BAB II

LANDASAN TEORI

2.1 Citra

Secara harafiah, citra (*image*) adalah gambar pada bidang dwimatra (dua dimensi). Ditinjau dari sudut pandang matematis, citra merupakan fungsi kontinu 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 optic, misalnya mata manusia, kamera, scanner, dsb, sehingga bayangan objek yang disebut citra tersebut terekam.

2.2 Representasi Citra Digital

Citra digital adalah sebuah citra f(x,y) yang telah di-diskretasi ke dalam koordinat spasial dan tingkat keabuan. Citra digital dinyatakan sebagai sebuah matrik NxM yang terdiri atas baris dan kolom untuk menyatakan sebuah titik pada citra dan elemen nilai matrik yang berupa nilai diskret menyatakan tingkat keabuan pada titik tersebut. Citra digital tiap elemen dikenal sebagai elemen gambar (picture element) atau pixel.

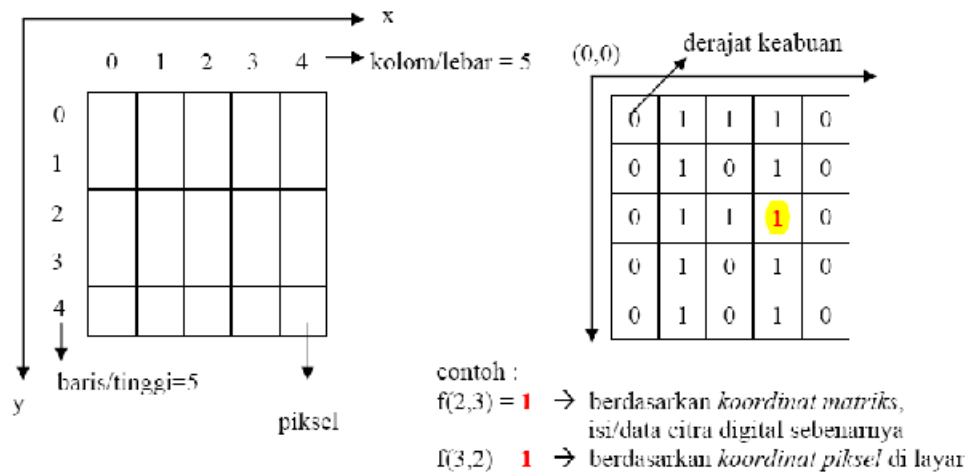
(baris/tinggi = N, kolom/lebar = M)

N = jumlah baris $0 = y = N - 1$

M = jumlah kolom $0 = x = M - 1$

L = maksimal warna intensitas (derajat keabuan /gray level) $0 = f(x,y) = L - 1$

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



Gambar 2.1. Matrik Citra Digital

Untuk mendapatkan suatu citra digital diperlukan suatu konversi sehingga dapat diproses oleh komputer. Proses konversi tersebut dengan membuat kisi-kisi arah horizontal dan vertical sehingga diperoleh gambar dalam bentuk array 2 dimensi. Proses tersebut disebut proses digitasi atau sampling.

Semakin tinggi resolusi berarti semakin kecil ukuran pixelnya, berarti semakin halus gambar yang diperoleh karena informasi yang hilang akibat pengelompokan tingkat keabuan pada proses pembuatan kisi-kisi semakin kecil. Tetapi membutuhkan tempat penyimpanan bit yang makin besar pula.

Proses selanjutnya adalah proses kuantisasi. Kuantisasi ialah tingkat keabuan setiap pixel dinyatakan dengan suatu harga integer. Jika digambarkan 3 bit untuk menyimpan harga integer maka diperoleh 8 tingkat keabuan. Makin besar jumlah tingkat keabuan yang digunakan makin baik gambar yang diperoleh. Tetapi sebagai konsekuensinya jumlah memori yang digunakan semakin besar.

Kelemahan data digital adalah kualitasnya yang cenderung menjadi lebih rendah bila dibandingkan data dalam bentuk analog, ini dikarenakan sifatnya yang berbentuk diskrit (fungsi tangga). Sehingga banyak bagian-bagian tertentu yang hilang atau mengalami pembulatan. Karena itu frekuensi sampling dan level kuantisasi yang diberlakukan dalam proses konversi sangat memegang peranan penting dalam menentukan kualitas data digital. Semakin tinggi frekuensi sampling yang digunakan dan semakin besar level kuantisasi maka semakin baik kualitas data digital tersebut.

Di samping kelemahan diatas, data dalam bentuk ini mempunyai banyak kelebihan dibandingkan data analog, sifatnya yang diskret menyebabkannya tidak dipengaruhi oleh gangguan (noise) dari luar sehingga kualitas data tetap terjaga, selain itu dengan mudah dapat dilakukan proses pemadatan data sehingga media penyimpanan lebih efisien, di samping itu dapat pula dilakukan proses restorasi apabila data tersebut mengalami kerusakan dan lain-lain.

2.3 File citra digital dengan format BMP

File citra dengan format BMP adalah salah satu format standar yang digunakan dalam sistem operasi Windows. File ini biasanya disimpan dengan ekstensi .BMP. Struktur dari file BMP adalah sebagai berikut:

Tabel 2.1. Struktur file bmp

BITMAPFILEHEADER
BITMAPINFOHEADER
Array RGBQUAD
Array warna

Struktur BITMAPFILEHEADER menunjukkan jenis file dan ukuran file. Struktur BITMAPINFOHEADER menunjukkan ukuran (lebar dan tinggi) dari citra, format warna citra, dan banyaknya warna yang digunakan di dalam citra. Struktur RGBQUAD berisi nilai intensitas warna yang dibagi menjadi komponen R, G, dan B (red, green, dan blue). Sedangkan array warna berisi informasi mengenai isi citra itu sendiri, yaitu pixel-pixel yang membentuk citra tersebut. Untuk mempercepat proses pengolahan citra digital, maka pixel-pixel yang ada di dalam file BMP dicopykan ke dalam variabel array. Dengan demikian pada proses selanjutnya tidak perlu dilakukan pembacaan file secara berulang-ulang. Deklarasi array untuk menyimpan citra tersebut adalah:

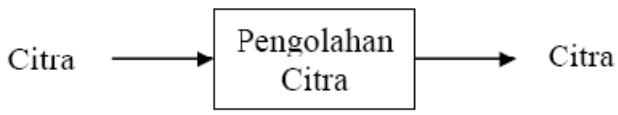
```
arCitra1 : array[1..256, 1..256] of byte;
arCitra2 : array[1..256, 1..256] of byte;
```

Secara lengkap, proses peng-copy-an pixel ini dapat dilihat pada *procedure BukaFileCitra* di listing program terlampir. Untuk menampilkan citra ke layar, digunakan component TImage dari Delphi.

2.4 Pengolahan Citra

Suatu citra digital melalui pengolahan citra digital (digital image processing) menghasilkan citra digital yang baru. Proses pengolahan citra digital ini contohnya adalah perbaikan citra (image restoration), peningkatan kualitas citra (image enhancement), pemfilteran, operasi cropping, perbesaran dan lain sebagainya yang berhubungan dengan operasi pengolahan citra. Dalam proses analisis citra digital (digital image analysis) menghasilkan suatu data yang termasuk di dalamnya adalah pengenalan pola (pattern recognitio) proses analisis ini contohnya adalah histogram.

Pengolahan citra bertujuan memperbaiki kualitas citra agar mudah diinterpretasi oleh manusia atau mesin (dalam hal ini komputer). Teknik-teknik pengolahan citra mentransformasikan citra menjadi citra lain. Jadi, masukannya adalah citra dan keluarannya juga citra, namun citra keluaran mempunyai kualitas lebih baik dari pada citra masukan.



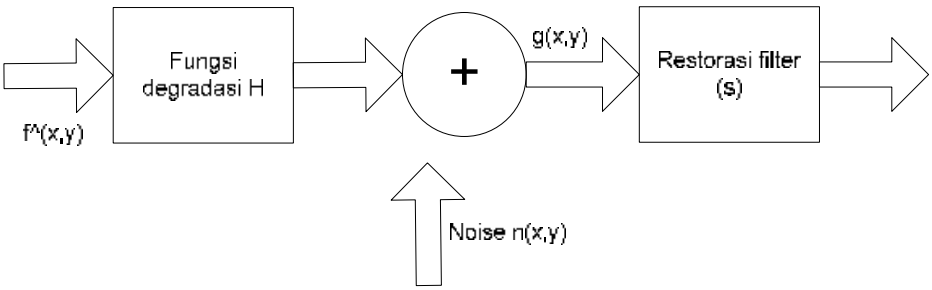
Gambar 2.2. Pengolahan citra

2.5 Restorasi citra

Tujuan terakhir dari teknik pemulihan gambar ini untuk memperbaiki gambar itu sendiri sesuai dengan beberapa pendefinisian rasa(sense). Teknik restorasi ini berorientasi pada memodelkan pendegradasian dan penggunaan proses kebalikan(inverse) dengan tujuan untuk memperoleh gambar aslinya.

Pendekatan ini umumnya berkaitan dengan memformulakan kriteria kebaikan yang akan menghasilkan perkiraan optimal dari hasil yang diinginkan. Perbedaannya, teknik mempertinggi sebenarnya adalah prosedur heuristic yang di desain untuk memanipulasi sebuah gambar dengan tujuan untuk mendapatkan keuntungan dari aspek psychophysical dari system penglihatan manusia.

Restorasi juga menjelaskan tentang memperbaiki kualitas sebuah citra dengan proses matematis dan probabilitas. Berbeda dengan image enhancement yang bersifat subjektif, karena proses perbaikan citranya berdasar pada perasaan manusia. Dibawah ini gambar dan persamaan matematis proses restorasi :



Gambar 2.3. Ilustrasi Proses Restorasi

$$g(x,y) = h(x,y) + n(x,y) \quad (2.1)$$

Keterangan:

g(x,y) = citra yang mengalami degradasi

f(x,y) = citra asli

n(x,y) = additive noise

h(x,y) = fungsi degradasi

2.6 Fungsi *Mumford Shah Image*

Masalah penting dalam image processing adalah masalah memperkirakan gambar utama dari sebuah citra yang mempunyai *blur* dan *noise*. Akhir-akhir ini, usaha keras telah dihabiskan dalam mendesain teknik alternative yang memelihara batas antara daerah, sedangkan masih dapat dilakukan tahap *regularization* yang kokoh dari *inverse problem.* Pada akhirnya, penulis mengimplementasikan sebuah algoritma untuk *deblurring* dan *denoising* citra.

Salah satu pendekatan restorasi citra adalah dengan teknik evolusi kurva. Tujuan dari teknik ini adalah untuk membatasi (yang direpresentasikan oleh kurva tertutup) dalam sebuah citra. Akhir-akhir ini, sebuah pendekatan kurva evolusi baru berdasarkan minimasi fungsi mumford shah telah diajukan untuk *smoothing* dari citra yang mempunyai blur dan noise. Citra smoothing yang diajukan pada tugas akhir ini adalah linear, dengan memelihara sisi global citra dengan kurva.

Dalam tugas akhir ini, penulis akan mendemontrasikan bagaimana memperluas teknik dalam memelihara sisi pada algoritma *denoising* dan *deblurring*. Hal yang utama adalah solusi dalam restorasi image pada kurva evolusi, yang menuju pada bagaimana cara merestorasi citra.

Persamaan fungsi mumford shah adalah sebagai berikut :

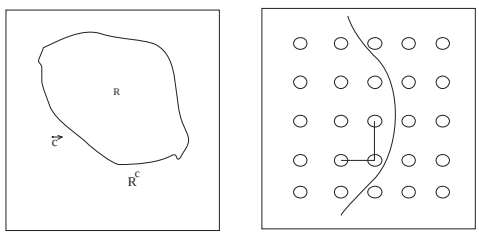
$$Eo(f,\vec{C})=\beta \iint_{\Omega} (f-g)^2\, dA+\alpha \iint_{\Omega\setminus\vec{C}} |\nabla f|^2\, dA+\gamma \oint_{\vec{C}} dS \qquad (2.2)$$

Dimana $\vec{C}$ merupakan kurva tertutup sederhana, g merupakan data yang sedang diamati, f merupakan bagian smooth approximation terhadap g yang diskontinuitas pada kurva $\vec{C}, \alpha$,ß dan ? adalah positive weighting scalar dan ? merupakan domain dari citra. Dari persamaan (2.1) dan (2.2) fungsi mumford shah menjadi:

$$E(f,\vec{C}) = \beta \iint_{\Omega} (h * f - g)^2\, dA + \alpha \iint_{\Omega\setminus\vec{C}} |\nabla f|^2\, dA + \gamma \oint_{\vec{C}} dS \qquad (2.3)$$

Persamaan fungsi diataslah yang akan diminimasi

2.6.1 Minimasi dari perluasan Fungsi Mumford shah



(a) (b)

Gambar 2.4. Ilustrasi kurva

gambar (a) mengilustrasikan kurva $\vec{C}$ daerah didalam kurva (R) , daerah diluar kurva (R^c) dan gambar (b) mengilustrasikan batas titik dalam dan luar yang dipisahkan oleh kurva.

Dari persamaan (2.3), penulis akan mengimplementasikan untuk meminimasi perluasan fungsi dari *mumford shah*. Pertama persamaan yang didapat akan didiskritkan yang dimodelkan dengan persamaan :

$$g = Hf + n \qquad (2.4)$$

dimana f dan g merupakan vector yang bernilai real (f dan g terdiri dari himpunan pixel pada citra yang diamati), H adalah matrik *blurring*, dan n adalah *white Gaussian noise.*

Persamaan diskrit dari perluasan fungsi mumford – shah adalah :

$$E(f, \vec{C}) = ß \| Hf - g \|^2 + \alpha \sum_{(i,j) \in S} f_i - f_{j+} ? \oint_{\vec{C}} dS \qquad (2.5)$$

dimana S = { (i,j) ¦ *i,j* adalah pixel yang bertetanggaan dan *i,j* di kurva yang sama. Pada gambar 2.3 (b) terlihat bahwa (i,j) dan (i,k) berada di S dan (i,l) dan (i,m) berada diluar S.

Masalahnya sekarang adalah untuk mencari f dan $\vec{C}$ untuk meminimasikan E ($f, \vec{C}$), penulis mengajukan skema optimasi iterasi dimana masing – masing terdiri dari dua langkah. Langkah pertama adalah menghitung f yang meminimasikan E ($f, \vec{C}$) dari kurva $\vec{C}$. langkah kedua adalah mengembangkan kurva dengan $\vec{C}$. algoritma diatas dapat disimpulkan sebagai berikut :

inisialisasi $\vec{C}$;

while (not konvergen)

{

solve f = arg min $_f$ E (f, $\vec{C}$) for fixed $\vec{C}$; //penjelasan di 2.6.2

evolve the curve with $\vec{C}_t$; // penjelasannya di 2.6.3

}

Dimana kriteria konvergen adalah

$\| f_{prev} - f_{new} \| < \epsilon$ dimana ϵ adalah konstanta positif yang amat kecil.

2.6.2 Estimasi dengan Persamaan Linear

Kita dapat mencari f dengan menghitung turunan pertama dari E terhadap f_i untuk setiap *i*. misal h_i adlah kolom ke- i dari matrik H, maka Hf = ? $_i$ hi fi .

Maka turunan pertama E terhadap f_i adalah sebagai berikut :

$$\frac{\partial E}{\partial fi} = 2 \beta \sum_j h_i{}^T h_j \; f_j - 2 \beta h_i{}^T g + 2 \alpha \sum_{j \; s.t. \; (i,j) \in S} (f_i - f_i) \qquad (2.6)$$

Kemudian pada kondisi $\partial H / \partial f_i$ = untuk setiap i adalah

$$\sum_j h_i{}^T h_j f_j - \frac{\alpha}{\beta} \sum_{j \; s.t. \; (i,j) \in S} (f_j - f_i) = h_i{}^T g \qquad (2.7)$$

akhirnya, kondisi pada f dapat diberikan (dalam persamaan linear)sebagai berikut, dimana baris ke – i dari f dapat dilihat dari persamaan (2.7) :

$H^T Hf - \frac{\alpha}{\beta} Af = H^T g,$

Dimana

$$Aij = \begin{cases} 1 & if(i,j) \in S \\ -\sum_{k\ s.t.(i,k)\in S} A_{ik} & if\ i = j \\ 0 & otherwise \end{cases}$$

Perhatikan bahwa pada matriks A adalah posisi sekarang pada kurva.

2.6.3 Evolusi Kurva

Ide dari evolusi kurva untuk memindahkan kurva pada sebuah titik diatas index *i* (dimana intensitasnya adalah f_i) disekitar kurva. Pada sebuah titik yang berada disekitar batas, terdapat persaingan antara *R* (daerah didalam kurva) dan R^c (daerah diluar kurva). Dengan kata lain, inti dari persaingan adalah untuk memutuskan menaruh titik *i* pada *R* atau R^c.

Perhatikan bahwa perpindahan kurva yaitu pada titik *i* brubah menjadi kedalam kurva akan merubah intensitas gambar dari titik itu dangan sebuah perluasan dari *f* yang berasal dari *R*, sedangkan kita harus menjaga intensitas gambar dari titik - titik yang lain supaya tidak berubah. Dapat dikatakan, memindah kurva yaitu pada titik *i* menjadi ke luar kurva menggantikan intensitas gambar dari titik itu dengan sebuah perluasan dari *f* yang berasal dari R^c. oleh karena itu, kontribusi dari *data – fidelity – term* kepada gradient energy terhadap kurva, didapat dari perbedaan antara dua evaluation dari *data - fidelity – term* ini untuk 2 pilihan kemungkinan f_i, sehingga kurva evolusinya adalah:

$$\overrightarrow{C} = \beta D\overrightarrow{N} + \alpha P\overrightarrow{N} - \gamma\kappa\overrightarrow{N}, \quad (2.8)$$

Dimana

$$D = \|Hf - g\|^2 \Big|_{f_i = f_{R^c}} - \|Hf - g\|^2 \Big|_{f_i = f_R}$$

$$P = \sum_{j \in R^C} (f_i - f_j)^2 - \sum_{j \in R} (f_i - f_j)^2$$

j is adjacent to i *j is adjacent to i*

? merupakan lengkungan dan $\vec{N}$ merupakan vector normal yang menghadap keluar dari kurva.

2.7 Image Inpainting

Objek dari *inpainting* ialah untuk menyusun kembali bagian gambar yang hilang atau rusak agar gambar itu dapat lebih terbaca dan mengembalikan keutuhannya.

Kebutuhan untuk memperbaiki citra secara tidak kentara telah berkembang dari lukisan ke fotografi dan film. Tujuannya tetap sama yaitu untuk mengembalikan dari degradasi (misal retakan atau guratan dalam fotografi), atau untuk menghilangkan elemen tertentu dari citra tersebut.

Teknik digital mulai dipergunakan secara luas dalam melakukan proses *inpainting*, mulai dari percobaan secara otomatis mendeteksi kerusakan, dan

menghilangkan guratan pada film hingga ke tool-tool *software* yang memungkinkan hasil yang bagus meskipun kebanyakan dikerjakan secara manual.

Dalam tugas akhir ini akan dijelaskan algoritma digital inpainting secara otomatis, area yang akan dilakukan proses inpainting secara otomatis akan diisi dengan berbagai struktur di sekitarnya. Teknik *inpainting* ini dapat dipergunakan juga untuk menghilangkan bagian tertentu dari suatu citra.

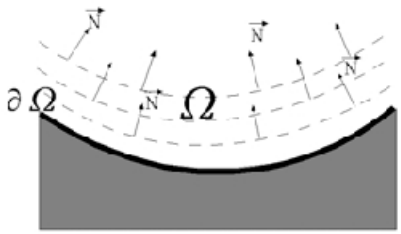
Dalam *image inpainting*, informasi utamanya berasal dari area di sekitar area yang akan dilakukan proses *inpainting*. Setidaknya ada dua usaha yang telah dilakukan dalam hubungannya dengan *digital inpainting*, yang pertama berhubungan dengan restorasi film dan yang kedua berhubungan dengan *texture syntesis.* Pada restorasi film digunakan perkiraan gerakan dan model *autoregressive* untuk menginterpolasi kerusakan dalam film dari *frame* di sebelahnya. Idenya adalah mengcopy piksel yang hilang dari *frame* di sebelahnya. Sedangkan dalam *texture syntesis* dengan cara menggabungkan domain frekuensi dan spatial untuk mengisi area yang kosong dengan tekstur yang dipilih. Tapi teknik ini memerlukan user untuk memilih tekstur yang akan dicopykan ke area yang akan diperbaiki. Untuk citra dimana area yang akan diperbaiki terdiri dari beberapa struktur yang berbeda, user akan memerlukan usaha yang tidak sedikit untuk memilih tekstur pengganti yang sesuai di dalam citra.

Algoritma dapat mengisi secara simultan area yang akan diperbaiki dengan berbagai latar belakang yang berbeda. Algoritma diperuntukan untuk

memperbaiki area yang berstruktur, tapi tidak untuk memproduksi area yang cukup luas.

2.7.1 Dasar Pemikiran

Dasar pemikiran dari proses *image inpainting* di ilustrasikan dengan gambar berikut ini :



Gambar 2.5 Area proses *inpainting*

Misal O adalah area yang akan dilakukan proses *inpainting*, dan dO adalah batas area yang akan dilakukan proses *inpainting*. Teknik yang digunakan di sini dengan memperpanjang garis *isophote* yang tiba pada dO. Proses pengisian dari dO ke dalam dilakukan dengan cara ini.

Teknik yang digunakan untuk memecahkan masalah inpainting secara garis besar dapat diketahui melalui tahap-tahap berikut :

1. Keseluruhan gambar menentukan bagaimana mengisi *gap*, tujuan inpainting ialah me*restore* area yang akan diproses.
2. Struktur dari area di sekitar O diteruskan ke dalam *gap*, garis kontur digambarkan melalui perpanjangan dari garis yang tiba pada dO

3. Area yang berbeda di dalam O sebagaimana yang didefinisikan oleh garis kontur diisi dengan warna yang sesuai dengan dO itu
4. Detail yang kecil di dalam O dilakukan proses inpaint, atau dengan kata lain ditambahkan tekstur.

Algoritma ini secara simultan dan iteratif melakukan langkah (2) dan (3) di atas. Algoritma secara *progresif* memperkecil *gap* O dengan mengisi ke dalam secara halus melalui garis yang tiba pada batas *gap* dO.

2.7.2 Algoritma Inpainting

Penerjemahan dari konsep manual *inpainting* di atas ke dalam bahasa matematik dan bahasa algoritma dapat dijelaskan pada bagian ini. Algoritma umum dari *inpainting* ditulis sebagai berikut :

$$I^{n+1}(i,j) = I^{n}(i,j) + ?tI^{n}_{t}(i,j),(i,j)? \ O \qquad (2.9)$$

dimana :

O : area yang akan dilakukan proses *inpainting*

n : waktu *inpainting*

(i.j) : koordinat piksel

?t : kecepatan perbaikan

$I^{n}_{t}(i,j)$: setiap piksel citra hasil perbaikan dalam area ***O***

$I^{n}(i,j)$: setiap piksel citra dalam area ***O***

Perhitungan hanya berjalan pada *O*, yang merupakan area yang akan diperbaiki. Dengan perhitungan ini citra $I^{n+1}(i,j)$ merupakan hasil perbaikan dari $I^{n}_{t}(i,j)$. dengan meningkatnya *n* maka kita mendapatkan citra yang lebih bagus. Kita sekarang perlu mendesain $I^{n}_{t}(i,j)$.

Sebagaimana disarankan dalam teknik *inpainting* manual, garis yang tiba pada batas dO dari area O yang akan di-*inpaint* perlu diteruskan. Dengan kata lain diperlukan propagasi informasi dari luar area O ke dalam O. Dengan $L^{n}(i.j)$ sebagai informasi yang akan dipropagasikan, dan $N^{n}(i,j)$ adalah arah propagasi, maka ini berarti :

$$\boldsymbol{I^{n}_{t}(i,j) = dL^{n}(i.j).\ N^{n}(i,j)} \qquad (2.10)$$

Dimana $dL^{n}(i.j)$ merupakan ukuran perubahan informasi dari $L^{n}(i.j)$. dengan perhitungan ini kita memperkirakan informasi $L^{n}(i.j)$ dari citra dan menghitung perubahannya menurut arah *N*. sekarang kita perlu mengekspresikan informasi *L* yang sedang dipropagasikan dan arah propagasi *N*.

Karena propagasi yang diinginkan berjalan dengan mulus, $L^{n}(i.j)$ harus merupakan *image smoothness estimator*. Untuk tujuan ini dapat dipergunakan implementasi diskrit *Laplacian* : $L^{n}(i.j) = I^{n}_{xx}(i.j) - I^{n}_{xx}(i.j)$. *Smoothness estimator* yang lain dapat dipergunakan meskipun hasil yang baik dapat dicapai dengan perhitungan ini.

Berikutnya adalah perhitungan nilai perubahan $dL^{n}(i.j)$ sepanjang *N*. Untuk melakukan ini harus mendefinisikan arah *N* untuk informasi propagasi yang akan dilakukan. Kemungkinannya ialah dengan mendefinisikan *N* sebagai normal dari

dO, misal pada tiap piksel *(i,j)* pada O, pada vektor *N(i,j)* akan merupakan normal dari dO dimana *(i,j)* berasal. Pilihan ini didasari bahwa normal propagasi pada batas area akan menuju kearah kontinuitas *isophotes* pada batas tersebut. *Smoothness estimator* ini akan diproyeksikan kearah *isophotes*, proyeksi ini akan digunakan untuk meng-*update* nilai dari citra di dalam area yang akan di-*inpaint.*

Untuk menjamin arah yang benar dalam tahap evolusi, proses *diffusi* akan dilakukan pada beberapa proses *inpainting*. Maka, setiap beberapa iterasi proses *inpainting* akan dilakukan proses *diffusi.*

2.8 NOISE

2.8.1 Additive Gaussian Noise

Additive Gaussian Noise merupakan salah satu metode pembangkitan *noise* yang akan menghasilkan *noise* yang berdistribusi normal. *Noise* yang dihasilkan dirumuskan sebagai:

a. n(n,y) =*s* .U + μ

b. g(x,y) = f(x,y) + n(x,y)

c. g(x,y) = f(x,y) + *s* . U + μ (2.10) (2.11)

dimana :

g(x,y) = citra setelah diberi *noise*

f(x,y) = citra asli

n(x,y) = white Gaussian *Noise*

μ = mean

s = standar deviasi

U = nilai acak yang terdistribusi normal dengan mean = 0 dan variansi=s2

2.8.2 Impulsive Noise

Metode *Impulsive Noise* akan memberikan *noise* dengan cara mengubah nilai *pixel* menjadi nilai intensitas maksimal atau minimal yang dimiliki oleh citra aslinya. Secara matematis *Impulsive Noise* dirumuskan

sebagai :

$$h(x,y)=f(x,y)+z(x,y) \quad (2.12)$$

$$\boldsymbol{h(x,y)} = \begin{cases} \boldsymbol{z(x,y)} \text{ dengan probabilitas } \boldsymbol{p} \\ \boldsymbol{f(x,y)} \text{ dengan probabilitas } \boldsymbol{1-p} \end{cases}$$

dimana :

h(x,y) = citra setelah diberi *noise*

z(x,y) = *Impulsive Noise*

f(x,y) = citra asli

p = probabilitas dari *Impulsive Noise*

2.8.3 Additive Laplacian Noise

Additive Laplacian Noise merupakan metode pembangkit *noise* yang memanfaatkan distribusi uniform dan konstanta laplace sebagai factor pengali. *Additive Laplacian Noise* menghasilkan citra *h*(x,y). *h*(x,y) dapat ditulis sebagai :

$$h(x,y) = f(x,y) + n(x,y) \quad (2.13)$$

$$n(x,y) = \begin{cases} V.ln(2U) * f(x,y) & \text{jika } 0 = U = ½ \\ V.(-ln(2-2U)) * f(x,y) & \text{jika } ½ < U \leq 1 \end{cases}$$

dimana :

h(x,y) = citra setelah diberi *noise*

n(x,y) = *Laplacian noise*

f(x,y) = citra asli

U = nilai acak yang terdistribusi uniform dari [0,1]

V = koefisien laplace (sebagai faktor pengali).

2.9 PARAMETER OBJEKTIF

2.9.1 Mean Square Error

MSE adalah rata-rata kuadrat nilai error antara citra asli dengan citra hasil rekonstruksi, secara matematis dapat dirumuskan sebagai berikut [9]:

$$MSE = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [f'(x,y) - f(x,y)]^2 \quad (2.14)$$

Dimana :

MSE = Nilai Mean Square Error dari citra hasil

M = Panjang citra hasil (dalam pixel)

N = Lebar citra hasil (dalam pixel)

(x,y) = Koordinat masing-masing pixel

f' = Nilai bit citra rekonstruksi pada koordinat x,y

f = Nilai bit citra asli pada koordinat x,y

2.9.2 Peak Signal to Noise Rasio

Parameter ukur yang digunakan untuk mengetahui performansi citra digital yang dihasilkan dari proses *restorasi* dalam Tugas Akhir ini adalah PSNR. PSNR merupakan nilai perbandingan antara harga maksimum dari citra hasil *filtering* dengan noise, yang dinyatakan dalam satuan desibel (dB).). Jika nilai PSNR > 30 dB, maka dapat dikatakan bahwa performansi citra hasil restorasi bagus. Apabila nilai PSNR > 50 dB maka dapat dikatakan bahwa performansi citra hasil restorasi sempurna mendekati citra asli. Secara matematis, nilai PSNR dapat dirumuskan sebagai berikut :

PSNR untuk citra yang terkena noise :

$$PSNR = 10^{10} \log \frac{f(x,y)^2}{(f(x,y) - h(x,y))^2}$$

(2.15)

PSNR untuk citra hasil restorasi :

$$PSNR = 10^{10} \log \frac{f(x,y)^2}{(f(x,y) - g(x,y))^2} \quad (2.16)$$

Dimana :

f(x,y) adalah citra asli

h(x,y) adalah citra yang telah diberi noise

g(x,y) adalah citra hasil restorasi

2.10 PARAMETER SUBJEKTIF

Untuk pengukuran secara subjektif criteria yang ditentukan berdasarkan pengamatan mata manusia karena sifat penglihatan manusia yang mmpu

melakukan pendeteksian langsung, sehingga kualitas subjektif tergantung kepada persepsi visual pengamat.

Krieria penilaian secara subjektif yang digunakan sebagai berikut :

1. *Unusable*, citra yang diamati memiliki kualitas yang sangat rendah, sehingga sudah tidak dapat dilihat lagi.
2. *Inferior*, citra yang diamati memiliki kualitas yang sangat rendah, tapi masih bias dilihat. Keberasaan interferensi benar – benar mengganggu.
3. *Marginal*, citra yang diamati memiliki kualitas yang sangat rendah, interferensi tarasa cukup mengganggu.
4. *Passable*, citra yang diamati memiliki kualitas yang cukup tinggi, dimana interferensi terasa agak mengganggu.
5. *Fine*, citra yang diamati memiliki kualitas yang tinggi, enek dilihat, dimana interferensi belum terasa mengganggu.
6. *Excellent*, citra yang diamati memiliki kualitas yang sangat tinggi, sebaik – baiknya kualitas sebagaimana yang diinginkan.

2.11 Sekilas tentang Borland Delphi 7.0

Borland Delphi 7.0 adalah paket bahasa pemrograman yang bekerja dalam sistem operasi windows. Delphi merupakan bahasa pemrograman yang mempunyai cakupan kemampuan yang luas dan sangat canggih. Berbagai jenis aplikasi dapat anda buat dengan Delphi, termasuk aplikasi untuk mengolah teks, grafik, angka, database dan aplikasi web.

Secara umum kemampuan Delphi adalah menyediakan komponen-komponen dan bahasa pemrograman yang andal, sehingga memungkinkan untuk membuat program aplikasi sesuai keinginan, dengan tampilan dan kemampuan yang canggih.

Untuk mempermudah pemrogram dalam membuat program aplikasi, Delphi meyediakan fasilitas pemrograman yang sangat lengkap. Fasilitas pemrograman tersebut dibagi dalam dua kelompok, yaitu objek dan bahasa pemrograman. Secara ringkas, objek adalah suatu komponen yang mempunyai bentuk fisik dan biasanya dapat dilihat (visual). Objek biasanya diapakaiuntuk melakukan tugas tertentu dan mempunyai batasan-batasan tertentu. Sedangkan bahasa pemrograman secara singkat dapat disebut sebagai sekumpulan teks yang mempunyai arti tertentu dan disusun dengan aturan tertentu serta untuk menjalankan tugas tertentu. Delphi menggunakan struktur bahasa pemrograman Object Pascal yang sudah sangat dikenal dikalangan pemrogram professional. Gabungan dari objek dan bahasa pemrograman ini sering disebut sebagai bahasa pemrograman berorientasi objek atau *Object Oriented Programming (OOP)*.

Khusus untuk pemrograman database, Delphi menyediakan objek yang sangat kuat, canggih dan lengkap, sehingga memudahkan pemrogram dalam merancang, membuat dan menyelesaikan aplikasi database yang diinginkan. Selain itu, Delphi juga dapat menangani data dalam berbagai format database, misalnya format MS-Access, SyBase, Oracle, FoxPro, Informix, DB2 dan lain-lain. Format database yang dianggap asli dari Delphi adalah Paradox dan dBase.