BAB II
TINJAUAN PUSTAKA

2.1 Digital *Image Processing*

Image processing adalah suatu metode yang digunakan untuk memproses atau memanipulasi gambar dalam bentuk 2 dimensi *image processing* dapat juga dikatakan segala operasi untuk memperbaiki, menganalisa, atau mengubah suatu gambar (Gonzalez, 2002).

Konsep dasar pemrosesan suatu objek pada gambar menggunakan pengolahan citra diambil dari kemampuan indera penglihatan manusia yang selanjutnya dihubungkan dengan kemampuan otak manusia. Dalam sejarahnya, pengolahan citra telah diaplikasikan dalam berbagai bentuk, dengan tingkat kesuksesan cukup besar. Seperti berbagai cabang ilmu lainnya, pengolahan citra menyangkut pula berbagai gabungan cabang-cabang ilmu, diantaranya adalah optik, elektronik, matematika, fotografi, dan teknologi komputer (Gonzalez, 2002).

Pada umumnya, objektifitas dari pengolahan citra adalah mentransformasi atau menganalisis suatu gambar sehingga informasi baru tentang gambar dibuat lebih jelas. Ada empat klasifikasi dasar dalam pengolahan citra yaitu point, area, geometric, dan frame (Gonzalez, 2002).

a. *Point* memproses nilai *pixel* suatu gambar berdasarkan nilai atau posisi dari pixel tersebut. Contoh dari proses *point* adalah *adding, substracting, contrast stretching* dan lainnya.
b. Area memproses nilai pixel suatu gambar berdasarkan nilai pixel tersebut beserta nilai *pixel* sekelilingnya. Contoh dari proses area adalah *convolution, dan blurring.*
c. Geometric digunakan untuk mengubah posisi dari pixel. Contoh dari proses *geometric* adalah *scaling, rotation*, dan *mirroring*.

d. *Frame* memproses nilai *pixel* suatu gambar berdasarkan operasi dari 2 buah gambar atau lebih. Contoh dari proses *frame* adalah *addition, substraction,* dan *and/or.*

Selain itu masih ada 3 tipe pengolahan citra yaitu:

a. *Low-level process*: proses-proses yang berhubungan dengan operasi primitif seperti *image pre-processing* untuk mengurangi *noise,* menambah kontras dan menajamkan gambar. Pada *low-level process, input* dan *output* - nya berupa gambar.
b. *Mid-level process:* proses-proses yang berhubungan dengan tugas-tugas seperti *segmentasi* gambar (membagi gambar menjadi objek-objek), pengenalan *(recognition)* suatu objek individu. Pada *mid-level process,* input pada umumnya berupa gambar tetapi output-nya berupa atribut yang dihasilkan dari proses yang dilakukan gambar tersebut seperti garis, garis *contour, dan objek-objek* individu.
c. *High-level process*: proses-proses yang berhubungan dengan hasil dari *mid-level process* (Gonzalez, 2002).

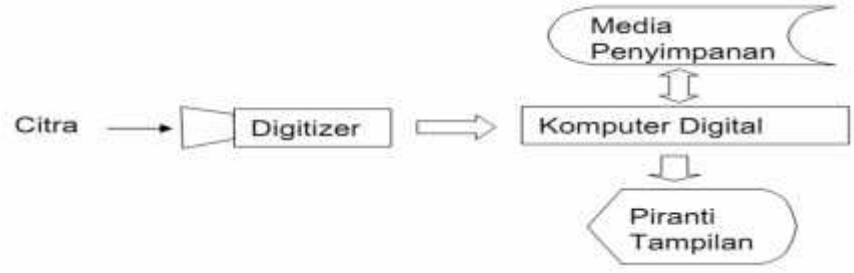
2.2 Pengolahan Citra Digital

Pengolahan citra digital merupakan sebuah teknologi *visual* yang digunakan untuk mengamati dan menganalisis suatu objek tanpa berhubungan langsung dengan objek yang diamati tersebut. Teknologi ini dapat digunakan untuk mengevaluasi mutu suatu produk tanpa merusak produk itu sendiri atau dikenal dengan istilah *non-destructive evaluation* (NDE) (Suhandy, 2003).

Proses pengolahan citra digital dan analisanya, banyak menggunakan persepsi visual. Data masukan dan keluaran yang dihsilkan oleh proses ini adalah dalam bentuk citra. Citra yang digunakan adalah citra digital, karena citra jenis ini dapat diproses oleh komputer digital. Citra digital diperoleh secara otomatis dari sistem penangkapan citra digital dan membentuk suatu matriks yang menyatakan intensitas cahaya pada suatu himpunan diskrit dari suatu titik atau citra masukan diperoleh melalui suatu kamera yang didalamnya terdapat suatu alat digitasi yang mengubah citra masukan berbentuk analog menjadi *citra digital* (Suhandy, 2003).

Alat digitasi ini dapat berupa penjelajahan *silod-state* yang menggunakan matriks sel yang sensitif terhadap cahaya yang masuk, dimana citra yang direkam maupun sensor yang digunakan mempunyai kedudukan atau posisi yang tetap alat masukan citra yang umum digunakan adalah kamera dimana sensor citra dari alat ini menghasilkan keluaran berupa citra analog sehingga dibutuhkan proses digitasi dengan menggunakan alat digitasi seperti yang telah disebutkan diatas. Komponen utama dari perangkat keras pengolahan citra secara digital adalah komputer dan alat peraga. Komputer tersebut bisa dari jenis komputer multiguna khusus yang dirancang untuk pengolahan citra digital. Pengolahan citra pada umumnya dilakukan dari pixel yang sifatnya paralel pipe lined (Suhandy, 2003).

2.3 Elemen Sistem Pemrosesan Citra Digital



Gambar 2.1. Grafik komputer dan pengolahan citra
(Sumber: Suhandy, 2003)

1. *Digitize*r (Digital *Acqusition* Sistem), sistem penamgkap citra digital yang melakukan penjelajahan citra dan mengkonversinya ke representasi numerik sebagai masukan bagi komputer digital. Hasil dari *digitizer* adalah matriks yang elemen-elemennya menyatakan nilai intensitas cahaya pada suatu titik. *Digitizer* terdiri dari 3 komponen dasar yaitu :
   a. Sensor citra yang bekerja sebagai pengukur intensitas cahaya
   b. Perangkat penjelajah yang berfungsi merekam hasil pengukuran intensitas pada seluruh bagian citra
   c. Pengubah analog ke digital yang berfungsi melakukan sampling dan kuantisasi.

2. Komputer digital, digunakan pada sistem pemroses citra, mampu melakukan berbagai fungsi pada citra digital resolusi tinggi.
3. Piranti tampilan, peraga berfungsi mengkonversi matriks intensitas tinggi merepresentasikan citra ke tampilan yang dapat diinterpretasi oleh manusia.
4. Media penyimpanan, piranti yang mempunyai kapasitas memori besar sehingga gambar dapat disimpan secara permanen agar dapat diproses lagi pada waktu yang lain (Gunadarma, 2006).

2.4 Elemen Dasar Citra Digital.

Ada beberapa elemen dasar citra digital diantaranya:

1. Kecerahan (*Brightness*)

   Kecerahan intensitas cahaya rata-rata dari suatu area yang melingkupinya.
2. Kontras (*Contrast*). Kontras sebaran terang *(lightness)* dan gelap *(darkness*) di dalam sebuah citra. Citra dengan kontras rendah komposisi citranya sebagian besar terang atau sebagian besar gelap. Citra dengan kontras yang baik, komposisi gelap dan terangnya, tersebar merata.
3. Kontur *(Contour)* merupakan keadaan yang ditimbulkan oleh perubahan intensitas pada pixel-pixel tetangga, sehingga kita dapat mendeteksi tepi objek di dalam citra.
4. Warna *(Color)* adalah persepsi yang dirasakan oleh sistem visual manusia terhadap panjang gelombang cahaya yang dipantulkan oleh objek. Warna-warna yang dapat ditangkap oleh mata manusia merupakan kombinasi cahaya dengan panjang berbeda. Kombinasi yang memberikan rentang warna paling lebar adalah *red* (R), *green* (G) dan *blue* (B) (Gunadarma, 2006).
5. Bentuk *(Shape)* adalah properti intrinsik dari objek tiga dimensi, dengan pengertian bahwa bentuk merupakan properti intrinsik utama untuk visual manusia. Umumnya citra yang dibentuk oleh manusia merupakan 2D, sedangkan objek yang dilihat adalah 3D.
6. Tekstur *(Texture)* adalah distribusi spasial dari derajat keabuan di dalam sekumpulan pixel-pixel yang bertetangga (Gunadarma, 2006).

2.5 Pembentukan citra

Citra ada dua macam diantaranya:

1. Citra Kontinu. Dihasilkan dari sistem optik yang menerima sinyal analog. Contoh : mata manusia, kamera analog
2. Citra diskrit atau citra digital. Dihasilkan melalui proses digitalisasi terhadap citra kontinu. Contoh : kamera digital, scanner (Gunadarma, 2006).

2.6 Tujuan Pengolahan Citra

1. Dapat memperbaiki kualitas citra agar mudah diinterpretasi oleh manusia atau komputer. Teknik pengolahan citra dengan mentrasfor-memastikan citra menjadi citra lain, contoh pemanfaatan citra (*image compression).* Pengolahan citra merupakan proses awal *(preprocessing)* dari komputer visi.
2. Pengenalan pola *(pattern recognition)* adalah Pengelompokkan data numerik dan simbolik (termasuk citra) secara otomatis oleh komputer agar suatu objek dalam citra dapat dikenali dan diinterpreasi. Pengenalan pola adalah tahapan selanjutnya atau analisis dari pengolahan citra (Idhawati, 2007).

2.7 Citra Warna

Model pengolahan warna telah banyak dikembangkan oleh para ahli, salah satu adalah model RGB. Model warna RGB menggunakan dasar tiga buah warna pokok yaitu *Red* (merah), *Green* (hijau), *Blue (*biru). Suatu citra warna yang disimpan dalam memori 8-bit, setiap pikselnyaakan mengandung informasi intensitas tiga buah warna tersebut (R, G, dan B) dengan selang nilai 0-255 (Idhawati, 2007).

Dalam model warna RGB, intensitas warna setiap piksel pada suatu citra dapat diubah dalam bentuk indeks warna, yaitu indeks warna merah (r), indeks warna hijau (h) dan indeks warna biru (b). Proses ini dinamakan normalisasi, dengan cara perhitungan seperti pada rumus dibawah ini:

Dimana:

R, G, B = nilai intensitas warna merah, hijau dan birur,

T, g, b = indeks warna merah, hijau dan biru

Model pengolahan warna yang lain adalah model wrna HIS. Model warna ini menggunakan dasar nilai *Hue* (corak), saturation (kejenuhan), dan intensity (kecerahan). Modelini dianggap sebagai cara sebenarnya manusia memandang suatu warna, yang biasanya melakukan penilaia warna secara kualitatif (Idhawati, 2007).

2.8 *Gray Level*

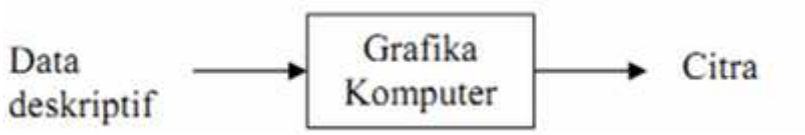
Gray-level adalah tingkat warna abu-abu dari sebuah *pixel*, dapat juga dikatakan tingkat cahaya dari sebuah *pixel*. Maksudnya nilai yang terkandung dalam *pixel* menunjukan tingkat terangnya pixel tersebut dari hitam ke putih (Idhawati, 2007).

Biasanya ditetapkan nilainya antara 0 hingga 255 (untuk 256-*graylevel*), dengan 0 adalah hitam dan 255 adalah putih. Karena hanya terbatas 1 byte saja maka untuk mempresentasikan nilai *pixel* cukup 8 bit saja. *Grayscale* adalah gambar yang memiliki *gray-level* sebagai nilai dari tiap *pixel*-nya. Oleh karena itu gambar digital dapat diwakili oleh format warna RGB *(Red-Green-Blue)* untuk setiap titiknya, di mana setiap komponen warna memiliki batasan sebesar 1 byte. Jadi untuk masing-masing komponen R, G, dan B mempunyai variasi dari 0 sampai 255. Total variasi yang dapat dihasilkan untuk sistem dengan format warna RGB adalah 256 x 256 x 256 atau 16.777.216 jenis warna. Karena setiap komponen warna memiliki batasan sebesar 1 byte atau 8 bit, maka total untuk mempresentasikan warna RGB adalah 8+8+8 = 24 bit (Gonzalez, 2002).

2.9 Analisis Citra

Analisa citra adalah kegiatan menganalisis citra sehingga menghasilkan informasi untuk menetapkan keputusan (biasanya didampingi bidang ilmu kecerdasan buatan/AI yaitu pengenalan pola *(pattern recognition)* menggunakan jaringan syaraf tiruan, logika fuzzy, dan lain-lain). Dalam ilmu komputer sebenarnya ada 3 bidang studi yang berkaitan dengan citra, tapi tujuan ketiganya berbeda, yaitu :

1. Grafika komputer



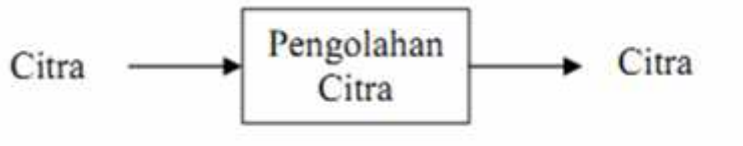
Gambar 2.2. Grafik komputer

(Sumber: Gonzalez, 2002)

Grafika komputer adalah proses untuk menciptakan suatu gambar berdasarkan deskripsi objek maupun latar belakang yang terkandung pada gambar tersebut. Merupakan teknik untuk membuat gambar objek sesuai dengan objek tersebut di alam nyata *(realism).* Bertujuan menghasilkan gambar/citra (lebih tepat disebut grafik atau *picture)* dengan primitif-primitif geometri seperti garis, lingkaran, tersebut (Gonzalez, 2002).

Primitif-primitif geometri tersebut memerlukan data deskriptif untuk melukis elemen-elemen gambar. Data deskriptif : koordinat titik, panjang garis, jari-jari lingkaran, tebal garis, warna, dan sebagainya. Grafika komputer berperan dalam visualisasi dan virtual *reality* (Idhawati, 2007).

2. Pengolahan Citra



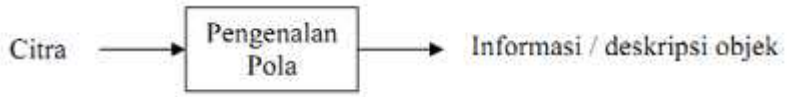
Gambar: 2.3. Pegolahan Citra

(Sumber: Idhawati, 2007)

Pengolahan citra yaitu perbaikan atau memodifikasi citra dilakukan untuk meningkatkan kualitas penampakan citra atau menonjolkan beberapa aspek informasi yang terkandung dalam citra (*image enhancement)* contoh : perbaikan kontras gelap atau terang, perbaikan tepian objek, penajaman, pemberian warna semu, dan lain-lain. Adanya cacat pada citra sehingga perlu dihilangkan atau diminimumkan *(image restoration)* contoh penghilangan kesamaran *(debluring),* citra tampak kabur karena pengaturan fokus lensa tidak tepat atau kamera goyang, penghilangan noise (Idhawati, 2007).

Elemen dalam citra perlu dikelompokkan, dicocokan atau diukur (*image segmentation)* Operasi ini berkaitan erat dengan pengenalan pola. Diperlukannya ekstraksi ciri-ciri tertentu yang dimiliki citra untuk membantu dalam pengidentifikasian objek *(image analysis).* Proses segementasi kadangkala diperlukan untuk melokalisasi objek yang diinginkan dari sekelilingnya. Contoh : pendeteksian tepi objek. Sebagian citra perlu digabung dengan bagian citra yang lain *(image reconstruction)* contoh, beberapa foto *rontgen* digunakan untuk membentuk ulang gambar organ tubuh. Citra perlu dimampatkan *(image compression)* contoh, suatu *file* citra berbentuk BMP berukuran 258 KB dimanfaatkan dengan metode *jpeg* menjadi berukuran 49 KB. Menyembunyikan data rahasia (berupa teks/citra) pada citra sehingga keberadaan data rahasia tersebut tidak diketahui orang *(steganografi* dan *watermarking)* (Idhawati, 2007).

3. Pengenalan Pola



Gambar 2.4. Pengenalan pola

(Sumber: Idhawati, 2007)

Pengenalan pola adalah mengelompokkan data numerik dan simbolik (termasuk citra) secara otomatis oleh mesin (komputer). Tujuan pengelompokkan adalah untuk mengenali suatu *objek* di dalam citra. Manusia bisa mengenali *objek* yang dilihatnya karena otak manusia telah belajar mengklasifikasi *objek-objek* di alam sehingga mampu membedakan suatu objek dengan *objek* lainnya (Idhawati, 2007).

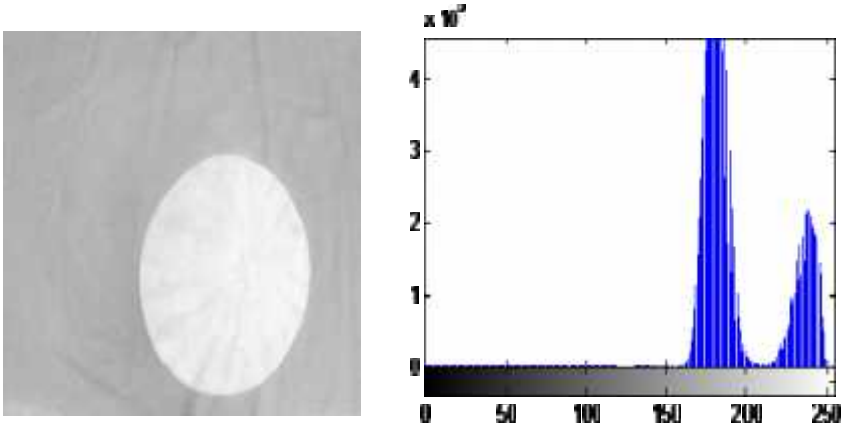
Kemampuan sistem visual manusia yang dicoba ditiru oleh mesin. Komputer menerima masukan berupa citra objek yang akan diidentifikasi, memproses citra tersebut dan memberikan keluaran berupa informasi atau deskripsi objek di dalam citra.

2.10 Histrogram

Histogram adalah grafik yang menunjukkan distribusi dari intensitas sebuah gambar Histogram dari sebuah gambar digital berupa sebuah fungsi r k = ^{n} k , dimana rk adalah nilai warna ke-k dan nk adalah jumlah pixel dalam gambar yang memiliki nilai tersebut. Pada *gray-level*, rk adalah tingkat gray-level ke-k. K = 0, 1, 2, …, L-1. L adalah batas maksimum nilai (Gonzalez, 2002).

Normalisasi dari histogram adalah dengan membagi tiap nilai nk dengan total pixel dari gambar, p (rk) = nk/n. Jumlah total nilai (p (rk)) dari normalisasi histogram adalah 1. Sumbu pada harizontal menujukan nilai gray level sedangkan sumbu vertikal menunjukan nilai jumlah piksel gray level tersebut $h(^{r}_{k)} = n_k/MN$ jika nilainya dinormalkan. Bahwa citra yang gelap histogramnya cenderung ke kiri (intensitas *gray level*nya rendah), citra terang cenderung kekanan (intensitas *gray levelnya* tinggi), untuk low contrast agak cenderung menjahui terang dan gelap, untuk high contrast histrogram merata pada semua gray level. Fungsi yang digunakan dalam matlab untuk menampilkan histrogram sebuah citra adalah: h = imhist (f, b). Di mana h merupakan histrogram dari citra, f merupakan variabel citra dan b adalah jmlah bins yang digunakan dalam membentuk histrogram. Jika tidak disebutkan maka matlab akan menggunakan nilai default, yaitu b=256 pada citra 8 bit. Contoh kode program histrogram yaitu (Prasetyo,2011).

```
i = imread (‘Semangka.jpg’);
i=rgb2gray(i);
x=imadjust (i, [],[],0.2);
figure,imshow(x);
figure, imhist (x);
```



Gambar 2.5. Histogram

Manipulasi dari histogram dapat digunakan secara efektif untuk *image enhancement* (peningkatan kualitas dari gambar). Selain itu juga berguna untuk aplikasi pengolahan citra lainnya seperti segmentasi, kompresi, dan lain-lain. Histogram juga mudah untuk dikalkulasikan dalam *software*. Hal tersebut membuat histogram menjadi sebuah tool yang populer untuk *real-time image processing.*

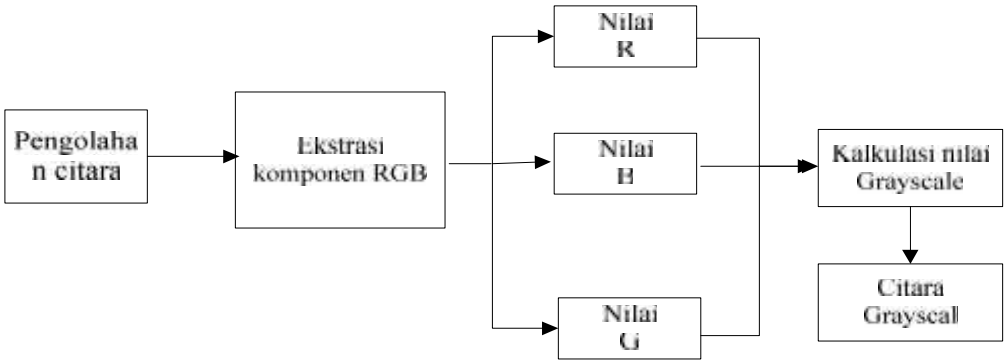
Ada 3 macam histogram processing :

a. *Histogram equalization* bertujuan untuk mengubah intensitas suatu gambar menjadi sebuah gambar dengan nilai histogram yang relatif sama di setiap levelnya. Nama lain *histogram equalization* adalah *histogram linearization.*
b. *Histogram matching (specification)* adalah hampir sama dengan *histogram equalization*, hanya saja pada *histogram matching*, dapat ditentukan sendiri bentuk dari histogram yang akan dihasilkan. *Histogram equalization* dan histogram matching dilakukan pada seluruh bagian dari gambar.
c. *Local enhancement* merupakan proses histogram *equalization* atau *histogram matching* yang dilakukan pada bagian atau daerah kecil pada gambar (Gonzalez, 2002).

2.11 *Grayscaling*

Grayscaling adalah proses perubahan nilai *pixel* dari warna (RGB) menjadi *gray-level.* Pada dasarnya proses ini dilakukan dengan meratakan nilai pixel dari 3 nilai rgb menjadi 1 nilai. Untuk memperoleh hasil yang lebih baik, nilai pixel tidak langsung dibagi menjadi 3 melainkan terdapat persentasi dari masing-masing nilai.

Salah satu persentasi yang sering digunakan adalah 29,9% dari warna merah *(Red*), 58,7% dari warna hijau *(Green),* dan 11,4% dari warna biru *(Blue).* Nilai pixel didapat dari jumlah persentasi 3 nilai tersebut (Gonzalez, 2002).



Gambar 2.6. Proses pengubahan citra RGB ke dalam citra *grayscale*
(Sumber: Idhawati, 2007)

2.12. Morfologi citra Grayscale

Perluasan morfologi di citra *grayscale* dalam hal ini adalah operasi dasar dilasi, *erosi, opening* dan *closing*. Operasi-operasi ini digunakan untuk mengembangkan bebera algoritma morfologi *grayscale* dasar. Juga dikembangkan algoritma untuk boundary extraction lewat operasi gradien morfologi, dan region partition berdasarkan *texstur content.*

Konvensi yang digunakan adalah fungsi citra digital dalam bentuk f(x,y) dan b(x,y), di mana f(x,y) adalah citra input dan b(x,y) adalah strel yang merupakan fungsi sub-*imgae*. Asumsinya adalah bahwa fungsi-fungsi ini diskrit. Z menyertakan himpunan real integer, asumsi bahwa (x,y) adalah integer dari ZxZ sedangkan f dan b adalah fungsi yang memberi nilai gray-level (angka real, dari

jumlah angka real, R) untuk setiap pasangan kordinat (x,y) yang berbeda. Jika *gray level* juga *integer*, Z menggantikan R (Prasetyo, 2011).

2.13. Algoritma-algoritma dengan dasar Morfologi citra *Grayscale*

2.13.1. marfologi penghalusan *(smoothing)*

Karena opening menekan detail terang yang lebih kecil dari pada strel yang ditetapkan, dan closing menekan datail gelap, keduanya sering digunakan secara kombinasi sebagai *morphological filters* untuk penghalusan citra dan menghilangkan *noise.* Objek yang terang dan beasr itu adalah objek dan komponen kecil dan terang adalah *noise.* Tujuannya adalah untuk menghilangkan noise. Dalam hal ini strel yang digunakan sebaiknya disk (Prasetyo, 2011).

2.13.2 Marfologi *Gradien*

Dilasi dan erosi digunakan dalam kombinasi dengan pengurangan citra untuk mendapatkan morfologi gradien citra, dinyatakan dengan g, difinisikan dengan: g = (f 0 b) – (f 0 b)

Dilasi akan menebalkan region dalam citra dan erosi menipiskannya. Perbedaan keduanya mempertebal boundary antara-region. Daerah homogenous tidak dipengaruhi (sepanjang strel relatif kecil) sehinga operasi pengurangan cenderung menghilangkannya. Hasilnya adalah citra dengan tepi yang *enhance* dan kontribusi daerah *homogenous* ditekan, maka menghasilkan efek *derivative-like* (gradien) (Prasetyo, 2011).

2.13.3 Transformaasi Top-Hat dan Bottom-Hat

Mengombinasikan pengukuran citra dengan opening dan closing akan menghasilkan transformasi *Top-Hat* dan *Bottom-Hat.* Transformasi *top-hat* citra *grayscale* f didefinisikan sebagai f dikurangi hasil opening:

$T_{hat}(f) = f-(f\ 0\ b)$

Sedangkan *transformasi bottom-hat* didefinisikan sebagai *closing* f dikurang dengan f:

$B_{hat}(f) = (f \cdot b) - f$

Satu aplikasi pokok ini dari *transformasi* ini adalah menghilangkan objek dari citra dengan menggunakan strel dalam operasi opening dan closing di mana objek tidak boleh dilepaskan. Perbedaan kedua operasi adalah citra di mana hanya terdapat sisa dari komponen yang dibuang. *Transformasi top-hat* digunakan untuk objek terang pada background gelap, sedangkan *transformasi bottom-hat* digunakan untuk objek gelap pada *background* terang. Untuk alasan ini, nama *white top-hat* dan *black top-hat* sering digunakan mereferensikan pada kedua transformasi ini (Prasetyo, 2011).

Penggunaan penting dari transformasi *top-hat* adalah memperbaiki efek ilumination yang tidak uniform, karena ilumination yang baik *(uniform)* mamainkan peran penting dalam proses pengekstrakan objek dari *backgrond*. Proses ini yang disebut dengan *segmentasi,* yaitu satu dari langkah awal yang dilakukan dalam analisis citra otomatis. Umumnya digunakan pendekatan segmentasi untuk *threshold* citra input (Prasetyo, 2011).

2.14 *Rekontruksi* Marfologi Citra Biner

Rekontruksi adalah tranformasi marfologi yang melibatkan dua citra dan sebuah strel (sebenarnya adalah sebuah citra dan sebuah strel). Citra yang satu adalah *marker,* adalah *starting point* untuk transformasi. Citra yang lain adalah *mask,* yang merupakan *constrain* transformasi. *Strel* yang digunakan untuk mendifinisikan konektivitas. Dalam sub-sub ini yang digunakan adalah 8-*connectivity (default)*, yang mengimplikasikan bahwa B adalah matriks 3x3 bernilai 1, dengan pusat yang difinisikan di koordinat (2,2). Jika g adalah mask dan f adalah marker, rekonstruksi g dari f, dinyatakan $R_g(f)$, didefinisikan dengan prosedur iteratif berikut (Prasetyo, 2011).

a. Inisialisasi h 1 menjadi marker citra f.
b. Buat strel B = ones (3)
c. Ulangi

$H_{k+1} = (h_k \oplus B) \cap g$

Sampai $h_{k+1} = h_k$

marker f harus menjadi bagian dari g; maka $f \dot{\subseteq} g$.

Algoritma ini berjalan dengan lebih cepat meskipun menggunakan formasi iteratif. Fungsi yang digunakan untuk melakukan "*fast hybrid reconstruction*" [Vincent, 1993] adalah *imreconstruct* dengan sintaks:

out = imreconstruct(marker, mask)

strel yang digunakan adalah square dengan ukuran 3 *piksel*. Untuk membuat marker dilakukan dengan membuat sebuah citra dengan ukuran yang sama dengan mask, dan bernilai 0 semua kecuali pada koordinat yang menjadi *starting point* deberi nilai 1 (Prasetyo, 2011).

2.14.1 Opening dengan Rekontruksi

Dalam morfologi opening, erosi biasanya menghilangkan objek kecil. Dilasi cenderung mengembalikan bentuk objek yang tersisa. Bagaimanapun akurasi dari restorasi ini tergantung kesamaan antara bentuk objek dengan strel. Metode *opening by reconstruction* secara pasti mengembalikan bentuk objek yang tersisa setelah erosi. Opening dengna rekontruksi f, menggunakan B, didefinissikan sebagai $R_1(f \theta B)$ (Prasetyo, 2011).

2.14.2 Filling Holes

Rekontruksi morfologi mempunyai spektum yang luas dalam aplikasi praktek. Setiap aplikasi ditentukan oleh pemilihan marker dan mask citra. Misalnya, andaikan dipilih marker citra $F_{m,}$ yang dilakukan adalah menjadikan 0 di setiap tempat kecuali border citra, di mana posisi tersebut diset menjadi 1-f:

$$F_m(x,y) = \begin{cases} 1 - f(x,y) & \\ \hline 2 & \text{jika (x,y) adalah border f lainya} \end{cases} \quad \text{.....(1)}$$

Maka $g = [R^c_f (f_m)]^c$ mempunya efek pengisian lubang dalam f. Fungsi yang digunakan untuk melakukan komputasi pengisian lubang adalah fungsi imfill dengan optimal argument "holes" yang digunakan g=imfill(f, holes)

Contoh sintks di bawah ini adalah efek pengisian lubang pada citra biner mengunakan fungsi imfill (Prasetyo, 2011).

```
>> i = imread ('key.jpg);
>> figure, imshow (i);
```

>> j = imfill(i, ‘holes’);

>> figure, imshow(j);

2.14.3 Membersikan Border Objek

Aplikasi rekontruksi lain yang berguna adalah menghilangkan objek yang menyetuh border citra. Kunci dalam aplikasi ini adalah memilih merker dan mask citra yang tepat untuk mendapatkan efek yang diharapkan. Dalam kasus ini digunakan citra sebagai mask dan marker citra $F_{m,}$ didefinisikan sebagai berikut:

F_m (x,)= $f\ x,y$ jika (x,y) adalah border f lainya.....(2)
0

Fungsi yang digunakan untuk melakukan prosedur ini secara otomatis adalah fungsi *imclearborder* dengan sintks:

g = imclearborder (f, conn)

di mana f adalah citar *input* dan g adalah hasilnya. Nilai untuk conn dapat diberi nilai 4 atau 8 (defult). Fungsi ini menekan struktur yang lebih terang dari pada sekitarnya dan yang cenderung ke citra. *Input* f boleh citra grayscale atau biner. Citra output adalah citra grayscale atau biner sesuai dengan inputnya (Prasetyo, 2011).

2.15 *Thresholding*

Thresholding ini di contohkan pada sebuah gambar, f(x,y) tersusun dari objek yang terang pada sebuah *background* yang gelap *Gray-level* milik objek dan milik *background* terkumpul menjadi 2 grup yang dominan. Salah satu cara untuk mengambil objek dari *background*nya adalah dengan memilih sebuah nilai *threshold* T yang memisahkan grup yang satu dengan grup yang lain. Maka semua pixel yang memiliki nilai > T disebut titik objek, yang lain disebut titik *background.* Proses ini disebut *thresholding* (Gonzalez, 2002).

Karena properti intutif dan kesederhanaannya dalam implementasi, *thresholding* citra menjadi titik pusat dalam aplikasi segmentasi citra. Di sini akan dibahas mengenai cara pemilihan nilai *thresholding* secara otomatis dan

memperhatikan metode untuk bermacam-macam *thresholding* menurut properti ketetanggaan citra lokal.

Andaikan bahwa histrogram intensitasnya yang ditunjukan pada gambar 2.5 yang berkaitan dengan citra *F*(x,y), yang terdiri dari obyek pada *background* mempunya *level* intensita yang dikelompokan kedalam dua mode domain. Satu cara yang jelas untuk mengekstrak obyek dari *background* adalah dengan memilih threshold T yang membagi mode-mode ini. Kemudian sembarang titik (x,y) untuk dimana *f(*x,y) ≥ T disebut *object point.* Sedangkan yang lain disebut *background point.* Dengan kata lain, citra yang di- *threshold* g(x,y) didefinisikan yaitu:

$$g(x,y) = \begin{matrix} 1 & f(x,y) \geq T \\ 2 & f\ x,y < T \end{matrix} \quad \text{.....(3)}$$

piksel yang diberi 1 berkaitan dengan obyek sedangkan *piksel* yang diberik diberi nilai 0 berkaitan dengan *background.* Ketika konstanta., pendekatan ini disebut *global thresholding* (Prasetyo, 2011).

2.16 Global *Thresholding*

Salah satu cara untuk memilih *thresholding* adalah dengan pemeriksaan visual histrogram citra. Histrogram dalam gambar 2.5 secara jelas mempunyai dua mode yang berbeda.sebagai hasilnya., mudah untuk memilih *threshold* T yang membaginya. Metode yang lain dalam pemilihan T adalah *trial and error,* mengambil beberapa *threshold* berbeda sampai satu nilai T yang memberikan hasil yang baik sebagai keputusan observer ditemukan. Hasil ini efektif dalam penentuan nilai *threshold* dalam *environment enteraktif,* seperti memperolehkan user untuk mengubah *threshold* menggunakan *widget (*kontrak grafis) semacam slider dan melihat hasilnya. Untuk pemilihan *threshold* secara otomatis, prosedur interaktifnya dijelaskan sebagai berikut:

1. Memilih perkiraan awal T. Disarankan perkiraan awal adalah titik tengah antara nilai intensitas minimum dan maksimum dalam citra.
2. Mensegmentasi citra menggunakan T. Ini akan menghasilkan dua kelompok piksel: G_1 yang berisi semua piksel dengan nilai intensitas ≥ T, dan G_2, yang berisi semua piksel dengan nilai intensitas < T.

3. Menghitung rata-rata intensitas μ_1 dan μ_2 untuk piksel-piksel dalam region G_1 dan G_2.
4. Menghitung nilai threshold yang baru:

   $T = (\mu_1 + \mu_2)$
5. Mengulangi langkah 2 sampai 4 sampai perbedaan T dalam iterasi yang berturut-turut lebih kecil dari pada parameter T_0 sebelumnya.

Toolbox matlab menyedikan fungsi *graythresh* yang menghitung *threshold* mengunakan metode otsu. Untuk mneguji formulasi metode berbasis histrogram ini, dimulai dengan normalisasi histrogram sebagai funngsi *probality discrete density*, yaitu:

$P_r (r_q) = \frac{nq}{n}$ q = 0, 1,2....., L -1

Di mana n adalah jumlah piksel dalam citra, n_q adalah jmlah piksel yang dipunyai level intensitas r_q dan L adalah total jumlah level intentitas yang tersedia dalam citra. Andaikan bahwa *threshold* k dipilih maka C_o adalah sekumpulan piksel dengan level [0, 1, ..., k – 1] dan C_1 adalah sekumpulan piksel dengan level [k, k + 1, ..., L – 1]. Metode otsu memilih nilai k yang memaksimalkan *between-class variance* σ^2 B**,** yang didefenisikan sebagai berikut:

$\sigma^2_b = \omega_o (\mu_o - \mu_T)^2 + \omega_1 (\mu_o - \mu_T)^2$

Di mana:

$\omega_o = \sum_{q=0}^{k-1} p_q (r_q)$ .....(4)

$\omega_1 = \sum_{q=k}^{L-1} p_q (r_q)$ ..... (5)

$\mu_o = \sum_{q=0}^{k-1} q p_q (r_q) / \omega_0$ .....(6)

$\mu_1 = \sum_{q=k}^{l-1} q p_q (r_q) / \omega_1$ .....(7)

$\mu_1 = \sum_{q=0}^{l-1} q p_q (r_q)$ .....(8)

Fungsi *graythresh* mengambil citra, menghitung histrogramnya, dan kemudian mencari nilai *threshold* yang memaksimalkan σ^2_B. *Threshold* dikembalikan sebagai nilai normal di antara 0.0 dan 1.0. sintaks untuk pemangilan fungsi *graythresh* adalah T = *graythresh* (f).

Di mana f adalah *input* dan T adalah *threshold* yang dihasilkan. Untuk melakukan segmentasi citra, digunakan T dalam fungsi im2bw. Karena *threshold* dinormalisasikan dalam range [0, 1], maka harus diskalakan ke range *gray level* yang benar dari citra sebelum digunakan. Misalnya jika f adalah kelas uint8, maka T dikalikan dengan 255 sebelum digunakan (Prasetyo, 2011).

2.17 Konvolusi

Secara umum konvolusi diidenfikasi sebagai cara untuk mengkombinasikan dua buah deret angka yang menghasilkan deret angka yang ketiga. Didalam dunia seismik deret-deret angka tersebut adalah wavelet sumber gelombang, reflektivitas bumi dan rekaman seismik. Secara matematis, konvolusi adalah integral yang mencerminkan jumlah lingkupan dari sebuah fungsi a yang digeser atas fungsi b sehingga menghasilkan fungsi c. Konvolusi dilambangkan dengan asterisk (*). Sehingga, a*b = c berarti fungsi a dikonvolusikan dengan fungsi b menghasilkan fungsi c (Rudy, 2005).

Konvolusi adalah salah satu proses *filtering image* yang sering dilakukan pada proses pengolahan gambar. Pada matlab terdapat banyak sekali cara yang dapat dilakukan untuk melakukan proses konvolusi. Proses konvolusi dilakukan dengan menggunakan matriks yang biasa disebut mask yaitu matriks yang berjalan sepanjang proses dan digunakan untuk menghitung nilai representasi lokal dari beberapa piksel pada *image* (Rudy, 2005).

Operasi konvolusi dilakukan perpixel dan untuk setiap *pixel* dilakukan operasi perkalian dan penjumlahan, sehinga memerlukan komputasi yang besar. Jika citra berukuran NxN dan kernel mxm, mk jumlah perkalian dlm orde N 2 m 2 Contoh, jk citra 512x512 dan kernel 16x16 maka ada sekitar 32 juta perkalian, tidak cocok untuk proses *real time*. Suatu cara mengurangi waktu komputasi adalah *mentransformasi* citra dan kernel ke dalam domain frekuensi dalam hal ini

Transf. *Fourier* keuntungan penggunaan domain *frekuensi* adalah proses konvolusi dapat diterapkan dalam bentuk perkalian langsung. g(x,y)= 1 2 1 2 4 2 1 2 1.

Pada *pixel-pixel* pinggir dapat diabaikan, tidak dikonvolusi (tetap). Duplikasi elemen citra, misalnya elemen kolom pertama disalin ke kolom M+1, begitu sebaliknya. Elemen kosong diasumsikan 0. Catatan: Ada masalah untuk pinggiran citra, hal ini dapat diatasi degan cara, dapat dilihat bahwa operasi konvolusi merupakan komputasi pada aras lokal, karena komputasi untuk suatu pixel pada citra keluaran melibatkan *pixel-pixel* tetangga pada citra masukannya. Konvolusi berguna pada proses pengolahan citra. Perbaikan kualitas citra Penghilangan derau Penghalusan atau pelembutan citra deteksi tepi, penajaman tepi dan lain-lain. Contoh: dilakukan konvolusi suatu citra fhoto hitam putih dengan penapis g*aussian* untuk mempertajam tepi-tepi di dlm citra. Penapis *gaussian* adalah sebuah mask berukuran 3x3 (Rudy, 2005).

Penapisan *(filtering)* termasuk pengolahan lokal yaitu dalam tranformasinya melibatkan:

1. Nilai-nilai pixel tetangganya
2. Nilai-nilai suatu sub-citra yang memiliki dimensi yangnilai nilai suatu sub citra yang memiliki dimensi yang sama. Sub-citra ini dikenal sebagai *filter, mask, kernel, template,* atau *window* atau *window*. Nilai dalam sub-citra tidak disebut sebagai nilai intensitas *pixel,* tetapi sebagai koefisien.

Untuk mengaplikasikan penapis pada citra, digunakan metode konvolusi. Konvolusi 2 fungsi f(x) dan g(x):

$$f\ x\ *\ g\ x\ = \int_{-\infty}^{\infty} f\ \alpha\ \ g\ x - \alpha\ \ d\alpha \ldots\ldots\ldots\ldots\ldots\ldots\ldots (9)$$

α = peubah bantu

Fungsi diskrit:

$$f\ x\ *\ g\ x\ = \sum\ f\ \alpha\ \ g\ x - \alpha\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots (10)$$

g(x) → convolution mask / filter / kernel atau template.

Notasi lain :

$$f(x,y) * g(x) = f(x,y) \circledast g(x) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (11)$$

Konvolusi bisa dinyatakan dalam matriks. Tiap elemen matriks penapisnya berupa koefisien konvolusi operasi menggeser karnel *pixel* per*pixel* -hasil disimpan dalam matriks baru (Rudy, 2005).

2.18 *Filter* atau Penapis

Filter atau penapis digunakan untuk proses pengolahan citra yaitu:

1. Perbaikan kualitas citra *(image enhancement)*
2. Penghilangan derau
3. Mengurangi erotan
4. Penghalusan atau pelembutan citra
5. Deteksi tepi, dan penajaman tepi

2.19 Segmentasi

Segmentasi gambar adalah pemisahan objek yang satu dengan objek yang lain dalam suatu gambar *(Ballerini).* Ada 2 macam segmentasi, yaitu *full segmentation* dan partial segmentation. *Full segmentation* adalah pemisahan suatu object secara individu dari *background* dan diberi ID (label) pada tiap-tiap segmen. Partial segmentation adalah pemisahan sejumlah data dari *background* dimana data yang disimpan hanya data yang dipisahkan saja untuk mempercepat proses selanjutnya (Rudy, 2005).

Ada 3 tipe dari segmentasi yaitu:

1. *Classification-based*: segmentasi berdasarkan kesamaan suatu ukuran dari nilai *pixel*. Salah satu cara paling mudah adalah thresholding. *Thresholding* ada 2 macam yaitu global dan lokal. Pada *thresholding* global, segmentasi berdasarkan pada sejenis histogram. Pada *thresholding* lokal, segmentasi dilakukan berdasarkan posisi pada gambar, gambar dibagi menjadi bagian-bagian yang saling melengkapi, jadi sifatnya dinamis.
2. *edge-based* yaitu mencari garis yang ada pada gambar dan garis tersebut digunakan sebagai pembatas dari tiap segmen.

3. *region-based* yaitu segmentasi dilakukan berdasarkan kumpulan *pixel* yang memiliki kesamaan (tekstur, warna atau tingkat warna abu-abu) dimulai dari suatu titik ke titik-titik lain yang ada disekitarnya (Rudy, 2005).

Segmentasi membangi citra ke dalam sejumlah region atau objek. Level untuk pembagian tergantung pada masalah yang diselesaikan. Meka, segmentasi seharusnya berhenti ketika objek yang diinginkan dalam aplkasi telah terisolasi. Misalnya, pemeriksaan otomatis pada rakitan produk elektronik, yang diinginkan adalah analisa citra produk dengan tujuan untuk mengetahui ada atau tidaknya penyimpangan tertentu, seperti salah komponen, atau lintasan hubungan yang putus.

Segmentasi citar *non-trival* adalah satu dari perkerjaan yang paling sulit dalam pengolahan citra. Akurasi segmentasi menentukan kemukinan sukses atau gagalnya komputerisasi prosedur analisis. Untuk alasan ini, perhatian seharusnya digunakan untuk meningkatkan kemukinan segmentasi yang kasar.

Algoritma segmentasi citra umumnya didasarkan pada satu dari dua properti nilai intensitas, diskontinuitas dan similaritas. Dalam katagori pertama, pendekatanya adalah memecah atau memilih citra berdasarkan perubahan kasar dalam intensitas, seperti tepi dalam citra. Pendekatan utama katagori kedua didasarkan pada pemecahan citra kedalam rigion yang sama menurut sejumlah kreteria yang didefinisikan, seperti *thresholding*, *region growing*, region *splitting* dan *merging* (Prasetyo, 2011).

2.19.1 Deteksi titik

Pendeteksian titik-titik terisolasi dalam citra adalah sederhana. Dengan menggunakan mask pada gambar, titik dideteksi pada lokasi di mana mask dipusatkan jika: $|R| \geq T$

di mana T adalah *threshold* non-negatif dan R diberikan oleh persamaan masking di atas. Dasarnya, formulasi ini mengukur perbedaan bobot antara titik pusat mask dan tetangganya. Ide bahwa *isolating point* (sebuah titik dimana gray levelnya secara signifikan berbeda dari *background*-nya dan ditempatkan dalam daerah *homogenous* atau dekat dengan *homogenous*) akan berbeda dari sekitarnya dan

dengan mudah dideteksi oleh mask jenis ini. Jumlah dari koefisien mask adalah nol. Artinya, akan memberikan nol pada daerah dengan nilai *gray level* kosntan (Prasetyo, 2011).

2.19.2 Deteksi Garis

Kompleksitas level berikutnya adalah deteksi garis. Jika mask pertama digerakan sepanjang citra, hasilnya akan lebih kuat pada garis (ketebalan satu piksel) yang berarah horizontal. Dengan bacgroun konstan, hasil maksimum akan didapatkan ketika garis yang dilewati sepanjang baris tengah dari mask (Prasetyo, 2011).

-1	-1	-1
2	2	2
-1	-1	-1

a. Horizontal

2	-1	-1
-1	2	-1
-1	-1	2

b. $+45^0$

-1	2	-1
-1	2	-1
-1	2	-1

c. vertikal

-1	-1	2
-1	2	-1
2	-1	-1

d. -45^0

Gambar 2.7. Mask untuk deteksi garis

(Sumber: Prasetyo, 2011)

jika R_1, R_2, R_3, dan R_4 menyatakan hasil mask pada gambar ditas dari kiri ke kanan, di mana nilai R diberikan oleh persamaan masking. Andaikan empat mask dijalankan secara individu terhadap citra. Jika titik tertentu dalam citra, $|Ri| > |Rj|$ untuk semua $j \neq i$, maka titik tersebut dikatakan lebih berasosiasi dengan garis dalam arah mask i. Misalnya, jika sebuah titik dalam citra $|Ri| > |Rj|$ untuk j = 2, 3, 4, bahwa titik tertentu dikatakan lebih berhubungan dengan garis horizontal. Alternatifnya, deteksi garis dapat dilakukan dalam arah tertentu. Dalam kasus ini, mask yang digunakan berasosiasi dengan arah dan threshold outputnya. Dengan

kata lain, jika ingin mendeteksi semua garis dalam citra dalam arah yang difinisikan mask, maka cukup hanya menjalankan mask sepanjang bidang citra dan men-threshold nilai absolut dari hasil. Titik-titik yang tersisa adalah hasil yang terkuat, di mana garis dengan ketabalan satu piksel berhubungan dengan arah yang difinisikan oleh mask (Prasetyo, 2011).

2.19.3 Deteksi Tepi

Meskipun deteksi titik dan garis penting untuk dibicarakan dalam segmentasi citra, deteksi tepi merupakan pendekatan yang paling umum untuk pendeteksian diskontiniutas nilai intensitas. Seperti diskontiniutas yang dideteksi oleh pengguna turunan pertama dan kedua. Pilihan turunan pertama dalam pengolahan citra adalah gradien. *Gradien* fungsi 2-D f(x,y) didefinisikan sebagai vektor.

$$\nabla f = \begin{matrix} Gx \\ Gy \end{matrix} = \begin{matrix} \frac{\partial}{\partial x} \\ \frac{\partial f}{\partial y} \end{matrix} \quad \text{.....(12)}$$

Jarak vektor ini adalah

$$\nabla f = amg\ \nabla f = G_x^2 + G_y^2 \quad \text{.....(13)}$$

$$= [\,(\partial f/\partial x)2 + (\partial f/\partial y)2\,]^2$$

Untuk menyederhanakan komputasi, persamaan ini kadang diperkirakan dengan mengabaikan operasi akar kuadrat:

$$\nabla f \approx G_x^2 + G_y^2 \quad \text{.....(14)}$$

Atau dengan absolutnya:

$$\nabla f \approx |G_x^2| + |G_x^2| \quad \text{.....(15)}$$

Perkiraan ini masih berlaku sebagai turunan, persamaan tersebut memberikan nilai nol pada daerah intensitas konstan dan nilainya proposonal terhadap tingkat perubahan intensitas dalam daerah di mana nilai piksel adalah

variabel. Umunya dalam praktek, penyebutan jarak gradien cukup dengan "gradien" saja (Prasetyo, 2011).

Properti dasar vektor gradien adalah bahwa titik-titik dalam arah rate maksimum dari perubahan f pada koordinat (x,y). Sudut di mana perubahan rate terjadi adalah :

$$\alpha\ x, y\ = \tan^{-1}\ \frac{Gx}{Gy} \quad \text{.....(16)}$$

Turunan kedua pegolahan citra umumnya dihitung menggunakan laplacian. Maka, fungsi laplacian 2-D *f(x,y)* dibentuk turunan kedua sebagai berikut:

$$\nabla^2 f(x,y) = \frac{\partial 2f(x,y)}{\partial x2} + \frac{\partial 2f(x,y)}{\partial y2} \quad \text{.....(17)}$$

Laplacian jarang digunakan sendirian untuk mendeteksi tepi, karena sebagai turunan kedua, laplacian tidak sensitif terhadap noise. Jaraknya menghasilkan tepi *double*, dan tidak bisa mendeteksi arah tepi. Tetapi laplacian bisa menjadi pelengkap yang powerfull ketika digunakan dalam kobimbinasi dengan teknik deteksi tepi yang lain. Misalnya, tepi *double* membuat ketidaksesuaian untuk deteksi tepi secara langsung. Properti ini dapat digunakan untuk *edge location* .

Ide dasar di balik deteksi tepi adalah untuk mencari tempat di dalam citra di mana intensitas berubah secara cepat, menggunakan satu dari dua kriteria umum berikut:

1. mencari tempat di mana turunan pertama intensitas lebih besar dalam jarak dari pada threshold yang ditetapkan.
2. Mencari tempat di mana turunan kedua dari intensitas mempunya *zero crossing*

-1	-2	-1
0	0	0
1	2	1

a. $G_{x=}(z_7+2z_8+z_9)-(z_1+2z_2+z_3)$

-1	0	1
-2	0	2
-1	0	1

b. $G_{y=}(z_3+2z_6+z_9)-(z_1+2z_4+z_7)$

-1	-1	-1
0	0	0
1	1	1

c. $G_{x=}(z_7+2z_8+z_9)-(z_1+2z_2+z_3)$

-1	0	1
-1	0	1
-1	0	1

d. $G_{y=}(z_3+2z_6+z_9)-(z_1+2z_4+z_7)$

-1	0
0	-1

e.$G_{x=}\ z_9+z_5$

0	-1
1	0

f.$G_{x=}\ z_8+z_6$

Gambar 2.8 *Detektor tepi: (a) – (b) Sobel; (c) – (d) Prewitt; (e) – (f) Robert*
(Sumber: Prasetyo, 2011)

Fungsi edge memberikan beberapa estimator turunan berdasarkan pada kriteria di atas. Beberapa estimator ini memukinkan untuk menspesifikasikan apakah detektor tepi sensitif terhadap tepi horizontal atau vertikal atau keduanya. Sintaks umum yang digunakan untuk fungsi ini adalah:

[g, t] = edge(f,'method', parameters)

Di mana f adalah citra input, method adalah satu dari pendekatan dalam tabel di atas, parameter adalah parameter tambahan. *Output* g adalah *array* logika dengan nilai 1 pada lokasi di mana titik-titik tepi terdeteksi dalam f dan nilai 0 untuk yang tidak terdeteksi. Parameter t adalah opsional; memberikan threshold yang digunakan oleh tepi untuk menentukan di mana gradien cukup kuat untuk disebut titik-titik tepi (Prasetyo, 2011).

2.19.4 Detektor Tepi sobel

Detektor Tepi sobel yang merupakan mask dalam gambar 2.8 (a) dan (b) untuk memperkirakan secara digital turunan G_x dan G_y. Dengan kata lain, gradien pada titik pusat ketetanggan dihitung oleh detektor sobel sebagai berikut:

$$g = \left[G_x^2 + G_y^2 \right]^{1/2} \quad \text{.....(18)}$$

$$g = \{[(Z_7 + 2_{z8} + z_9) - (z_1 + 2z_2 + z_3)]^2$$
$$+ [(z_{3+} 2_{z6} + z_9) - (z_1 + 2z_4 + z_7)]^2\}^{1/2}$$

Maka dapat dikatakan bahwa piksel pada lokasi (x,y) adalah piksel tepi jika $g \geq T$ pada lokasi tersebut, di mana T adalah threshild yang ditentuka. Deteksi sobel dapat diimplementasikan oleh pemilteran citra f (menggunakan imfilter) dengan mask pada gambar 2.8 (a), pengkuadratan nilai piksel dari setiap citra filter, menambahkan dua hasilnya dan menghitung akar kuadratnya. Fungsi edge adalah paket fungsi beberapa operasi dan menambah *figure* lain, seperti menerima nilai *threshold*, penetuan *threshold* secara otomatis. Fungsi edge berisi teknik deteksi tepi yang tidak diimplementasikan secara langsung dengan *imfilter*.

Sintaks umum detektor sobel:

[g, t] = edge(f,'sobel',T,dir) (Prasetyo, 2011).

Di mana f adalah citra input, T adalah *threshold*, dan dir menetapkan arah yang lebih disukai pada citra terdeteksi, 'horizontal', 'vertical', atau 'both' (default). Sedangakan g adalah citra biner yang berisi nilai 1 pada lokasi dimana tepi terdeteksi dan 0 untuk yang baik. Parameter t pada output bersifat opsional, t adalah nilai *threshold* yang digunakan oleh fungsi edge. Jika T ditetapkan, maka t = T. Selain itu, jika T tidak ditetapkan (atau kosong, []), fungsi *edge* mengisi t sama dengan *thershold* yang ditentukan secara otomatis dan kemudian menggukannya untuk deteksi tepi. Satu alasan utama untuk memasukkan t dalam argumen output adalah untuk mendapatkan nilai awal *threshold*. Fungsi *edge* menggunakan detektor sobel sebagai default jika sintaks g= edge (f) atau [g,t] = edge(f) yang digunakan (Prasetyo, 2011).

2.20 *Slope Histogram*

Slope Histogram adalah histogram yang mencatat *gradien* atau kemiringan dari tiap pixel berwarna hitam dalam sebuah gambar. *Slope Histogram* dapat digunakan untuk mengenali objek dengan cara membandingkan *histogra*m dari data asli dengan *histogram* dari objek yang ingin dikenali (Rudy, 2005)

Untuk dapat menggunakan *slope histogram*, sebuah gambar sebaiknya diubah dahulu menjadi biner. Hal ini dapat dilakukan dengan menggunakan *threshold*. Setelah gambar siap, maka gradien atau kemiringan tiap *pixel* yang berwarna hitam dicari. Di sini dapat digunakan matriks yang memetakan *pixel* tersebut dengan pixel-pixel berwarna hitam yang ada disekelilingnya. Matriks yang digunakan adalah 5 x 5.

Persamaan garis dalam matematika adalah sebagai berikut:

$$ax+by+c=0$$

di mana a, b, dan c adalah parameter yang menjelaskan posisi garis dan orientasinya, dan x,y adalah koordinat horisontal dan vertikal. Nilai a dapat dicari dengan menggunakan rumus cos(θ + π/2) dan b dapat dicari dengan (1 – a * a)1/2 Sedangkan konstanta c dapat dicari dari c = -ax –by. Garis yang ”terbaik” didapat dengan mencoba semua sudut antara -90 hingga 90 derajat dengan asumsi bahwa garis harus melalui paling sedikit satu dari *pixel* berwarna gelap dalam matriks 5x5. Nilai error didapatkan dari jumlah ax+by+c untuk setiap x dan y yang pixel-nya berwarna hitam. Jika semua *pixel* berwarna hitam berada pada 1 garis maka nilai error-nya adalah 0 dan dengan memilih garis dengan nilai error paling minimum, maka garis yang ”terbaik” dapat dipilih. Nilai pada histogram milik sudut dari garis yang ”terbaik” di-increment dan dengan demikian terbentuklah *slope histogram* (Rudy, 2005).

Untuk membandingkan 2 *slope histogram* dapat digunakan distance yang didefinisikan:

$$D = \frac{a \bullet b}{a \bullet b + b \bullet b - a \bullet b}$$

Dimana *dot product* dari a dan b didefinisikan:

$$a \bullet b = \sum_{i-1}^{N} a_{i\,bi}$$

Nilai d dapat dikatakan nilai kemiripan dari a dan b (Rudy, 2005).

2.21 Sistem Pakar

Secara umum, sistem pakar merupakan sistem yang mengadopsi pengetahuan manusia ke dalam komputer sehingga komputer dapat digunakan untuk

menyelesaikan suatu masalah sebagaimana yang dilakukan oleh seorang pakar. Sistem pakar dibuat pada wilayah pengetahuan tertentu dan untuk suatu keahlian tertentu yang mendekati kemampuan manusia di salah satu bidang khusus. Sistem pakar mencoba mencari solusi yang memuaskan sebagaimana yang dilakukan seorang pakar dan dapat memberikan penjelasan terhadap langkah yang diambil serta memberikan alasan atas kesimpulan yang diambil (Rudy, 2005).

2.22 Keuntungan Sistem Pakar

Secara garis besar, ada banyak keuntungan bila menggunakan sistem pakar, diantaranya (Arhami, 2005).

1. Menjadikan pengetahuan dan nasihat lebih mudah didapat.
2. Meningkatkan output dan produktivitas.
3. Menyimpan kemampuan dan keahlian pakar.
4. Meningkatkan penyelesaian masalah yaitu menerusi paduan pakar, penerangan, sistem pakar khas.
5. Meningkatkan reliabilitas.
6. Memberikan *respons* (jawaban) yang cepat.
7. Merupakan panduan yang intelligence (cerdas).
8. Dapat bekerja dengan informasi yang kurang lengkap dan mengandung ketidakpastian.
9. *Intelligence* database (basis data cerdas), bahwa sistem pakar dapat digunakan untuk mengakses basis data dengan cara cerdas.

2.23 Kelemahan Sistem Pakar

Disamping memiliki beberapa keuntungan, sistem pakar juga memiliki beberapa kelemahan, antara lain:

1. Biaya yang diperlukan untuk membuat dan memeliharanya sangat mahal.
2. Sulit dikembangkan system pakar yang benar-benar berkualitas tinggi. Hal ini tentu saja erat kaitannya dengan ketersediaan pakar di bidangnya.
3. Sistem pakar tidak dapat 100% bernilai benar.

4. Terkadang sistem tidak dapat membuat keputusan.
5. Pengetahuan tidak selalu didapat dengan mudah karena pendekatan tiap pakar berbeda.

2.24 Karakteristik Sistem Pakar

Sistem pakar mempunyai beberapa karakteristik dasar yang membedakan dengan program komputer biasa umumnya, yaitu (Turban, 1995).

1. Mempunyai kepakaran dalam menyelesaikan masalah bukan hanya mendapatkan solusi yang benar saja, namun juga bagaimana mendapatkan pemecahan dengan cepat dan mahir.
2. Domain tertentu sistem pakar mengutamakan kedalaman mengenai bidang tertentu.
3. Memiliki kemampuan mengolah dat yang mengandung ketidak pastian kadang-kadang data yang tersedia tidak lengkap sistem harus dapat memberikan pemecahan sesuai data yang tersedia dengan memberikan pertimbangan, saran atau anjuran sesuai dengan kondisi yang ada.
4. Dirancang untuk dapat dikembangkan secara bertahap program komputer dirancang untuk memberikan jawaban yang tepa setiap waktu. Sedangkan sistem pakar dirancang untuk berlaku sebagai seoran pakar, kadang memberikan jawaban yang benar, dan suatu saat mungkin tidak tepat *(Expert system makes mistake).*

2.25 Komponen Sistem Pakar

Komponen-komponen yang terdapat dalam sistem pakar adalah :

1. Antarmuka pengguna *(User Interface)*

   User interface merupakan mekanisme yang digunakan untuk pengguna dan sistem pakar untuk berkomunikasi. Antarmuka menerima informasi dari pemakai dan mengubahnya ke dalam bentuk yang dapat diterima oleh sistem. Selain itu antarmuka menerima informasi dari system dan menyajikannnya dalam bentuk yang dapat dimengerti oleh pemakai. Menurut McLeod (1995), pada bagian ini terjadi dialog antara program

dan pemakai, yang memungkinkan sistem pakar menerima instruksi dan informasi *(input)* dan pemakai juga memberikan informasi *(output)* kepada pemakai.

2. Basis pengetahuan

   Basis pengetahuan mengandung pengetahuan untuk pemahaman, formulasi, dan penyelesaian masalah. Komponen sistem pakar ini disusun atas dua elemen dasar, yaitu fakta dan aturan. Fakta merupakan informasi tentang *obyek* dalam area permasalahan tertentu, sedangkan aturan merupakan informasi tentang cara bagaimana memperoleh fakta baru dari fakta yang telah diketahui (Arhami, 2005).

3. Akuisisi pengetahuan

   Akusisi pengetahuan adalah akumulasi, transfer dan transformasi keahlian dalam menyelesaikan masalah dari sumber pengetahuan ke dalam program komputer. Dalam tahap ini knowledge *engineer* berusaha menyerap pengetahun untuk selanjutnya ditransfer ke dalam basis pengetahuan. Pengetahuan diperoleh dari pakar, dilengkapi dengan buku, basis data, laporan penelitian dan pengalaman pemakai (Arhami, 2005).

Akuisisi pengetahuan dilakukan sepanjang proses pembangunan sistem. Menurut (Firebaugh 1989). Proses akuisisi pengetahuan dibagi ke dalam enam tahap, yaitu :

1. Tahap identifikasi

   Tahap identifikasi meliputi penentuan komponen-komponen kunci dalam sistem yang sedang dibangun. Komponen kunci ini adalah *knowledge engineer*, pakar, karakterisitik masalah, sumber daya, dan tujuan. *Knowledge engineer* dan pakar bekerja bersama untuk menentukan berbagai aspek masalah, seperti lingkup dari proyek, data input yang dimasukkan, bagian-bagian penting dan interaksinya, bentuk dan isi dari penyelesaian, dan kesulitan-kesulitan yang mungkin terjadi dalam pembangunan sistem. Mereka juga harus menentukan sumber pengetahuan seperti basis data, system informasi manajemen, buku teks, serta prototipe masalah dan contoh. Selain menentukan sumber

pengetahuan, pakar juga mengklarifikasi dan menentukan tujuan-tujuan sistem dalam proses penentuan masalah.

2. Tahap konseptualisasi

   Konsep-konsep kunci dan hubungannya yang telah ditentukan pada tahap pertama dibuat lebih jelas dalam tahap konseptualisasi.

3. Tahap formalisasi

   Tahap ini meliputi pemetaan konsep-konsep kunci, sub-masalah dan bentuk aliran informasi yang telah ditentukan dalam tahap-tahap sebelumnya ke dalam representasi formal yang paling sesuai dengan masalah yang ada.

4. Tahap implementasi

   Tahap ini meliputi pemetaan pengetahuan dari tahap sebelumnya yang telah diformalisasi ke dalam skema representasi pengetahuan yang dipilih.

5. Tahap pengujian

   Setelah prototipe sistem yang dibangun dalam tahap sebelumnya berhasil menangani dua atau tiga contoh, prototipe sistem tersebut harus menjalani serangkaian pengujian dengan teliti menggunakan beragam *sampel* masalah. Masalah-masalah yang ditemukan dalam pengujian ini biasanya dapat dibagi dalam tiga kategori, yaitu kegagalan *input/output*, kesalahan logika dan strategi kontrol.

6. Revisi prototipe

   Suatu unsur penting pada semua tahap dalam proses akuisisi pengetahuan adalah kemampuan untuk kembali ke tahap-tahap sebelumnya untuk memperbaiki sistem.

4. Mesin inferensi

   Komponen ini mengandung mekanisme pola pikir dan penalaran yang digunakan oleh pakar dalam menyelesaikan suatu masalah. Mesin inferensi adalah program komputer yang memberikan metodologi untuk penalaran tentang informasi yang ada dalam basis pengetahuan dan

dalam workplace, dan untuk memformulasikan kesimpulan (Turban, 1995).

Inferensi merupakan proses menghasilkan kesimpulan berdasarkan fakta atau pengetahuan yang diketahui atau diasumsikan. Terdapat dua pendekatan untuk mengontrol inferensi dalam sistem pakar berbasis aturan yaitu pelacakan ke depan *(forward chaining).*

Pelacakan ke Depan *(Forward Chaining).* Pada metode *forward chaining* di artikan sebagai pendekatan yang dimotori data. Dalam pendekatan ini pelacakan dimulai dari informasi masukan, dan selanjutnya mencoba menggambarkan kesimpulan. Sehingga metode ini juga sering disebut "Data *driven"*.

Forward Chaining merupakan pencocokan fakta atau pernyataan dimulai dari bagian sebelah kiri dulu *(IF* dulu). Dengan kata lain penalaran dimulai dari fakta terlebih dahulu untuk menguji kebenaran hipotesis (Kusumadewi, 2013).

2.26 Konsep Dasar Sistem Pakar

Konsep dasar sistem pakar mengandung (Kusumadewi, 2013).

1. Keahlian.

   Keahlian bersifat luas dan merupakan penguasaan pengetahuan dalam bidang khusus yang diperoleh dari pelatihan, membaca atau pengalaman. Contoh bentuk pengetahuan yang termasuk keahlian:

   Teori, fakta, aturan-aturan pada lingkup permasalahan tertentu

   Strategi global untuk menyelesaikan masalah

2. ahli/pakar.

   Seorang ahli adalah seseorang yang mampu menjelaskan suatu tanggapan, mempelajari hal-hal baru seputar topik permasalahan, menyusun kembali pengetahuan jika dipandang perlu, memecahkan masalah dengan cepat dan tepat.

3. pengalihan keahlian.

   Tujuan dari sistem pakar adalah untuk mentransfer keahlian dari seorang pakar ke dalam komputer kemudian ke masyarakat. Proses ini meliputi 4 kegiatan, yaitu:

a. Perolehan pengetahuan (dari para ahli atau sumber-sumber lainnya),
b. Representasi pengetahuan ke komputer.
c. Kesimpulan dari pengetahuan dan
d. Pengalihan pengetahuan ke pengguna.

4. Mengambil keputusan.

   Hal yang unik dari sistem pakar adalah kemampuan untuk menjelaskan dimana keahlian tersimpan dalam basis pengetahuan.

   Kemampuan komputer untuk mengambil kesimpulan dilakukan oleh komponen yang dikenal dengan mesin inferensi yaitu meliputi prosedur tentang pemecahan masalah.

5. Aturan.

   Sistem pakar yang dibuat merupakan sistem yang berdasarkan pada aturan–aturan dimana program disimpan dalam bentuk aturan-aturan sebagai prosedur pemecahan masalah. Aturan tersebut biasanya berbentuk *IF – THEN*.

6. kemampuan menjelaskan.

   Keunikan lain dari sistem pakar adalah kemampuan dalam menjelaskan atau memberi saran/rekomendasi serta juga menjelaskan mengapa beberapa tindakan atau saran tidak direkomendasikan (Kusumadewi, 2013).

2.27 Komponen-Komponen yang Terdapat Dalam Arsitektur atau Struktur Sistem Pakar :

1. Antarmuka Pengguna *(User Interface)*

   Merupakan mekanisme yang digunakan oleh pengguna dan sistem pakar untuk berkomunikasi.

2. Basis Pengetahuan

   Basis pengetahuan mengandung pengetahuan untuk pemahaman, formulasi, dan penyelesaian masalah.

   Komponen sistem pakar ini disusun atas 2 elemen dasar, yaitu :

a. fakta : informasi tentang obyek dalam area permasalahan tertentu.
b. aturan : informasi tentang cara bagaimana memperoleh fakta baru dari fakta yang telah diketahui.

3. Akuisisi Pengetahuan *(Knowledge Acquisition)*

Akuisisi pengetahuan adalah akumulasi, transfer, dan transformasi keahlian dalam menyelesaikan masalah dari sumber pengetahuan ke dalam program komputer..

4. Mesin atau motor *Inferensi (inference engine).*

Komponen ini mengandung mekanisme pola pikir dan penalaran yang digunakan oleh pakar dalam menyelesaikan suatu masalah.

5. *Workplace / Blackboard*

Workplace merupakan area dari sekumpulan memori kerja *(working memory)*, digunakan untuk merekam kejadian yang sedang berlangsung termasuk keputusan sementara.

6. Fasilitas Penjelasan

Adalah komponen tambahan yang akan meningkatkan kemampuan sistem pakar.

7. Perbaikan Pengetahuan

Pakar memiliki kemampuan untuk menganalisis dan meningkatkan kinerjanya serta kemampuan untuk belajar dari kinerjanya (Kusumadewi, 2013).

2.28 Basis Pengetahuan *(Knowledge Base)*

Basis pengetahuan berisi pengetahuan-pengetahuan dalam penyelesaian masalah. Ada 2 bentuk pendekatan basis pengetahuan :

1. Penalaran berbasis aturan *(rule-based reasoning)* Pengetahuan direpresentasikan dengan menggunakan aturan berbentuk *IF-THEN.*
2. Penalaran berbasis kasus *(case-based reasoning)*

   Pada penalaran berbasis kasus, basis pengetahuan akan berisi solusi-solusi yang telah dicapai sebelumnya, kemudian akan diturunkan suatu

solusi untuk keadaan yang terjadi sekarang (fakta yang ada) (Kusumadewi, 2013).

2.29 Manfaat Sistem Pakar

1. Memungkinkan orang awam bisa mengerjakan pekerjaan para ahli
2. Bisa melakukan proses secara berulang secara otomatis
3. Menyimpan pengetahuan dan keahlian para pakar
4. Mampu mengambil dan melestarikan keahlian para pakar (terutama yang termasuk keahlian langka)
5. Mampu beroperasi dalam lingkungan yang berbahaya
6. Memiliki kemampuan untuk bekerja dengan informasi yang tidak lengkap dan mengandung ketidakpastian. Pengguna bisa merespon dengan jawaban ’tidak tahu’ atau ’tidak yakin pada satu atau lebih pertanyaan selama konsultasi dan sistem pakar tetap akan memberikan jawaban.
7. Tidak memerlukan biaya saat tidak digunakan
8. Dapat digandakan (diperbanyak) sesuai kebutuhan dengan waktu yang minimal dan sedikit biaya
9. Dapat memecahkan masalah lebih cepat daripada kemampuan manusia dengan catatan menggunakan data yang sama.
10. Menghemat waktu dalam pengambilan keputusan
11. Meningkatkan kualitas dan produktivitas karena dapat memberi nasehat yang konsisten dan mengurangi kesalahan
12. Mampu menyediakan pelatihan. Pengguna pemula yang bekerja dengan sistem pakar akan menjadi lebih berpengalaman. Fasilitas penjelas dapat berfungsi sebagai guru (Kusumadewi, 2013).

2.30 Struktur Sistem Pakar

Ada 2 bagian utama sistem pakar :

1. lingkungan pengembangan *(development environment):* digunakan untuk memasukkan pengetahuan pakar ke dalam lingkungan sistem pakar

2. lingkungan konsultasi *(consultation environment)*: digunakan oleh pengguna yang bukan pakar untuk memperoleh pengetahuan pakar (Kusumadewi, 2013).