cdots & f(1,N-1) \\ f(2,0) & f(2,1) & \cdots\cdots & f(2,N-1) \\ f(3,0) & f(3,1) & \cdots\cdots & f(N-1,N-1) \end{bmatrix} \qquad (2.2\text{-}1$$

---

[2] Gonzale, Rafael C, **DIGITAL IMAGE PROCESSING**, Addison-Wesley Publishing Company, 1987, hal. 22.

Bagian sisi kanan dari persamaan diatas disebut sebagai *digital image*, sementara setiap elemen yang terdapat di dalam array disebut *piksel* (*image element*).

Pada proses pengolahan citra digital diperlukan informasi tentang besarnya ukuran suatu citra serta nilai maksimum dari gray level, yang dapat ditunjukkan dalam bentuk,

$$N = 2^n \quad \text{dan} \tag{2.2-2}$$

$$G = 2^m \tag{2.2-3}$$

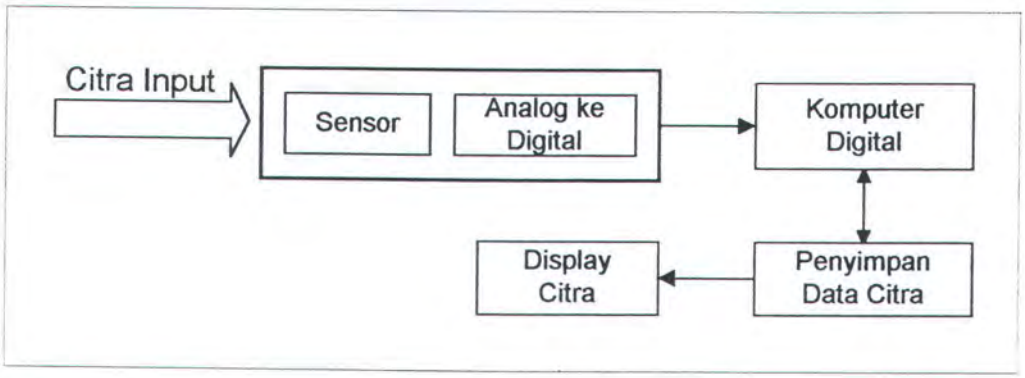
dimana G menunjukkan nilai maksimum gray level, terletak pada range 0-L, N adalah ukuran dimensi citra yang berbentuk bujursangkar N-kali-N, ukuran ini yang sering disebut *resolusi piksel*, dan m, adalah panjang bit yang digunakan untuk menyimpan nilai intensitas setiap piksel. Jika parameter b menyatakan jumlah bit yang diperlukan untuk proses pengolahan digital dari suatu citra, maka b dapat dinyatakan sebagai :

$$b = N \times N \times m \tag{2.2-4}$$

Sebagai contoh, jika citra berukuran 128 x 128 dengan 64 gray level maka untuk melakukan proses digital diperlukan alokasi memori sebesar 98,304 bit. Besar-kecilnya nilai gray level yang digunakan akan menentukan kualitas gambar yang dihasilkan.

Sistem penangkapan citra digital sendiri terdiri dari tiga komponen dasar, yaitu sensor citra yang bekerja sebagai pengukur intensitas cahaya, perangkat penjelajah yang bertugas merekam hasil pengukuran intensitas

pada seluruh bagian citra, dan pengubah analog ke digital yang berfungsi untuk mengubah harga kontinu menjadi harga diskrit sehingga dapat diproses mesin komputer. Proses penangkapan citra dapat dilihat pada gambar 2.1.



Gambar 2.1 Tahapan-tahapan pada sistem sistem pengolahan citra

2.3 DASAR-DASAR MODEL HUBUNGAN ANTAR PIKSEL

Dalam masalah pengolahan citra digital, hubungan antar piksel merupakan hal yang sangat penting. Sebuah piksel p pada koordinat (x,y) mempunyai 4 tetangga horisontal dan vertikal, yang koordinat-koordinatnya adalah sebagai berikut :

(x+1,y), (x-1,y), (x,y+1), dan (x,y-1)

Kumpulan dari piksel-piksel diatas yang disebut *4-neighbors of p* dapat dinyatakan sebagai $N_4(p)$, kecuali jika p(x,y) posisinya terletak di garis batas gambar, sehingga jumlah piksel tetangga tidak terdiri dari 4 (empat) tetangga. Selain 4 tetangga diatas, p juga mempunyai 4 tetangga diagonal, yaitu :

(x+1,y+1), (x+1,y-1), (x-1,y+1), (x-1,y-1)

Piksel-piksel diatas dinyatakan sebagai $N_D(p)$. Gabungan dari $N_4(p)$ dan $N_D(p)$ didefinisikan sebagai *8-neighbors of p* dan dinyatakan sebagai $N_8(p)$.

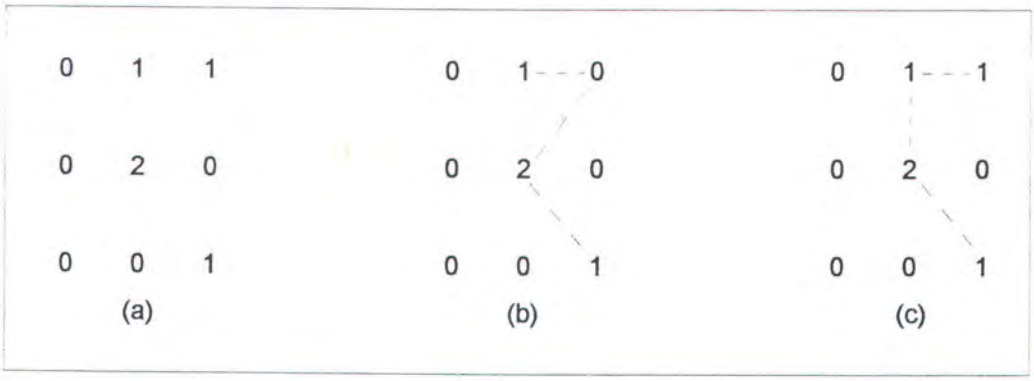
Hubungan antar piksel merupakan suatu konsep yang sangat penting, yang digunakan untuk mendefinisikan batas-batas dari suatu objek serta bagian-bagian daerah kecil dari suatu gambar. Sebagai pertimbangan apakah dua piksel dihubungkan atau tidak diperlukan beberapa kriteria. Diantaranya adalah apakah kedua piksel tersebut mempunyai prinsip kedekatan yang sesuai dengan konsep yang telah ditentukan, seperti konsep 4-neighbors atau 8-neighbors. Kemudian yang kedua, apakah kedua piksel itu mempunyai nilai gray level yang sesuai dengan kriteria yang kita inginkan. Sebagai contoh, jika dua piksel mempunyai nilai masing-masing 0 dan 1 dan keduanya merupakan bagian dari 4-neighbors, maka dinyatakan bahwa kedua piksel tersebut tidak ada hubungan, hal ini karena keduanya mempunyai nilai yang berbeda.

Jika didefinisikan bahwa hubungan antar piksel dapat terjadi apabila memenuhi kriteria yang ditentukan, yakni mempunyai nilai gray level dari salah satu himpunan V{59,60,61), maka ada 3 (tiga) tipe hubungan :

1. *4-conectivity*, yaitu dua piksel p dan q yang masing-masing mempunyai nilai gray level dari V, akan berhubungan, jika q adalah anggota dari $N_4(p)$.

2. *8-conectivity*, yaitu dua piksel p dan q yang masing-masing mempunyai nilai gray level dari V, akan berhubungan jika q adalah anggota dari $N_8(p)$.
3. *m-conectivity*[3], yaitu dua piksel p dan q yang masing-masing mempunyai nilai gray level dari V, akan berhubungan jika,
   (i) q adalah anggota dari $N_4(p)$ atau
   (ii) q anggota dari $N_D(p)$ dan kumpulan dari $N_4(p) \cap N_4(q)$ adalah kosong.

   Hubungan antar piksel tipe ini dapat dilihat dalam gambar 2.2(b) dan 2.2(c).



Gambar 2.2 (a) Susunan piksel-piksel pada bidang spasial dua dimensi
(b) 8 - tetangga dari piksel '2'
(c) m - tetangga dari piksel '2'

INSTITUT TEKN
SEPULUH - NOP

[3] Ibid, hal. 30.

2.4 DOMAIN SPASIAL - FREKUENSI

Dalam sistem pengolahan citra digital, metode pengambilan data yang akan diolah dapat dikategorikan dalam dua macam, yaitu metode *domain spasial* dan *domain frekuensi.*

Domain spasial menunjuk pada sekumpulan piksel yang membentuk suatu citra, dan metode domain spasial merupakan suatu prosedur yang dioperasikan secara langsung pada piksel. Fungsi pengolahan citra pada domain spasial adalah sebagai berikut

$$g(x,y) = T[f(x,y)] \qquad (2.4\text{-}1)$$

dimana $f(x,y)$ adalah citra input, $g(x,y)$ adalah citra hasil olahan, serta T adalah operator pada fungsi f, yang didefinisikan melalui nilai tetangga dari (x,y).

Pendekatan yang digunakan untuk mendefinisikan tetangga dari (x,y) dapat berbentuk daerah bujursangkar atau persegipanjang yang berpusat di koordinat (x,y), seperti yang telah dijelaskan pada bagian sebelumnya. Selanjutnya daerah ini disebut dengan jendela. Jendela ini digerakkan sepanjang daerah citra mulai dari posisi kiri atas sampai dengan kanan bawah untuk menghasilkan nilai g pada tiap lokasi piksel.

Jika terdapat jendela dengan koefisien w_1, w_2, ..., w_g, dengan *8-neighbors* pada titik (x,y) maka operasi yang akan terjadi adalah :

$$T[f(x,y)] = w_1 f(x-1,y-1) + w_2 f(x-1,y) + w_3 f(x-1,y+1) + w_4 f(x,y-1 w_5 f(x,y) +$$

$$w_6 f(x,y+1) + w_7 f(x+1,y-1) + w_8 f(x+1,y) + w_9 f(x+1,y+1) \qquad (2.4\text{-}2)$$

Pada domain frekuensi dinyatakan dalam bentuk transformasi *Fourier* dari citra yang akan diolah. Dasar dari teknik domain frekuensi adalah sebagai berikut :

$$g(x,y) = h(x,y)*f(x,y) \qquad (2.4\text{-}3)$$

atau dalam bentuk teorema konvolusi bisa dianalogikan dengan

$$G(u,v) = H(u,v)*F(u,v) \qquad (2.4\text{-}4)$$

dimana G, H dan F adalah transformasi Fourier dari g, h dan f

2.5 FILTER FREKUENSI RENDAH (LOW-PASS FILTER)

Pada proses pengambilan gambar melalui scanner, mutu gambar yang dihasilkan tidak akan sebaik gambar aslinya. Untuk itu perlu dipakai beberapa metode yang dapat memperbaiki mutu gambar tersebut, salah satunya adalah penghilangan *noise*, yang akan kami jelaskan lebih lanjut.

Proses filtering citra yang bersifat spatial dapat digunakan dengan operator jendela yang elemennya terdiri dari faktor pembobotan yang menentukan nilai intensitas suatu piksel berdasarkan nilai intensitas piksel-piksel tetangganya.

Filter frekuensi rendah mempunyai karakteristik menyalurkan dan memperkeras bagian frekuensi rendah. Proses filtering suatu citra dengan filter frekuensi rendah mempunyai efek pemerataan tingkat keabuan, sehingga gambar yang diperoleh akan tampak agak kabur kontrasnya.

Noise dalam suatu citra digital memiliki spektrum frekuensi spatial yang lebih tinggi dibanding komponen citra normal. Hal ini disebabkan

karena keterpisahaannya. Oleh karena itu dengan filter frekuensi rendah, noise yang ada akan dapat dikurangi. Citra output Q, M dimensi (MxM), dibentuk melalui konvolusi diskrit array citra input F, N x N, dengan konvolusi array H, L x L, dengan persamaan

$$Q(m_1,m_2) = \sum_{n1}\sum_{n2} F(n_1 n_2) H(m_1 - n_{1+1}, m_2 - n_{2+1}) \qquad (2.5\text{-}1)$$

H merupakan filter frekuensi rendah dengan komponen-komponen seperti dibawah ini,

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \qquad \frac{1}{10}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \qquad \frac{1}{16}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Setiap matriks filter diatas dinormalisasi agar tidak terjadi bias. Karena pada citra yang dipakai dalam pembuatan perangkat lunak ini hanya memiliki dua warna, yaitu hitam dan putih, maka matriks filter yang digunakan berbentuk

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

2.6 REGION-ORIENTED SEGMENTATION

Segmentasi adalah proses pembagian suatu gambar ke dalam region-region berdasarkan klasifikasi nilai-nilai tertentu. Nilai-nilai ini ditentukan melalui suatu nilai *threshold* yang didistribusikan ke semua piksel dalam citra, seperti nilai intensitas, atau warna.

Bila R menyatakan total region dalam suatu citra, maka dalam proses segmentasi, R akan dibagi ke dalam subregion-subregion sebanyak n yaitu R_1, R_2, ... R_n.

Pada proses segmentasi akan ada beberapa kondisi, yakni :

(a) $\bigcup_{i=1}^{n} R_i = R$

(b) R_i akan dihubungkan ke suatu region i = 1,2, ... ,n.

(c) $R_i \cap R_j = 0$ untuk semua i dan j, $i \neq j$.

(d) $P(R_i)$ = True, untuk i = 1,2, ...,n.

(e) $P(R_i \cup R_j)$ = False, untuk $i \neq j$

Kondisi (a) menandakan bahwa proses segmentasi harus lengkap, yaitu semua piksel harus didefinisikan ke salah satu region. Pada (b), anggota semua piksel dari suatu region harus dihubungkan ke dalam region tersebut. Pada kondisi (c), region-region yang berbeda harus dipisahkan satu sama lain. Sedang kondisi (d), menyatakan bahwa semua piksel yang terdapat dalam region R_i mempunyai nilai intensitas yang sama. Dan kondisi terakhir (e), menyatakan bahwa anggota dari region R_i dan R_j mempunyai nilai intensitas yang berbeda.

2.6.1 Region Growing by Pixel Aggregation

Region Growing adalah suatu prosedur yang menggabungkan piksel-piksel atau subregion ke dalam region yang lebih besar. Dalam

prosedur ini dilakukan pendekatan dengan cara *pixel aggregation*[4], yaitu memilih beberapa bibit (*seed*) piksel yang nilai (gray level, color, tekstur) piksel-piksel tetangganya adalah 'berdekatan'.

Sebagai contoh, dapat dilihat pada gambar 2.3. Kita pilih piksel pada koordinat (3,2) dan (3,4) sebagai seed, masing-masing ke dalam region R_1 dan R_2. Selanjutnya untuk menentukan piksel lainnya ke dalam region tertentu (R_1 atau R_2), kita gunakan nilai obsolut perbedaan antara gray level piksel dengan nilai seed yang ada sebagai parameternya. Nilai absolut ini kemudian kita bandingkan nilai threshold, lalu kita pilih perbandingan yang terkecil. Pada gambar (b) dapat dilihat hasil proses segmentasi dengan nilai threshold T=3. Sedang pada nilai threshold T=6 hasil segmentasi dapat dilihat di gambar (c).

	1	2	3	4	5
1	0	0	5	6	7
2	1	1	5	8	7
3	0	1	6	7	7
4	0	0	7	6	6
5	0	1	5	6	5

(a)

[4] Ibid, hal. 369.

	1	2	3	4	5
1	a	a	b	b	b
2	a	a	b	b	b
3	a	a	b	b	b
4	a	a	b	b	b
5	a	a	b	b	b

(b)

	1	2	3	4	5
1	a	a	a	a	a
2	a	a	a	a	a
3	a	a	a	a	a
4	a	a	a	a	a
5	a	a	a	a	a

(c)

Gambar 2.3 (a) Susunan piksel-piksel dalam array pada citra asli

(b) Citra hasil segmentasi dengan T=3

(c) Citra hasil segmentasi dengan T=6

2.7 TEKSTUR

Dari beberapa karakteristik citra, isi tekstur merupakan salah satu karakteristik citra yang mempunyai bentuk yang khas. Karakteristik dari isi tekstur inilah yang digunakan untuk menggambarkan suatu daerah pada citra. Meskipun secara formal tidak ada definisi khusus tentang tekstur, tetapi karakteristik tekstur dapat digambarkan dari sejumlah properti-propertinya seperti kehalusan (*smoothness*), kekasaran (*coarseness*) dan sifat keberaturan (*regularity*).

Pada proses pengolahan citra, metode yang digunakan untuk menggambarkan suatu tekstur, secara prinsip dibagi menjadi tiga[5], yaitu pertama, secara *statistical*, kedua, secara *spectral* dan yang ketiga, *structural*. Dalam tugas akhir ini digunakan prinsip pendekatan dengan metode structural.

2.7.1 Metode Statistik

Pada metode statistik, penggambaran suatu tekstur dapat dilihat dari beberapa karakteristik seperti kehalusan, kekasaraan dan butiran-butiran. Metode ini ditekankan pada tingkat gray level piksel dari suatu citra dengan menggunakan analisa tekstur untuk proses pendeteksian citra. Salah satu pendekatan sederhana untuk menggambarkan tekstur adalah dengan menggunakan momen-momen dari *histogram* gray level pada sebuah daerah/citra.

2.7.2 Metode Spektral

Dalam metode ini karakteristik tekstur didasarkan pada properti-properti dari spektrum *Fourier*, yang digunakan untuk mendeteksi bentuk/pola periodik secara umum pada citra, caranya yaitu dengan mengidentifikasi *high-energy* dalam spektrum Fourier. Ada tiga *feature* dari spektrum Fourier yang dapat digunakan untuk analisa tekstur yaitu, pertama, puncak-puncak spektrum yang terlihat menonjol yang merupakan

---

[5] Ibid, hal. 414.

penggambaran dari pola-pola tekstur, kedua, lokasi dari puncak-puncak spektrum yang merupakan titik ruang dari pola-pola tadi, dan ketiga, dengan mengeliminasi semua komponen yang mempunyai pola secara periodik melalui suatu proses filtering, maka elemen-elemen pada citra yang bersifat non-periodik akan masih tetap ada dalam citra. Dari sisa-sisa elemen non-periodik yang masih ada dalam citra ini, selanjutnya dengan metode statistik akan dapat menggambarkan bentuk suatu tekstur.

2.7.3 Metode Struktural

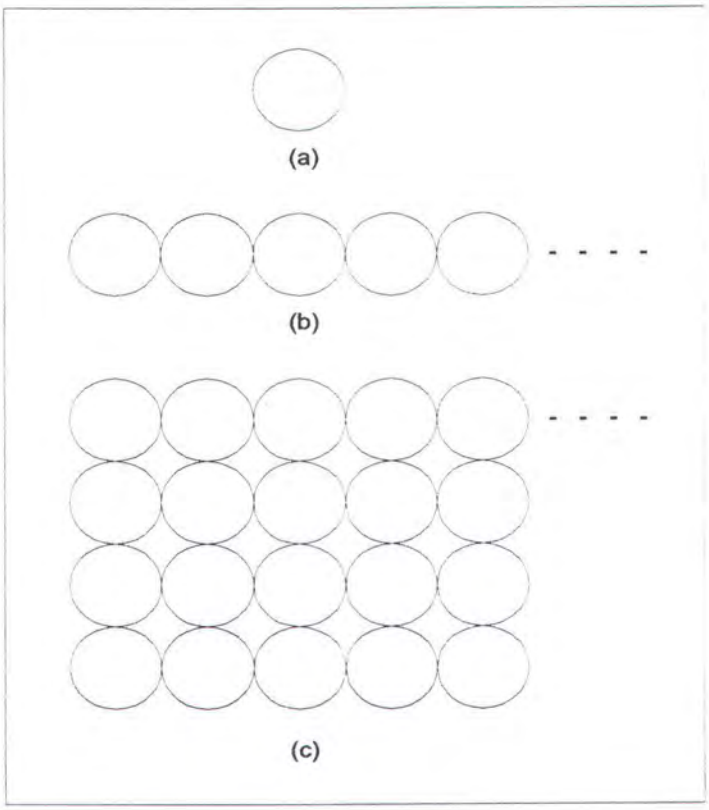
Konsep pada metode ini ditekankan pada karakteristik dari tekstur yang merupakan perulangan pola dari elemen-elemen yang terdapat pada daerah citra. Gambar 2.4 menggambarkan suatu bentuk perulangan pola elemen dari citra yang dinyatakan dalam beberapa aturan. Pada gambar dimisalkan kita mempunyai aturan $S \rightarrow aS$ yang berarti bentuk S dapat dinyatakan sebagai aS. Dan 'a' menyatakan sebuah perulangan dan diartikan sebagai perulangan ke samping kanan '*circles to the right*' ke dalam string 'aaa', dan dimisalkan bentuk aturan $S \rightarrow aS$ dapat di-generate ke dalam pola-pola tekstur suatu citra. Maka dengan menambahkan beberapa aturan lagi akan didapatkan bentuk seperti dalam gambar 2.4(c).[6]

Dari beberapa aturan sebelumnya, jika kita tambahkan beberapa aturan $S \rightarrow bS$, $A \rightarrow cA$, $A \rightarrow c$, $S \rightarrow bS$, $S \rightarrow a$, dimana 'b'

[6] Ibid, hal. 417.

SEPULUH - NOPEMB

menyatakan perulangan ke arah bawah '*circle down*', 'c', menyatakan perulangan ke arah sebelah kiri '*circle to the left*', maka dari semua aturan yang ada kita dapat men-generate ke dalam bentuk string '*aaabccbaa*', yang dapat dinyatakan sebagai matriks perulangan berukuran 3x3. Gambar 2.4(c) merupakan pola-pola tekstur yang dikembangkan dengan aturan seperti diatas.



Gambar 2.4

(a) Bentuk asli dari pola tekstur.

(b) Bentuk tekstur yang dihasilkan dari aturan S $\rightarrow$ aS

(c) Bentuk tekstur yang dihasilkan dari penggunaan semua aturan seperti yang dijelaskan diatas.

BAB III

JARINGAN SARAF BUATAN

BAB III

JARINGAN SARAF BUATAN

Jaringan saraf buatan merupakan salah satu cabang bidang ilmu terbaru dari Kecerdasan Buatan, yang pada saat ini menjadi salah satu bidang ilmu menarik yang banyak menjadi objek penelitian. Hal ini dikarenakan banyak aplikasi-aplikasi yang ternyata dapat diselesaikan dengan menggunakan jaringan saraf buatan. Selain itu metode jaringan saraf ini mempunyai kelebihan-kelebihan dibandingkan dengan metode/algoritma yang lain, terutama dalam hal kecepatan waktu komputasi.

Dalam kehidupan sehari-hari sering kita menghadapi persoalan-persoalan, seperti pengenalan suatu objek, pemilihan jalur terpendek pada suatu rute, pendeteksian suatu gambar dan lain sebagainya, kesemuanya itu bisa kita selesaikan dengan memodelkan masalah tersebut ke dalam sistem jaringan saraf buatan, yang kemudian dapat kita selesaikan dengan menggunakan salah satu algoritma jaringan saraf buatan.

Kelebihan utama pada jaringan saraf buatan ini adalah mampu mengolah informasi dalam jumlah yang cukup besar dibandingkan dengan jaringan saraf manusia, serta proses perhitungan yang relatif lebih cepat. Banyak aplikasi-aplikasi yang sekarang bisa diselesaikan dengan jaringan

saraf buatan seperti pengenalan suara, pengenalan pola, sistem kontrol, diagnosa suatu penyakit dalam bidang kodekteran, segmentasi dan klasifikasi citra.

3.1 DEFINISI DAN KONSEP JARINGAN SARAF BUATAN

Jaringan saraf buatan adalah suatu sistem pengolahan informasi yang cara kerjanya menirukan cara kerja jaringan saraf manusia[7]. Tetapi meskipun demikian tidak semua proses dan fungsi pada jaringan saraf manusia dapat diterapkan ke dalam jaringan saraf buatan. Salah satu fungsi dari jaringan saraf buatan yang diadopsi dari jaringan saraf manusia adalah kemampuannya untuk mempelajari hal-hal yang baru dikenal, kemudian mengingat hal-hal yang baru dikenal dan dipelajari tadi, serta memberikan respon terhadap suatu rangsangan. Sifat-sifat neuron serta hubungan antar neuron menjadi dasar dari pembentukan jaringan saraf buatan.

Jaringan saraf buatan tersusun atas beberapa elemen pemroses (processing element atau PE) yaitu neuron, unit, sel atau node yang saling terhubung dalam bentuk *directed graph* melalui jalur sinyal searah yang disebut dengan koneksi. Setiap PE mempunyai satu koneksi yang terhubung ke sinyal output yang bercabang ke beberapa koneksi yang lain, dimana setiap koneksi membawa sinyal yang sama. Pengolahan

---

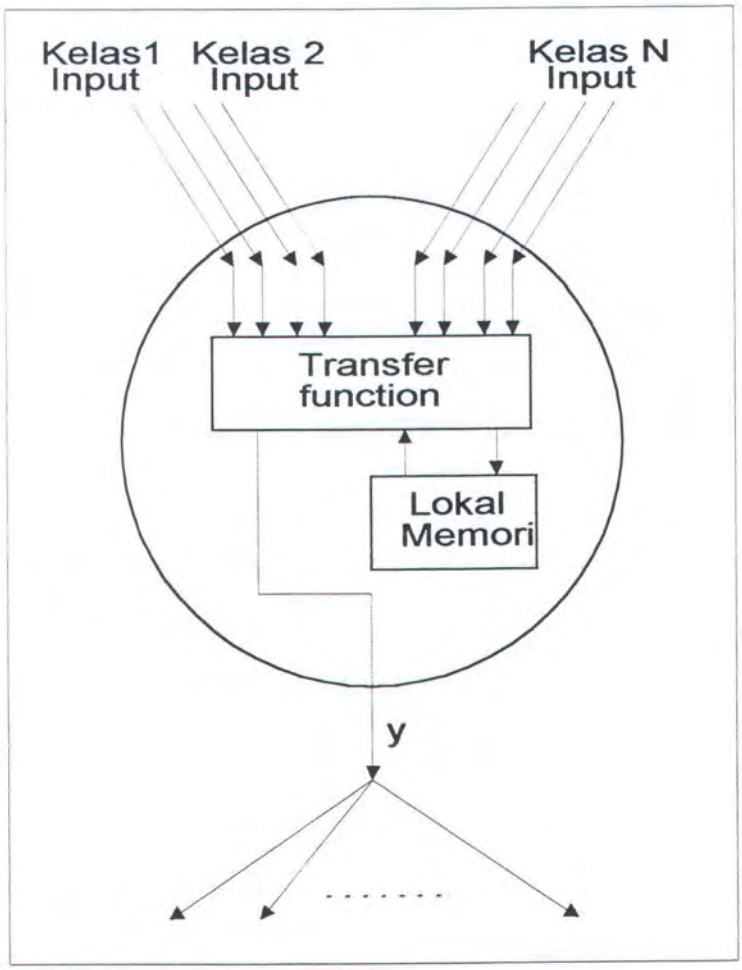
[7] Laurene Fausett, **FUNDAMENTAL OF NEURAL NETWORK**, Prentice-Hall International, Inc., New Jersey, 1994, hal. 3.

informasi yang terjadi pada setiap PE bergantung pada sinyal input di dalam PE dan nilai yang tersimpan di memori lokal setiap PE.

Directed graph yang telah disebut diatas adalah objek geometris yang terdiri atas sekumpulan titik (node) yang dihubungkan oleh sekumpulan garis terarah. Dalam jaringan saraf, struktur pengolahan informasi akan mengikuti bentuk graf terarah dengan beberapa definisi, yaitu :

1. Node pada graf disebut dengan elemen pemroses (PE).
2. *Link* pada graf disebut dengan koneksi. Setiap koneksi berfungsi sebagai jalur sinyal konduksi searah.
3. Setiap PE dapat menerima sejumlah input.
4. Setiap elemen pemroses dapat memiliki beberapa output dimana setiap output mengirimkan sinyal yang sama.
5. Setiap elemen pemroses memiliki memori lokal.
6. Setiap elemen pemroses memiliki fungsi transfer (*transfer function*) yang dapat menggunakan dan mengubah isi memori lokal, memakai sinyal output dari PE. Dengan kata lain, masukan dari fungsi transfer yang diperbolehkan adalah nilai yang tersimpan pada memori lokal dan sinyal masukan PE pada waktu perhitungan fungsi transfer dilakukan. Keluaran fungsi transfer yang diperbolehkan adalah nilai yang akan disimpan dalam memori lokal dan sinyal output dari PE. Fungsi transfer akan beroperasi secara episodik, sehingga ada suatu masukan

yang disebut '*aktifkan*' yang akan menyebabkan fungsi transfer beroperasi dengan perhitungan yang melibatkan sinyal masukan pada waktu itu dan nilai pada memori lokal dan memproduksi sinyal output yang telah diperbaharui serta dapat juga mengubah isi memori lokal. Sinyal 'aktifkan' tersebut datang dari hubungan antara PE penjadwal yang merupakan bagian dari keseluruhan jaringan. PE dengan fungsi transfer yang kontinyu selalu beroperasi.



Gambar 3.1 Bentuk Elemen Pemroses (PE)

7. Sinyal input dari luar sistem saraf buatan yang menuju sistem tersebut datang dari hubungan-hubungan yang berasal dari dunia luar sistem. Sinyal output dari sistem ke dunia luar sistem merupakan hubungan-hubungan yang meninggalkan sistem.

3.2 ARSITEKTUR JARINGAN SARAF

Suatu jaringan saraf minimal tersusun atas *input layer* dan *output layer*. Dalam beberapa tipe jaringan diantara input layer dan output layer terdapat *hidden layer*. Hal ini berarti bahwa semua neuron pada input layer akan terhubung ke semua neuron dalam hidden layer yang selanjutnya setiap unit pada hidden layer nantinya akan dihubungkan ke semua neuron di output layer. Pada setiap layer biasanya neuron mempunyai fungsi keaktifan serta pola hubungan ke neuron lain yang sama. Pada kebanyakan sistem jaringan saraf, ada dua kemungkinan hubungan antar neuron yang terletak pada sebuah layer, yaitu satu sama lain akan terhubung atau sama sekali tidak ada hubungan antar neuron.

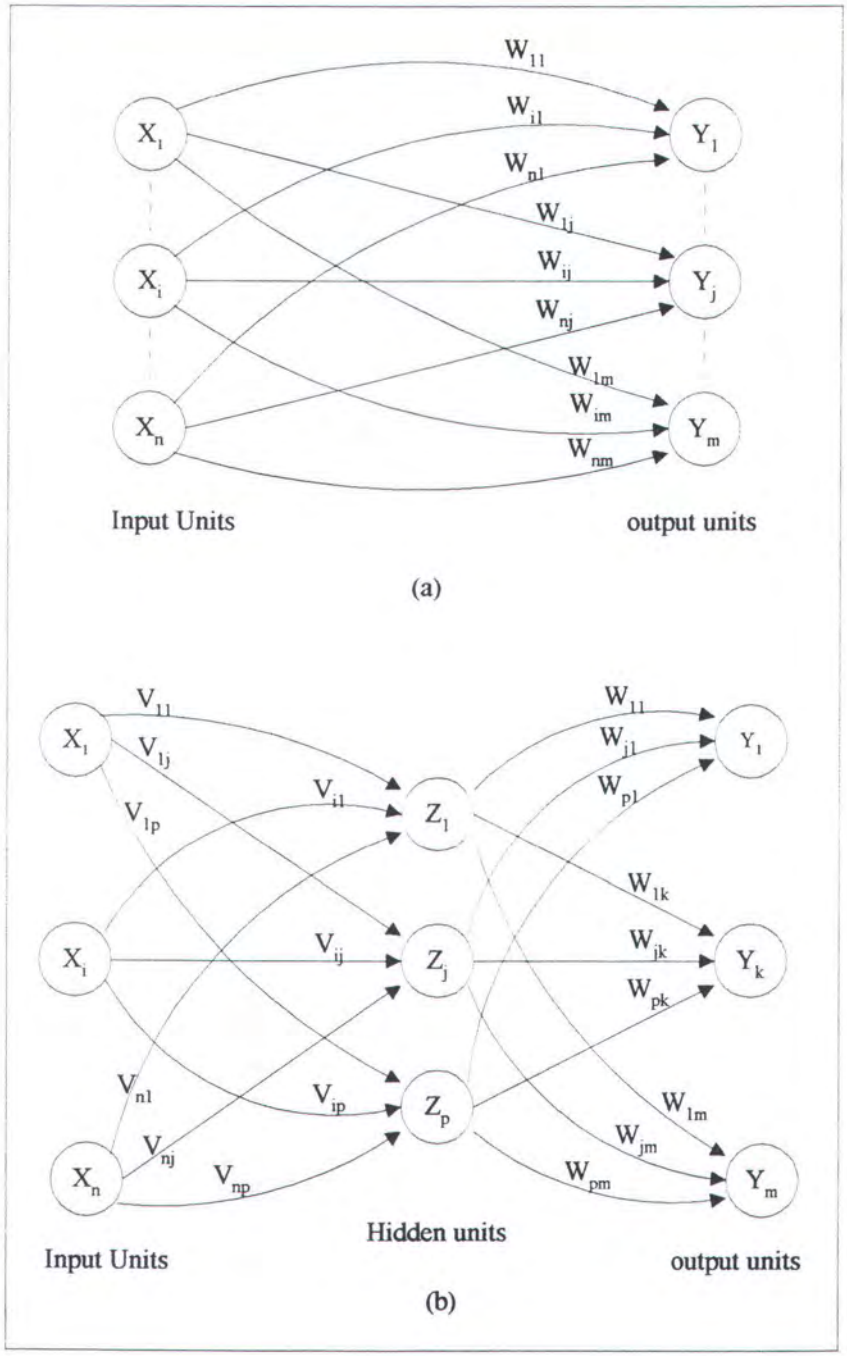
Penyusunan neuron-neuron ke dalam suatu layer dan pola hubungan neuron baik yang di dalam satu layer maupun pola hubungan antar layer disebut *net architecture*[8]. Dalam jaringan input layer tidak dapat dianggap sebagai suatu layer, karena pada input layer tidak ada proses perhitungan. Dan hidden layer juga disebut sebagai *processing unit*, terdiri

[8] Ibid, hal. 12.

atas neuron-neuron yang bertugas untuk mengolah informasi yang berasal dari input dan kemudian diteruskan ke output unit.

Berdasarkan jumlah layer, jaringan dapat dibedakan menjadi *single-layer* dan *multi-layer*. Gambar 3.2(a) menggambarkan jaringan yang berbentuk single-layer, yaitu jaringan yang hanya mempunyai satu layer saja. Sedangkan gambar 3.2(b) menggambarkan jaringan yang berbentuk multi-layer, yaitu jaringan yang terdiri lebih dari satu layer.

Jaringan juga dapat dibedakan berdasarkan arah aliran sinyal, yaitu jaringan bersifat *feedforward* dan *recurrent*. Jaringan feedforward adalah jaringan yang arah aliran sinyal input unitnya-nya menuju ke output unit, pada gambar 3.2(a) dan 3.2(b) merupakan jaringan feedforward. Sedangkan jaringan recurrent, yaitu jika terdapat lintasan yang berasal dan kembali ke unit yang sama.



Gambar 3.2 (a) Bentuk arsitektur jaringan single-layer

(b) Bentuk arsitektur jaringan multi-layer

Salah satu masalah yang timbul dalam menyusun arsitektur jaringan adalah menentukan jumlah dari processing unit, yang akan dihubungkan dengan input dan output unit, dimana pada setiap hubungan tersebut mempunyai nilai tersendiri, dan digunakan untuk melatih jaringan dengan sejumlah contoh persoalan yang akan dihadapi.

3.3 ELEMEN PEMROSES

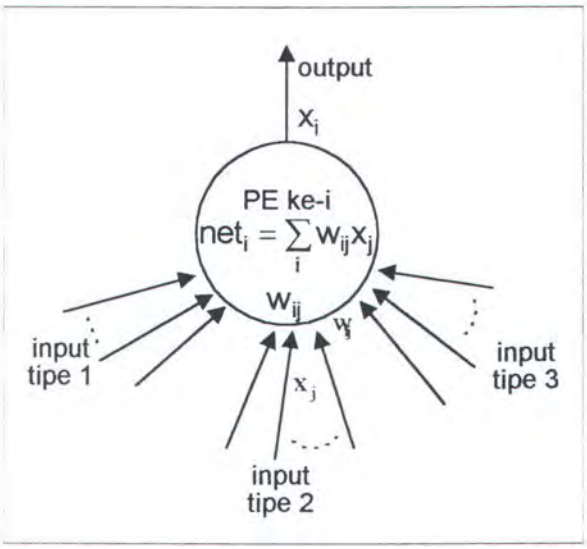
Elemen pemroses (PE) pada jaringan saraf buatan dapat dianalogikan seperti neuron pada saraf biologi. Pada elemen pemroses ini, satu atau lebih input yang berasal dari satu atau lebih elemen pemroses yang lain akan diproses untuk menghasilkan satu output yang dikirim ke satu atau lebih elemen pemroses.

Elemen pemroses mengatur beberapa fungsi dasar dari jaringan saraf. Elemen pemroses akan mengevaluasi sinyal input dan menentukan kekuatan dari sinyal-sinyal input tersebut. Menghitung total perkalian antara input dengan bobot dari masing-masing input serta membandingkan total perkalian dengan suatu nilai batas (*threshold*). Selanjutnya elemen pemroses akan menentukan nilai outputnya.

Model dari elemen pemroses dapat dilihat pada gambar 3.3. Dari gambar dapat dilihat bahwa input pada elemen pemroses ke-i yang berasal dari elemen pemroses ke-j dilambangkan dengan x_{ij}, yang juga merupakan output dari elemen pemroses ke-j. Setiap hubungan ke elemen pemroses ke-i mempunyai suatu nilai bobot tertentu. Bobot pada

PERPUS
INSTITUT TEKNIK
SEPULUH - NOPEMB

hubungan dari elemen pemroses j ke elemen pemroses i dilambangkan dengan W_{ij}.



Gambar 3.3 Struktur Elemen Pemroses pada jaringan saraf[9].

Pada setiap elemen pemroses akan dihitung nilai net-input. Dengan mengabaikan hubungan khusus, nilai net-input dihitung dengan menjumlahkan nilai input dikalikan dengan bobot yang bersesuaian. Nilai net-input untuk elemen ke-i dihitung dengan persamaan sebagai berikut :

$$net_i = \sum_i x_j w_i \qquad (3.3\text{-}1)$$

dimana index j menunjukkan semua hubungan node input yang berhubungan dengan elemen pemroses ke-i. Nilai net-input didasarkan pada semua hubungan input ke suatu elemen pemroses.

---

[9] James A. Freeman and David M. Sakapura, **NEURAL NETWORK 'Algorithms, Application and Programming Techniques'**, Addison-Wesley Publishing Company, 1991, hal. 18.

Untuk setiap elemen pemroses, nilai net-input dikonversikan menjadi nilai keaktifan dengan persamaan,

$$a_i(t) = F_i(a_i(t-1), net_i(t)) \tag{3.3-2}$$

Nilai keaktifan yang sekarang bergantung terhadap nilai keaktifan sebelumnya, a(t-1). Pada kebanyakan kasus, nilai keaktifan sama dengan nilai net-input. Setelah nilai keaktifan dihitung, nilai output untuk setiap elemen pemroses dihitung dengan menggunakan suatu fungsi output,

$$x_i = f_i(a_i) \tag{3.3-3}$$

Karena $a_i = net_i$, fungsi diatas menjadi

$$x_i = f_i(net_i) \tag{3.3-4}$$

Fungsi keaktifan digunakan untuk menunjukkan fungsi f_i, yang merubah nilai net-input menjadi output x_i.

3.4 NILAI BOBOT

Pada jaringan saraf, proses belajar merupakan proses untuk merubah nilai bobot pada elemen pemroses. Proses belajar memegang peranan yang sangat penting dalam jaringan saraf dan operasi-operasi yang berlaku merupakan operasi matriks.

Pada jaringan saraf dengan N elemen pemroses dengan indeks 1 sampai dengan N, memiliki vektor bobot yang dapat ditulis sebagai berikut,

$$w = (w_{11}, w_{12}, \ldots, w_{1n}, w_{21}, w_{22}, \ldots, w_{2n}, \ldots, w_{N1}, w_{N2}, \ldots W_{Nn})^T$$

$$= (w_1^T, w_2^T, \ldots, w_N^T) \qquad (3.4\text{-}1)$$

Vektor ini dibentuk dengan menggabungkan semua bobot dari semua elemen pemroses. Vektor w_1, w_2, ..., w_N merupakan vektor bobot dari elemen pemroses 1,2, ..., N. Pada persamaan diatas, setiap elemen pemroses memiliki n buah elemen pemroses sebagai input dengan indek 1 sampai dengan n. Vektor bobot dari elemen pemroses dapat ditulis sebagai berikut,

$$w_1 = \begin{pmatrix} w_{11} \\ w_{12} \\ \vdots \\ w_{1n} \end{pmatrix} = (w_{11}, w_{12}, \ldots, w_{1n})^T$$

$$w_2 = \begin{pmatrix} w_{21} \\ w_{22} \\ \vdots \\ w_{2n} \end{pmatrix} = (w_{21}, w_{22}, \ldots, w_{2n})^T$$

$$w_N = \begin{pmatrix} w_{N1} \\ w_{N2} \\ \vdots \\ w_{Nn} \end{pmatrix} = (w_{N1}, w_{N2}, \ldots, w_{Nn})^T$$

3.5 METODE BELAJAR

Proses belajar merupakan proses untuk merubah nilai bobot pada elemen pemroses. Proses ini sangat berpengaruh terhadap karakteristik suatu jaringan saraf. Terdapat dua macam metode belajar, yaitu *supervised* dan *unsupervised*. Selain itu terdapat pula jaringan dengan nilai bobot yang sudah ditetapkan tanpa melalui proses belajar.

3.5.1 Metode Supervised

Dalam metode ini, output dari jaringan dibandingkan dengan output yang dikehendaki. Nilai bobot, yang ditetapkan secara acak pada proses belajar, diatur oleh jaringan sehingga pada iterasi berikutnya dapat mendekati output yang diinginkan.

Dengan metode ini, jaringan harus dilatih terlebih dahulu sebelum dioperasikan. Untuk pelatihan diperlukan data input dan output yang disebut dengan *training set*. Setiap nilai input bersesuaian dengan nilai output yang dikehendaki. Pelatihan akan berhenti apabila jaringan menghasilkan output yang diperlukan untuk suatu nilai input.

Tujuan akhir dari semua proses belajar adalah meminimalisasi kesalahan antara output jaringan saraf dengan output yang dikehendaki, dengan cara mengubah nilai bobot secara terus-menerus.

Jaringan saraf menentukan hubungan antara input dan output dengan melihat beberapa contoh dari pasangan input-output. Kemampuan untuk menentukan bagaimana suatu data akan diproses disebut dengan

pengaturan sendiri (*self-organization*). Proses pengaturan sendiri disebut dengan '*adaptasi*' atau '*belajar*'.

Pasangan input-output yang digunakan untuk mengajar atau melatih jaringan disebut dengan *training set*. Untuk mengetahui output yang diharapkan dari suatu input, jaringan belajar dengan mengukur kekuatan hubungan antar elemen pemroses. Metode yang digunakan untuk mengatur proses tersebut disebut dengan '*aturan belajar*' atau (*learning rule*).

3.5.2 Metode Unsupervised

Metode ini disebut juga *self-supervised*. Pada jaringan ini tidak ada pengaruh luar yang mempengaruhi bobot jaringan. Jaringan mencari sifat-sifat beraturan atau kecenderungan dari sinyal input, dan membuat adaptasi menurut fungsi dari jaringan.

Algoritma metode belajar *unsupervised* menekankan pada kerjasama antar *cluster* dari elemen pemroses. Cluster bekerja bersama-sama dan mencoba saling memberikan rangsang. Jika suatu input diaktifkan oleh sembarang node pada cluster, aktivitas cluster secara keseluruhan akan meningkat. Sebaliknya, jika suatu input ke node pada cluster dikurangi, maka aktivitas cluster secara keseluruhan akan berkurang.

Kompetisi antar elemen pemroses merupakan dasar dari metode belajar ini. Pelatihan pada elemen pemroses dapat menjelaskan respon

dari suatu kelompok terhadap rangsang tertentu serta menghubungkan kelompok tersebut satu sama lain dan menghubungkannya dengan respon yang tepat .

BAB IV

JARINGAN SARAF HOPFIELD

BAB IV

JARINGAN SARAF HOPFIELD

Dalam bab IV ini akan digambarkan dan dijelaskan sedikit tentang salah satu bentuk jaringan saraf buatan, yaitu jaringan saraf Hopfield. Pada tugas akhir ini untuk proses klasifikasi citra tekstur kami gunakan arsitektur jaringan saraf Hopfield, tetapi sebagai tambahan agar lebih jelas dalam memahami jaringan Hopfield akan kami jelaskan beberapa keterangan tambahan.

Jaringan Hopfield diperkenalkan serta dikembangkan oleh John Hopfield pada tahun 1982 - 1984. Dari penjelasan pada bab sebelumnya, jaringan saraf dapat dibedakan berdasarkan arah tujuan sinyal, yakni *feedworward* dan *recurrent*. Pada jaringan '*feedforward*', sinyal yang berasal dari suatu neuron akan bergerak menuju ke semua neuron lainnya, dan tidak ada lintasan sinyal yang bergerak dari suatu neuron menuju ke neuron tempat asal sinyal tadi. Sebaliknya, jaringan '*recurrent*' dimungkinkan adanya lintasan sinyal yang berasal dan menuju ke neuron yang sama.

Jaringan saraf Hopfield ini dibagi dalam 2 (dua) bentuk, yaitu Jaringan Hopfield Diskrit dan Jaringan Hopfield Kontinyu, hal ini

bergantung pada fungsi nilai output unitnya apakah merupakan fungsi diskrit atau kontinyu[10].

4.1 JARINGAN HOPFIELD DISKRIT

Pada jaringan Hopfield diskrit, setiap unit saling terhubung satu sama lain. Jaringan ini mempunyai nilai bobot yang bersifat *square* dan *symmetric*, yang berarti untuk nilai bobot W_{ij} akan bernilai sama dengan nilai bobot W_{ji}. Hal ini dikarenakan pada setiap 2 (dua) layer di jaringan akan mempunyai jumlah unit yang sama, dan bobot unit ke-n dari hubungan antara layer 1 menuju ke layer 2 akan mempunyai nilai bobot yang sama dengan unit ke-n dari hubungan antara layer 2 menuju ke layer 1. Dan juga, jaringan ini akan mengabaikan hubungan antara suatu unit dengan unit itu sendiri (*no self-conection*), nilai bobot pada hubungan seperti ini adalah 0. Sehingga dapat kita definisikan, untuk semua hubungan unit nilai bobotnya adalah :

$$W_{ij} = W_{ji} \quad \text{dan} \quad W_{ii} = 0 \qquad (4.1\text{-}1)$$

Jaringan Hopfield diskrit mempunyai dua karateristik yang unik, yakni :

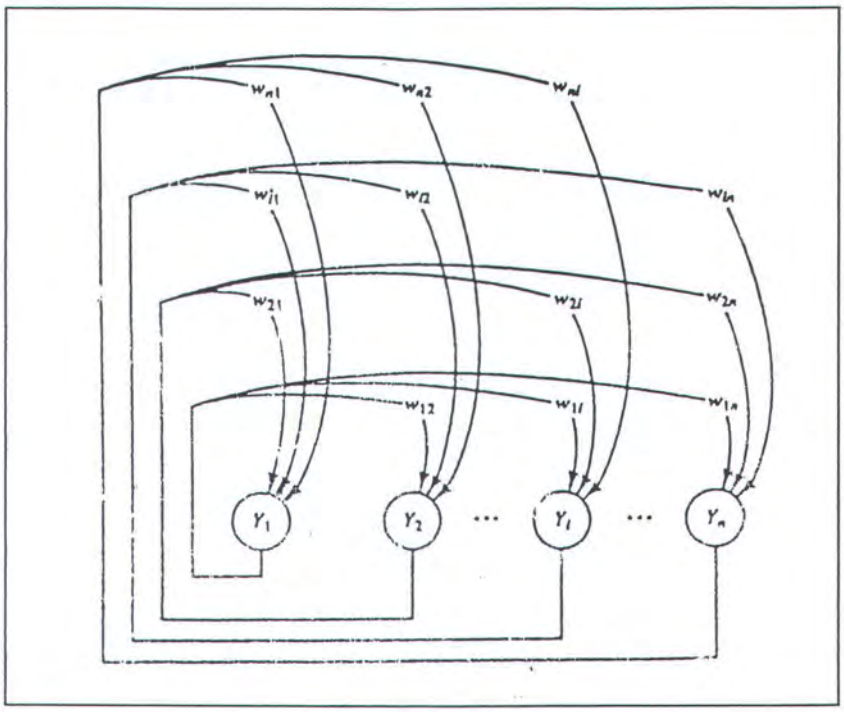
1. Hanya ada 1 (satu) unit yang meng-update nilai keaktifannya pada setiap satuan waktu atau iterasi (nilai keaktifannya ini bergantung pada sinyal yang diterima dari unit-unit lain).

---

[10] Ibid, hal. 141.

2. Setiap unit selain menerima sinyal dari unit-unit lain pada jaringan, juga menerima sinyal dari luar (I_i - input bias).

$$net_i = \sum_{j=1}^{n} w_{ij} x_j + I_i \qquad (4.1\text{-}2)$$



Gambar 4.1

Arsitektur Jaringan Hopfield Diskrit

Untuk menentukan apakah kondisi jaringan sudah stabil (*konvergen*), kita dapat menggunakan suatu fungsi energi atau disebut fungsi *Lyapunov* pada jaringan. Lyapunov (1857 - 1918), adalah seorang ahli matematika dan insinyur mesin, dialah yang pertamakali memperkenalkan fungsi energi tersebut. Dalam beberapa kali iterasi, pada akhirnya nanti, nilai keaktifan pada semua unit akan tidak berubah untuk

iterasi berikutnya. Pada kondisi inilah dinyatakan bahwa jaringan telah mencapai kondisi yang stabil atau konvergen.

Bentuk arsitektur jaringan Hopfield diskrit dapat dilihat pada gambar 4.1 di halaman sebelumnya. Dari gambar dapat dilihat bahwa tidak ada hubungan yang berasal dan menuju ke unit yang sama. Jaringan Hopfield diskrit ini biasanya digunakan untuk menyelesaikan persoalan yang berhubungan dengan *associative memory*.

Algoritma pada aplikasi jaringan Hopfield diskrit adalah sebagai berikut :

Step 0. Inisialisasi semua nilai bobot.

$$W_{ij} = \sum_p [2s_i(p) - 1][2s_j(p) - 1] \quad \text{for } i \neq j$$

Cek semua nillai keaktifan pada jaringan. Jika belum konvergen lakukan langkah2 no 1-7

Step 1. Untuk setiap input vector x, lakukan langkah2 no 2-6

Step 2. Inisialisasi nilai keaktifan dalam jaringan dengan nilai input vector dari luar.

$$y_i = x_i, \quad (i = 1,.....n)$$

Step 3. Lakukan langkah no 4-6 untuk setiap unit Y_i

(urutan unit yang aktif dipilih secara acak)

Step 4. Lakukan perhitungan input pada jaringan :

$$y_in_i = x_i + \sum_j y_j w_{ji}$$

Step 5. Cek nilai nilai keaktifannya (sinyal output)

$$y_i = \begin{cases} 1 & \text{if} \quad y_in_i > \theta_i \\ y_i & \text{if} \quad y_in_i = \theta_i \\ 0 & \text{if} \quad y_in_i < \theta_i \end{cases}$$

nilai threshold, θ, dapat diberi nilai sembarang, tetapi biasanya diberi nilai 0.

Step 6. Masukkan nilai y_i ke semua unit

Step 7. Cek apakah jaringan sudah konvergen/belum.

Pada kasus sederhana, jaringan Hopfield diskrit dapat digunakan untuk men-generate suatu vector input ke dalam vector yang kita inginkan. Sebagai contoh kita masukkan vector (1,1,1,0) atau (dalam bentuk bipolar (1,1,1,-1)) ke dalam jaringan. Kemudian kita berikan vector input (0,0,1,0). Dari persoalan sederhana ini, kita akan proses vector input (0,0,1,0) dengan menggunakan algoritma jaringan Hopfield diskrit sehingga dihasilkan vector yang kita inginkan, dalam hal ini yaitu, (1,1,1,0).

Analisis dan solusi masalah :

Step 0. Inisialisasi nilai bobot

$$W = \begin{bmatrix} 0 & 1 & 1 & -1 \\ 1 & 0 & 1 & -1 \\ 1 & 1 & 0 & -1 \\ -1 & -1 & -1 & 0 \end{bmatrix}$$

Step 1. Vector input x = (0,0,1,0)

Step 2. y = (0,0,1,0)

MILIK PERPUSTA
INSTITUT TEKNOL
SEPULUH – NOPEMB

Step 3. Aktifkan unit Y_1

Step 4. $y_in_1 = x_1 + \sum_j y_j w_{j1} = 0 + 1 = 1$

Step 5. $y_in_1 > 0$

$y_1 = 1$

Step 6. $y = (1,0,1,0)$

Step 3. Aktifkan unit Y_4

Step 4. $y_in_4 = x_4 + \sum_j y_j w_{j4} = 0 + (-2) = -2$

Step 5. $y_in_4 < 0$

$y_4 = 0$

Step 6. $y = (1,0,1,0)$

Step 3. Aktifkan unit Y_3

Step 4. $y_in_3 = x_3 + \sum_j y_j w_{j3} = 1 + 1 = 2$

Step 5. $y_in_3 > 0$

$y_3 = 1$

Step 6. $y = (1,0,1,0)$

Step 3. Aktifkan unit Y_2

Step 4. $y_in_2 = x_2 + \sum_j y_j w_{j2} = 0 + 2 = 2$

Step 5. $y_in_2 > 0$

$y_2 = 1$

Step 6. $y = (1,1,1,0)$

Step 7. Cek, apakah kondisi jaringan sudah konvergen

Pada kondisi ini jika proses dilanjutkan, maka tidak ada perubahan nilai keaktifan di semua unit, sehingga dinyatakan bahwa jaringan sudah konvergen. Dan, dari analisa diatas dapat diamati bahwa kemampuan jaringan untuk konvergen tidak dipengaruhi oleh apakah nilai input dan nilai keaktifannya dalam bentuk biner atau bipolar, tetapi ditentukan oleh perubahan besarnya nilai bobot[11].

4.1.1 Fungsi Energi Jaringan Hopfield Diskrit

Salah satu cara untuk menentukan apakah suatu jaringan sudah mencapai konvergen pada jaringan Hopfield diskrit adalah dengan menggunakan fungsi energi (atau fungsi Lyapunov). Pengertian fungsi energi disini adalah suatu fungsi yang akan dinyatakan berhenti (konvergen) jika nilai fungsi tersebut tidak bertambah pada semua unit. Jadi jika nilai suatu fungsi energi sudah '*ditemukan*' pada sebuah iterasi, maka dinyatakan bahwa jaringan sudah *konvergen*.

Pada contoh kasus seperti diatas, dapat kita buat suatu fungsi energi jaringan Hopfield diskrit sebagai berikut :

$$E = -0.5 \sum_{i \neq j} \sum_{j} y_i y_j w_{ij} - \sum_{i} x_i y_i + \sum_{i} \theta_i y_i \qquad (4.1\text{-}3)$$

Jadi jika nilai keaktifan pada jaringan berubah sebesar Δy_i, maka energi akan berubah sebesar

[11] Laurene Fausett, op.cit., hal. 140.

$$\Delta E = -\ [\sum_j y_j w_{ij} + x_i - \theta_i]\ \Delta y_i \qquad (4.1\text{-}4)$$

Dari persamaan diatas, jika y_i positif dan y_i berubah ke nilai 0, maka

- jika $x_i + \sum_j y_j w_{ji} < \theta_i$

  maka perubahan pada y_i adalah negatif, sehingga $\Delta E < 0$

Tetapi jika y_i adalah 0 dan y_i berubah ke nilai positif, maka

- jika $x_i + \sum_j y_j w_{ji} > \theta_i$

  maka perubahan pada y_i adalah positif, sehingga $\Delta E < 0$

Dari uraian diatas dapat kita ambil kesimpulan, bahwa kondisi jaringan akan konvergen, yaitu apabila Δy_i positif dan $[x_i + \sum_j y_j w_{ji}]$ juga positif, dan bila Δy_i negatif maka $[x_i + \sum_j y_j w_{ji}]$ juga harus negatif, sehingga perubahan fungsi energi tidak bertambah besar.

4.2 JARINGAN HOPFIELD KONTINYU

Jaringan Hopfield kontinyu merupakan pengembangan dari bentuk jaringan Hopfield diskrit. Jaringan Hopfield kontinyu ini mempunyai nilai fungsi output yang bersifat kontinyu. Dan juga, jaringan Hopfield kontinyu juga dapat digunakan untuk menyelesaikan persoalan yang berhubungan dengan *associative memory*. Seperti pada jaringan Hopfield diskrit,

jaringan Hopfield kontinyu juga mempunyai bentuk nilai bobot yang simetris, $w_{ij} = w_{ji}$.

Dalam jaringan Hopfield kontinyu, kita definiskan u_i sebagai input jaringan pada elemen pemroses (*Processing Element - PE*) yang ke-i. Sedangkan fungsi output PE ke-i adalah :

$$v_i = g_i(\lambda u_i) = \frac{1}{2}(1 + \tanh(\lambda u_i)) \qquad (4.2\text{-}1)$$

dimana x adalah konstanta *gain parameter.*

Bila dalam jaringan Hopfield kontinyu, diberikan fungsi energi :

$$E = 0.5 \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} v_i v_j + \sum_{i=1}^{n} \theta_i v_i \qquad (4.2\text{-}2)$$

jaringan akan konvergen atau dalam kondisi stabil, yang merupakan nilai minimum dari fungsi energi, bila

$$\frac{d}{dt} E \leq 0 \qquad (4.2\text{-}3)$$

Untuk bentuk fungsi energi seperti ini, jaringan akan konvergen jika keaktifan dari setiap neuron berubah menurut waktu dengan memenuhi persamaan differensial :

$$\frac{d}{dt} u_i = -\frac{\partial E}{\partial v_i} = -\sum_{j=1}^{n} w_{ij} v_j - \theta_i \qquad (4.2\text{-}4)$$

Di dalam karya tulisnya yang diterbitkan pada tahun 1984, Hopfield memperkenalkan suatu fungsi energi untuk jaringan Hopfied kontinyu yakni,

$$E = -0.5 \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} v_i v_j - \sum_{i=1}^{n} \theta_i v_i + \frac{1}{\tau} \sum_{i=1}^{n} \int_0^{v_i} g_i^{-1}(v) dv \qquad (4.2\text{-}5)$$

4.2.1 Fungsi Energi Jaringan Hopfield Kontinyu

Fungsi energi dari jaringan Hopfield kontinyu adalah :

$$E = -\frac{1}{2} \sum_i \sum_j T_{ij} v_i v_j + \frac{1}{\lambda} \sum_i \frac{1}{R_i} \int_0^{v_i} g_i^{-1}(v) dv - \sum_i I_i v_i \qquad (4.2\text{-}6)$$

Pada persamaan diatas $g^{-1}(v) = u$ merupakan fungsi invers dari $v = g^{-1}(u)$. Gambar 4.2 menyatakan suatu fungsi $g^{-1}(v) = u$, serta integral dari $g^{-1}(v)$ sebagai fungsi v.

Untuk menjelaskan bahwa fungsi diatas adalah merupakan fungsi Lyapunov dari sistem, maka persamaan diatas akan kita turunkan dan diasumsikan bahwa T_{ij} adalah simetris, sehingga :

$$\frac{dE}{dt} = -\sum_i \frac{dv_i}{dt} \left(\sum_j T_{ij} v_j - \frac{u_i}{R_i} + I_i \right) \qquad (4.2\text{-}7)$$

kemudian

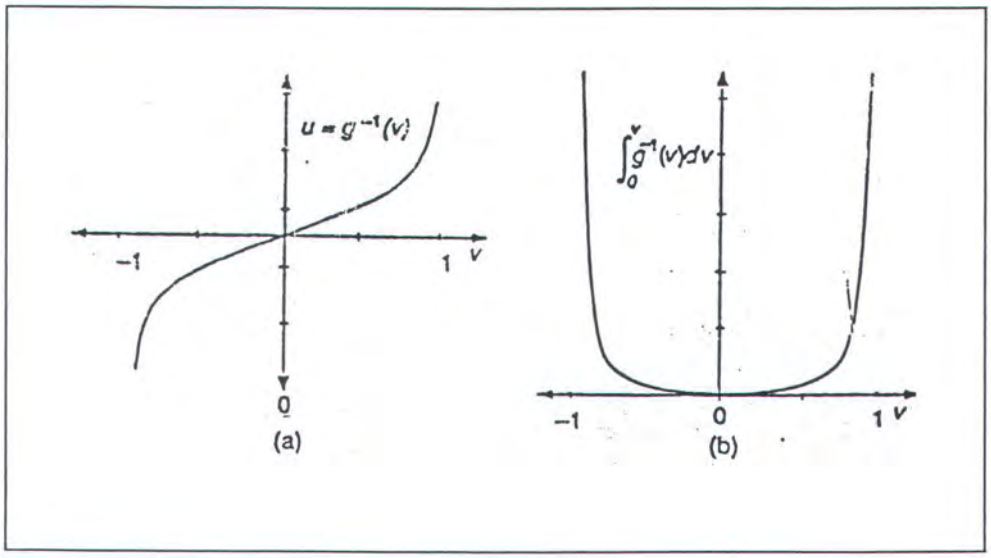
$$\frac{dE}{dt} = -\sum_i C \frac{dv_i}{dt} \frac{du_i}{dt} \qquad (4.2\text{-}8)$$

karena $u_i = g_i^{-1}(v_i)$, dengan menggunakan teorema rantai maka persamaan dapat ditulis menjadi :

$$\frac{du_i}{dt} = \frac{dg_i^{-1}(v_i)}{dv_i} \frac{dv_i}{dt} \quad \text{dan} \qquad (4.2\text{-}9)$$

$$\frac{dE}{dt} = -\sum_i C \frac{dg_i^{-1}(v_i)}{dv_i} \left(\frac{dv_i}{dt} \right)^2 \qquad (4.2\text{-}10)$$

Gambar 4.2 menunjukkan bahwa $g_i^{-1}(v_i)$ adalah fungsi dari v_i yang bertambah secara monotonik, sehingga turunannya selalu bernilai positif. Semua faktor dalam partambahan pada persamaan akan bernilai positif, sehingga $\frac{dE}{dt}$ akan menurun. Sistem akan stabil, apabila $\frac{dE}{dt} = \frac{dv_i}{dt} = 0$. Dalam sistem ini, diasumsikan bahwa nilai E adalah terbatas, sehingga pada suatu iterasi nanti akan mempunyai nilai E yang tidak berubah.



Gambar 4.2

Grafik dari fungsi $g^{-1}(v)$ dan fungsi integral dari $g^{-1}(v)$

BAB V

PERACANGAN DAN PEMBUATAN

BAB V

PERANCANGAN DAN PEMBUATAN PERANGKAT LUNAK

Pada bab ini akan dibahas mengenai sistem perangkat lunak klasifikasi tekstur, algoritma program beserta penjelasan konsep-konsep teori yang dipakai pada sistem klasifikasi tekstur ini, serta penjelasan struktur data secara global yang digunakan dalam pembuatan perangkat lunak.

Perangkat lunak ini dibuat dengan menggunakan bahasa pemrograman Delphi versi 1.0, yang dijalankan pada sistem operasi Windows 3.1 dengan prinsip *obejct-oriented programming* (OOP). Kelebihan utama dalam compiler Borland Delphi ini adalah dalam hal kemudahannya untuk memanfaatkan Windows API (*Application Programming Interface*), yang digunakan sebagai interaksi antar muka dengan pemakai, karena fasilitas ini sudah di*handle* dalam class libaray Delphi.

5.1 KEBUTUHAN SISTEM

Perangkat lunak yang dibuat hanya dapat dijalankan dibawah sistem operasi Windows 3.1 atau windows 95. Sistem operasi windows ini

mempunyai manajemen memori yang sangat bagus, yang tidak hanya menggunakan memori RAM saja, tetapi juga dapat menggunakan ruang di dalam Hard-disk sebagai *virtual-memori*. Pada perangkat lunak ini juga dibuat beberapa file database yang digunakan untuk menyimpan data.

5.2 SISTEM KLASIFIKASI TEKSTUR

Persoalan klasifikasi tekstur merupakan salah satu persoalan optimasi dari sejumlah piksel yang tersusun pada bidang spasial dua dimensi dari sebuah citra ke dalam kelas-kelas/objek-objek tertentu, dimana pada citra input tersebut tersusun atas region-region dari beberapa macam tekstur. Sebelum dilakukan proses klasifikasi dari sebuah citra input, harus diketahui terlebih dahulu jumlah kelas serta tipe kelasnya. Selanjutnya dengan menggunakan metode tertentu kita akan dapat menentukan kelas dari tiap-tiap piksel dalam citra input tadi, ke dalam kelasnya masing-masing.

Ada beberapa batasan dari persoalan ini, seperti yang telah saya sebutkan di bagian pertama (pendahuluan), dimana jumlah tekstur yang terdapat dalam citra input maksimal terdiri dari sepuluh macam, juga tekstur yang dapat kita kenali hanya terbatas pada tekstur yang bersifat *periodik* baik yang keperiodikannya berbentuk terstruktur maupun tidak terstruktur. Batasan ini diperlukan karena metode yang kami gunakan untuk pemodelan tekstur citra hanya mampu mengolah data pada tipe

tekstur tersebut. Pada tugas akhir ini metode yang digunakan untuk memodelkan tekstur adalah Markov Random Field.

Tujuan dari pemodelan data tekstur adalah untuk menganalisa dan menggambarkan bentuk karakteristik dari suatu citra tekstur, sehingga didapatkan informasi-informasi dari masing-masing tekstur. Informasi-informasi ini selanjutnya digunakan sebagai parameter untuk proses klasifikasi. Selain itu dari informasi-informasi ini pula kita akan dapat membuat suatu citra tekstur tiruan dengan menggunakan suatu algoritma tertentu. Data-data yang diperlukan pada metode MRF ini yaitu nilai intensitas piksel dari suatu citra. Salah satu pendekatan sederhana untuk menganalisa nilai intensitas suatu piksel dari suatu citra, ditekankan pada nilai intensitas dari tetangga-tetangganya, dengan aturan hubungan antar tetangga yang akan kami jelaskan pada bagian berikutnya.

Ada empat dasar pokok dalam proses pengolahan citra yang berkaitan dengan klasifikasi tekstur[12], yaitu :

1. Proses segmentasi dalam suatu citra.
2. Proses klasifikasi pada tiap-tiap piksel dalam suatu citra input.
3. Mentransformasikan semua proses pada persoalan ini serta menampillkan informasi ke dalam grafik komputer.
4. Melaksakanan proses pengkodean ulang dari suatu citra.

---

[12] George R. Cross and Anil K. Jain, **IEEE TRANSACTION ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE**, Vol. PAMI-5, No. 1, January 1983, hal. 25.

Proses segmentasi diperlukan untuk mendapatkan bentuk secara kasar dari objek-objek pada suatu citra input, sehingga proses untuk klasifikasiannya akan dapat dilakukan lebih mudah. Dan dari bentuk segmentasi secara kasar tadi selanjutnya kita lakukan proses klasifikasi pada semua piksel ke dalam kelas-kelasnya secara random. Perlu diketahui bahwa untuk mendapatkan suatu hasil yang optimal dari proses klasifikasi pada setiap piksel dalam suatu citra adalah sangat sulit, hal ini dikarenakan besarnya ukuran area dalam pendeteksian citra. Sebagai gambaran, jika ukuran dari citra input adalah 255-kali-256 serta terdapat 4 macam tekstur dalam citra, maka terdapat $2^{4^{16}}$ kemungkinan konfigurasi label dalam proses klasifikasi. Metode yang kami gunakan untuk proses klasifikasi adalah dengan teknik *Deterministic Relaxation* yang diimplementasikan ke dalam *Energi Hopfield*.

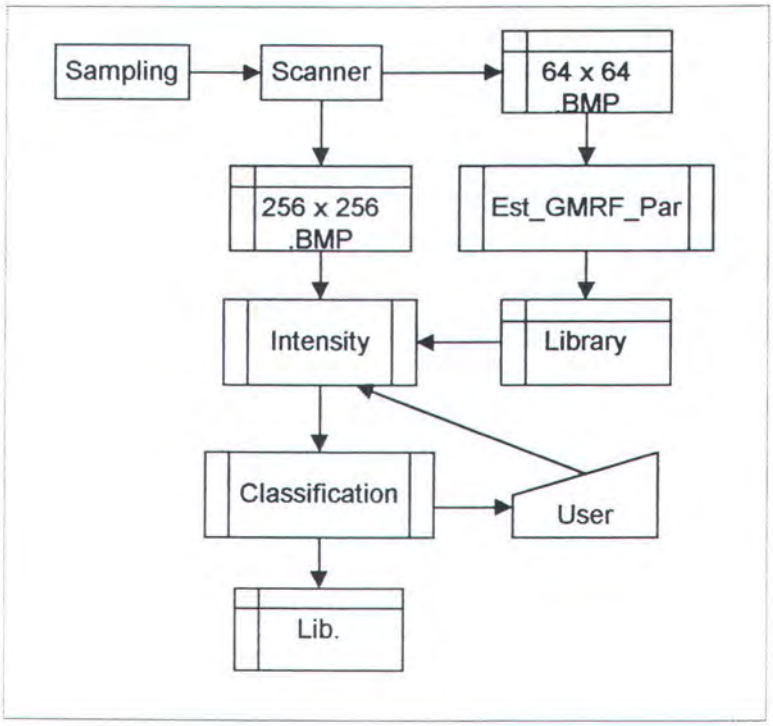
5.3 GAMBARAN UMUM PERANGKAT LUNAK

Perangkat lunak ini pada dasarnya merupakan alat bantu untuk memvisualisasikan suatu citra tekstur serta citra yang merupakan hasil dari proses klasifikasi ke dalam kelas-kelas tekstur tertentu. Masukan yang diperlukan adalah suatu citra tekstur dalam bentuk file .BMP baik yang dihasilkan dari proses scanning suatu objek atau file .BMP hasil olahan dari software-software khusus yang menyediakan file citra tekstur. Dan keluaran yang dihasilkan adalah suatu citra yang telah dibagi dan dikelompokkan ke dalam kelas-kelas tertentu.

Untuk mengimplementasikan tugas-tugas diatas perangkat lunak ini dibagi atas beberapa modul utama, yaitu : *Est_GMRF_Par*, *Intensity* dan *Classification*. Masing-masing modul kemudian dikembangkan secara independen. Modul-modul tersebut serta interaksinya dapat dilihat pada gambar 5.1.

Proses diawali dengan pengambilan citra tekstur melalui proses scanning yang disimpan dalam file berformat .BMP. Citra tekstur yang diperlukan adalah citra yang berukuran 64 x 64 dan 256 x 256 yang masing-masing dikonversi ke dalam bentuk *gray-level* 8 bit. Tahap berikutnya adalah melakukan proses komputasi untuk mendapatkan nilai parameter-parameter GMRF dari citra input yang berukuran 64 x 64. Dari parameter-parameter GMRF tersebut selanjutnya dilakukan pemrosesan pada citra input yang berukuran 256 x 256 untuk mendapatkan nilai intensitas dari setiap piksel dengan menggunakan parameter-parameter GMRF tersebut sebagai variabelnya. Setelah didapatkan nilai intensitas pada setiap piksel, maka proses klasifikasi dapat dilakukan, dan hasil dari proses klasifikasi dapat diperlihatkan kepada user.

MILIK PERPUSTA
INSTITUT TEKNOL
SEPULUH – NOPEMBE



Gambar 5.1

Modul-modul yang terdapat dalam perangkat lunak

5.4 PEMODELAN TEKSTUR DENGAN MARKOV RANDOM FIELD (MRF)

Pada proses pengolahan citra, citra input dinyatakan sebagai sekumpulan piksel-piksel dengan nilai tertentu pada suatu arary persegi panjang atau bujursangkar. Dalam tugas akhir ini nilai dari setiap piksel pada citra hanya dibatasi pada tingkat gray level sebesar 8 bit, sehingga nilai gray levelnya terletak antara 0 sampai dengan 255 (0-255). Misalkan Ω menyatakan sekumpulan piksel yang terletak pada bidang koordinat dua dimensi dari suatu citra yang berukuran M x M, sehingga Ω dapat dinyatakan secara matematis sebagai berikut :

$$\Omega = \{(i,j),\ 1 \le i,j \le M\} \qquad (5.4\text{-}1)$$

Pada metode pemodelan Markov Random Field, nilai intensitas suatu piksel dari citra sangat bergantung pada nilai intensitas dari tetangga-tetangganya, kecuali jika terdapat beberapa noise dalam citra. Jika dimisalkan $X(i,j)$ adalah '*brightness level*' pada suatu titik dalam citra M x M, dan pengkodean suatu kelas dari $X(i,j)$ dinyatakan sebagai $X(i)$ dimana $i = 1,2,....,M^2$. Suatu piksel pada titik j dinyatakan sebagai tetangga dari piksel titik i jika probabilitas dari '*brightness level*' titik i, yakni

$$p(X(i) \mid i = 1,2,3,....., M^2)$$

bergantung pada X(j) '*brightness level*' di titik j. Dari teorema-teorema diatas, maka Markov Random Field dapat didefinisikan[13] sebagai berikut :

1. *Positivity* : yaitu $p(X) > 0$ untuk semua X.
2. *Markovianity* :

   $p(X(i) \mid$ semua titik yang terdapat dalam citra kecuali titik i)

   $= p(X(i) \mid$ semua tetangga dari i)
3. *Homogeneity* : $p(X(i) \mid$ semua tetangga dari i) hanya bergantung pada tetangga yang konfigurasinya telah ditentukan.

Untuk menentukan anggota tetangga dari suatu titik s dapat kita gunakan aturan dari *Gaussian-Markov Random Field* (*GMRF*). Struktur hirarki dari model GMRF dapat dilihat pada gambar 5.2.

---

[13] Ibid, hal. 26.

		7	6	7		
	5	4	3	4	5	
7	4	2	1	2	4	7
6	3	1	X	1	3	6
7	4	2	1	2	4	7
	5	4	3	4	5	
		7	6	7		

Gambar 5.2 Struktur hirarki dari model GMRF

Dari gambar 5.2 dapat dilihat bahwa angka '*order*' dari model struktur GMRF adalah relatif terhadap lokasi pusat yakni X dan bersifat simetris. Model hirarki GMRF yang digunakan dalam tugas akhir ini adalah model '*fourth order*'[14].

Ada beberapa pendefinisian dalam struktur model GMRF ini, yaitu:

- *Ns* menyatakan '*symmetric fourth-order neighborhood*' dari titik s.
- *N** menyatakan sekumpulan dari '*one-sided shift vector*' yang '*berkorespondensi*' terhadap '*fourth-order neighborhood*'.

Dimana *N** adalah sebagai berikut :

$$N^* = \{\tau_1, \tau_2, \tau_3, \tau_4, \tau_5, \tau_6, \tau_7, \tau_8, \tau_9, \tau_{10}\}$$

$$= \{(-1,0), (0,1), (-1,1), (1,1), (-2,0), (0,2), (-1,2), (1,2), (-2,1), (2,1)\}$$

dan

[14] Bart Kosko, op.cit., hal. 40.

$$Ns = \{r : r = s \pm \tau, \ \tau \in N^*\} \qquad (5.4\text{-}2)$$

dalam hal ini $s + \tau$ didefinisikan sebagai

$s = (i,j)$ dan $\tau = (x,y)$ sehingga,

$s + \tau = (i+x, j+y)$

5.4.1 Estimasi Parameter GMRF

Ada banyak metode yang telah dikembangkan untuk mengestimasikan parameter GMRF. Tetapi tidak satupun diantara metode-metode tersebut yang dapat menjamin ke-*konsisten*-an[15] (perkiraan konvergen ke dalam suatu parameter) dan ke-stabil-an[16] (suatu matriks yang menyatakan nilai *joint probability density* dari MRF adalah pasti positif) dapat dicapai secara bersamaan. Secara umum pengoptimalan dari algoritma ini befungsi untuk memperkirakan kondisi yang stabil. Parameter GMRF ini digunakan untuk mendapatkan beberapa ukuran parameter untuk proses klasifikasi dan bukan bertujuan untuk mendapatkan tekstur tiruan. Oleh sebab itu metode ini dianggap sudah cukup memadai dalam mendapatkan parameter-parameter untuk proses segmentasi dan klasifikasi meskipun kita tidak dapat mengestimasikan *stability* dari parameter GMRF tersebut.

Proses pengolahan citra untuk mendapatkan parameter GMRF akan dilakukan pada citra yang terdiri dari satu macam tekstur, sehingga

---

[15] B.S. Manjunath and R. Chellappa, IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, Vol. 13, No. 5, May 1991, hal. 479.

[16] Ibid, hal. 479.

masing-masing tekstur akan mempunyai nilai parameter GMRF. Citra yang akan diproses mempunyai resolusi 64 x 64 dengan nilai gray level dari 0 sampai dengan 255. Ada dua macam parameter GMRF, yang masing-masing kita lambangkan sebagai $\hat{\Theta}$ dan $\hat{\sigma}$. Diagram aliran data untuk proses mendapatkan nilai parameter GMRF pertama, $\hat{\Theta}$, dapat dilihat pada gambar 5.3. Fungsi persamaan parameter GMRF pertama[17] adalah sebagai berikut,

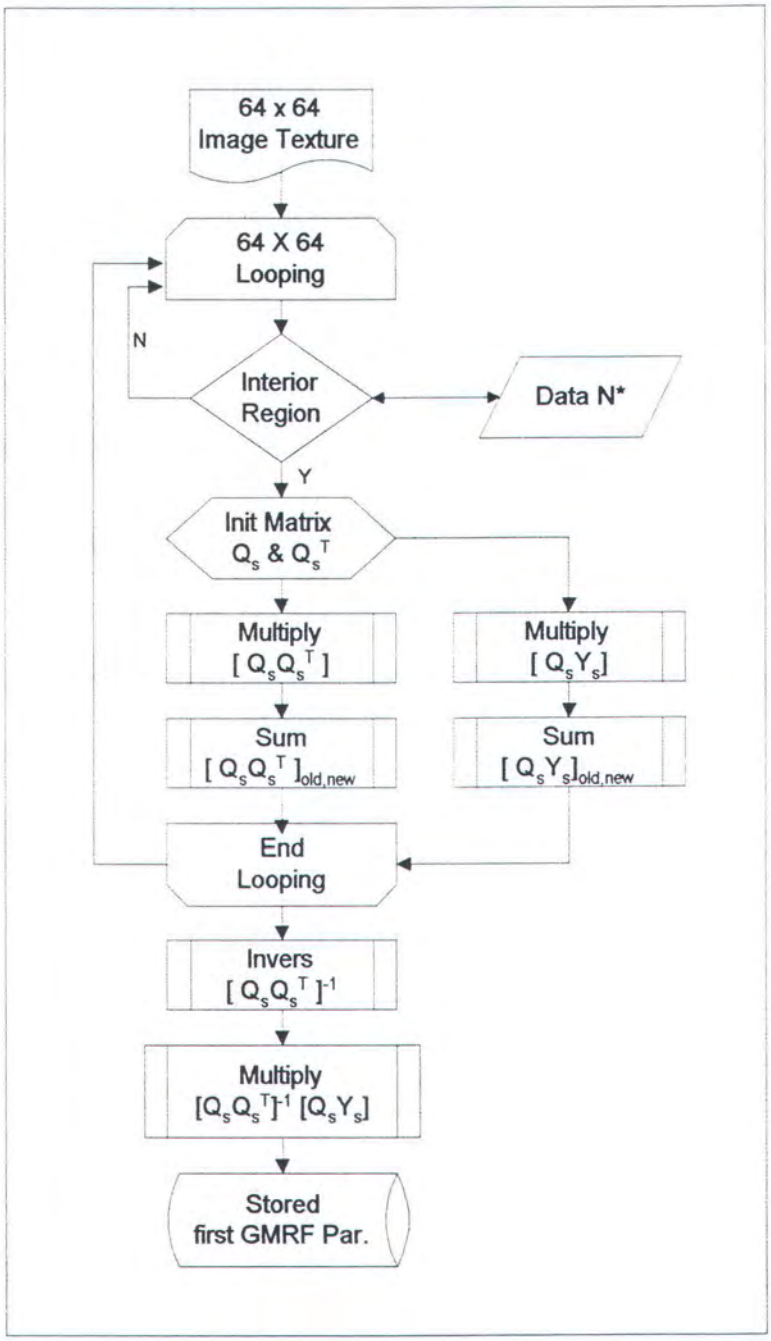
$$\hat{\Theta} = \left[\sum_{\Omega_I} Q_s Q_s^T\right]^{-1} \left[\sum_{\Omega_I} Q_s y_s\right] \qquad (5.4\text{-}3)$$

Q_s adalah matriks berukuran 10-kali-10, sedangkan Q_s^T adalah matriks transpose dari Q_s. Simbol *s* pada lambang Q_s, merupakan titik koordinat dua dimensi pada citra yang sedang diproses, sedangkan y_s menyatakan nilai gray level pada koordinat *s*. Posisi koordinat s pada persamaan diatas hanya meliputi *interior region* yang dilambangkan sebagai Ω_I, dimana Ω_I adalah merupakan hasil pengurangan Ω dan Ω_B atau,

$$\Omega_I = \Omega - \Omega_B,$$

dan $\Omega_B = \{s=(i,j),$ dengan $s \in \Omega$ dan paling tidak ada satu atau lebih $s \pm \tau \notin \Omega$, dimana $\tau \in N^*\}$

[17] Bart Kosko, op.cit., hal. 42.



Gambar 5.3

Diagram aliran data proses etimasi paramater GMRF pertama

Sedangkan fungsi matriks Q_s adalah sebagai berikut :

$$Q_s = \left[y_{s+\tau_1} + y_{s-\tau_1}, y_{s+\tau_2} + y_{s-\tau_2}, \ldots\ldots\ldots, y_{s+\tau_{10}} + y_{s-\tau_{10}} \right]^T \qquad (5.4\text{-}4)$$

Fungsi inisialisasi Q_s diatas memberikan nilai elemen matriks Q_s pada kolom dua sampai dengan sepuluh sama dengan 0. Sebaliknya pada matriks Q_s^T elemen-elemen pada baris dua sampai dengan 10 akan bernilai 0. Susunan elemen-elemen matriks Q_s dapat dilihat pada gambar 5.4.

$$Q_s = \begin{bmatrix} y_{s+\tau_1} + y_{s-\tau_1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ y_{s+\tau_2} + y_{s-\tau_2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ y_{s+\tau_3} + y_{s-\tau_3} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ y_{s+\tau_4} + y_{s-\tau_4} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ y_{s+\tau_5} + y_{s-\tau_5} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ y_{s+\tau_6} + y_{s-\tau_6} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ y_{s+\tau_7} + y_{s-\tau_7} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ y_{s+\tau_8} + y_{s-\tau_8} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ y_{s+\tau_9} + y_{s-\tau_9} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ y_{s+\tau_{10}} + y_{s-\tau_{10}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Gambar 5.4

Susunan elemen-elemen matriks Q_s

Elemen-elemen matriks Q_s pada kolom pertama merupakan penjumlahan dari nilai gray level tetangga dari posisi s, dengan penentuan kriteria tetangga seperti yang telah dijelaskan pada bagian sebelumnya. Proses ini akan dilakukan untuk semua piksel yang terdapat pada *interior region*. Sebagai gambaran jika dilakukan proses pada koordinat s (50,80) maka matriks Q_s adalah sebagai berikut :

$$Q_s = \begin{bmatrix} y_{49,80_1} + y_{51,80_1} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ y_{50,80} + y_{50,79} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ y_{49,81} + y_{51,79} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ y_{51,81} + y_{49,79} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ y_{48,80} + y_{52,80} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ y_{50,82} + y_{50,78} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ y_{49,82} + y_{51,78} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ y_{51,82} + y_{49,78} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ y_{48,81} + y_{52,79} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ y_{52,81} + y_{48,79} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Gambar 5.5 Matriks Q_s pada posisi s(50,80)

Matriks yang merupakan hasil penjumlahan dari perkalian vektor antara matriks Q_s dan Q_s^T yang diproses pada semua piksel dalam interior region pada citra, kemudian dilakukan proses *invers* pada matriks tersebut, yang selanjutnya disebut matriks A. Pada proses matematis lainnya juga dihasilkan suatu matriks yang merupakan hasil penjumlahan

dari perkalian skalar antara matriks Q_s dan nilai gray level di posisi s, y_s, yang selanjutnya disebut matriks B. Dengan melakukan proses perkalian vektor antara matriks A dan matriks B maka akan dihasilkan parameter GMRF pertama. Parameter GMRF pertama ini terdiri dari 10 elemen yang terdapat dalam matriks hasil perkalian matriks A dan matriks B tersebut. Parameter-parameter GMRF tersebut adalah $\Theta_1^\ell, \Theta_2^\ell, \ldots\ldots, \Theta_{10}^\ell$, dimana simbol ℓ menyatakan tipe tekstur atau tipe kelasnya.

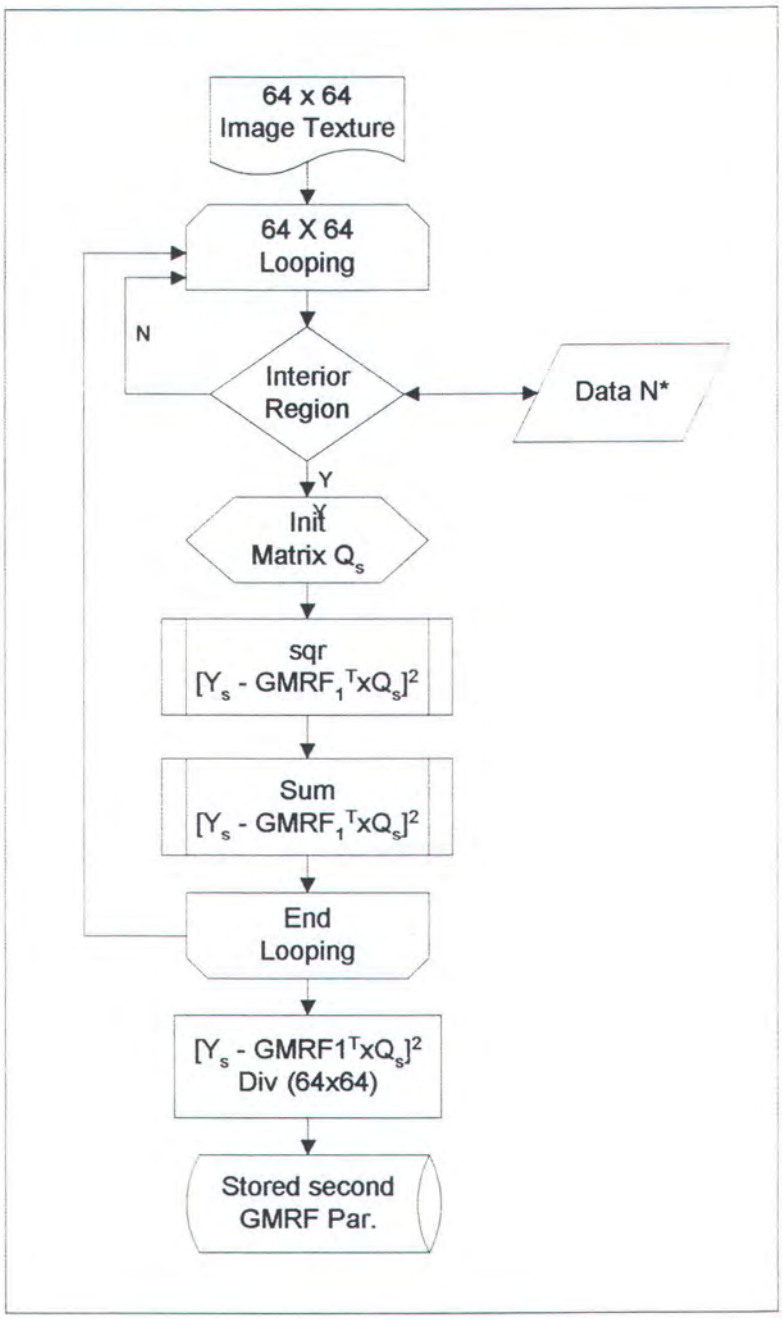
Parameter GMRF kedua, $\hat{\sigma}$, persamaan matematisnya adalah sebagai betikut[18] :

$$\sigma^2 = \frac{1}{N^2} \sum_{\Omega_i} [y_s - \Theta^T Q_s]^2 \qquad (5.4\text{-}5)$$

Sama seperti proses pada parameter GMRF pertama, pada fungsi ini, proses juga dilakukan pada semua piksel yang posisinya terletak pada interior region. N menyatakan resolusi dari citra tekstur yang akan diproses, dalam hal ini N=64. Diagram aliran data untuk proses pada parameter GMRF kedua dapat dilihat pada gambar 5.6.

INSTITUT TEKNOLOG
SEPULUH - NOPEMBER

[18] Bark Kosko, op.cit., hal 42.



Gambar 5.6

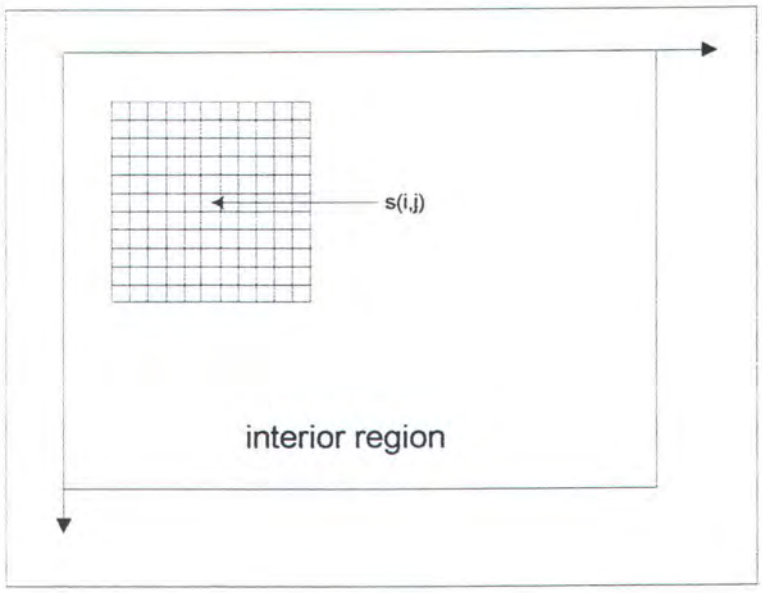
Diagram aliran data proses estimasi parameter GMRF kedua

5.4.2 Proses Intensitas dan Label

Seperti yang telah dijelaskan pada bagian 5.2, bahwa proses segmentasi dan klasifikasi memerlukan dua variabel Y_s dan L_s yang masing-masing berkaitan dengan nilai intensitas dan pe-label-an kelas setiap piksel. Dalam memodelkan suatu citra yang terdiri lebih dari satu tekstur, variabel Y_s dan L_s ini sangat dibutuhkan. Pada bagian ini, untuk proses distribusi nilai intensitas dari piksel digunakan model GMRF dengan memformulasikan ke dalam fungsi energi Gibbs.

5.4.2.1 Proses Intensitas

Pada bagian ini, model untuk proses intensitas adalah dengan Gaussian Markov Random Field. Disini, citra digambarkan sebagai sekumpulan window Ws berukuran k x k yang saling overlapping. Window-window Ws dalam citra ini berpusat pada setiap piksel di titik s, dimana $s \in \Omega_I$. Window k x k ini digerakkan sepanjang *interior region* mulai dari posisi kiri atas sampai kanan bawah untuk menghasilkan nilai intensitas pada setiap lokasi piksel, seperti ditunjukkan pada gambar 5.7.



Gambar 5.7

Window k x k berpusat di posisi s pada Ω_I dari citra

Besarnya ukuran window W_s adalah 11 x 11. Dalam setiap window, diasumsikan bahwa label tekstur adalah *homogeneous*[19], yang berarti bahwa semua piksel dalam window W_s mempunyai label tekstur yang sama dan model dari distribusi intensitas adalah *fourth-oder stationery GMRF*.

Jika Y_s^* adalah vector dua dimensi yang menyatakan '*zero mean intensity array in window W_s*'[20], maka dengan menggunakan formulasi fungsi energi Gibbs akan didapatkan '*joint probality density*' dalam window W_s, yang persamaannya adalah sebagai berikut :

---

[19] Bart Kosko, op.cit., hal. 42.
[20] Bart Kosko, op.cit., hal. 42.

$$P(Y_s^* = y_s^* \mid L_s = \ell) = \frac{e^{-U_1(y_s^* \mid L_s = \ell)}}{Z_1(\ell)}, \tag{5.4-6}$$

dimana $Z_1(\ell)$ adalah partition function dari fungsi Gibbs dan

$$U_1 = (y_s^* \mid L_s = \ell) = \frac{1}{2\sigma_\ell^2} \sum_{r \in W_s} \left\{ y_r^2 - \sum_{\tau \in N^* \mid r+\tau \notin W_s} \Theta_\tau^\ell y_r (y_{r+\tau} + y_{r-\tau}) \right\} \tag{5.4-7}$$

Proses komputasi dari persamaan (5.4-7) dilakukan pada setiap piksel dalam interior region dan diproses terhadap semua tekstur yang terdapat dalam citra. Untuk menghindari *efek boundary citra* , maka nilai U_1 pada posisi *boundary* diset dengan nilai U_1=0.

5.4.2.2 Proses Label

Model dari distribusi label pada setiap piksel didefinisikan sebagai *fourth-order discrete MRF*. Pada tugas akhir ini proses pengklasifikasian yang digunakan adalah dalam bentuk *unsupervised*, sehingga tidak diperlukan estimasi parameter tertentu untuk menentukan label-label pada setiap piksel yang berguna untuk mendapatkan gambaran secara kasar dari segmentasi citra. Daerah-daerah label hanya dikelompokkan oleh sebuah parameter tunggal, yaitu β, yang berfungsi untuk menentukan batasan antara region-region yang berbeda dalam citra. Dalam proses klasifikasi nantinya parameter β ini diperlukan dalam proses untuk

menentukan kelas suatu piksel dari citra. Untuk menentukan nilai paramater β yang cocok pada proses klasifikasi didapatkan dari cara *trial and error*[21]. Nilai β berada pada range antara 0.2 sampai dengan 3.

5.5 KLASIFIKASI TEKSTUR

Tahap terakhir dari perangkat lunak ini adalah melaksanakan proses klasifikasi setiap piksel ke dalam kelas-kelasnya. Pada bagian ini ada dua modul utama untuk melaksanakan proses klasifikasi, yaitu modul untuk mengecek kestabilan sistem jaringan serta modul untuk mencari nilai keaktifan suatu neuron yang paling maksimum.

Proses klasifikasi ini dilakukan untuk menentukan setiap piksel ke dalam kelas-kelas tertentu, sesuai dengan kriteria yang telah ditetapkan. Dan proses klasifikasi dari citra ini dilakukan setelah didapat parameter-parameter yang didapatkan dari hasil pemodelan citra tekstur yang telah diproses sebelumnya.

Metode yang digunakan untuk proses klasifikasi ini adalah dengan Deterministic Relaxation yang diimplementasikan ke dalam jaringan saraf buatan, dalam bentuk fungsi energi Hopfield. Pada metode ini untuk mendapatkan hasil yang maksimal sangat sensitif terhadap konfigurasi awal dari nilai parameter-parameter pada fungsi energi Hopfield di setiap piksel. Juga, pencapaian suatu kondisi yang stabil atau *konvergen* dalam suatu iterasi hanya bersifat lokal.

---

[21] Bart Kosko, op.cit., hal. 52.

5.5.1 Fungsi Energi

Fungsi energi yang merupakan minimalisasi dari jaringan didapatkan dari pemodelan tekstur seperti yang telah dijelaskan pada bagian sebelumnya. Dengan mengetahui tipe kelas dan jumlah tekstur pada citra serta mengetahui parameter untuk pemodelan citra teksturnya, maka persoalan klasifikasi tekstur ini akan dapat diformulasikan kedalam suatu fungsi minimalisasi energi. Pada fungsi energi, notasi $U_1(i,j,\ell) = U_1(Y_s^*, L_s = \ell) + \omega(\ell)$, dimana s=(i,j) menyatakan posisi piksel dalam citra. Parameter $U_1(i,j,\ell)$ pada jaringan ini berisi informasi yang berkaitan dengan intensitas dan nilai paramater lainnya dari piksel pada posisi $s=(i,j) \in \Omega$, seperti yang telah didefinisikan pada persamaan 5.4-7. Parameter $U_1(i,j,\ell)$ ini diproses pada setiap titik piksel serta setiap kelas ℓ dari citra. Jaringan terdiri dari *K* layer, dan setiap layer tersusun atas M x M array, dimana K adalah merupakan jumlah kelas tekstur pada citra input dan M merupakan ukuran dimensi dari citra input.

Elemen-elemen neuron dalam jaringan dinyatakan dalam bentuk *biner* dan disusun dalam format *(i,j, ℓ)*, notasi (i,j) atau s merupakan posisi piksel dalam citra dan notasi ℓ menyatakan tipe layer pada jaringan.

Suatu neuron (i,j, ℓ) disebut ON jika mempunyai output $V_{ij\ell} = 1$, yang menandakan bahwa suatu piksel pada posisi s=(i,j) dari citra input mempunyai kelas tekstur ℓ, dan untuk setiap piksel pada posisi s=(i,j) hanya satu $V_{ij\ell}$ yang mempunyai nilai 1 (satu), sedangkan yang lainnya

bernilai sama dengan 0 (nol). Hal ini berarti bahwa setiap piksel dari citra hanya mempunyai satu macam kelas, yaitu ℓ. Notasi $T_{ij\ell;i'j'\ell'}$ menyatakan kekuatan hubungan antar neuron-neuron (i,j, ℓ), dan (i',j', ℓ') pada jaringan, sedangkan notasi $I_{ij\ell}$ menyatakan *bias input*.

Secara umum fungsi energi Hopfield dalam persoalan klasifikasi tekstur ini adalah didefinisikan sebagai berikut :

$$E = \frac{1}{2}\sum_{i=1}^{M}\sum_{j=1}^{M}\sum_{\ell=1}^{K}\sum_{i'=1}^{M}\sum_{j'=1}^{M}\sum_{\ell'=1}^{K} T_{ij\ell;i'j'\ell'} V_{ij\ell} V_{i'j'\ell'} - \frac{1}{2}\sum_{i=1}^{M}\sum_{j=1}^{M}\sum_{\ell=1}^{K} I_{ij\ell} V_{ij\ell} \qquad (5.5\text{-}1)$$

Dengan mencocokkan pada fungsi energi Gibbs, maka persamaan diatas dapat ditulis sebagai :

$$E = \frac{1}{2}\sum_{i=1}^{M}\sum_{j=1}^{M}\sum_{\ell=1}^{K} U_1(i,j,\ell) V_{ij\ell} - \frac{\beta}{2}\sum_{i=1}^{M}\sum_{j=1}^{M}\sum_{\ell=1}^{K} \sum_{(i',j')\in N_{ij}} V_{i'j'\ell} V_{ij\ell} \qquad (5.5\text{-}2)$$

Pada persamaan (5.5-2), N_{ij} adalah merupakan anggota tetangga dari posisi (i,j) yang didefinisikan sebagai *fourth-order neighborhood* dari (i,j), (sama dengan N_{ij} seperti yang telah dijelaskan sebelumnya). Dari persamaan (5.5-2) secara implisit menyatakan bahwa setiap posisi piksel mempunyai sebuah label yang unik, artinya bahwa hanya ada satu neuron yang aktif dalam setiap kolom pada jaringan.

Dalam persamaan (5.5-1) dan (5.5-2), kita dapat mengidentifikasi parameter-parameter dalam jaringan sebagai berikut :

$$T_{ij\ell;i'j'\ell'} = \begin{cases} \beta & \text{if}(i',j') \in N_{ij} \quad \forall \ell = \ell' \\ 0 & \text{otherwise} \end{cases} \qquad (5.5\text{-}3)$$

sedangkan bias input dirumuskan sebagai berikut :

$$I_{ij\ell} = U_1(i, j, \ell) \qquad (5.5\text{-}4)$$

Persamaan (5.4-3) mengindikasikan bahwa nilai paramater $T_{ij\ell i'j'\ell'}$ pada semua $\ell = \ell'$ akan bernilai β jika piksel pada posisi (i',j') merupakan anggota tetangga dari piksel pada posisi (i,j), sebaliknya jika piksel di posisi (i',j') bukan anggota tetangga dari piksel pada posisi (i,j) maka $T_{ij\ell i'j'\ell'}$ bernilai 0.

Fungsi energi ini akan berguna untuk melakukan pengecekan apakah sistem jaringan sudah mencapai *konvergen*. Selama jaringan belum mencapai konvergen, proses klasifikasi akan terus dilakukan, sampai pada suatu iterasi dimana sistem jaringan sudah dinyatakan konvergen. Pada sistem ini secara prinsip kondisi konvergen hanya bersifat lokal, sehingga untuk menentukan nilai yang konvergen adalah dengan melihat nilai energi di setiap *stage* pada setiap iterasi, jika pada setiap stage nilai energinya tidak mengalami peningkatan atau tidak ada perubahan, berarti sistem sudah konvergen.

5.5.2 Algoritma Deterministic Relaxation

Pada metode ini menggambarkan bahwa nilai maksimum dari suatu neuron ditentukan oleh kelas-kelas dari piksel tetangganya serta nilai intensitas dari piksel tersebut. Kelas-kelas dari neuron ini nilai awalnya didapatkan dari proses *Maximazing* dari nilai intensitasnya, yaitu

dengan menjumlahkan nilai intensitas U1(.) ditambah dengan nilai $Z_1(\ell)$ yang didapatkan dengan cara *trial and error* , cara ini sering disebut sebagai *Maximum Likelihood Estimation (MLE)*[22].

Pada persamaan (5.5-3) dan (5.5-4), parameter-parameter dalam jaringan merupakan parameter dari model citra. Hubungan matriks pada sistem jaringan diatas adalah simetris dan bersifat '*no self-feedback*'. $T_{ij\ell;ijl} = 0$, untuk semua nilai i, j, ℓ atau $\forall\ i, j, \ell$. Jika $u_{ij\ell}$ merupakan *potential neuron* (i, j, ℓ), dengan asumsi bahwa ℓ merupakan nomor layer yang menyatakan kelas tekstur yang ke-ℓ maka persamaan $u_{ij\ell}$ dapat dinyatakan sebagai berikut :

$$u_{ijl} = \sum_{i'=1}^{M}\sum_{j'=1}^{M}\sum_{\ell'=1}^{K} T_{ij\ell;i'j'\ell'} V_{i'j'\ell'} + I_{ij\ell} \qquad (5.5\text{-}5)$$

Seperti yang telah dijelaskan sebelumnya bahwa hanya terdapat satu $V_{ij\ell}$ yang bernilai sama dengan satu sedangkan $V_{ij\ell}$ lainnya sama dengan 0. Maka untuk menentukan nilai $V_{ij\ell}$ digunakan suatu persamaan yaitu,

$$V_{ij\ell} = \begin{cases} 1 & \text{jika } u_{ij\ell} = \min\limits_{\ell'}\{u_{ij\ell'}\} \\ 0 & \text{otherwise} \end{cases} \qquad (5.5\text{-}6)$$

Metode *deterministic relaxation* seperti ini merupakan metode sederhana yang menggunakan sebuah sirkuit '*winner-takes-all*' pada setiap kolom, sehingga neuron yang menerima input secara maksimum akan bersifat ON, sedangkan yang lainnya akan bersifat OFF. Hal ini

[22] Ibid, hal. 478.

MILIK PERPUSTAKAAN
INSTITUT TEKNOLOGI
SEPULUH – NOPEMBER

dapat dilihat dari persamaan (5.4-5) dan (5.4-6) yang menyatakan bahwa untuk mencari suatu neuron (i, j, ℓ) yang ON yaitu dengan mencari nilai $u_{ij\ell}$ kepada masing-masing kelas ℓ. Dari nilai-nilai $u_{ij\ell}$ dari masing-masing kelas ℓ pada setiap piksel ini kemudian dipilih nilai $u_{ij\ell}$ yang paling maksimum, ini menandakan bahwa piksel tersebut mempunyai kelas ℓ.

Proses *updating* seperti ini akan memastikan bahwa pada setiap *stage* nilai fungsi energi akan selalu berkurang. Sebelum melakukan proses updating, diperlukan inisialisasi nilai awal $V_{ij\ell}$ pada setiap piksel. Nilai awal $V_{ij\ell}$ ini didapatkan dari mengestimasikan parameter-parameter yang telah diproses sebelumnya.

Pada proses updating ini, jika dalam suatu iterasi sistem jaringan belum stabil maka nilai $V_{ij\ell}$ lama yang terdapat pada setiap kolom dalam fungsi energi, akan diganti dengan nilai $V_{ij\ell}$ baru yang didapatkan dari proses dengan deterministic ini, yang kemudian dilanjutkan pada iterasi berikutnya sampai dicapai kondisi yang konvergen. Untuk mendapatkan hasil yang optimal secara umum biasanya kondisi konvergen akan dapat dicapai dalam 20 sampai dengan 30 iterasi.

5.6 STRUKTUR FILE DATABASE

Beberapa file database dibuat dalam sistem perangkat lunak ini. Tujuan pembuatan file database dalam perangkat lunak ini adalah selain data yang diolah cukup besar, sehingga memori internal komputer tidak

cukup untuk menampungnya, juga data perlu disimpan secara permanen kedalam *external storage* untuk keperluan pengolahan lebih lanjut.

Pada sistem perangkat lunak ini dibuat dua file database untuk keperluan penyimpanan data. Struktur data file database yang pertama adalah sebagai berikut,

```
Parameter_GMRF = record
                   File_Text_64    : string[10];
                   first_GMRF      : array[1..10] of real;
                   second_GMRF     : real;
                 end;
```

File database ini untuk menyimpan data-data paramater GMRF dari semua file citra. File ini terdiri dari 3 field dimana field pertama *File_Text_64* merupakan nama file citra tekstur yang tipenya adalah string dan file-file citra ini berukuran 64-kali-64 serta hanya mengandung 1 macam tekstur. Field kedua *first_GMRF* merupakan field untuk menyimpan data parameter GMRF pertama yang terdiri dari 10 elemen, yang ber*tipe*kan real. Kemudian field terakhir, *second_GMRF* untuk menyimpan data parameter GMRF kedua yang tipenya real.

Selanjutnya untuk menyimpan data nilai intensitas piksel dari citra maka dibuat suatu file database yang kedua, dimana struktur datanya adalah,

```
Data_Pixel = record
               File_Textures          : string[10];
               XPixel, YPixel         : byte;
               Intensity_Pix[tot_class] : real;
               Output_V[tot_class]    : byte;
               Energy_Pix             : real;
             end;
```

File_Textures pada struktur data diatas merupakan file citra yang akan diklasifikasikan, yang terdiri lebih dari 1 macam tekstur serta berukuran 128-kali-128 atau 256-kali-256. *Intensity_Pix[tot_class]* menyatakan nilai intensitas dari suatu piksel yang posisi koordinatnya adalah (*XPixel, YPixel*) dan *tot_class* menyatakan jumlah kelas tekstur pada citra. Selanjutnya field *Output_V[tot_class]* berfungsi untuk menentukan kelas tekstur dari piksel(*XPixel, YPixel*), field ini bernilai 0 atau 1 seperti yang telah dijelaskan pada bab V. selanjutnya field yang terakhir, *Energy_Pix* menyatakan nilai energi Hopfield dari piksel(*XPixel, YPixel*).

BAB VI

UJI COBA DAN EVALUASI PERANGKAT LUNAK

BAB VI

UJI COBA DAN EVALUASI PERANGKAT LUNAK

Dalam bab ini akan dibahas tentang hasil uji coba perangkat lunak dan evaluasi algoritma yang digunakan pada perangkat lunak ini. Juga akan diperlihatkan jalannya/tahap-tahap proses dalam sistem perangkat lunak klasifikasi tekstur ini.

Disini juga akan ditampilkan beberapa output yang dihasilkan dari sistem perangkat lunak. Sehingga dari analisa beberapa output ini nantinya dapat ditarik kesimpulan dari jalannya program yang ada.

Langkah-langkah untuk sistem perangkat lunak ini, antara lain :

Langkah 1: Memasukkan nama *file* citra yang akan diproses untuk mendapatkan nilai-nilai parameternya. Langkah ini dapat dilakukan melalui menu-menu yang telah disediakan, dan proses ini dilakukan terhadap semua file citra tekstur.

Langkah 2: Setelah me-*load* file citra seperti pada langkah 1 diatas, langkah berikutnya adalah mencari nilai paramater GMRF dari file citra tersebut, yaitu dengan memilih menu "*MRF Model - Estimate GMRF*". Hasil proses ini akan disimpan dalam suatu file database.

Langkah 3: Masukkan file citra input yang akan dilakukan proses klasifikasi. File-file citra ini, sebelumnya telah dilakukan proses editing untuk mendapatkan beberapa model citra yang terdiri lebih dari satu tekstur.

Langkah 4: Proses selanjutnya adalah mencari nilai intensitas setiap piksel dari file citra input tadi terhadap masing-masing kelas tekstur. Proses ini dapat dilakukan pada menu "*MRF Model-intensity*" yang didalamnya terdapat beberapa submenu. Sebelum melakukan proses *Intensity*, harus diinputkan terlebih dahulu jumlah kelas tekstur serta tipe kelasnya yang terdapat pada file citra. Nilai-nilai intensitas setiap piksel yang dihasilkan dari proses ini akan disimpan kedalam file database.

Langkah 5: Memilih nilai '*bias term*' dengan cara menginputkan dari keyboard melalui menu '*MRF model - Bias input*'. Untuk mendapatkan nilai *bias term* yang valid, dilakukan dengan cara *trial and error*.

Langkah 6: Pada tahap ini dilakukan proses klasifikasi dengan nilai-nilai parameter yang telah didapatkan dari proses sebelumnya. Proses ini dapat dilakukan pada menu '*Classification-process*'. Jika nantinya hasil dari proses ini kurang baik proses dapat diulangi dengan mencari nilai-nilai *bias-term* yang valid seperti pada langkah 5, dan proses klasifikasi

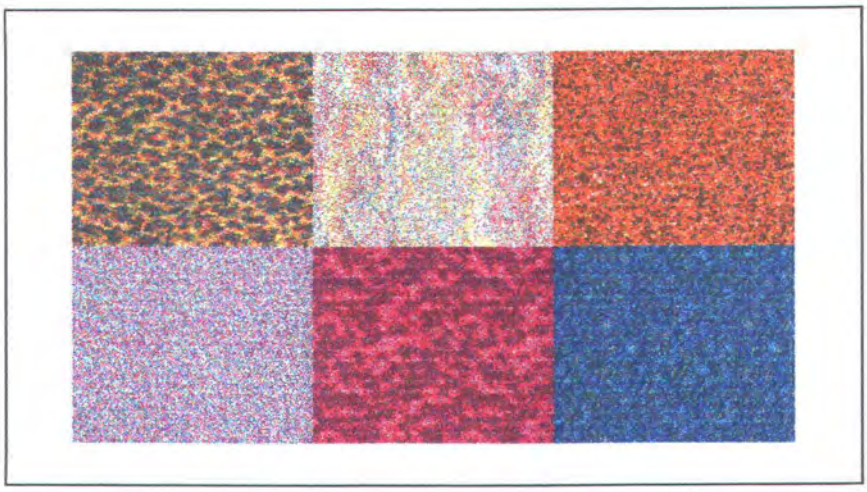
dapat dilakukan lagi sampai didapatkan hasil yang memuaskan.

Langkah 6: Menampilkan file citra yang merupakan hasil dari proses klasifikasi.

6.1 UJI COBA DAN EVALUASI ALGORITMA PEMODELAN TEKSTUR

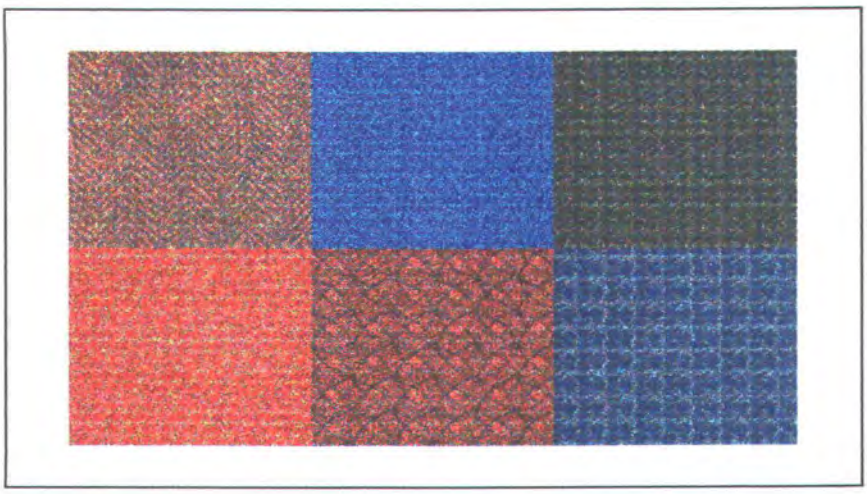
Seperti yang telah dibahas pada bab 5, bahwa masing-masing citra tekstur dicari nilai parameter GMRF-nya. Citra tekstur yang diolah dalam hal ini adalah berukuran 64-kali-64 dalam bentuk gray level 8 bit. Parameter GMRF yang dihasilkan adalah Θ_ℓ dan σ_ℓ, yang menggunakan model *fourth-order* dalam strukur GMRF.

Pada Tabel 6.1 dan tabel 6.2 dapat dilihat nilai parameter GMRF dari beberapa macam citra tekstur. Tabel 6.1 adalah nilai parameter GMRF dari citra tekstur yang berbentuk '*terstruktur*', sedangkan pada tabel 6.2 adalah nilai parameter GMRF yang berbentuk '*tidak terstruktur*'. Gambar 6.1 dan 6.2 adalah beberapa contoh model citra tekstur yang digunakan dalam sistem perangkat lunak ini. Dari hasil komputasi seperti yang terdapat pada tabel 6.1 dapat dilihat bahwa nilai dari hasil penjumlahan parameter Θ_ℓ yang terdiri dari 10 elemen Θ_ℓ rata-rata adalah sama dengan 0,49.



Gambar 6.1 Model citra tekstur "tidak terstruktur"

bagian atas (dari sebelah kiri ke kanan): tekstur 1, tekstur 2, tekstur 3
bagian bawah (dari kiri ke kanan) : tekstur 4, tekstur 5, tekstur 6



Gambar 6.2 Model Citra tekstur "terstruktur"

bagian atas (dari sebelah kiri ke kanan): tekstur 7, tekstur 8, tekstur 9
bagian bawah (dari kiri ke kanan) : tekstur A, tekstur B, tekstur C

Tabel 6.1

Nilai-nilai parameter GMRF dari citra tekstur seperti pada gambar 6.1

	tekstur 1	***tekstur 2***	***tekstur 3***	***tekstur 4***	***tekstur 5***	***tekstur 6***
Θ_1	0.4419	0.0138	0.0773	-0.0084	0.3890	-0.1325
Θ_2	0.1273	0.1470	0.0899	0.2113	0.3565	0.2302
Θ_3	0.0251	0.0888	0.0486	0.0676	-0.1455	0.2020
Θ_4	0.0123	0.0506	0.0548	-0.0472	0.0375	0.0557
Θ_5	0.1283	0.0018	0.0344	0.0245	-0.0589	-0.0226
Θ_6	-0.0119	0.0835	0.0325	0.0688	-0.0953	0.0281
Θ_7	-0.0076	0.0483	0.0245	0.0399	0.0230	0.0254
Θ_8	-0.0027	0.0698	0.0330	0.0572	0.0067	0.0491
Θ_9	0.0268	0.0123	0.0555	0.0114	0.0545	0.0226
Θ_{10}	0.0152	-0.0162	0.0480	0.0747	-0.0680	0.0415
σ^2	934.68	396.36	1170.31	693.23	128.67	320.19

Tabel 6.1

Nilai-nilai parameter GMRF dari citra tekstur seperti pada gambar 6.2

	tekstur 7	*tekstur 8*	*tekstur 9*	*tekstur A*	*tekstur B*	*tekstur C*
Θ_1	-0.1423	0.0588	0.0653	0.1588	-0.0248	0.1695
Θ_2	0.1091	0.0582	0.1691	0.1693	0.0395	0.1608
Θ_3	0.0387	0.0352	0.3686	0.1842	0.2515	0.1064
Θ_4	0.1156	0.0372	-0.0522	0.0399	0.1924	0.0606
Θ_5	0.1434	0.0854	0.1710	0.0179	-0.1538	0.0022
Θ_6	-0.2067	0.1381	0.0000	-0.0432	-0.0270	0.0821
Θ_7	0.1156	-0.0267	-0.1544	0.0057	0.0279	-0.0462
Θ_8	0.0661	0.1059	0.0717	0.0124	0.0880	0.0483
Θ_9	0.1350	0.0190	-0.0887	-0.0555	0.0411	-0.0327
Θ_{10}	0.1246	-0.0114	-0.0557	0.0215	0.0652	-0.0522
σ^2	418.20	238.89	653,64	385.54	521.37	693.23

6.2 UJI COBA DAN EVALUASI PROSES INTENSITAS

Seperti telah dijelaskan pada bab 5, bahwa sebelum proses klasifikasi dilakukan, maka sebelumnya harus dilakukan proses *intensity*. Proses Intensity ini adalah untuk mencari nilai intensitas setiap piksel dari

citra input terhadap masing-masing kelas tekstur yang terdapat dalam citra input tersebut.

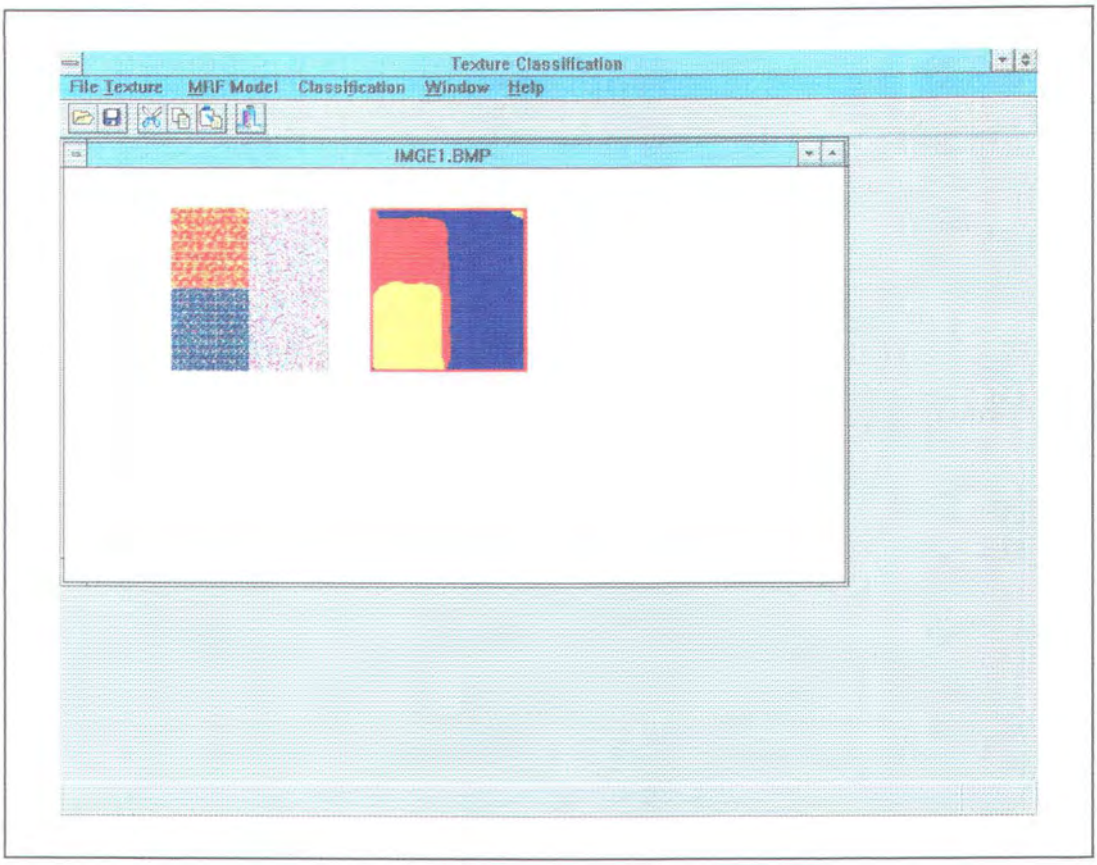
Nilai intensitas suatu piksel didapatkan dari *local mean* dalam window berukuran 11-kali-11 yang diproses terhadap semua piksel dari citra input, yang terdapat pada *interior region*. Untuk menghindari dari efek *boundary* maka nilai intensitas piksel, $U_1(.)$ pada *boundary* diberi nilai $U_1(.)=0$. Karena nilai intensitas suatu piksel ditekankan pada distribusi nilai gray-level dari semua tetangga dari piksel tersebut, maka piksel-piksel yang posisinya terdapat/berdekatan dengan daerah tepi citra, nilai intensitas yang dihasilkan kurang sempurna. Hal ini dikarenakan distribusi gray level dari tetangga piksel tersebut tidak lengkap. Demikian juga dengan piksel-piksel yang posisinya berdekatan dengan daerah-daerah yang mempunyai kelas tekstur yang berbeda dengan piksel tersebut, hal ini juga akan dapat memberikan nilai intensitas piksel yang kurang sempurna. Kurang sempurnanya nilai intensitas piksel tersebut, karena dalam proses distribusi gray level tetangga ada beberapa piksel yang merupakan piksel dari kelas tekstur yang lain, sehingga secara otomatis '*kekuatan*' intensitas piksel terhadap kelasnya akan sedikit berkurang dengan adanya distribusi dari piksel lain yang mempunyai kelas tekstur yang berbeda dengan piksel tersebut.

Untuk meminimalisasi kesalahan dari distrbusi nilai intensitas seperti pada keterangan yang telah dijelaskan diatas, maka pemilihan besarnya ukuran window sangatlah penting. Pada sistem perangkat lunak

ini, penulis telah melakukan beberapa kali percobaan untuk mencari besarnya ukuran window yang tepat. Dari beberapa kali percobaan ternyata ukuran window 11-kali-11 yang digunakan dalam proses ini, akan memberikan hasil yang paling optimal. Jika ukuran window diperkecil maka hal ini akan mengurangi kekuatan intensitas dari suatu piksel. Sebaliknya jika ukuran window diperbesar maka akan memperbesar faktor '*interferensi*' dari piksel yang mempunyai kelas tekstur yang berbeda. Dengan mempertimbangkan kedua hal diatas serta hasil percobaan yang telah dilakukan, penulis akhirnya memutuskan ukuran window yang digunakan dalam algoritma ini adalah 11-kali-11.

6.3 UJI COBA DAN EVALUASI PROSES KLASIFIKASI

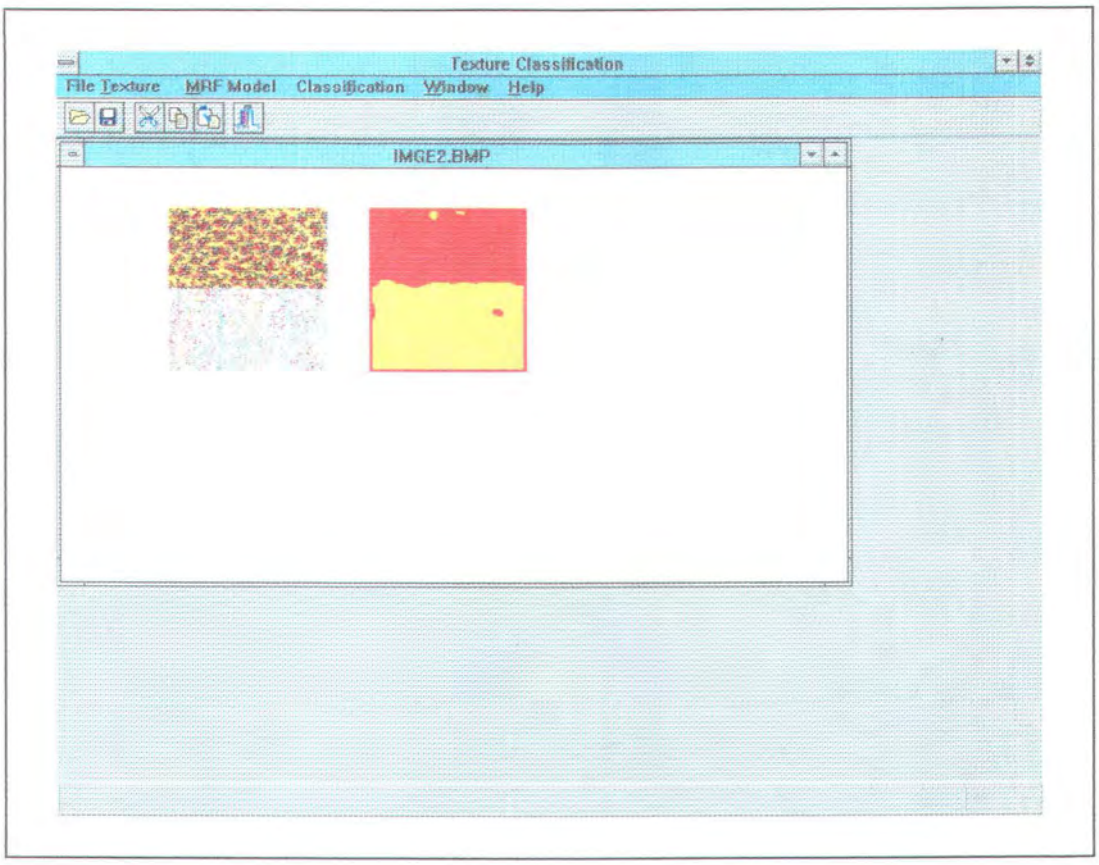
Tahapan yang paling akhir dalam sistem perangkat lunak ini adalah proses klasifikasi. Proses klasifikasi ini dilakukan terhadap semua piksel dari citra input. Proses '*updating*' piksel pada citra input dilakukan secara *simultaneously* dan bersifat random. Proses *updating* ini akan dilakukan kembali jika belum dicapai kondisi yang konvergen.



Gambar 6.3

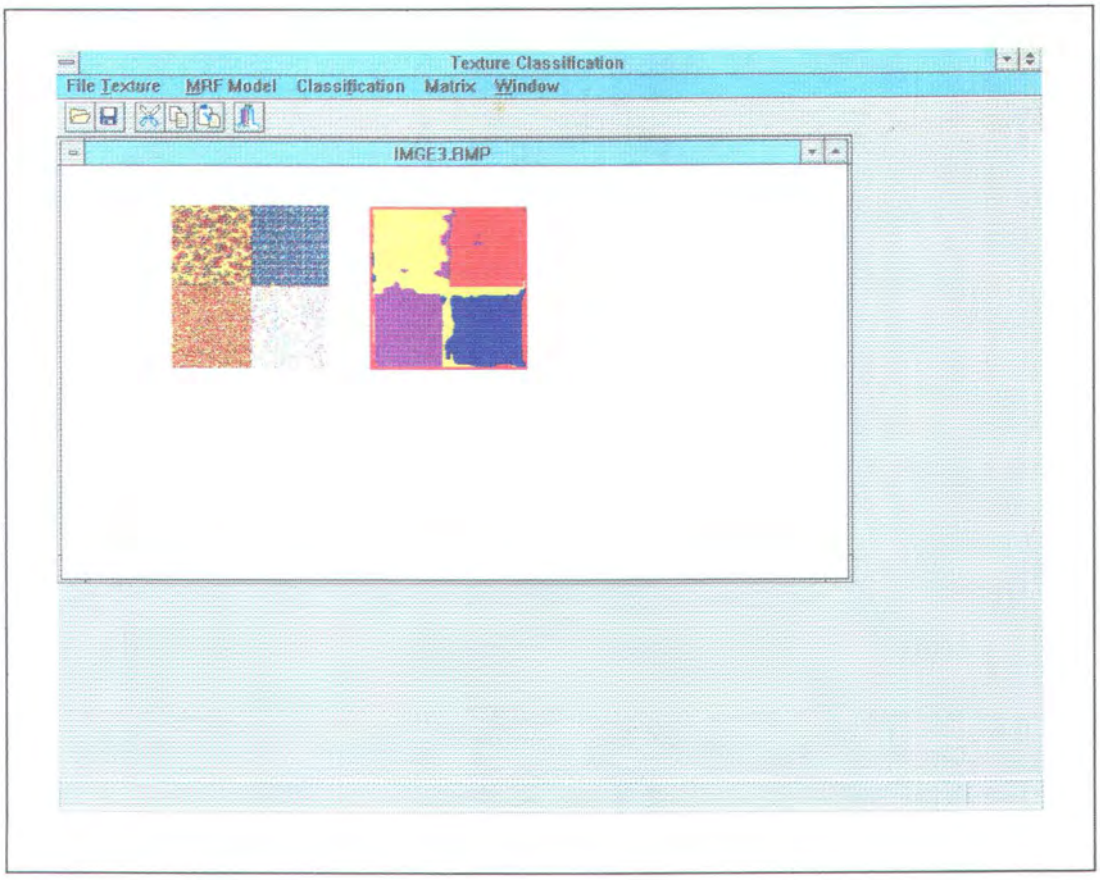
Citra ouput hasil proses klasifikasi dengan 3 kelas

MILIK PERPUSTAKAAN
INSTITUT TEKNOLOGI
SEPULUH - NOPEMBER



Gambar 6.4

Citra ouput hasil proses klasifikasi dengan 2 kelas



Gambar 6.5

Citra ouput hasil proses klasifikasi dengan 4 kelas

Gambar 6.3, 6.4 dan 6.5 memperlihatkan citra input dan citra yang merupakan hasil dari proses klasifikasi. Untuk mendapatkan hasil yang optimal dalam proses klasifikasi ini, besarnya nilai *bias term*, $\omega(\ell)$, yang diinputkan akan sangat berpengaruh, dan untuk mendapatkan nilai *bias term* yang tepat, cara satu-satunya adalah dengan melakukan percobaan yang berulang-ulang dengan mempertimbangkan nilai dari parameter GMRF-nya. Nilai *bias term* ini dapat bernilai positif atau negatif serta berlainan untuk masing-masing kelas tekstur. Nilai *bias term* ini sangat bergantung dengan jumlah kelas dalam citra tekstur input tersebut, serta bentuk teksturnya.

Pada gambar 6.3 citra input terdiri dari 3 kelas tekstur yaitu tekstur A (pada daerah kiri atas), tekstur C (pada daerah kiri bawah) dan tekstur 4 (pada sisi kanan) dengan nilai *bias term* berturut-turut adalah 7.8, 9.1 dan 1.6. Gambar 6.4 citra input terdiri dari 2 kelas yaitu tekstur 1 (pada bagian atas) dan tekstur 2 (pada bagian bawah) dengan *nilai bias* term masing-masing adalah 9.7 dan 4. Sedangkan Gambar 6.5 citra input terdiri dari 4 kelas yaitu tekstur 2 (pada bagian kiri atas), tekstur 3 (pada bagian kanan bawah), tekstur 7 (pada bagian kiri bawah) dan tekstur C (pada bagian kanan atas) yang berturut-turut nilai bias termnya adalah 6, 2.9, 0.2 dan 5.5.

Dari hasil ke-3 percobaan seperti terlihat pada gambar diatas dapat dilihat bahwa terdapat beberapa piksel yang mengalami ***misclassification***. Kesalahan terjadi biasanya terdapat pada beberapa

titik piksel yang posisinya berdekatan dengan daerah *boundary* citra atau berdekatan dengan batas tepi dari daerah kelas tekstur yang lain. Hal ini dikarenakan kekuatan neuron pada titik piksel tersebut menjadi lemah karena letaknya yang berdekatan dengan tepi citra. Juga kekuatan neuron akan menjadi berkurang karena adanya *interferensi* dari beberapa piksel yang berdekatan yang mempunyai kelas tekstur yang berbeda. Kondisi seperti diatas akan mempengaruhi ke-*valid*-an dari nilai kelas tekstur yang dihasilkan. Hal ini seperti yang telah dijelaskan pada keterangan sebelumnya yaitu karena pendekatan yang dilakukan pada algoritma ini adalah berdasarkan distribusi dari nilai-nilai gray level tetangganya yang terdapat dalam window dan dimodelkan sebagai *fourth-order neighborhood*. Dari pendekatan ini maka kekuatan neuron dari suatu titik piksel akan mempengaruhi kekuatan neuron dari piksel tetangganya.

BAB VII

PENUTUP

BAB VII

PENUTUP

7.1 KESIMPULAN

Dari tugas akhir ini setelah membahas secara luas tentang persoalan klasifikasi tekstur beserta algoritma untuk penyelesaiannya, sekaligus perancangan dan pembuatan perangkat lunak untuk klasifikasi tektur, beberapa hal yang dapat diambil kesimpulan adalah sebagai berikut :

1. Metode Markov Random Field yang digunakan untuk memodelkan citra tekstur masih mempunyai banyak kelemahan antara lain dengan menggunakan struktur hirarki Gaussian-MRF, maka untuk mendapatkan nilai intensitas yang '*baik*' pada suatu piksel sesuai dengan model/struktur teksturnya maka letak piksel tersebut harus berada minimum jaraknya sama dengan ukuran lebar/tinggi window yang digunakan. Hal ini karena nilai intensitas suatu piksel sangat bergantung terhadap distribusi gray-level semua tetangganya.
2. Pada proses klasifikasi secara prinsip kondisi konvergen pada sistem jaringan hanya bersifat lokal, yang berarti bahwa

nilai ke-konvergennya dilihat serta ditentukan oleh nilai konvergen tiap-tiap piksel dari citra input.

3. Semakin besar ukuran citra input dalam sistem perangkat lunak ini, maka semakin baik pula citra output yang dihasilkan, meskipun untuk itu proses komputasinya menjadi bertambah lama.
4. Prosentage terbesar kesalahan dalam menentukan kelas suatu piksel adalah pada piksel yang letaknya berdekatan dengan garis batas citra dan piksel-piksel yang letaknya agak berdekatan dengan daerah-daerah yang mempunyai kelas yang berbeda.
5. Hal mendasar yang menjadi kelemahan pada sistem perangkat lunak klasifikasi tekstur dengan menggunakan jaringan saraf ini adalah adanya variabel *bias term* yang nilainya untuk tiap-tiap citra input adalah berbeda-beda, serta untuk mendapatkan nilai *bias term* yang '*valid*' hanya dapat dilakukan dengan cara *trial and error*.

6.2 SARAN

Dalam pembuatan Tugas Akhir ini masih terdapat banyak kekurangan yang perlu diperbaiki untuk pengembangan lebih lanjut perangkat lunak klasifikasi tektur ini, antara lain :

MILIK PERPUS
INSTITUT TEKN
SEPULUH – NOP

1. Sistem perangkat lunak ini dapat dikembangkan menjadi perangkat lunak yang mampu mengolah citra sampai dengan 16 juta warna.
2. Agar mendapatkan hasil yang lebih baik dapat digunakan sistem jaringan saraf buatan dengan menggunakan metode stochastic untuk proses learningnya.

DAFTAR PUSTAKA

DAFTAR PUSTAKA

1. James A. Freeman, David M. Sakapura, ***Neural Network 'Algorithms, Applications and Programming Techniques'***, Addison-Wesley Publishing Company, Inc,. 1991.
2. Laurene Fausett, ***Fundamental Of Neural Network***, Prentice Hall International, Inc,. 1994.
3. Bart Kosko, ***Neural Network For Signal Processing***, Prentice Hall International Inc., 1992.
4. Gonzales Rafael C., ***Digital Image Processing***, Addison-Wesley Publishing Company, Inc., 1987.
5. George R. Cross and Anil K. Jain, ***IEEE Transcations On Pattern Analysis And Machine Intelligence***, Vol. PAMI-5, No. 1, 1983.
6. B.S. Manjunath and R. Chellappa, ***IEEE Transcations On Pattern Analysis And Machine Intelligence***, Vol. 13, 1991.