BAB 2
LANDASAN TEORI

2.1 Tinjauan Pustaka

Pengolahan citra terutama perbaikan kualitas citra dewasa ini telah banyak dilakukan. Perbaikan kualitas citra (*image enhancement*) mulai merebak, baik untuk citra warna maupun citra *gray-scale*. Kualitas citra yang kurang bagus dapat diperbaiki, baik dari segi ukuran, warna, fokus, kecerahan, kontras, dan lain-lain.

Perbaikan kualitas citra untuk tugas akhir di Universitas Atma Jaya Yogyakarta yang telah dibuat antara lain "*Pembangunan Aplikasi Defocusing Citra Dengan Metode Adaptive Highpass Filter*" (Nugroho, 2006). Aplikasi di bangun dengan menggunakan *Visual Basic 6.0*. *Adaptive Highpass Filter* dibuat dengan membandingkan seluruh nilai piksel dengan nilai varians dari nilai seluruh piksel. Piksel yang mempunyai nilai dibawah nilai varians akan mengalami penyesuaian nilai piksel, dengan mengganti nilai piksel varians sehingga mengalami perubahan tingkat kefokusan yang lebih baik, hanya pada bagian yang kurang fokus, sesuai dengan besarnya nilai varians keseluruhan.

Mengacu pada uraian diatas, maka pada tugas akhir ini penyusun memilih untuk membangun suatu aplikasi pengolahan citra untuk memperbaiki kontras citra *X-Ray* dengan menggunakan *C#. NET*. Aplikasi ini digunakan untuk memperbaiki kontras citra *X-Ray* agar dapat terlihat lebih jelas.

2.2 Pengolahan Citra Digital

2.2.1 Citra

Citra (*image*) merupakan salah satu komponen multimedia yang memegang peranan penting sebagai bentuk informasi visual. Citra mempunyai karakteristik yang tidak dimiliki oleh data teks, karena citra kaya dengan informasi. Ada peribahasa yang berbunyi "*a picture is more than a thousand words*" atau sebuah gambar bermakna lebih dari seribu kata. Maksudnya adalah sebuah gambar dapat memberikan informasi yang lebih banyak daripada informasi tersebut disajikan dalam bentuk kata-kata (tekstual).

Citra adalah suatu representasi, kemiripan, atau imitasi/tiruan dari suatu objek atau benda (Kamus Webster). Secara harafiah, citra (*image*) adalah gambar pada bidang dua dimensi. Ditinjau dari segi matematis, citra merupakan fungsi kontinyu dari intensitas cahaya pada bidang dua dimensi. Sumber cahaya menerangi objek, objek memantulkan kembali sebagian dari berkas cahaya tersebut. Pantulan cahaya ditangkap oleh alat-alat optik (mata manusia, kamera, *scanner*, dan sebagainya), sehingga bayangan objek yang disebut citra akan terekam. Citra keluaran dari sistem perekaman data bersifat optik berupa foto, analog berupa sinyal video seperti gambar pada layar monitor televisi, dan digital yang dapat langsung disimpan pada suatu pita magnetik.

Citra juga dapat dibagi menjadi 2 yaitu citra diam (*still images*) dan citra bergerak (*moving images*). Citra diam adalah citra tunggal yang tidak bergerak. Sedangkan citra bergerak adalah rangkaian citra diam yang ditampilkan secara beruntun (sekuensial) sehingga

memberi kesan pada mata sebagai gambar yang bergerak. Citra yang akan diolah dalam tugas akhir ini nantinya adalah citra diam (*still images*).

2.2.2 Definisi Pengolahan Citra

Pengolahan citra adalah pemrosesan citra, khususnya dengan menggunakan komputer, menjadi citra yang kualitasnya lebih baik. Agar citra yang mengalami gangguan mudah diinterpretasi baik oleh manusia maupun mesin, maka citra tersebut perlu dimanipulasi menjadi citra lain yang kualitasnya lebih baik.

Umumnya, operasi-operasi pada pengolahan citra diterapkan pada citra apabila (Jain, 1989) :

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

Pengolahan citra bertujuan untuk memperbaiki kualitas citra agar mudah diinterpretasikan oleh manusia dan mesin (komputer). Teknik-teknik pengolahan citra mentransformasikan citra menjadi citra lain. Jadi masukannya adalah citra dan keluarannya juga citra, tetapi citra keluaran mempunyai kualitas yang lebih baik dari pada citra masukan. Pengolahan citra yang akan dibahas dalam tugas akhir ini adalah pengubahan kontras citra, dimana pengubahan kontras citra merupakan salah satu contoh operasi pengolahan citra.

2.2.3 Klasifikasi Pengolahan Citra

Operasi-operasi yang dilakukan di dalam pengolahan citra bermacam-macam, namun secara umum operasi pengolahan citra dapat diklasifikasikan dalam beberapa jenis, yaitu :

1. *Image Enhancement* (Peningkatan Kualitas Citra)
   Image Enhancement merupakan salah satu teknik peningkatan citra yang akan memetakan tiap *gray-scale* ke *gray-scale* yang lain dengan transformasi yang telah ditentukan. Jenis operasi ini bertujuan untuk memperbaiki kualitas citra dengan cara memanipulasi parameter-paramenter citra. Dengan operasi ini, ciri-ciri khusus yang terdapat di dalam citra lebih ditonjolkan. Contoh operasi image enhancement ini adalah perbaikan kontras gelap/terang (*contrast enhancement*), perbaikan tepian objek (*edge enhancement*), penajaman (*sharpening*), pemberian warna semu (*pseudocoloring*), dan penapisan derau (*noise filtering*).

2. *Image Restoration* (Pemugaran Citra)
   Operasi ini bertujuan untuk mengurangi, meminimisasi atau membuang degradasi derau yang telah diketahui dari suatu citra. Tujuan *image restoration* hampir sama dengan *image enhancement*, perbedaannya adalah *image restoration* menyebabkan degradasi gambar diketahui. Contoh operasi *image restoration* ini adalah menghilangkan kekaburan (*blurring*) dari suatu citra yang disebabkan karena keterbatasan kualitas sensor atau lingkungannya, penapisan derau, koreksi

bentuk geometris yang disebabkan karena ketidaklinearan sensor.

3. *Image Compression* (Pemampatan Citra)

   Teknik kompresi data citra difokuskan pada pengurangan jumlah bit yang diperlukan untuk mentransmisikan citra tanpa kehilangan informasi yang berarti. Jenis operasi ini dilakukan agar citra dapat direpresentasikan dalam bentuk yang lebih kompak sehingga memerlukan memori yang lebih sedikit. Hal yang perlu diperhatikan dalam pemampatan citra adalah hasil pemampatan harus mempunyai kualitas gambar yang bagus. Contoh metode pemampatan citra adalah metode JPEG.

4. *Image Analysis* (Pengorakan Citra)

   Operasi ini bertujuan untuk menghitung besaran kuantitatif dari citra untuk menghasilkan deskripsinya. Teknik ini mengekstraksi ciri-ciri tertentu yang membantu dalam identifikasi objek. *Image analysis* dapat juga diterapkan pada pengendalian lengan robot untuk menggerakkan suatu objek setelah mengidentifikasi objek tersebut. Salah satu contohnya adalah teknik segmentasi yang dapat digunakan untuk memisahkan objek yang diinginkan dengan citra asalnya. Contoh operasi *image analysis* adalah pendeteksian tepi objek (*edge detection*), ekstraksi batas (*boundary*), dan representasi daerah (*region*).

5. *Image Segmentation* (Segmentasi Citra)

   Operasi ini bertujuan untuk memecah suatu citra ke dalam beberapa segmen dengan suatu kriteria tertentu. Jenis operasi ini berkaitan erat dengan pengenalan pola (*pattern recognition*) (Munir, 2004).

2.3 Contrast Stretching

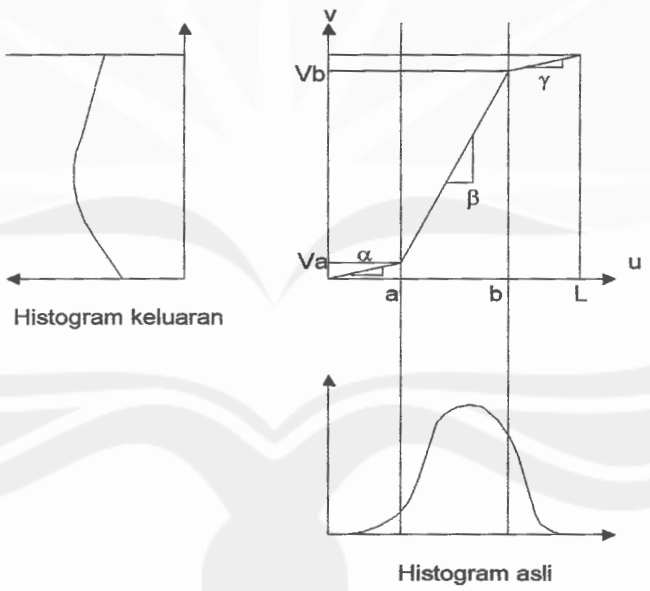
Kontras menyatakan sebaran terang (*lightness*) dan gelap (*darkness*) di dalam sebuah citra. Citra dapat dikelompokkan ke dalam 3 kategori kontras yaitu citra kontras-rendah (*low contrast*), citra kontras-bagus (*good contrast* atau *normal contrast*), dan citra kontras-tinggi (*high contrast*). Citra kontras-rendah dicirikan dengan sebagian besar komposisi citranya adalah terang atau sebagian besar gelap. Citra kontras-bagus memperlihatkan jangkauan nilai keabuan yang lebar tanpa ada suatu nilai keabuan yang mendominasi. Citra kontras-tinggi memiliki jangkauan nilai keabuan yang lebar, tetapi terdapat area lebar yang didominasi oleh warna gelap dan area yang lebar yang didominasi oleh warna terang. Citra dengan kontras-rendah dapat diperbaiki kualitasnya dengan operasi perentangan kontras (*contrast stretching*). Salah satu teknik peningkatan citra (*image enhancement*) adalah perentangan kontras (*contrast stretching*). Teknik perentangan kontras (*contrast stretching*) ada bermacam-macam. Tetapi dalam tugas akhir ini teknik yang akan dibahas ada 4, yaitu *Linear Contrast Stretching*, *Histogram Equalization*, *Lee's Algorithm*, dan *Logarithmic Image Processing* (LIP).

2.3.1 Linear Contrast Stretching

Linear Contrast Stretching atau perentangan kontras linear meupakan cara untuk meningkatkan kontras yang cukup sederhana. Peningkatan kontras dilakukan dengan merentangkan kisaran nilai intensitas yang sempit menjadi kisaran intensitas yang diinginkan [0,255]. Perentangan kontras dapat dinyatakan dalam persamaan (2.1).

$$v = \begin{cases} \alpha u & ,0 \le u < a \\ \beta(u-a)+V_a & ,a \le u < b \\ \gamma(u-b)+V_b & ,b \le u < L \end{cases} \qquad (2.1)$$

Secara grafis proses diatas dapat digambarkan seperti pada gambar 2.1.

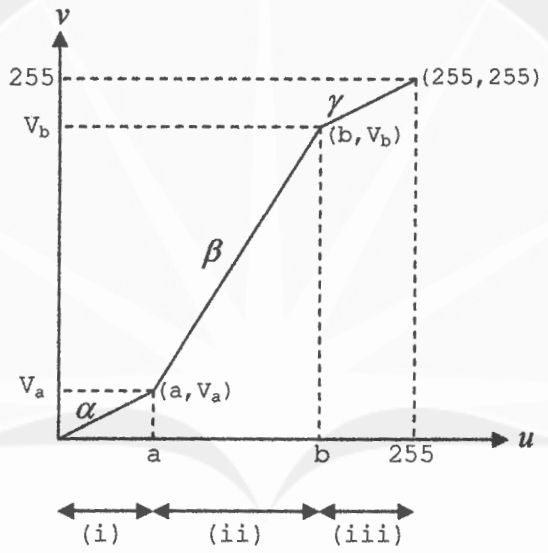


Gambar 2.1 *Linear Contrast Stretching*

Pemilihan parameter a, b, V_a, dan V_b ditentukan berdasarkan histogram citra. Proses ini tidak merentangkan kontras pada semua aras keabuan, tetapi

hanya pada selang aras keabuan yang diinginkan. Input dan output aras keabuan terdistribusi antara [0,L] dimana L = 255. Kemiringan α,β,γ menentukan relatif perentangan kontras (Jain, 1995).

Penjabaran area pada metode Linear Contrast Stretching dapat dilihat pada gambar 2.2, dimana terdapat 3 area yaitu $\alpha,\beta,$ dan γ.



Gambar 2.2 Penjabaran Area *Linear Contrast Stretching*

Untuk mengetahui nilai α,β,γ maka persamaan (2.1), dapat diturunkan sebagai berikut :

Dengan persamaan matematis $\frac{y-y_1}{y_2-y_1}=\frac{x-x_1}{x_2-x_1}$ maka diperoleh

(i) Untuk $0 \le u < a$

$$\frac{v-0}{V_a-0}=\frac{u-0}{a-0} \Leftrightarrow \frac{v}{V_a}=\frac{u}{a} \Leftrightarrow v=\frac{V_a}{a}\cdot u \Leftrightarrow v=\alpha\cdot u$$

$$\therefore \alpha=\frac{V_a}{a} \qquad (2.2)$$

(ii) Untuk $a \le u < b$

$$\frac{v - V_a}{V_b - V_a} = \frac{u - a}{b - a} \Leftrightarrow v - V_a = \frac{(V_b - V_a)}{(b - a)} \cdot (u - a)$$

$$\Leftrightarrow v = \frac{V_b - V_a}{b - a} \cdot (u - a) + V_a \Leftrightarrow v = \beta \cdot (u - a) + V_a$$

$$\therefore \beta = \frac{V_b - V_a}{b - a} \qquad (2.3)$$

(iii) Untuk $b \le u < 255$

$$\frac{v - V_b}{255 - V_b} = \frac{u - b}{255 - b} \Leftrightarrow v - V_b = \frac{(255 - V_b)}{(255 - b)} \cdot (u - b)$$

$$\Leftrightarrow v = \frac{255 - V_b}{255 - b} \cdot (u - b) + V_b \Leftrightarrow v = \gamma \cdot (u - b) + V_b$$

$$\therefore \gamma = \frac{255 - V_b}{255 - b} \qquad (2.4)$$

Dari penurunan persamaan (2.1) maka didapat rumusan untuk mencari α yaitu pada persamaan (2.2), rumusan untuk mencari β pada persamaan (2.3), dan rumusan untuk mencari γ pada persamaan (2.4), dimana :

a, b, V_a, V_b : input dari user,

u : citra input,

v : citra output,

L : 255.

2.3.2 Histogram Equalization

Histogram Equalization atau penyamaan histogram merupakan metode untuk memodifikasi kisaran dinamis dan kontras citra dengan mengubah histogramnya sehingga

intensitas di dalam citra diubah sehingga penyebarannya seragam (*uniform*). Histogram citra adalah grafik yang menggambarkan penyebaran nilai-nilai intensitas piksel dari suatu citra atau bagian tertentu dari suatu citra. Tujuan dari *histogram equalization* adalah memperoleh penyebaran histogram yang merata, sedemikian sehingga setiap derajat keabuan memiliki jumlah piksel yang relatif sama.

Proses penyamaan ditunjukkan melalui hubungan-hubungan seperti pada persamaan (2.5),

$$b(x,y) = f[c(x,y)] \tag{2.5}$$

dimana :

c : citra dengan histogram belum disamakan,

f : fungsi yang mengalihkan citra c ke citra b dengan histogram yang rata.

Persamaan (2.6) dibawah ini merupakan fungsi rapat peluang dari suatu piksel a,

$$p_1(a) = \frac{1}{Area_1} H_1(a) \tag{2.6}$$

dimana :

$p_1(a)$: peluang menemukan suatu piksel dengan nilai a dalam citra,

$Area_1$: jumlah piksel dalam citra,

$H_1(a)$: histogram citra.

Persamaan (2.7) menunjukkan fungsi rapat-kumulatif (*Cumulative-Density Function*, CDF) untuk

piksel dengan nilai a. CDF merupakan jumlah semua fungsi rapat peluang sampai dengan a.

$$p_1(a) = \frac{1}{Area_1} \sum_{i=0}^{a} H_1(i) \qquad (2.7)$$

Persamaan (2.8) menunjukkan bentuk fungsi penyamaan histogram yang diinginkan,

$$f(a) = D_m \frac{1}{Area_1} \sum_{i=0}^{a} H_c(a) \qquad (2.8)$$

dimana :

$f(a)$: fungsi penyamaan histogram yang diinginkan,

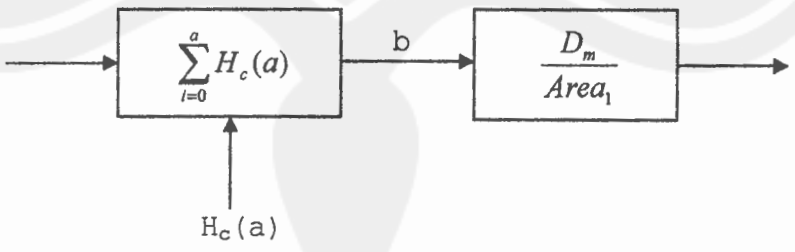
$H_c(a)$: histogram citra asli,

D_m : jumlah aras keabuan dalam citra baru, b.

$D_m = \frac{1}{p(a)}$, untuk semua nilai piksel bernilai a dalam citra b. Citra b memiliki histogram yang rata, yaitu :

H(0)=H(1)=H(2)=...

karena peluang tiap nilai piksel telah sama. Proses penyamaan histogram diatas dapat digambarkan dalam diagram blok pada gambar 2.3.



Gambar 2.3 Diagram Blok Penyamaan Histogram

Langkah-langkah yang dilakukan untuk melakukan penyamaan histogram adalah sebagai berikut :

1. Menghitung histogram untuk mendapatkan jumlah titik yang mempergunakan suatu aras keabuan.
2. Menghitung jumlah histogram.
3. Alihragam citra masukan ke citra keluaran,

$$Citra\,keluaran = \frac{D_m}{Area} \times jumlah\,histogram$$

dimana :

Area = jumlah piksel dalam citra,

D_m = jumlah aras keabuan pada citra keluaran (Dwiandiyanta, 2004).

2.3.3 Lee's Algorithm

Lee's Algorithm adalah salah satu algoritma peningkatan kualitas citra yang diimplementasikan oleh Lee. Algoritma ini nanti akan dikembangkan oleh Deng menjadi *Logarithmic Image Processing*. Algoritma Lee ditunjukkan oleh persamaan (2.9),

$$F'(i,j) = \alpha A(i,j) + \beta(F(i,j) - A(i,j)) \qquad (2.9)$$

dimana :

$F'(i,j)$: nilai hasil/nilai F yang baru,

α : nilai input dari user,

$A(i,j)$: rerata piksel dalam suatu jendela $n \times n$ citra asli yang berpusat pada piksel (i,j),

β : nilai input dari user,

$F(i,j)$: nilai piksel asli pada koordinat (i,j).

Syarat F′ ditunjukkan dengan persamaan (2.10) dibawah ini :

$$F' = \begin{cases} F' < 0 \ , F' = 0 \\ F' > 255, F' = 255 \end{cases} \qquad (2.10)$$

Apabila nilai hasil (F′) kurang dari 0 maka F′ diset sama dengan 0, tetapi apabila nilai hasil (F′) lebih besar dari 255 maka F′ diset sama dengan 255 (Weisenberck, 2004).

2.3.4 Logarithmic Image Processing (LIP)

Logarithmic Image Processing (LIP) merupakan suatu algoritma peningkatan citra yang didasarkan pada pengolahan citra logaritmis yang diusulkan oleh Deng. Algoritma ini adalah suatu implementasi baru tentang algoritma peningkatan citra Lee dan didasarkan pada suatu struktur matematis *Logarithmic Image Processing* yang dikembangkan oleh Pinoli dan Jourlin. Teknik ini mampu secara simultan meningkatkan keseluruhan kontras dan kejelasan citra.

Pada model LIP, intensitas suatu citra secara lengkap dimodelkan dengan fungsi aras keabuannya f. Contoh fisik dari fungsi aras keabuan adalah fungsi penyerapan pada suatu tapis cahaya dengan *opacity* diketahui pada tiap posisi. Fungsi aras keabuan terdefinisi pada interval [0,M], dengan M selalu positif. Nilai nol pada fungsi aras keabuan menyatakan daerah yang paling terang, sedangkan nilai M menyatakan daerah yang paling gelap.

Pada LIP, penjumlahan fungsi aras keabuan f dan g serta perkalian f oleh suatu bilangan nyata α didefinisikan dalam persamaan (2.11) dan persamaan (2.12) seperti dibawah ini :

$$f \mathbin{\overset{+}{\triangle}} g = f + g - \frac{fg}{M} \qquad (2.11)$$

dan

$$\alpha \mathbin{\overset{x}{\triangle}} f = M - M\left(1 - \frac{f}{M}\right)^{\alpha} \qquad (2.12)$$

Jourlin dan Pinolli menemukan jika fungsi aras keabuan terdefinisi pada $[-\infty, M]$, maka pengurangan fungsi aras keabuan dapat didefinisikan seperti pada persamaan (2.13).

$$f \mathbin{\overset{-}{\triangle}} g = M \frac{f - g}{M - g} \qquad (2.13)$$

Jourlin dan Pinolli juga mendefinisikan bahwa kontras antara piksel berdekatan f dan g ditunjukkan seperti pada persamaan (2.14).

$$c(f,g) = \mathrm{Max}(f,g) \mathbin{\overset{-}{\triangle}} \mathrm{Min}(f,g) \qquad (2.14)$$

Namun dapat pula digunakan definisi lain seperti pada persamaan (2.15),

$$c(f,g) = P(f \mathbin{\overset{-}{\triangle}} g) \qquad (2.15)$$

dengan P(f) adalah fungsi pemetaan positif, yang ditunjukkan pada persamaan (2.16) dibawah ini :

$$P(f) = \begin{cases} f & ; jika\ f > 0 \\ 0 \mathbin{\overset{-}{\triangle}} f & ; jika\ f < 0 \end{cases} \qquad (2.16)$$

Dengan menggunakan model LIP, algoritma Lee dapat dimodifikasi seperti pada persamaan (2.17) dibawah ini :

$$f'(i,j) = \alpha \,\triangle_{x}\, a(i,j) \,\triangle_{+}\, \beta \,\triangle_{x}\, [f(i,j) \,\triangle_{-}\, a(i,j)] \qquad (2.17)$$

dengan f(i,j) dan f'(i,j) adalah fungsi aras keabuan citra asli dan citra hasil pengolahan, dan α, β adalah berupa bilangan nyata serta a(i,j) adalah nilai rerata fungsi aras keabuan pada jendela N x N yang berpusat pada piksel (i,j). Suku a(i,j) dapat didefinisikan dengan persamaan (2.18).

$$a(i,j) = \frac{1}{nxn} \,\triangle_{x}\, \triangle_{\Sigma}^{i+n/2}_{i-n/2} \triangle_{\Sigma}^{j+n/2}_{j-n/2} f(k,l) \qquad (2.18)$$

Dengan simbol $\triangle_{\Sigma}$ merupakan kependekan dari penjumlahan menggunakan operasi penjumlahan khusus.

Algoritma diatas dapat disederhanakan dengan alihragam sederhana yang ditemukan oleh Jourlin dan Pinolli untuk operasi penjumlahan dan perkalian dalam kawasan fungsi aras keabuan. Alihragam untuk fungsi aras keabuan f disimbolkan sebagai $\overline{f}$, yang didefinisikan seperti pada persamaan (2.19).

$$\overline{f} = 1 - \frac{f}{M} \qquad (2.19)$$

Karena alihragam ini membawa dari fungsi aras keabuan ke fungsi aras keabuan negatif ternormalisasi, maka disebut alihragam komplemen ternormalisasi. Sifat-sifat alihragam ini ditunjukkan pada persamaan (2.20).

$$\overline{f \triangle_{+} g} = \overline{f}\ \overline{g}$$

$$\overline{\alpha \triangle_{x} f} = \overline{f}^{\alpha}$$

$$\overline{f \triangle_{-} g} = \frac{\overline{f}}{\overline{g}} \qquad (2.20)$$

Alihragam komplemen ternormalisasi berguna untuk penyederhanaan baik analisis dan implementasi algoritma yang berdasarkan model LIP.

Karena fungsi aras keabuan ternormalisasi adalah selalu lebih besar daripada nol, maka jelas bahwa alihragam logaritmis dapat mengubah operasi-operasi perkalian dan pemangkatan menjadi penjumlahan dan perkalian. Dari persamaan (2.17) dapat ditulis menjadi seperti persamaan (2.21).

$$\overline{f'(i,j)} = \overline{\alpha \triangle_{x} a(i,j) \triangle_{+} \beta \triangle_{x} [f(i,j) \triangle_{-} a(i,j)]} \qquad (2.21)$$

Dengan menggunakan sifat-sifat alihragam komplemen ternormalisasi, persamaan (2.21) dapat ditulis menjadi persamaan (2.22).

$$\overline{f'(i,j)} = \overline{\alpha \triangle_{x} a(i,j)}\ \ \overline{\beta \triangle_{x} [f(i,j) \triangle_{-} a(i,j)]} \qquad (2.22)$$

Sehingga persamaan (2.22) disederhanakan menjadi seperti persamaan (2.23).

$$\overline{f'(i,j)} = \overline{a(i,j)}^{\alpha}\ \overline{[f(i,j) \triangle_{-} a(i,j)]}^{\beta} \qquad (2.23)$$

Atau dapat disederhanakan lagi menjadi persamaan (2.24).

$$\overline{f'(i,j)} = \overline{a(i,j)}^{\alpha} \frac{\overline{f(i,j)}^{\beta}}{\overline{a(i,j)}} \qquad (2.24)$$

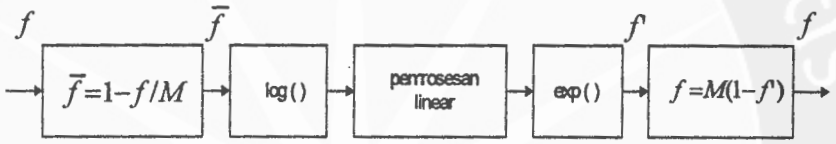
Kedua ruas dikenakan operasi logaritma, sehingga didapatkan persamaan (2.25).

$$\log \overline{f'(i,j)} = \alpha \log \overline{a(i,j)} + \beta \left[\log \overline{f(i,j)} - \log \overline{a(i,j)} \right] \qquad (2.25)$$

Dengan persyaratan seperti ditunjukkan pada persamaan (2.26) dibawah ini :

$$\log \overline{a(i,j)} = \frac{1}{nxn} \sum_{k=i-n/2}^{i+n/2} \sum_{l=j-n/2}^{j+n/2} \log \overline{f(k,l)} \qquad (2.26)$$

Implementasi algoritma diatas dapat dinyatakan dengan diagram blok seperti pada gambar 2.4.



Gambar 2.4 Diagram Blok Model LIP.

Untuk citra 8 bit, M = 255, karena intensitas citra pertama-tama diubah menjadi fungsi aras keabuan dan kemudian diubah dengan alihragam komplemen ternormalisir dan operasi logaritmis, maka operasi-operasi fungsi aras keabuan dapat diubah menjadi operasi-operasi penjumlahan dan perkalian. Dengan menggunakan alihragam balik, didapatkan fungsi aras keabuan ternormalisasi negatif, kemudian diperoleh fungsi aras keabuan dan intensitas citra.

Meskipun diagram blok pada gambar 2.4 diturunkan dari algoritma diatas, tetapi dapat pula digunakan untuk implementasi lain algoritma pengolahan citra berbasis model LIP, hanya dengan menggunakan operasi-

operasi yang yang didefinisikan dalam persamaan (2.20). Perlu dicatat pula bahwa dalam gambar 2.4 yang menggunakan alihragam komplemen ternormalisasi, algoritma yang berbasis LIP dapat diimplementasikan menggunakan struktur yang sama sebagai tapis homomorfik multiplikatif. Hal ini menunjukkan bahwa logaritma alami atas model LIP dan akan menjamin algoritma implementasi yang cepat. Dapat dilihat pula pada gambar 2.4 bahwa struktur tapis homomorfik multiplikatif adalah salah satu bagian dari algoritma yang berdasarkan model LIP.

Algoritma pengolahan citra ini dapat mengefektifkan peningkatan detail dalam area citra yang sangat gelap atau sangat terang, sehingga dapat digunakan untuk meningkatkan citra yang tidak terlihat (*underexposed*) atau citra yang terlalu terang (*overexposed*). Algoritma ini juga dapat digunakan untuk melakukan penyesuaian kejelasan citra.

2.4 Citra Medis X-Ray

2.4.1 Sejarah Penemuan X-Ray

Sejarah penemuan sinar-X oleh *Wilhelm Conrad Roentgen* pada 8 November 1895 tidak bisa terlepas dari penelitian sinar katoda. Sinar katoda timbul karena adanya lucutan listrik melalui gas di dalam tabung bertekanan rendah. Untuk menimbulkan bunga api listrik antara katoda dan anoda di udara pada tekanan 1 atmosfer (Atm) diperlukan beda tegangan listrik yang sangat besar, kira-kira 30.000 Volt/cm.

Peristiwa-peristiwa yang terjadi di dalam sinar katoda diselidiki oleh beberapa peneliti sekitar tahun 1870. Dengan menggunakan tabung khusus yang disebut tabung Crookes, William Crookes (1832-1919) memasang rintangan antara katoda dan dinding tabung yang dapat berpendar di depan katoda itu. Meskipun dari penelitian ini diketahui sinar katoda merambat lurus, Crookes belum berhasil mengidentifikasi apakah sinar katoda berupa partikel atau gelombang cahaya.

Penyelidikan yang lain berhasil mengungkapkan bahwa sinar katoda dibelokkan oleh medan magnet maupun medan listrik. Dengan bantuan sebidang tabir yang dilapisi sulfida seng yang dapat mengeluarkan pendar berwarna biru, akan terlihat perjalanan berkas sinar katoda yang membelok saat didekatkan sebuah magnet batang. Pembelokan ini juga terlihat bila sinar katoda dilewatkan di antara dua bidang kondensator bermuatan listrik. Dari penyelidikan ini dapat disimpulkan bahwa sinar katoda terdiri atas partikel-partikel bermuatan negatif. Inilah penelitian-penelitian awal yang membekali *Roentgen* ke arah penemuan sinar-X (Akhadi, 2001).

2.4.2 X-Ray Dalam Dunia Medis

'Melihat tembus' keadaan dalam tubuh manusia tanpa harus melakukan operasi pertama kali berhasil setelah ditemukan sinar-X oleh Roentgen pada tahun 1895. Dengan karakternya yang mampu menembus jaringan tubuh manusia Sinar-X membuat tubuh manusia yang tidak tembus cahaya menjadi 'transparan'. Film hasil rontgen yang merupakan citra bayangan (proyeksi) dari obyek

yang dikenai sinar-X pada prinsipnya sama dengan sebuah bayangan obyek yang terbuat dari kaca pada pencahayaan dengan sinar matahari. Sifat ini dimanfaatkan di dunia kedokteran khususnya untuk melakukan visualisasi struktur tulang dan jaringan lainnya terhadap tubuh pasien untuk keperluan diagnosa (Warsito, 2005).

Dalam dunia medis sinar-X terutama dimanfaatkan untuk diagnosis. Dengan penemuan sinar-X ini, informasi mengenai tubuh manusia menjadi mudah diperoleh tanpa perlu melakukan pembedahan. Gambar terbentuk karena adanya perbedaan intensitas sinar-X yang mengenai permukaan film setelah terjadinya penyerapan sebagian sinar-X oleh bagian tubuh manusia. Daya serap tubuh terhadap sinar-X sangat bergantung pada kandungan unsur-unsur yang ada di dalam organ.

Perkembangan dalam bidang teknologi, terutama setelah ditemukannya beberapa jenis pemantau radiasi dan metode proses pembentukan bayangan gambar dengan komputer, memungkinkan proses pembentukan bayangan gambar pada film diubah dengan cara merekonstruksi bayangan gambar dengan komputer. Dengan teknik ini, bayangan gambar dapat diperoleh dengan segera. Kemampuan untuk membedakan antara jaringan yang satu dengan lainnya juga mengalami peningkatan. *CT-scan*, misalnya, mampu membedakan antara dua jaringan yang sangat mirip dalam otak manusia, yaitu antara *substansia grisea* dengan *substansia alba*.

Untuk meningkatkan kualitas gambar dalam *radiodiagnostik*, seringkali digunakan media kontras dengan cara memasukkan substansi yang bisa menyerap sinar-X lebih banyak ke dalam tubuh yang sedang

didiagnosis. Bahan yang sering dimanfaatkan sebagai media kontras adalah Barium (Ba) dan Iodium (I) (Akhadi, 2001).

2.5 Visual C# .NET

2.5.1 Pengenalan Visual C# .NET

Visual Studio .NET 2003 merupakan paket bahasa pemrograman baru yang diluncurkan oleh *Microsoft*. Salah satu bahasa yang ada di dalam paket tersebut adalah *Visual C# .NET*. *Visual C# .NET* adalah bahasa pemrograman terbaru yang memadukan kelebihan *Java*, *C*, dan *C++*. *Visual C# .NET* merupakan suatu bahasa pemrograman yang handal, cepat, mendukung penuh *Object Oriented Programming* (OOP), serta fasilitas GUI. Kelebihan *Visual C# .NET* dibandingkan *Visual Basic .NET* adalah dalam hal kemampuan mewarisi sifat-sifat yang unggul dari *Java*, *C++*, dan kecepatan eksekusi yang lebih tinggi dibandingkan dengan *Visual Basic .NET*. Selain lebih kuat, produktif dan stabil, keunggulan lain *Microsoft Visual C# .NET* adalah sebagai :

1. *Visual C# .NET* mengatasi semua masalah yang sulit di sekitar pengembangan aplikasi berbasis *windows* dan menghilangkan penggunaan dll serta versi komponen, apalagi mewarisi sifat *C++* dan berbau *Java*.
2. *Visual C# .NET* mempunyai fasilitas penanganan *bug* yang hebat dan *real time background compiler*, membuat *developer Visual C# .NET* dapat mengetahui kesalahan kode yang terjadi secara *up to date*.

3. *Windows form designer* memungkinkan *developer* memperoleh aplikasi *desktop* dalam waktu yang singkat.
4. *Visual C# .NET* menyediakan bagi *developer* model pemrograman data akses *ActiveX Data Object* (ADO) yamg sudah dikenal dan diminati, ditambah dengan XML baru yang berbasis *Microsoft ADO.NET*. Dengan *ADO.NET*, *developer* akan memperoleh akses ke komponen yang lebih *powerful*, seperti *control DataSet*.
5. *Visual C# .Net* menghasilakan *"Visual C#* untuk *Web"*. Menggunakan *Form web* yang baru, dengan mudah membangun *thin-client* aplikasi berbasis *web* yang secara cerdas jalan di *browser* dan *platform* mana pun.
6. *Visual C# .NET* mendukung pembangunan aplikasi *Client-Server*, terdistribusi, serta aplikasi yang berbasis *windows* serta *web* (Budiharto, 2004).

2.5.2 Framework .NET

Framework .NET merupakan suatu komponen *Windows* yang terintegrasi dan dibuat agar dapat menjalankan berbagai macam aplikasi *.NET*, termasuk pengembangan aplikasi *Web Service* (XML). *Microsoft Framework .NET* menyediakan semua tool dan teknologi yang diperlukan untuk membangun aplikasi terdistribusi.

Framework .Net terdiri dari dua elemen penting, yaitu *Common Language Runtime* dan *Framework Class Library*. *Common Language Runtime* (CLR) adalah sebuah lingkungan untuk menjalankan program saat *runtime*.

Framework Class Library (FCL) merupakan kumpulan *library* yang dibutuhkan dalam pengembangan aplikasi dan sangat terintegrasi dengan komponen CLR, FCL diletakkan di atas CLR dan menyediakan layanan yang dibutuhkan oleh aplikasi *.NET.* Ketika sebuah aplikasi dijalankan dengan target *Framework .NET*, maka secara langsung *Class Library* akan saling berinteraksi, kemudian CLR akan memberikan layanan sebagai penyedia mesin untuk menjalankan aplikasi.

Framework .NET diciptakan untuk memenuhi beberapa tujuan sebagai berikut :

1. Menyediakan *environment* kerja yang handal dan konsisten, mampu menjalankan bahasa pemrograman berorientasi objek (OOP), baik program dieksekuai secara lokal maupun terdistribusi dalam jaringan.
2. Memberikan kemudahan kepada *developer* untuk mengembangkan aplikasi dengan berbagai variasi, misalnya aplikasi dapat *web service*, *Windows Form*, atau aplikasi *Console*.
3. Membangun komunikasi di dalam aplikasi sehingga sebuah aplikasi dapat dibangun dengan berbagai kode bahasa.
4. Menyediakan *environtment* yang lebih aman sehingga aplikasi sangat bagus diletakkan dalam jaringan (Jaenudin, 2006).

2.5.3 GDI+

GDI+ (Graphics Device Interface) digunakan untuk memanipulasi warna, huruf, bentuk geometris dan gambar

bitmap. GDI+ juga memerlukan pemahaman beberapa tipe relasi seperti *brush*, *pen*, *color*, *point*, dan *rectangle*.

Framework .NET menyediakan sejumlah *namespace* untuk membuat grafik 2-Dimensi. Dalam penambahan fungsionalitas dasarnya ada paket grafis (*color*, *font*, *pen*, *brush*, dan *image manipulation*) dan juga tipe *geometric transform*, *antialias*, *pallete blending*, dan *support document printing*. Ada beberapa inti GDI+ *namespace* yaitu :

1. System.Drawing
   Merupakan inti GDI+ *namespace*, mendefinisikan beberapa tipe untuk membuat dasar (*font*, *pen*, *basic brushes*, dll) seperti halnya semua tipe grafis.
2. System.Drawing.Drawing2D
   Namespace ini menawarkan tipe-tipe yang digunakan untuk lebih mengedepankan fungsionalitas grafik 2-Dimensi (misalnya, *gradient*, *brushes*, *geometric transform*, dll).
3. System.Drawing.Imaging
   Namespace ini mendefinisikan tipe-tipe yang secara langsung memanipulasi gambar grafik (misalnya, mengganti palet, mengekstraksi gambar *metadata*, memanipulasi *metafile*, dll).
4. System.Drawing.Printing
   Namespace ini mendefinisikan tipe-tipe untuk membuat gambar agar halaman bisa dicetak, interaksi dengan printer, dan memunculkan format pencetakan.
5. System.Drawing.Text

Namespace ini menyediakan manipulasi koleksi huruf. Sebagai contoh tipe *FontCollection* yang menyediakan huruf dalam mesin.

Ketika menggunakan GDI+, set telebih dahulu *System.Drawing.dll assembly*. Tetapi apabila menggunakan *Windows Application Project Workspace* baru *Visual Studio .NET* maka *System.Drawing* diset secara otomatis. Mayoritas tipe yang digunakan ketika memprogram aplikasi GDI+ menggunakan *System.Drawing namespace*. *Class* ini merepresentasikan *image*, *brush*, *pen*, dan *font*. *System.Drawing* juga mendefinisikan sejumlah tipe relasi seperti *Color*, *Point* dan *Rectangle*. *System.Drawing namespace* terdiri dari beberapa tipe yaitu :

a. Bitmap

   Meng-encapsulasi file gambar dan mendefinisikan sejumlah metode untuk memanipulasi data grafis.

b. Brush, Brushes

   Objek *brush* digunakan untuk mengisi bagian dalam (*interior*) bentuk grafis seperti segiempat, elip, dan poligon.

c. SolidBrush, SystemBrushes, TextureBrush

   Tipe ini merepresentasikan sejumlah variasi *brush* dengan fungsi *brush* kelas dasar abstrak (*abstract base class*) untuk *remaining type*. Biasanya tipe *brush* didefinisikan dalam *System.Drawing.Drawing2D namespace*.

d. Color, SystemColor, ColorTranslator

Struktur warna mendefinisikan sejumlah *static field* yang dapat digunakan untuk mengkonfigurasikan warna huruf, *brush*, dan *pen*. Tipe *ColorTranslator* menyediakan pembangunan tipe warna *.NET* baru dari representasi warna lain (*Win32, OLE_COLOR type, HTML color constant*, dll).

e. Font, FontFamily

Tipe huruf meng-encapsulasi/membungkus karakteristik huruf (misalnya, *type name, bold, italic, point size*, dll).

f. Graphic

Inti dari kelas ini merepresentasikan permukaan gambar yang valid, seperti sejumlah metode untuk membuat teks, gambar, dan pola geometris. Tipe yang terdapat dalam *.NET* sama dengan *WIN32 HDC*.

g. Icon, SystemIcons

Kelas ini merepresentasikan berbagai *icon*, seperti sistem standar penyedia *icon*.

h. Image, ImageAnimator

Image adalah kelas abstrak dasar yang menyediakan fungsionalitas untuk *Bitmap, Icon*, dan tipe *cursor*. *ImageAnimator* menyediakan cara untuk mengiterasi sejumlah tipe gambar yang diperoleh dalam beberapa interval khusus.

i. Pen, Pens, SystemPens

Pens adalah objek yang digunakan untuk menggambar garis dan kurva. Tipe *Pens* mendefinisikan sejumlah

properties statis yang mengembalikan suatu *pen* baru dari warna yang diberikan.

j. Point, PointF

Struktur ini merepresentasikan koordinat (x,y) dipetakan ke *integer* atau *float* (berturut-turut).

k. Rectangle, RectangleF

Struktur ini merepresentasikan suatu dimensi segiempat (dipetakan kembali ke *integer* atau *float*).

l. Size, SizeF

Struktur ini merepresentasikan *height/width* (dipetakan kembali ke *integer* atau *float*).

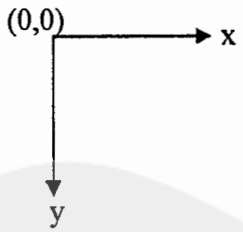
m. StringFormat

Tipe ini digunakan untuk meng-encapsulasi beberapa fitur layout tekstual (misalnya, *alignment*, *linespacing*, dll).

n. Region

Mendeskripsikan bagian dalam gambar geometris terdiri dari *rectangle* dan *path*.

GDI+ menyediakan pilihan dari macam-macam sistem koordinat. Default titik (0,0) terdapat dibagian kiri atas dan menempatkan bagian kiri atas dengan sumbu x mengarah ke kanan dan sumbu y mengarah ke bawah seperti pada gambar 2.5.



Gambar 2.5 Default Sistem Koordinat GDI+

System.Drawing.Image mendefinisikan sejumlah metode dan *properties* yang mencakup berbagai macam informasi bit mengenai dasar set piksel. Sebagai contoh, *Image Class* menyediakan properties *Width*, *Height*, dan *size* untuk mendapatkan kembali dimensi gambar. *Properties* lainnya untuk mengakses palet dasar. Sejumlah tipe didefinisikan dengan *System.Drawing.Imaging namespace*, yang mendefinisikan keseluruhan tipe yang memberi fasilitas beberapa transformasi gambar tingkat lanjut. Beberapa macam *Image Type*, misalnya seperti dibawah ini :

a. *FromFile()*, metode statis ini menciptakan gambar dari file khusus.

b. *FromHBitmap()*, menciptakan *Bitmap* dari *Window* (juga statis).

c. *FromStream()*, menciptakan gambar dari *data stream* khusus (juga statis).

d. *Height*, *Width*, *PhisicalDimensions*, *HorizontalResolution*, *VerticalResolution*, *properties* ini mengembalikan informasi mengenai dimensi gambar.

e. *Pallete, properties* ini mengembalikan tipe data *ColorPallete* yang merepresentasikan dasar paket yang digunakan untuk gambar.

f. *GetBounds()*, mengembalikan *Rectangle* yang merepresentasikan *current size* dalam gambar.

g. *Save()*, menyimpan gambar dalam bentuk file.

Ketika membuat gambar *Bitmap* secara langsung ke tipe *Control-Derived*, akan ditemukan lebih banyak kontrol dan fungsionalitas yang dapat dipilih untuk menciptakan tipe *PictureBox* untuk menempatkan gambar. Tipe *PictureBox* diperoleh dari kontrol, pewarisan sebagian besar fungsionalitasnya seperti kemampuan untuk meng-*capture* gambar, memberi *tool tip/context* menu dan memberi sejumlah detail. Ada beberapa macam *PictureBoxSizeMode* yaitu :

1. *AutoSize*, ukuran *PictureBox* sama dengan ukuran isi gambar.
2. *CenterImage*, gambar ditampilkan di tengah, jika *PictureBox* lebih besar dari gambar. Apabila gambar lebih besar dari *PictureBox*, gambar ditempatkan di tengah *PictureBox* dan tepi luar terpotong.
3. *Normal*, gambar ditempatkan di sudut kiri atas dalam *PictureBox*. Jika *PictureBox* lebih kecil dari gambar, gambar akan terpotong (Troelsen, 2001).