BAB II

TEORI PENUNJANG

2.1 Aksara Jepang Katakana

Pada umumnya tulisan Jepang dibagi menjadi tiga jenis tulisan diantaranya adalah *hiragana, katakana,* dan *kanji*. Perberdaan tulisan tersebut memiliki fungsi masing masing seperti digunakan dalam serapan bahasa asing, nama orang, kata penghubung, dan kata baku. Salah satunya adalah yang akan diangkat dalam tugas akhir ini yaitu tulisan *katakana.*

Katakana untuk melambangkan suku kata tunggal, tetapi mempunyai fungsi yang berbeda dengan huruf *hiragana*. Huruf *katakana* selain digunakan untuk menulis kata-kata yang berasal dari bahasa asing, juga digunakan untuk penekanan suatu kata yang berasal dari Jepang asli. *Katakana* bentuk hurufnya terkesan kaku, karena setiap coretannya bersudut tajam sehingga katakana disebut sebagai huruf laki-laki. Huruf *katakana* berjumlah 48.

ア a	イ i	ウ u	エ e	オ o
カ ka	キ ki	ク ku	ケ ke	コ ko
サ sa	シ shi	ス su	セ se	ソ so
タ ta	チ chi	ツ tsu	テ te	ト to
ナ na	ニ ni	ヌ nu	ネ ne	ノ no
ハ ha	ヒ hi	フ fu	ヘ he	ホ ho
マ ma	ミ mi	ム mu	メ me	モ mo
ヤ ya		ユ yu		ヨ yo
ラ ra	リ ri	ル ru	レ re	ロ ro
ワ wa				ヲ (w)o
ン n				

Gambar 2.1 Huruf *Katakana*

Fungsi huruf *katakana* adalah :

- Menuliskan kata-kata yang berasal dari bahasa asing
- Menuliskan nama orang dan tempat asing
- Menuliskan telegram

2.2 Definisi Citra

Secara harafiah citra (*image*) adalah gambar pada bidang dwimatra (dua dimensi). Gambar 2.2 (a) adalah citra seorang gadis model yang bernama Lena saat masih muda, dan Gambar 2.2 (b) adalah citra kapal di sebuah pelabuhan. Ditinjau dari sudut pandang matematis, citra merupakan fungsi menerus (*continue*) dari intensitas cahaya pada bidang dwimatra. Sumber cahaya menerangi objek, objek memantulkan kembali sebagian dari berkas cahaya tersebut. Pantulan cahaya ini ditangkap oleh alat-alat optik, misalnya mata pada manusia, kamera, pemindai (*scanner*), dan sebagainya, sehingga bayangan objek yang disebut citra tersebut terekam.

Citra sebagai keluaran dari suatu sistem perekaman data dapat bersifat:

1. Optik berupa foto
2. Analog berupa sinyal video seperti gambar pada monitor televisi
3. Digital yang dapat langsung disimpan pada suatu pita magnetik

(a) (b)

Gambar 2.2 (a) Contoh bentuk citra Lena, (b) Contoh bentuk citra kapal

Pengolahan citra adalah pemrosesan citra, khususnya dengan menggunakan komputer, menjadi citra yang kualitasnya lebih baik. Sebagai contoh citra burung nuri pada Gambar 2.3 (a) sebelah kiri tampak gelap lalu dengan operasi pengolahan citra kontrasnya diperbaiki sehingga menjadi lebih terang dan tajam, seperti yang diperlihatkan pada Gambar 2.3 (b).

(a) (b)

Gambar 2.3 (a) Citra awal, (b) Citra yang telah diperbaiki pengolahan citra

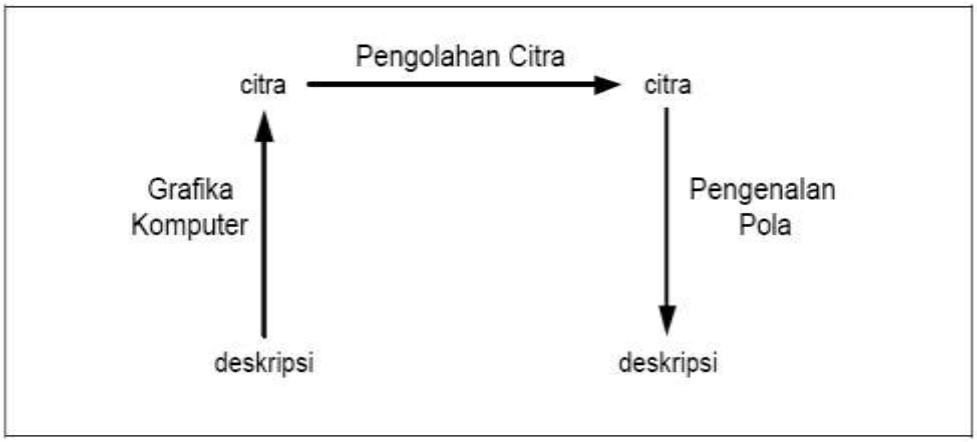
Umumnya, operasi-operasi pada pengolahan citra diterapkan pada citra jika:

1. Perbaikan atau memodifikasi citra perlu dilakukan untuk meningkatkan kualitas penampakan atau untuk menonjolkan beberapa aspek informasi yang terkandung dalam citra.
2. Elemen dalam citra perlu dikelompokan, dicocokan, atau diukur.
3. Sebagian citra perlu digabung dengan bagian citra yang lain.

Pada bidang komputer, sebenarnya ada tiga bidang studi yang berkaitan dengan data citra, namun tujuan ketiganya berbeda, yaitu :

1. Grafika Komputer (*Computer Graphics*).
2. Pengolahan Citra (*Image Processing*).
3. Pengenalan Pola (*Pattern Recognition/Image Interpretation*).

Hubungan antar ketiga bidang tersebut ditunjukan pada Gambar 2.4



Gambar 2.4 Tiga bidang studi yang berhubungan dengan citra

Grafika Komputer bertujuan untuk menghasilkan citra dengan primitif-primitif geometri seperti garis, lingkaran, dan sebagainya. Primitif-primitif geometri tersebut memerlukan data deskriptif untuk melukis elemen-elemen gambar. Contoh data deskriptif adalah koordinat titik, panjang garis, jari-jari lingkaran, tebal garis, warna

dan sebagainya. Grafika komputer memainkan peran penting dalam visualisasi dan *virtual reality*.

Gambar 2.5 Alur Grafika Komputer

Pengolahan citra bertujuan untuk memperbaiki kualitas citra agar mudah diinteparsi oleh manusia atau mesin. Teknik-teknik pengolahan citra mentransformasikan citra menjadi citra yang lain. Jadi, masukannya adalah citra begitu juga keluaranya adalah berupa citra, namun citra keluaran mempunyai kualitas lebih baik daripada citra masukan. Termasuk ke dalam bidang ini juga adalah penempatan citra (*image compression*).

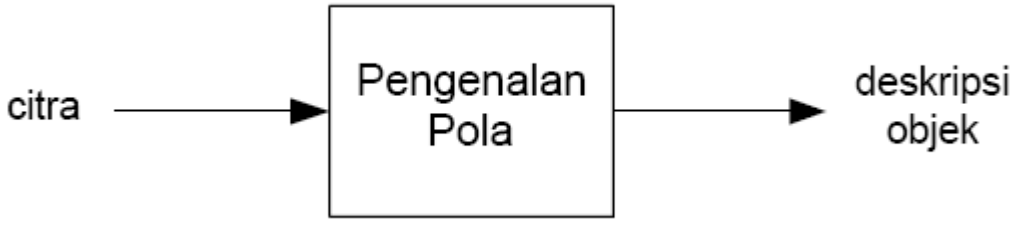
Gambar 2.6 Alur Pengolahan Citra

Pengubahan kontras citra seperti pada Gambar 2.6 adalah contoh operasi pengolahan citra. Contoh operasi pengolahan citra lainnya adalah penghilangan derau (*noise*) pada citra Lena Gambar 2.7 Citra Lena sebelah kiri mengandung derau berupa bintik-bintik putih (derau). Dengan operasi penapisan (*filtering*), derau pada citra masukan ini dapat dikurangi sehingga dihasilkan citra Lena yang kualitasnya lebih baik.

(a) (b)

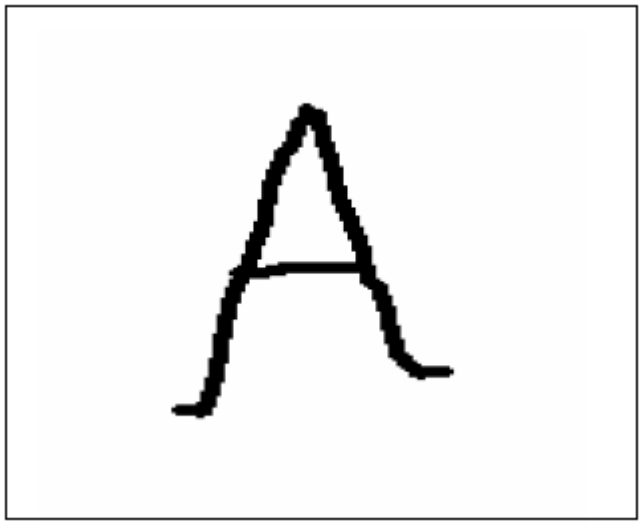
Gambar 2.7 (a) Citra Lena yang mengandung derau (b) hasil dari operasi penapisan derau

Pengenalan pola mengelompokan data numerik dan simbolik secara otomatis oleh mesin. Tujuan pengelompokan adalah untuk mengenali suatu objek di dalam citra. Manusia dapat mengenali objek yang dilihatnya karena otak manusia telah belajar mengklasifikasi objek-objek di alam sehingga mampu membedakan suatu objek dengan objek lainnya. Kemampuan sistem visual manusia inilah yang dicoba ditiru oleh mesin. Komputer menerima masukan berupa citra objek yang akan diidentifikasi, memproses citra tersebut, dan memberikan keluaran berupa deskripsi objek dalam citra.



Gambar 2.8 Alur pengenalan pola

Contoh pengenalan pola misalnya citra pada Gambar 2.9 adalah tulisan tangan yang digunakan sebagai data masukan untuk mengenali karakter “A”. Dengan menggunakan suatu algoritma pengenalan pola, diharapkan komputer dapat mengenali bahwa karakter tersebut adalah ‘A’.



Gambar 2.9 Citra karakter ‘A’ yang digunakan sebagai masukan untuk pengenalan huruf

2.4 Aplikasi Pengolahan Citra dan Pengenalan Pola

Pengolahan citra mempunyai aplikasi yang sangat luas dalam berbagai bidang kehidupan. Di bawah ini disebutkan beberapa aplikasi dalam beberapa bidang :

1. Bidang perdagangan seperti pembacaan kode batang (*bar code*) yang tertera pada barang, mengenali huruf atau angka pada suatu formulir secara otomatis
2. Bidang militer seperti mengenali sasaran peluru kendali melalui sensor visual, mengidentifikasi jenis pesawat musuh.
3. Bidang kedokteran seperti pengolahan citra sinar X untuk mammografi, NMR (*Nuclear Magnetic Resonance*), mendeteksi kelamin tubuh dari sinar X, rekonstruksi foto janin hasil USG

4. Bidang biologi seperti pengenalan jenis kromosom melalui gambar mikroskopik
5. Komunikasi data seperti pemampatan citra yang ditransmisi
6. Hiburan seperti pemampatan video (MPEG)
7. Robotika seperti visually-*guided autonomous navigation*
8. Pemetaan seperti klasifikasi penggunaan tanah melalui foto udara
9. Geologi seperti mengenali jenis batu-batuan melalui foto udara
10. Hukum seperti pengenalan sidik jari, pengenalan foto narapidana

2.5 Citra Digital

Sebuah citra digital terdiri dari sejumlah elemen yang berhingga, dimana masing masing mempunyai lokasi dan nilai tertentu. Elemen-elemen ini disebut sebagai *picture element, image element, pels* atau *pixels*. Bidang digital *image processing* meliputi pengolahan *digital image* dari suatu komputer digital. Gambar dihasilkan dari seluruh *spectrum* elektromagnetik mulai dari gamma sampai gelombang radio.

Ada tiga tipe dalam pengolahan citra :

- *Low-level process*

  Low-level process meliputi operasi dasar seperti *image processing* :

  - ✓ *Reduce noise*
  - ✓ *Contrast enhancement*
  - ✓ *Image sharpening*

  Pada level ini baik input maupun output adalah berupa gambar.

- *Mid-level process*

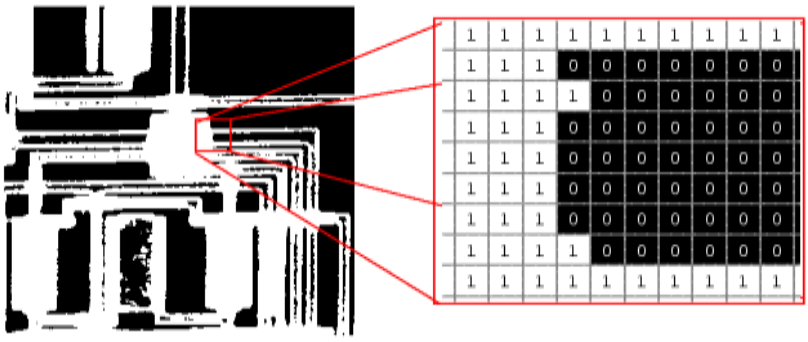
Mid-level processing meliputi segmentasi (membagi sebuah gambar dalam region atau object). Mendeskripsikan objek tersebut untuk direduksi dalam bentuk yang diinginkan dan klasifikasi (*recognition*) dari objek tersebut. *Input* dari proses ini berupa gambar, dan *output*-nya berupa atribut yang diambil dari gambar tersebut (misal : *edge, counters* dan identitas dari objek tertentu)

- *High-level process*

  High-level processing meliputi pemberian arti dari suatu rangkaian objek-objek yang dikenali dan akhirnya menampilkan fungsi-fungsi kognitif secara normal sehubungan dengan penglihatan.

2.5.1 Citra Biner

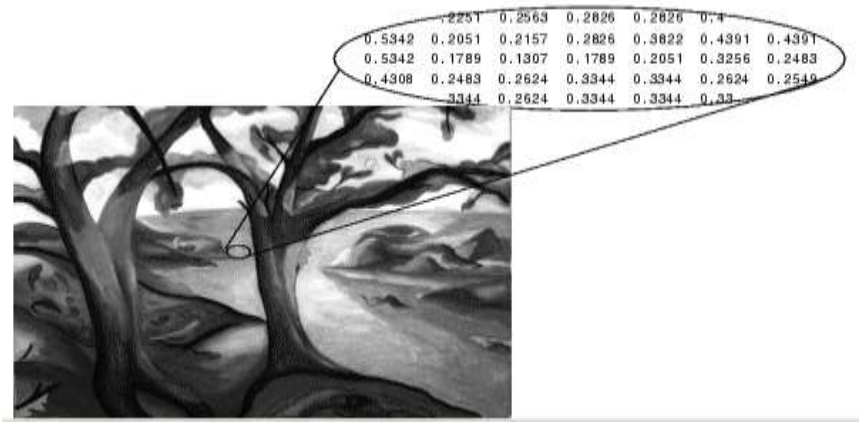
Citra biner adalah citra yang hanya mempunyai dua nilai derajat keabuan yaitu hitam dan putih. Piksel-piksel objek bernilai satu dan piksel-piksel latar belakang bernilai nol. Dimana setiap piksel adalah hitam atau putih. Karena hanya ada dua kemingkinan nilai pada setiap piksel, maka yang diperlukan hanya satu bit per piksel. Citra seperti ini sangat efisien untuk penyimpanan. Citra seperti ini bisa dimanfaatkan untuk representasi biner dari teks, tanda tangan, sidik jari, atau rancangan arsitektur.



Gambar 2.10 Citra Biner

2.5.2 Citra Abu-Abu

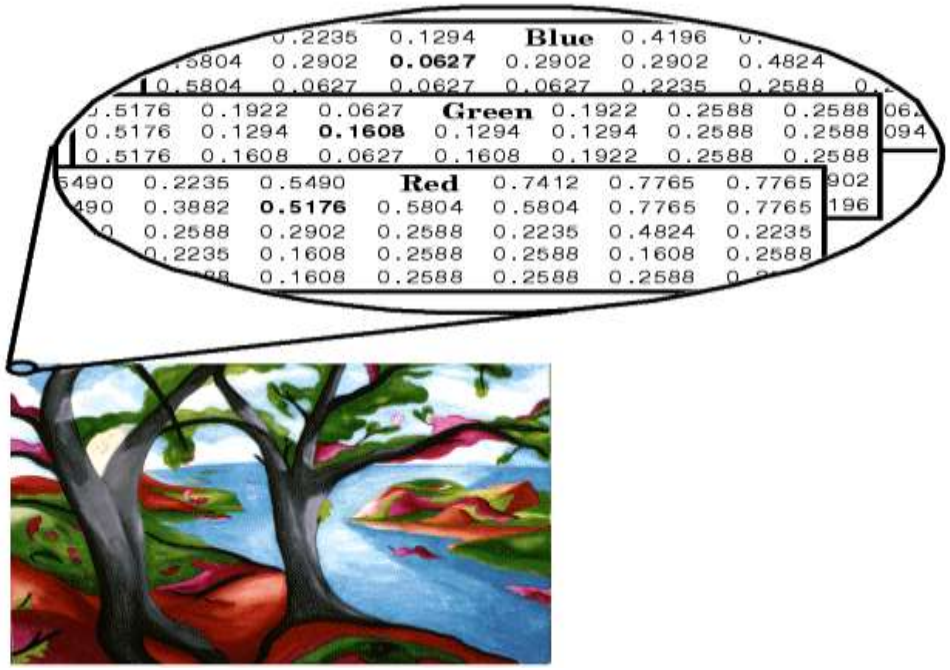
Setiap piksel merupakan bayagan abu-abu, yang memiliki nilai intensitas 0 (hitam) sampai 255 (putih). Rentang ini berarti bahwa setiap piksel dapat direpresentasikan oleh delapan bit, atau satu *byte*. Citra abu-abu ini bisa digunakan untuk merepresentasikan citra medis (sinar-X), tulisan atau buku, dan lainnya. Citra abu-abu dengan 256 level abu-abu dapat digunakan untuk mengenali kebanyakan objek.



Gambar 2.11 Citra Abu-abu

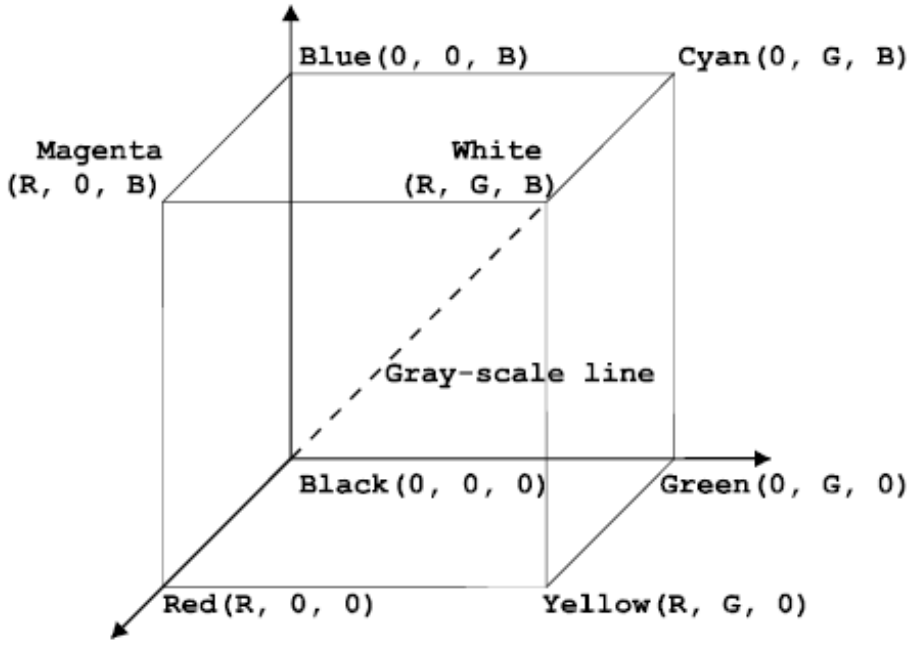
2.5.3 Citra warna atau *Red, Green, Blue* (RGB)

Pada citra warna atau RGB setiap piksel memiliki suatu warna khusus. Warna tersebut dideskripsikan oleh jumlah warna merah (R,*red*), green (G,*green*), dan biru (B,*blue*). Jika setiap komponen warna tersebut memiliki rentang intensitas 0-255, maka terdapat sejumlah 255^3 = 16.777.216 kemungkinan jenis warna pada citra ini. Karena dibutuhkan 24-bit per piksel, maka citra ini disebut pula dengan citra warna 24-bit.



Gambar 2.12 Citra warna *Red, Green, Blue* (RGB)

Pada model RGB, semua warna dimodelkan dalam suatu kubik warna dengan panjang sisi satu, seperti yang tertera pada Gambar 2.13. Warna-warna di sepanjang diagonal hitam-putih (*black-white*) ditampilkan dengan garis putus-putus, yaitu titik-titik yang dalam hal ini nilai-nilai R,G, dan B sama.



Gambar 2.13 Model warna RGB

2.6 Teknik *Grayscaling*

Grayscaling merupakan teknik mengubah citra warna menjadi citra abu-abu. Teknik ini merupakan proses awal dalam pengolahan citra. Dalam pemrograman MATLAB teknik *grayscaling* dapat digunakan dengan menggunakan fungsi *rgb2gray* sehingga yang awalnya citra warna dikonversi menjadi citra yang terdiri dari hitam dan putih atau biasa disebut citra *grayscale.* Teknik ini merupukan proses awal dalam pengolahan citra agar dapat melanjutkan pada proses berikutnya.

Gambar 2.14 Teknik Grayscaling

2.7 Segmentasi Citra

Segmentasi merupakan proses membagi suatu citra kedalam komponen-komponen region atau objek. Algoritma segmentasi secara umum didasarkan pada salah satu dari sifat dasar nilai intensitas, yaitu :

- *Discontinuity* : Pendekatan dengan membagi citra berdasarkan perubahan besar pada nilai intensitasnya seperti tepi citra.
- *Similarity* : Pendekatan dengan membagi citra ke dalam region-region yang serupa sesuai dengan kriteria awal yang diberikan.

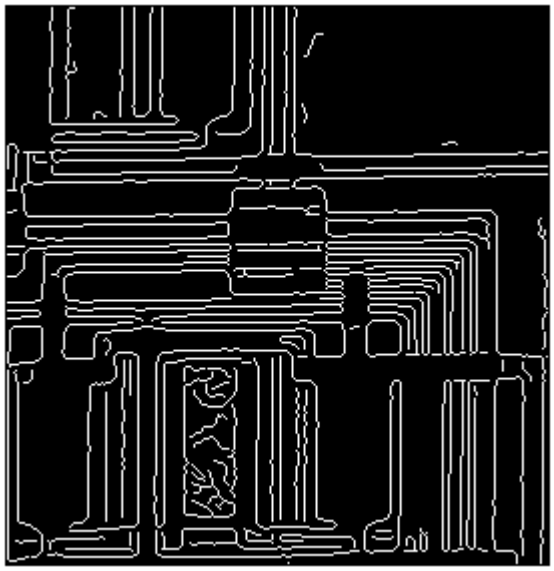
Segmentasi ditujukan pada operasi untuk mempartisi citra menjadi komponen-komponen, atau objek-objek yang terpisah. Pada bagian ini terdapat dua bagian yang penting diantaranya adalah pengambang-batasan dan pendeteksian tepi.

2.7.1 Deteksi Tepi

Tepi adalah sebuah himpunan dari piksel-piksel yang terhubung dan berada pada batas (*boundary*) diantara dua *region*. Definisi tepi membutuhkan kemampuan untuk mengukur transisi *gray-level* dengan cara yang tepat.

Suatu model tepi yang ideal merupakan suatu himpunan *connected pixel* yang masing-masing berada pada suatu perubahan langkah tegak lurus (*orthogonal*) dalam *grey-level*. Tapi kenyataanya, hasil contoh atau akuisisi citra belum tentu menghasilkan citra yang sesempurna itu, misalkan citra lebih kabur dan sebagainya. Akibatnya tepi lebih mendekati model citra model '*ramplik*e'. Kemiringan (*slope*) dari *ramp* sesuai dengan tingkat kekaburan dari tepi.

Ketebalan dari tepi ditentukan oleh panjang dari *ramp*, yang merupakan transisi dari *gray-level* awal sampai *gray-level* akhir. Panjang *ramp* ditentukan oleh *slope*, yang ditentukan juga oleh tingkat kekaburan. Berarti bahwa tepi yang kabur cenderung tebal dan tepi yang tajam cenderung tipis.



Gambar 2.15 Deteksi Tepi

2.7.2 Teknik *Thresholding* (Pengambang-batasan)

Thresholding atau bisa disebut dengan pengambang-batasan yaitu sebuah teknik untuk mengubah suatu citra abu-abu menjadi citra biner (putih dan hitam)

Pengambang-batasan terdapat dua jenis :

- Pengambang-batasan Tunggal
- Pengambang-batasan Ganda

2.7.2.1 Pengambang-batasan tunggal

Suatu citra abu-abu diubah menjadi citra biner (putih dan hitam) dengan cara memilih suatu level keabuan T dalam citra asli, dan kemudian mengubah setiap piksel hitam atau putih tergantung nilai keabuan lebih besar atau kurang dari T. Suatu piksel menjadi :

- Putih jika level keabuannya > T
- Hitam jika level keabuannya <= T

Pengambang-batasan merupakan hal yang vital dalam segmentasi citra digital, dimana diinginkan untuk mengisolasi objek-objek dari latar belakang citra. Hal ini juga merupakan bidang yang penting dalam robotika. Pengambang batasasan dapat dilakukan secara sederhana dalam Matlab. Jika diasumsikan dimiliki suatu citra 8 bit dan disimpan dalam suatu variabel X, maka perintahnya X>T.

Citra yang dihasilkan kemudian dapat diproses untuk mencari jumlah bintik atau ukuran rata-rata bintik. Untuk melihat hal ini bekerja, ingat bahwa dalam Matlab, suatu operasi pada suatu angka, ketika diterapkan pada suatu matriks, diinpretasikan sebagai penerapan terhadap semua elemen-elemen matriks. Perintah X>T akan memberikan nilai balik 1 (*true*) untuk semua piksel yang memiliki intensitas keabuan yang lebih besar dari T, dan nilai balik 0 (*false*) untuk semua piksel yang memiliki intensitas keabuan yang lebih kecil atau sama dengan T.

Selain metode diatas, Matlab juga memiliki fungsi *im2bw*, yang mengambang-batas suatu citra dengan sembarang tipe data. Karena level merupakan parameter bernilai antara 0 sampai 1, yang mengindikasikan fraksi nilai-nilai keabuan yang diubah menjadi putih. Perintah ini bisa dipakai untuk citra abu-abu, citra berwarna

maupun untuk citra indeks dengan tipe data uint8, uint16, ataupun double. Fungsi *im2bw* secara otomatis menskalakan nilai level menjadi suatu nilai keabuan yang cocok dengan tipe citra, dan kemudian melakukan pengambang-batasan seperti yang telah dijelaskan sebelumnya.

2.7.2.2 Pengambang-batasan Ganda

Disini dipilih dua nilai T1 dan T2 dan diterapkan suatu operasi pengambang-batasan sebagai berikut. Suatu piksel menjadi :

- Putih jika level keabuan antara T1 dan T2
- Hitam jika level diluar rentang ini

Operasi ini bisa diimplementasikan dengan suatu variasi sederhana dari pengambang-batasan tunggal X>T1 & X<T2. Karena simbol *ampersand* berperan sebagai operasi logikal, hasilnya akan menghasilkan nilai 1 jika memenuhi kedua pertidak-samaan itu

Gambar 2.16 (a) Citra awal (b) Citra setelah proses terambang-batasan

Terlihat pada Gambar 2.16 merupakan contoh proses perambang-batasan dari citra awal menjadi citra biner

2.7.2.3 Penerapan Pengambang-batasan

Pengambang-batasan bermanfaat pada situasi situasi berikut ini :

1. Ketika diinginkan untuk menghapus detail-detail yang tidak diinginkan dalam citra untuk memfokuskan pada yang essensial.
2. Ketika dimaksudkan untuk menyingkapkan aspek-aspek tersembunyi dalam suatu citra.
3. Ketika diinginkan untuk menghapus derau yang bervariasi pada suatu citra teks

.

2.8 Morfologi Citra

Morfologi merupakan suatu cabang pengolahan citra digital yang secara khusus berguna untuk menganalisis bentuk dalam citra. Matlab memiliki banyak fungsi untuk morfologi biner dalam pengolahan citra. Tetapi fungsi-fungsi tersebut juga bisa digunakan untuk morfologi abu-abu. Teori morfologi dapat dikembangkan dalam berbagai cara. Berikut dipaparkan salah satu operasi yang dikenal dalam morfologi citra. yang digunakan dalam tugas akhir ini yaitu dilasi.

Dilasi adalah sebuah operasi yang “menebalkan” atau “menipiskan” objek pada citra biner. Jika diberikan dua himpunan piksel A dan B, maka dilasi A dan B, yang dinotasikan sebagai A $\oplus$ B, didefinisikan sebagai

$$A \oplus B = \{z|(\hat{B})_z \cap A \neq \emptyset\} \quad \text{.....} \quad 2.1$$

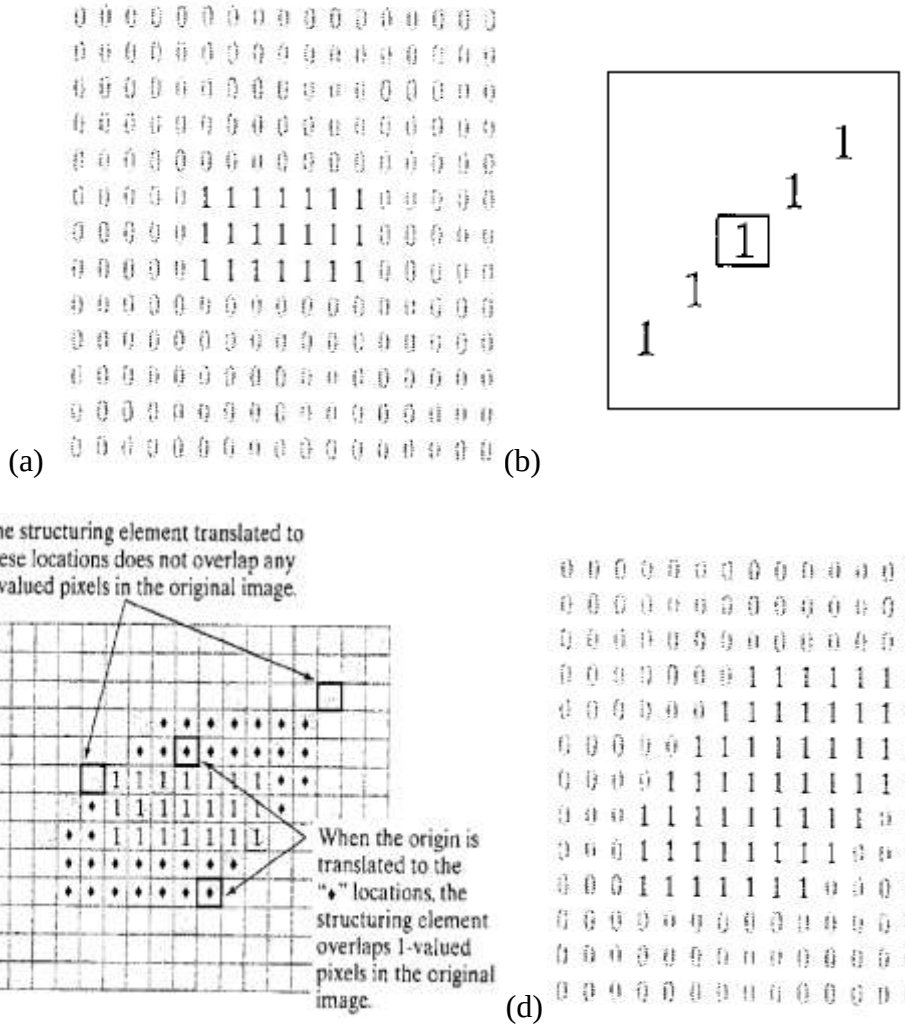
Hal ini berarti bahwa untuk setiap B, A ditranslasi sejauh koordinat-koordinat tersebut dan kemudian dilakukan operasi union atas semua translasi yang dilakukan. Definisi dilasi dapat juga diartikan sebagai

$$A \oplus B = \{(x,y) + (u,v) : (x,y) \varepsilon A,(a,b) \varepsilon B\} \quad \text{.....} \quad 2.2$$

Dari definisi diatas, dapat diketahui bahwa operasi dilasi bersifat komutatif yaitu

$A \oplus B = B \oplus A$…….. 2.3

Dilasi mempunyai efek membesarkan ukuran objek. Tidak selalu benar bahwa objek asli A selalu berada dalam dilasi $A \oplus B$. tergantung pada koordinat B, bisa jadi $A \oplus B$ terletak jauh dari objek asli A.



Gambar 2.17 (a) Citra asli, (b) Struktur elemen dengan lima piksel tersusun dalam garis diagonal, (c) Struktur elemen ditranslasikan ke beberapa lokasi pada citra, (d) hasil citra

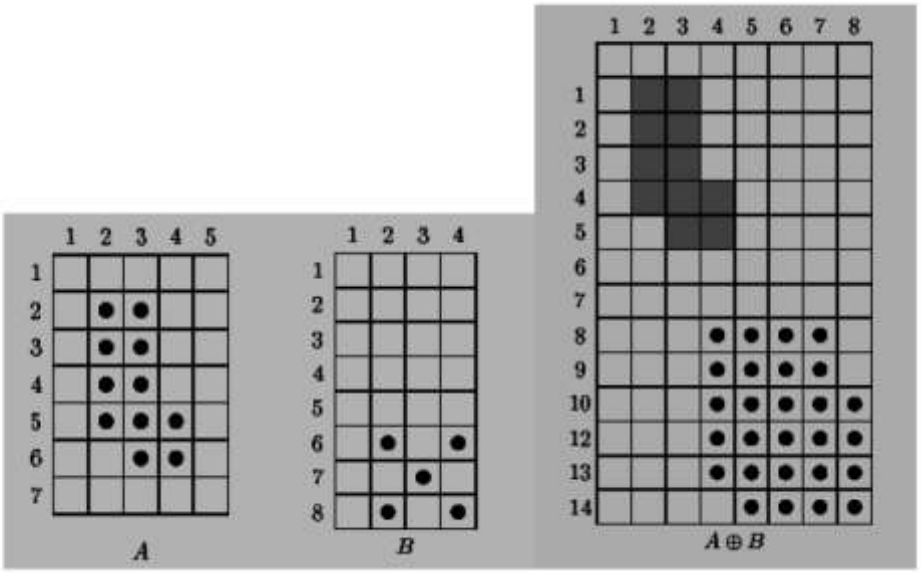
Pada dasarnya dilasi A⊕B dapat dilakukan dengan menggantikan setiap titik (x,y) pada A dan B (titik(0,0) pada B diletakan pada (x,y)). Atau ekivalennya yaitu mengganti setiap titik (u,v) pada bilangan B dengan A. Dilasi juga dikenal dengan sebutan *Minkowski Addition*.

Pada contoh berikut maka A⊕B berada jauh dari A karena

B = {(7,3),(6,2),(6,4),(8,2),(8,4)}…….. 2.4

Sehingga :

$$A \oplus B = A_{(7,3)} \cup A_{(6,2)} \cup A_{(6,4)} \cup A_{(8,2)} \cup A_{(8,4)} \quad \text{……..} \quad 2.5$$



Gambar 2.18 Dilasi A⊕ B

Untuk dilasi maka pada umumnya diasumsikan bahwa A adalah citra yang akan diolah dan B adalah suatu himpunan piksel. Himpunan piksel B sering disebut *structuring element* atau *kernel*

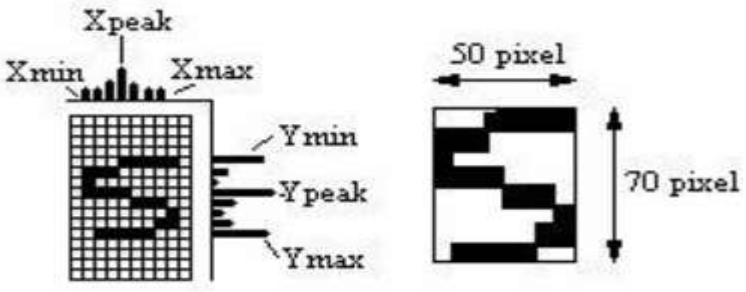
(a) (b)

Gambar 2.19 (a) Citra awal (b) Citra setelah proses Dilasi

Terlihat pada gambar 2.19 bahwa hasil dilasi membuat garis tepi menjadi lebih tebal.

2.9 *Cropping*, Normalisasi dan Ekstraksi Ciri

Cropping pada pengolahan citra adalah memotong satu bagian citra untuk memperoleh citra yang di inginkan, sedangkan normalisasi yaitu mentransformasikan citra ke bentuk citra normal yang sesuai dengan kebutuhan. Sistem ini menggunakan penskalaan dari citra semula ke bentuk citra normalisasi.



(a) (b)

Gambar 2.20 (a) Pemotongan Citra (*crop*), (b) Penyesuaian ukuran citra (normalisasi)

Ekstraksi ciri terbagi menjadi tiga macam yaitu :

1. Ekstraksi fitur bentuk

   Bentuk dari suatu objek adalah karakter konfigurasi permukaan yang diwakili oleh garis dan kontur, fitur bentuk dikategorikan bergantung pada teknik yang digunakan. Kategori tersebut adalah berdasarkan batas (*boundary-based*) dan berdasarkan daerah (*region-based*). Teknik berdasarkan batas menggambarkan bentuk daerah dengan menggunakan karakteristik eksternal, contohnya adalah piksel sepanjang batas objek, sedangkan teknik berdasarkan daerah menggambarkan bentuk wilayah dengan menggunakan karakteristik internal, contohnya adalah piksel yang berada dalam suatu wilayah. Fitur bentuk yang biasa digunakan adalah :

   - Wilayah (area) yang merupakan jumlah piksel dalam wilayah digambarkan oleh bentuk (*foreground*).
   - Lingkar (*parimeter*) adalah jumlah dari piksel yang berada pada batas dari bentuk, *parimeter* didapatkan dari hasil deteksi tepi.
   - Kekompakan (*compactness*)
   - *Euler number* atau faktor E adalah perbedaan antara jumlah dari *connected component* (C) dan jumlah lubang (H) pada citra.

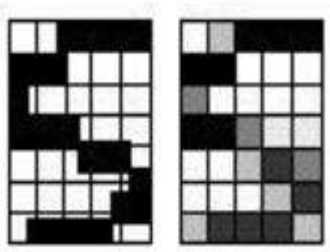
2. Ekstraksi fitur tekstur

   Pada ekstraksi fitur ini, fitur pembeda adalah tekstur yang merupakan karakteristik penentu pada citra. Teknik statistik yang terkenal untuk ekstraksi fitur adalah *gray-level co-occurrence*, teknik tersebut dilakukan dengan melakukan pemindaian untuk mencari jejak derajat keabuan setiap dua buah piksel yang dipisahkan dengan jarak d dan sudut θ yang tetap. Biasanya sudut yang digunakan adalah 0°, 45°, 90°, dan 135°.

3. Ekstraksi fitur warna

   Pada ekstraksi fitur warna, ciri pembeda adalah warna, biasanya ekstraksi fitur ini digunakan pada citra berwarna yang memiliki komposisi warna RGB (*red green blue*).

Ekstraksi ciri dalam pengolahan citra adalah mengubah nilai-nilai intensitas koordiant piksel yang terdapat dalam citra menjadi susunan kode nilai pada setiap piksel. Nilai setiap titik koordinat piksel huruf diberi kode (1) untuk warna hitam dan kode (-1) untuk warna putih. Pengkodean tersebut dilakukan agar sesuai dengan fungsi aktifasi pada Jaringan Syaraf Tiruan.



Gambar 2.21 Citra disesuaikan kembali agar memenuhi *input* dari jaringan syaraf tiruan

Bisa dilihat pada gambar 2.21 nilai setiap piksel berubah agar dapat terhubung kepada Jaringan Syaraf Tiruan yang memiliki rentan -1 hingga 1.

2.10 Jaringan Syaraf Tiruan

Jaringan Syaraf tiruan merupakan salah satu upaya manusia untuk memodelkan cara kerja atau fungsi sistem syaraf manusia dalam melaksanakan tugas tertentu. Pemodelan ini didasari oleh kemampuan otak manusia dalam mengorganisasikan sel-sel penyusunnya yang disebut *neuron*, sehingga mampu melaksanakan tugas-tugas tertentu, khususnya pengenalan pola dengan efektifitas yang sangat tinggi.

Jaringan Syaraf Tiruan mempunyai struktur tersebar paralel yang sangat besar dan mempunyai kemampuan belajar, sehingga bisa melakukan *generalization* atau diterjemahkan sebagai generalisasi, yaitu bisa menghasilkan *output* yang benar untuk *input* yang belum pernah dilatihkan. Dengan kedua kemampuan pemrosesan informasi ini, Jaringan Syaraf Tiruan mampu menyelesaikan masalah-masalah yang sangat kompleks.

Pada prakteknya, Jaringan Syaraf Tiruan tidak bisa memberikan solusi dengan bekerja sendiri, tetapi perlu diintegrasikan ke dalam pendekatan rekayasa sistem yang kosisten. Setiap masalah yang kompleks didekomposisi kedalam sejumlah tugas sederhana, dan Jaringan Syaraf Tiruan ditugaskan sebagai bagian dari tugas-tugas tersebut (seperti pengenalan pola, *assosiative memory*, kontrol, dan sebagainya) yang sesuai dengan kemampuannya.

2.10.1 Fungsi Aktivasi

Fungsi aktivasi yang dinotasikan dengan f(x) mendefinisikan nilai output dari suatu *neuron* dalam level aktivasi tertentu berdasarkan nilai *output* pengkombinasi linier. Ada beberapa macam fungsi aktivasi yang biasanya digunakan pada Jaringan Syaraf Tiruan, diantaranya adalah :

- *Threshold function*

  f(x) = 1 jika x >= 0…….. (2.6a)

  f(x) = 0 jika x < 0…… (2.6b)

- *Piecewise linear function*

  f(x) = 1 x >= $\frac{1}{2}$……..... (2.7a)

  f(x) = x > $\frac{1}{2}$ > x > $-\frac{1}{2}$…….. (2.7b)

  f(x) = 0 x <= $\frac{1}{2}$…… (2.7c)

- *Sigmoid function*

  f(x) = $\frac{1}{1+\exp(-ax)}$ …… (2.8)

2.10.2 Arsitektur Jaringan

Pola dimana *neuron-neuron* pada Jaringan Syaraf Tiruan disusun berhubungan erat dengan algoritma belajar (*training*) yang digunakan utuk melatih jaringan. Secara umum, kita bisa membagi arsitektur jaringan menjadi empat, yaitu :

1. *Single-Layer Feedforward Networks*

Suatu Jaringan Syaraf Tiruan berlapis adalah jaringan *neuron* yang diorganisasikan dalam bentuk lapisan lapisan.

2. *Multi-Layer Feedforward Networks*

Kelas kedua dari *feedforward neural network* adalah jaringan dengan satu atau lebih lapis tersembunyi (*hidden layer*), dengan *computation nodes* yang berhubungan disebut *hidden neuron* atau *hidden untis*.

3. *Recurrent Networks*

   Recurrent neural network merupakan jaringan yang mempunyai minimal satu *feedback loop*

4. *Lattice Structure*

   Sebuah *lattice* (kisi-kisi) terdiri dari satu dimensi, dua dimensi, atau lebih array *neuron* dengan himpunan *node* sumber yang bersesuaian yang memberikan sinyal *input* ke array; dimensi *lattice* mengacu pada jumlah dimensi ruang dimana *graph* berada.

2.10.3 *Standard Backpropagation*

Multi-Layer Perceptron merupakan model Jaringan Syaraf Tiruan yang paling banyak digunakan dalam bidang pendidikan dan aplikasi, arsitektur *Multi-Layer Perceptron* merupakan salah satu Jaringan Syaraf Tiruan yang menggunakan *Multi-Layer feedforward Network*. Banyak algoritma pelatihan yang diusulkan untuk melatih Multi-Layer Perceptron. Salah satu yang popular adalah algoritma pelatihan *Back Propagation* atau yang biasa disebut Propagasi Balik. Sesuai dengan namanya, algoritma ini melakukan dua tahap perhitungan, yaitu perhitungan mundur yang mempropagasikan balik galat tersebut untuk memperbaiki bobot-bobot sinaptik pada semua *neuron* yang ada. Berikut ini adalah langkah-langkah detail dari algoritma pelatihan Propagasi balik (*Back Propagation*)

Algoritma Pelatihan Propagasi Balik :

1. Definisikan masalah, misalkan matriks (P) dan matriks target (T).
2. Inisialisasi, menentukan arsitektur jaringan nilai ambang MSE sebagai kondisi berhenti, *learning rate*, serta menetapkan nilai-nilai bobot sinaptik melalui pembangkitan nilai acak dengan interval nilai sembarang. Kita bisa membangkitkan nilai acak dalam interval [-1,+1] atau [-0.5,+0.5] ataupun lainnya. Tidak ada aturan yang baku mengenai interval ini.

3. Pelatihan jaringan

- Perhitungan Maju

Dengan menggunakan bobot-bobot yang telah ditentukan pada inisialisasi awal (W1), kita dapat menghitung keluaran dari *hidden layer* berdasarkan persamaan berikut (misal kita gunakan fungsi aktivasi *sigmoid*)

$A1 = \frac{1}{1+ e^{(W1*P+B1)}}$……………………………………………. (2.9)

Hasil keluaran *hidden layer* (A1) dipakai untuk mendapatkan keluaran dari *output layer*, seperti pada persamaan berikut :

$A2 = W2 * A1 + B2$……………………………………………… (2.10)

Keluaran dari jaringan (A2) dibandingkan dengan target yang diinginkan. Selisih nilai tersebut adalah *error* (galat) dari jaringan, seperti pada persamaan berikut :

$E = T - A2$…………………………………………………….. (2.10)

Sedangkan nilai galat keseluruhan dinyatakan oleh persamaan berikut :

$SSE = \Sigma \Sigma \mathrm{E}^2$…………………………………………………….. (2.11)

- Perhitungan Mundur

Nilai galat yang didapat dipakai sebagai parameter dalam pelatihan. Pelatihan akan selesai jika galat yang diperoleh sudah dapat diterima. Galat yang didapat dikembalikan lagi ke lapis-lapis yang berada di depannya. Selanjutnya, *neuron* pada lapis tersebut akan memperbaiki nilai-nilai bobotnya. Perhitungan perbaikan bobot diberikan pada persamaan-persamaan berikut:

$D2 = (1 - A2^2) * E$ …………………………………………….. (2.12a)

$D1 = (1 - A1^2) * (W2 * D2)$…………………………………… (2.12b)

$dW1 = dW1 + (lr * D1 * P)$…………………………………..... (2.12c)

$$dB1 = dB1 + (lr * D1)$$ ..... (2.12d)

$$dW2 = dW2 + (lr * D2 * P)$$ ..... (2.12e)

$$dB2 = dB2 + (lr * D2)$$ ..... (2.12f)

- Perbaikan Bobot Jaringan

  Setelah *neuron-neuron* mendapatkan nilai yang sesuai dengan kontribusinya pada galat keluaran, maka bobot-bobot jaringan akan diperbaiki agar galat dapat diperkecil. Perbaikan bobot jaringan diberikan oleh persamaan-persamaan berikut :

  $$TW1 = W1 + dW1$$ ..... (2.13a)

  $$TB1 = B1 + dB1$$ ..... (2.13b)

  $$TW2 = W2 + dW2$$ ..... (2.13c)

  $$TB2 = B2 + dB2$$ ..... (2.13d)

- Presentasi Bobot Jaringan

  Bobot-bobot yang baru, hasil perbaikan, dipakai kembali untuk mengetahui apakah bobot-bobot tersebut sudah cukup baik bagi jaringan. Baik bagi jaringan berarti bahwa dengan bobot-bobot tersebut, galat yang akan dihasilkan sudah cukup kecil. Pemakaian nilai bobot-bobot yang baru diperlihatkan pada persamaan-persamaan berikut :

  $$TA1 = \frac{1}{1 + e^{(TW1 * P + TB1)}}$$ ..... (2.14a)

  $$TA2 = TW2 * TA1 + TB2$$ ..... (2.14b)

  $$TE = T - TA2$$ ..... (2.14c)

  $$TSSE = \Sigma \Sigma TE^2$$ ..... (2.14d)

  Kemudian bobot-bobot sinapsis jaringan diubah menjadi bobot-bobot baru:

  W1 = TW1; ..... (2.15)

B1 = TB1;…….. (2.16)

W2 = TW2 ;…….. (2.17)

B2 = TB2;……. (2.18)

A1 = TA1;…… (2.19)

A2 = TA2;…… (2.20)

E = TE;……. (2.21)

SSE = TSSE;…… (2.22)

4. Langkah-langkah diatas adalah untuk satu kali siklus pelatihan (satu *epoch*). Biasanya, pelatihan harus diulang-ulang lagi hingga jumlah siklus tertentu atau telah tercapai SSE (*Sum Square Error*) atau MSE (*Mean Square Error*) yang diinginkan.

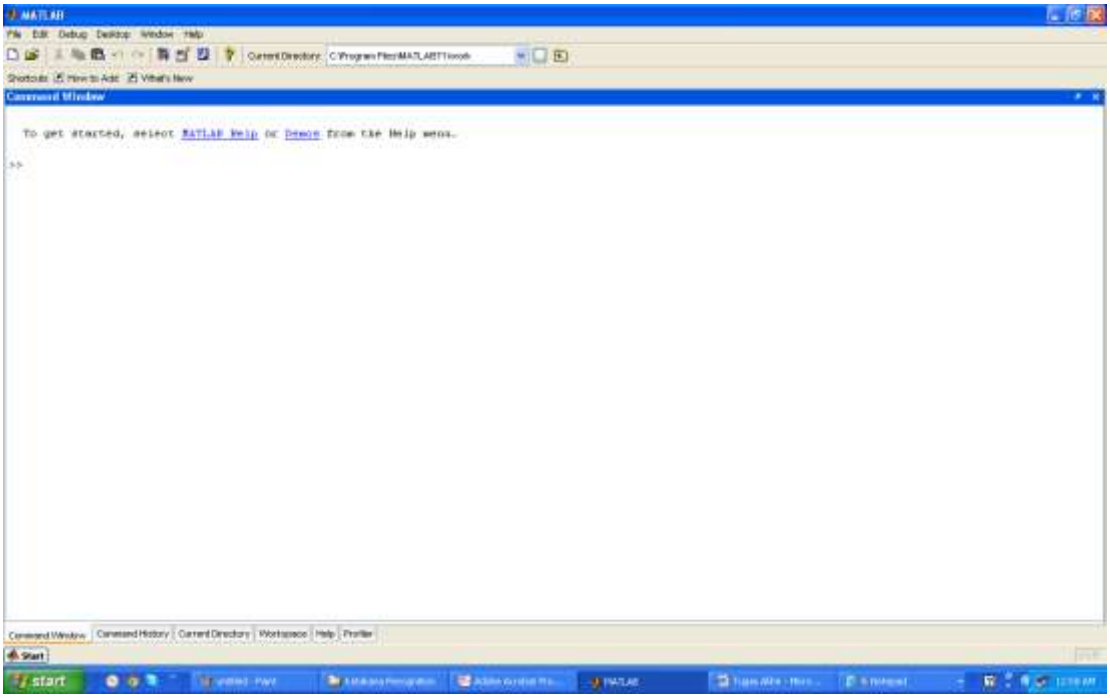
5. Hasil akhir dari pelatihan jaringan adalah bobot-bobot W1,W2,B1 dan B2.

Pada prakteknya, perancangan arsitektur *Multi-Layer Perceptron* sangat tergantung pada masalah yang akan diselesaikan. Untuk himpunan masukan dengan jumlah dimensi yang besar dan jumlah kelas keluaran yang diinginkan juga besar, maka diperlakukan jumlah *node* pada *hidden layer* yang lebih besar juga atau diperlakukan dua *hidden layer* dengan jumlah *node* yang lebih sedikit pada masing-masing *hidden layer*. Tetapi tentu saja ada batas optimumnya untuk kedua parameter tersebut.

2.11 Pemrograman Matlab

Matlab merupakan suatu paket perangkat lunak yang memampukan *user* untuk melakukan komputasi matematika, menganalisis data, mengembangkan algoritma, melakukan simulasi dan pemodelan, dan menghasilkan tampilan grafik dan antarmuka grafikal.

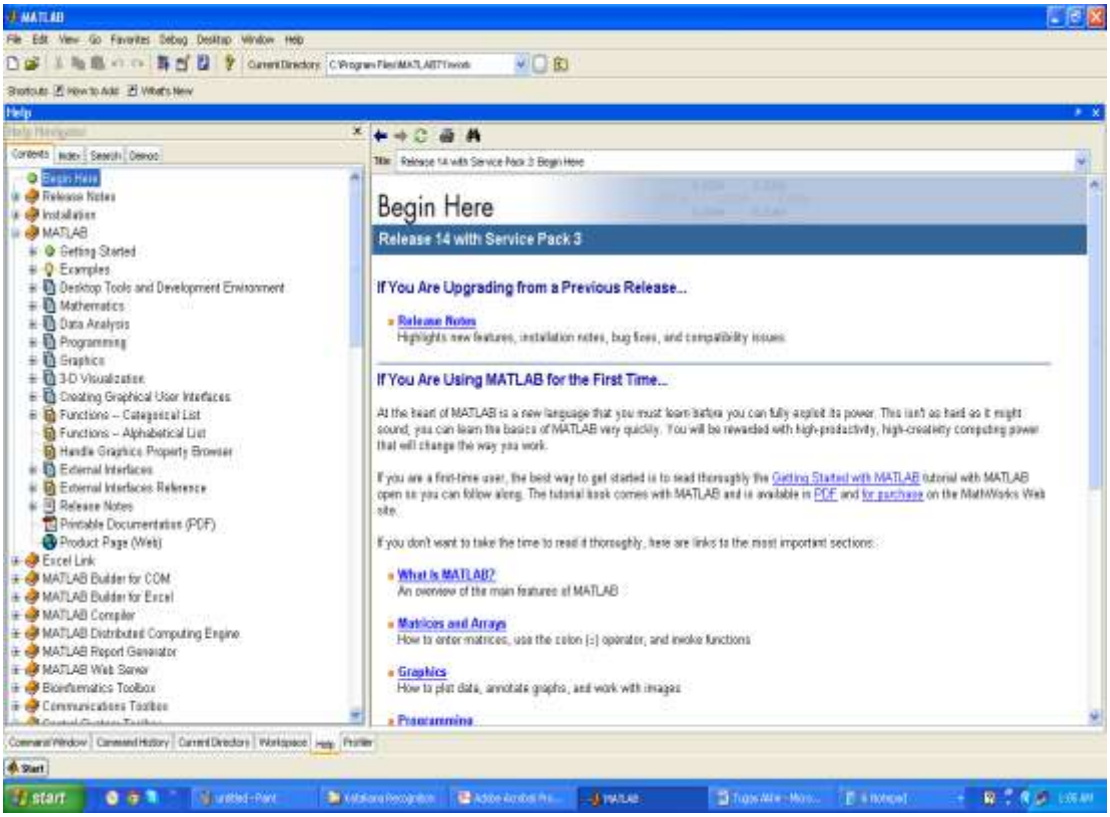
Untuk menjalankan Matlab pada suatu komputer pribadi, *user* hanya perlu mengklik dua kali ikon Matlab. Untuk menggunakan Matlab, *user* juga hanya perlu mengetikan perintah atau *variable* pada *Command Window*. Kemunculan tanda *prompt* (>>) menandakan bahwa Matlab siap menerima perintah dari *user*. Pada Gambar 2.22 ditunjukan *Command Window* pada Matlab, dalam hal ini tertampil tanda *prompt* yang menyatakan bahwa Matlab siap menerima perintah yang akan *user* berikan.



Gambar 2.22 Tampilan *Command Window* Matlab

Untuk mengakhiri sesi Matlab, *user* diminta untuk mengetikan *quit* atau *exit* pada *prompt* Matlab, untuk menginterupsi atau mengaborsi proses komputasi tanpa harus keluar dari Matlab, *user* dapat menekan *Ctrl+C* (menekan kunci *Ctrl* bersamaan dengan C). *User* juga bisa mengetikan perintah *help* pada *prompt* Matlab untuk meminta pertolongan Matlab dalam mendeskripsikan sintaks atau contoh fungsi atau perintah Matlab, atau *user* bisa mengetikan *helpwin* untuk mendapatkan

pertolongan deskripsi Matlab pada jendela yang terpisah, seperti yang tertera pada Gambar 2.23



Gambar 2.23 Jendela *help* pada Matlab