BAB II

TINJAUAN PUSTAKA

2.1 Pendahuluan

Bab ini memuat bahasan mengenai landasan teori yang relevan dengan permasalahan penelitian, penelitian terdahulu yang terkait dengan penelitian yang akan dikembangkan, dan hipotesis penelitian.

2.2 Penelitian Terdahulu

Penelitian mengenai peningkatan kualitas citra berwarna yang sudah pernah dilakukan sebelumnya yaitu yang dilakukan oleh Hanmandlu, Verma, Kumar dan Kulkarni (2009) menggunakan metode *novel optimal fuzzy system using bacterial foraging*. Metode ini dapat menentukan sebuah *region* pada suatu citra kurang atau terlalu terang dengan menetukan *entropy* dan *visual factor* dan menerapkan *bacterial foraging* untuk mengoptimalkan prosesnya . Yang diproses adalah nilai *Hue Saturation and Value* (HSV) yang didapat dari konversi nilai *Red, Green and Blue* (RGB) dimana nilai *hue* dipertahankan untuk menjaga komposisi warna asli.

Penelitian selanjutnya mengenai peningkatan kualitas citra berwarna dengan mempertajam warnanya dilakukan oleh Ching-Chung Yang (2011) menggunakan metode *modified mask-filtering approach* dimana derivatif antara piksel target dan tetangganya ditransfer oleh akar kubik dari fungsi sinusoidal bukan yang linear tradisional. Gambar akhir yang diperoleh akan memiliki kualitas yang lebih jelas dan berkarakteristik baik tapi lebih sedikit *overshoot* yang biasanya ditemukan dalam teknik konvensional.

Dalam penelitian mengenai *dictionary learning for remote sensing big data* Wang,Geng, Liu, Lu, Kolodziej, Ranjan dan Zomaya (2014) menggunakan *Particle Swarm Optimization* (PSO) untuk meningkatkan akurasi algoritmanya dalam meperbarui semua atom sekaligus dan menentukan arah optimasi dimana dalam iterasinya cukup dipilih atom khusus didalam kamus untuk diperkenalkan PSO ke tahap memperbarui atom dari model *dictionary learning.* Selanjutnya untuk menentukan arah optimasi data referensi lebih dahulu diperkenalkan ke model PSO. Hasilnya dimensi pergerakan partikel cukup terbatas dan meningkatkan efektivitas kamus, tapi tanpa beban komputasi berat.

Metode *Particle Swarm Optimization* (PSO) memiliki kelebihan dibandingkan dengan algoritma pada penelitian sebelumnya yaitu mudah diimplementasikan dan tidak memebutuhkan banyak variabel sehingga lebih efisien dan tidak membutuhkan banyak proses komputasi (Sathya, 2010).

Berdasarkan penelitian sebelumnya, maka pada tugas akhir ini akan diterapkan *Fuzzy Logic* dan modifikasi *Particle Swarm Optimization.* Penelitian ini diharapkan mampu menghasilkan citra yang lebih baik dan sesuai keinginan pengguna.

2.3 Citra

Citra didapatkan sebagai hasil Citra merupakan sesuatu yang terdiri dari kumpulan piksel atau titik-titik yang memiliki warna dan berbentuk dua dimensi (Pearson., 1991). Citra secara matematis didefinisikan sebagai fungsi *f(x,y)*,

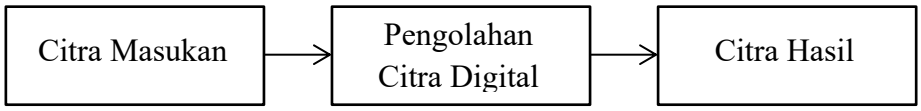
dimana *x* dan *y* merupakan koordinat spasial dan *f* merupakan sepasang koordinat yang disebut intensitas atau level keabuan (Gonzales dan Woods, 2010).

2.3.1 Citra Digital

Citra digital adalah larik atau *array* yang nilai x, y dan nilai intensitas f nya memiliki batas dan jumlahnya diskrit. (Gonzalez dan Woods, 2010). Atau dapat pula dikatakan citra digital merupakan matriks yang indeks baris dan kolomnya merupakan representasi dari letak titik citra tersebut.

2.3.2 Pengolahan Citra

Suatu citra terkadang tidak sesuai dengan harapan karena citra dapat mengalami penurunan mutu. Citra perlu diolah agar mudah diinterpretasikan baik oleh mesin ataupun manusia (Munir,2004). Pengolahan citra merupakan serangkaian tindakan atau operasi yang dilakukan pada citra untuk mendapatkan citra hasil yang diinginkan (Shih, 2010). Skema umum pengolahan citra digital dapat dilihat pada Gambar II.1.



Gambar II.1. Skema Umum Pengolahan Citra Digital

2.4 Jenis-jenis Citra Digital

2.4.1 Citra Biner

Citra biner merupakan citra yang hanya memiliki nilai derajat keabuan hitam dan putih. Pada citra biner, setiap titik hanya berniali 0 atau 1 yang merepresentasikan warna hitam atau putih (Munir,2004).

Gambar II.2. Citra Biner (Munir, 2004)

2.4.2 Citra *Grayscale*

Citra *grayscale* merupakan citra yang berisikan warna abu-abu yang memungkinkan memberi warna lebih banyak daripada citra biner. Banyaknya kemungkinan nilai maksimum dan minimum pada citra tergantung pada jumlah bit yang umumnya pada citra grayscale memiliki 8 bit. Contohnya skala keabuan untuk citra 8 bit jumlah kemungkinan nilainya adalah $2^8 = 256$ dan nilai maksimum $2^8 - 1 = 255$ dengan nilai minimumnya 0 (Putra, 2010).

Gambar II.3. Citra *Grayscale* (Niemietz, 2007)

2.4.3 Citra Berwarna

Citra berwarna merupakan citra yang memiliki setiap piksel didalamnya memiliki warna yang spesifik. Komponen warna dari citra warna dinyatakan dengan tiga warna dasar yaitu, merah, hijau dan biru. Citra warna terdiri atas citra warna 8 bit, 16 bit, dan 24 bit dimana setiap piksel dari citra warna diwakili oleh masing-masing bit tersebut (Munir, 2004).

Gambar II.4. Citra Berwarna (Gonzalez, 2010)

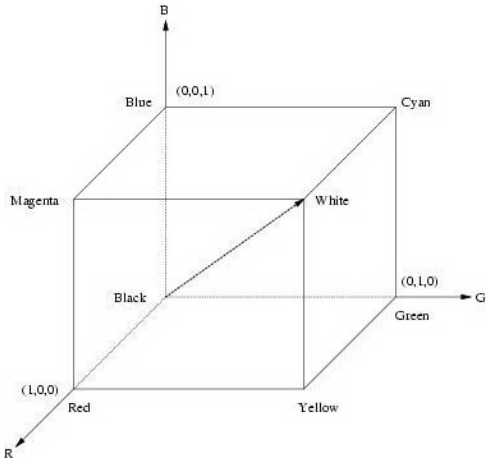
2.5 Model Warna

Model warna merupakan model matematika abstrak yang mendeskripsikan warna dalam bentuk angka, biasanya direpresentasikan dalam tiga hingga empat nilai atau komponen warna. Ketika model ini dikaitkan dengan deskripsi yang tepat mengenai bagaimana komponen harus ditafsirkan, kumpulan warna yang dihasilkan disebut color space.

2.5.1 Model Warna RGB

Model warna RGB merupakan suatu *additive color model*. Dalam hal ini, warna merah, hijau, dan biru dikombinasikan untuk mereproduksi spektrum warna yang luas. Tujuan utama dari model warna RGB adalah untuk menampilkan gambar dalam sistem elektronik, seperti layar TV dan monitor komputer dan juga digunakan dalam fotografi digital.

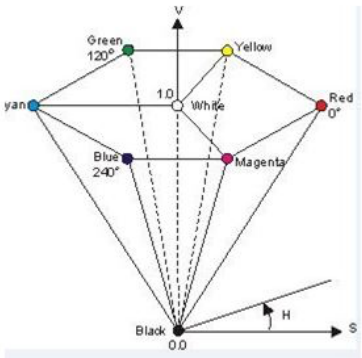
Untuk menciptakan suatu warna dengan RGB, tiga cahaya (merah, hijau, biru) harus ditumpangkan. Tanpa intensitas, masing-masing dari ketiga warna dianggap sebagai warna hitam, sedangkan dengan intensitas penuh mengarahkan ke persepsi warna putih. Perbedaan intensitas menghasilkan hue dari warna, sementara perbedaan antara intensitas tertinggi dan terendah dari warna menghasilkan warna menjadi lebih atau kurang jenuh.



Gambar II.5. Model Warna RGB (Gonzalez, 2010)

2.5.2 Model Warna HSV

HSV, yang merupakan singkatan dari hue, saturation, dan value, menggambarkan warna tiga dimensi. HSV berusaha menggambarkan hubungan antara warna, dan memperbaiki model warna RGB. Apabila HSV digambarkan sebagai roda, sumbu pusat bergerak dari warna putih yang berada di atas ke warna hitam yang berada di bawah, dengan warna-warna netral lain berada diantaranya. Sudut dari sumbu axis menggambarkan hue, jarak dari sumbu menggambarkan saturation, dan jarak sepanjang sumbu axis menggambarkan value.



Gambar II.6. Model Warna HSV

2.6 Peningkatan Kualitas Citra

Salah satu operasi pengolahan citra adalah peningkatan kualitas citra. Menurut Munir (2004) yang disebut proses peningkatan kualitas citra adalah proses untuk mengubah citra menjadi citra hasil yang lebih mudah di interpretasi manusia. Atau dengan kata lain peningkatan kualitas citra adalah operasi dalam pengolahan citra untuk menghasilkan citra hasil yang lebih baik dari citra aslinya.

Salah satu bentuk peningkatan kualitas citra yaitu perbaikan terhadap kecerahan sebuah citra. Citra dapat dikategorikan menjadi tiga kategori kecerahan yaitu terlalu cerah (*overexposed*), citra dengan tingkat kecerahan baik dan citra kurang cerah (*underexposed*). Citra *undrexposed* dan *overexposed* memiliki karakteristik yaitu sebagian besar komposisi citranya adalah terang atau sebagian besar gelap. Dari histogramnya terlihat jika pengelompokan nilai-nilai pixel berada di bagian kiri (berisi nilai keabuan yang rendah), citranya cenderung gelap. Jika pengelompokan nilai-nilai pixel berada di bagian kanan (berisi nilai keabuan yang tinggi), citranya cenderung terang. Citra dengan tingkat kecerahan baik memperlihatkan jangkauan nilai keabuan yang hampir merata tanpa ada nilai keabuan yang mendominasi. Histogram citranya memperlihatkan sebaran nilai keabuan yang cenderung setara (Munir, 2004).

(a) (b) (c)

Gambar II.7. Kecerahan Citra : (a) citra underexposed, (b) citra overexposed, (c) citra dengan kecerahan yang baik (Munir, 2004)

2.7 Logika Fuzzy

Logika fuzzy pertama kali dikembangkan oleh Lotfi A. Zadeh seorang ilmuwan Amerika Serikat berkebangsaan Iran dari Universitas California di Berkeley. Logika fuzzy merepresentasikan cara berpikir manusia ke dalam bentuk algoritma agar dapat diterjemahkan oleh mesin. Berbeda dengan nilai kebenaran pada logika klasik yang bernilai biner 0 (salah) atau 1 (benar). Logika fuzzy mempunyai nilai kebenaran real dalam selang [0,1]. Logika fuzzy merupakan suatu cara untuk memetakan suatu ruang *input* ke dalam ruang *output* (Sri Kusumadewi, 2002). Adapun bagian-bagian dalam sistem logika fuzzy yaitu :

a. Variable fuzzy yang merupakan variabel yang akan dibahas dalam suatu sistem fuzzy
b. Himpunan fuzzy yang merupakan suatu grup yang mewakili suatu kondisi tertentu dalam suatu variabel fuzzy
c. Semesta pembicaraan yang merupakan keseluruhan nilai yang diperbolehkan untuk dioperasikan dalam suatu variabel fuzzy. Semesta pembicaraan merupakan himpunan bilangan real yang senantiasa naik (bertambah) secara monoton dari kiri ke kanan. Nilai semesta pembicaraan

dapat berupa bilangan positif maupun negatif. Adakalanya nilai semesta pembicaraan ini tidak dibatasi batas atasnya.

d. Domain himpunan fuzzy yang merupakan keseluruhan nilai yang diijinkan dalam semesta pembicaraan dan boleh dioperasikan dalam suatu himpunan fuzzy. Seperti halnya semesta pembicaraan, domain merupakan himpunan bilangan real yang senantiasa naik (bertambah) secara monoton dari kiri ke kanan. Nilai domain dapat berupa bilangan positif maupun negatif.

2.7.1 Fungsi Keanggotaan

Fungsi Keanggotaan (*membership function*) adalah suatu kurva yang menunjukkan pemetaan titik-titik input data ke dalam nilai keanggotaannya atau derajat keanggotaan yang memiliki interval antara 0 sampai 1. Salah satu cara yang dapat digunakan untuk mendapatkan nilai keanggotaan adalah dengan melalui pendekatan fungsi.

2.8 Mengubah Ruang Warna Citra RGB ke HSV

Untuk mengubah ruang warna citra RGB menjadi HSV diasumsikan koordinat-koordinat R, G, B [0,1] adalah berurutan merah, hijau, biru dalam ruang warna RGB, dengan max adalah nilai maksimum dari nilai red, green, blue, dan min adalah nilai minimum dari nilai red, green, blue. Untuk memperoleh sudut hue [0,360] yang tepat untuk ruang warna HSV

Rumus untuk mendapatkan nilai h :

$$h\,(hue) = \begin{cases} 0,\ jika\ max = min \\ 60°\ x\ \left(\frac{G-B}{max-min} mod\ 6\right), jika\ max = R \\ 60°\ x\ \left(\frac{B-R}{max-min} + 2\right), jika\ max = G \\ 60°\ x\ \left(\frac{R-G}{max-min} +\ 4\right), jika\ max = B \end{cases} \quad \text{(II.1)}$$

Rumus untuk mendapatkan nilai s dan v :

$$s\,(saturation) = \begin{cases} 0, jika\ max = min \\ \frac{max-min}{V}\ , dalam\ keadaan\ lain \end{cases} \quad \text{(II.2)}$$

$$v\,(value) = max \quad \text{(II.3)}$$

Karena rumus di atas akan menghasilkan nilai value dan saturation dalam jangkauan RGB [0,1] maka harus dikalikan dengan 255 untuk memperoleh nilai dengan jangkauan RGB [0,255].

2.9 Histogram

Informasi penting mengenai isi citra digital dapat dengan baik diketahui dengan membuat histogram citra. Histogram citra adalah grafik yang menggambarkan penyebaran nilai-nilai intensitas pixel dari suatu citra atau bagian tertentu dalam citra. Dari sebuah histogram dapat diketahui frekuensi kemunculan nisbi (relative) dari intensitas pada citra tersebut. Histogram juga dapat menunjukan banyak hal tentang kecerahan (brightness) dan kontras (contrast) pada sebuah citra.

Misalkan sebuah citra digital memiliki L derajat keabuan, yaitu dari nilai 0 sampai L-1 (misalnya pada citra dengan kuantisasi derajat keabuan 8-bit, nilai derajat keabuan dari 0 sampai 255).

Rumus menghitung histogram :

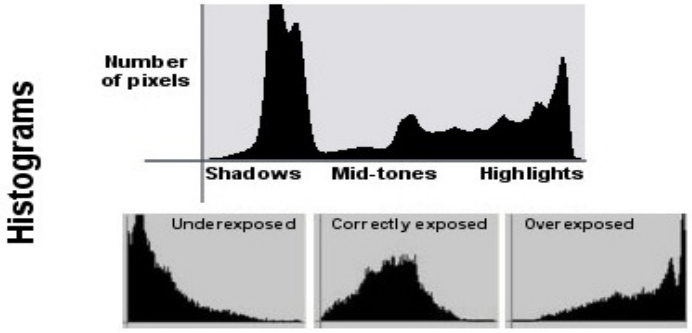
$$h_i = n_i/n\,, i = 0,1,2 \ldots . L-1 \qquad \text{(II.4)}$$

Dimana :

n_i = jumlah pixel yang memiliki derajat keabuan i ;

n = jumlah seluruh pixel di dalam citra;

Tinggi histogram merepresentasikan intensitas pixel yang menonjol sedangkan lebar histogram menunjukkan rentang kontras dari gambar (Munir, 2004).



Gambar II.8. Grafik Histogram Citra (Dennis P. Curtin, 2011)

2.10 *Exposure*

Untuk pembagian kategori histogram menjadi dua kategori yaitu kurang terang dan terlalu terang digunakan parameter *exposure. Exposure* juga digunakan sebagai nilai awal untuk poros variable lain (Hanmandlu dkk, 2009).

Rumus *Exposure* :

$$Exposure = \left(\frac{1}{L}\right)\left(\frac{(\sum_{v=1}^{L} p(v)xv)}{\sum_{v=1}^{L} p(v)}\right) \qquad \text{(II.5)}$$

Dimana :

v = nilai keabuan pixel pada citra;

p(v) = nilai histogram;

L = nilai keabuan dari citra;

Karena parameter tunggal tidak dapat mencirikan kurang terang, campuran dan terlalu terang didalam wilayah sebuah citra maka digunakan *upper threshold* (UT) dan *lower threshold* (LT). Tingkat keabuan dibawah UT diasumsikan sebagai wilayah kurang terang, tingkat kebuan diatas LT diasumsikan wilayah terlalu terang, dan sisanya termasuk wilayah campuran.

Rumus *upper threshold* (UT) dan *lower threshold* (LT) :

$$UT = L(\gamma - u) \qquad \text{(II.6)}$$

$$LT = L(\gamma + l) \qquad \text{(II.7)}$$

Dimana :

γ = poros;

u = terletak pada kisaran 0 ke γ;

l = terletak pada kisaran 0 ke (1-γ);

Nilai awal dari poros diatur *exposure* dan nilai optimumnya ditentukan oleh modifikasi *particle swarm optimization* (PSO). Untuk menyederhanakan proses perhitungan nilai *u* dan *l* diatur 0,1.

2.11 Fuzzifikasi

Fuzzifikasi merupakan tahapan mengubah sebuah variabel non *fuzzy* menjadi variabel *fuzzy*. Sebuah citra berukuran R x C yang memiliki tingkat intensitas *v* pada jangkauan [0,L-1].

Representasi fuzzy dari citra adalah :

$$I = \cup \{\mu(v_{rc})\} = \cup \left\{\frac{\mu_{rc}}{v_{rc}}\right\} \qquad \text{(II.8)}$$

r= 1, 2, 3…R;c= 1, 2, 3…C

Dimana :

R= Jumlah baris dalam matriks citra;

C= Jumlah kolom dalam matriks citra;

$\mu(v_{rc})$ atau $\frac{\mu_{rc}}{v_{rc}}$ = fungsi keanggotaan μ_{rc};

2.12 Fungsi Keanggotaan *Gaussian*

Karena citra dibagi menjadi tiga wilayah berdasarkan nilai UT dan LT, fuzzifikasi dilakukan untuk tiga wilayah yang berbeda, Untuk fuzzifikasi di daerah kurang cerah atau terlalu cerah diassumsikan menjadi Gaussian (Hanmandlu dkk, 2009).

Rumus untuk mencari nilai awal dari fh adalah :

$$f_h^2 = \left(\frac{1}{2}\right)\left(\frac{\sum_{v=0}^{L-1}((v_{max}-v)^4)p(v)}{\sum_{v=0}^{L-1}((v_{max}-v)^2)p(v)}\right) \qquad \text{(II.9)}$$

Dimana:

v= Tingkat keabuan citra;

v_{max}= Nilai maksimum tingkat keabuan dari citra;

p(v) = nilai histogram;

Fuzzifikasi pada fungsi (5) beroperasi di bawah UT dan fungsi (6) beroperasi diatas LT. Sedangkan daerah antara UT dan LT tidak tersentuh.

Rumus fungsi kenaggotaan Gaussian untuk wilayah kurang cerah :

$$\mu_u(v) = e^{-\left[\left(v_{max}-\left(v_{avg}-v\right)\right)/\left(\sqrt{2}fh\right)\right]^2} \qquad \text{(II.10)}$$

Rumus fungsi keanggotaan Gaussian untuk wilayah terlalu cerah untuk citra $v \geq$ LT :

$$\mu_o(v) = e^{-\left[\left(v_{max}-\left(v_{avg}-(L-v)\right)\right)/\left(\sqrt{2}fh\right)\right]^2} \qquad \text{(II.11)}$$

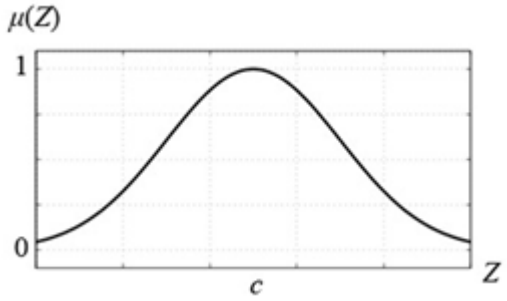
Dimana:

v= Tingkat keabuan citra di wilayah kurang cerah;

v_{avg}= Nilai tingkat keabuan rata-rata;

v_{max}= Nilai maksimum tingkat keabuan dari citra;

fh= fuzzifier;



Gambar II.9. Grafik Fungsi Keanggotaan Gaussian (Kusumadewi, 2002)

2.13 Intensifikasi *Fuzzy*

Intensifikasi *fuzzy* merupakan proses memodifikasi nilai keanggotaan dengan menggunakan bentukan fungsi sigmoid. Seperti pada rumus untuk menentukan fungsi keanggotaan, digunakan dua fungsi sigmoid berbeda untuk peningkatan nilai keanggotaan wilayah kurang cerah dan terlalu cerah (Hanmandlu dan D. Jha, 2006).

Rumus intensifikasi nilai fungsi keanggotaan untuk wilayah kurang cerah

$$\mu'_u(v) = \left\{\frac{1}{1+e^{-t(\mu_u(v)-\mu_{cu})}}\right\} \qquad \text{(II.12)}$$

Rumus intensifikasi nilai fungsi keanggotaan untuk wilayah terlalu cerah:

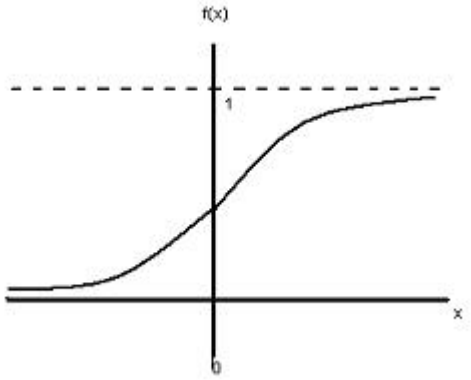
$$\mu'_o(v) = \left\{\frac{1}{1+e^{-g(\mu_o(v)-\mu_{co})}}\right\} \qquad \text{(II.13)}$$

Dimana:

t dan g = parameter intensifikasi;

μ_{co}= titik silang untuk wilayah terlalu cerah diatur 0,5;

μ_{cu}= titik silang untuk wilayah kurang cerah diatur 0,5;



Gambar II.10. Grafik Fungsi Sigmoid (Puspaningrum, 2006)

2.14 *Average fuzzy contrast and fuzzy contrast*

Pendekatan *fuzzy* digunakan untuk mengukur kontras dan *visual factor* karena sangat sulit untuk menentukan ukuran kualitas yang sesuai dan cocok untuk persepsi visual manusia. Kinerja dari pendekatan ini dapat diukur secara objektif dengan analisis kualitatif dari citra dan secara subjektif dengan analisis visual.

Rumus *fuzzy contrast* untuk wilayah kurang cerah :

$$C_u = \left(\frac{1}{UT}\right)\left(\sum_{v=0}^{UT-1}\left(\mu'_u(v) - \mu_{cu}\right)^2\right) \qquad \text{(II.14)}$$

Rumus *average fuzzy contrast* untuk wilayah kurang cerah :

$$\bar{C}_u = \left(\frac{1}{UT}\right)\left(\sum_{v=0}^{UT-1}\left(\mu'_u(v) - \mu_{cu}\right)\right) \qquad \text{(II.15)}$$

Rumus *fuzzy contrast* untuk wilayah terlalu cerah :

$$C_o = \left(\frac{1}{L-LT}\right)\left(\sum_{v=LT}^{L-1}\left(\mu'_o(v) - \mu_{co}\right)^2\right) \qquad \text{(II.16)}$$

Rumus *average fuzzy contrast* untuk wilayah terlalu cerah :

$$\bar{C}_o = \left(\frac{1}{L-LT}\right)\left(\sum_{v=LT}^{L-1}\left(\mu'_o(v) - \mu_{co}\right)\right) \qquad \text{(II.17)}$$

Rumus *initial fuzzy contrast* dan *initial average fuzzy contrast* yang merupakan perhitungan *average fuzzy contrast* dan *fuzzy contrast* sebelum nilai fungsi keanggotaan ditingkatkan.

Rumus *initial fuzzy contrast* untuk wilayah kurang cerah :

$$C_{ui} = \left(\frac{1}{UT}\right)\left(\sum_{v=0}^{UT-1}(\mu_u(v) - \mu_{cu})^2\right) \qquad \text{(II.18)}$$

Rumus *initial average fuzzy contrast* untuk wilayah kurang cerah :

$$\bar{C}_{ui} = \left(\frac{1}{UT}\right)\left(\sum_{v=0}^{UT-1}(\mu_u(v) - \mu_{cu})\right) \qquad \text{(II.19)}$$

Rumus *initial fuzzy contrast* untuk wilayah terlalu cerah :

$$C_{oi} = \left(\frac{1}{L-LT}\right)\left(\sum_{v=LT}^{L-1}(\mu_o(v) - \mu_{co})^2\right) \qquad \text{(II.20)}$$

Rumus *initial average fuzzy contrast* untuk wilayah terlalu cerah :

$$\bar{C}_{oi} = \left(\frac{1}{L-LT}\right)\left(\sum_{v=LT}^{L-1}(\mu_o(v) - \mu_{co})\right) \qquad \text{(II.21)}$$

Average fuzzy contrast menunjukkan intensitas keseluruhan gambar sedangkan *fuzzy contrast* mengukur penyebaran gradien keanggotaan yang berhubungan dengan titik silang.

2.15 Quality Factor

Quality factor merupakan rasio mutlak dari average fuzzy contrast dan fuzzy contrast. *Quality factor* harus memberikan ukuran kualitas citra sebagai karakteristik citra.

Rumus menghitung *quality factor* :

$$Q_{u/o} = \left|\frac{\bar{C}_{u/o}}{C_{u/o}}\right| \qquad \text{(II.22)}$$

Rumus menghitung *quality factor* sebelum peningkatan nilai keanggotaan :

$$Q_{ui/oi} = \left|\frac{\bar{C}_{ui/oi}}{C_{ui/oi}}\right| \qquad \text{(II.23)}$$

2.16 Visual Factor

Visual factor merupakan normalisasi *quality factor* untuk menilai kualitas citra. *Visual factor* dihitung secara terpisah untuk wilayah kurang cerang dan terlalu cerah.

Rumus menghitung *visual factor* untuk wilayah kurang cerah :

$$V_u = \frac{Q_u}{Q_{ui}} \qquad \text{(II.24)}$$

Rumus menghitung *visual factor* untuk wilayah terlalu cerah :

$$V_o = \frac{Q_o}{Q_{oi}} \qquad \text{(II.25)}$$

dimana

V_u= *visual factor* untuk wilayah kurang cerah;

V_o= *visual factor* untuk wilayah terlalu cerah;

Q_u= *quality factor* untuk wilayah kurang cerah;

Q_o= *quality factor* untuk wilayah terlalu cerah;

Q_{ui}= *initial quality factor* untuk wilayah kurang cerah;

Q_{oi}= *initial quality factor* untuk wilayah terlalu cerah;

visual factor secara keseluruhan merupakan kombinasi antara *visual factor* wilayah kurang cerah dan *visual factor* wilayah terlalu cerah:

Rumus *visual factor* keseluruhan :

$$V_f = \left(\frac{UT}{L}\right) V_u + \left(1 - \frac{LT}{L}\right) V_o \qquad \text{(II.26)}$$

Secara empiris *visual factor* dinyatakan sebagai berikut :

$$V_{sf} \rightarrow 1.5 - \left(\frac{0.9}{255}\right) \gamma_i \qquad \text{(II.27)}$$

Dimana :

γ_i = nilai awal dari poros;

V_{sf}= berada pada jangauan 0,6-1,5;

2.17 Particle Swarm Optimization

Particle Swarm Optimization (PSO) terinspirasi pada perilaku sekawanan burung atau ikan. Algoritma PSO meniru perilaku sosial organisme ini. Perilaku sosial terdiri dari tindakan individu dan pengaruh dari individu-individu lain

dalam suatu kelompok. Kata partikel berarti individu di dalam sebuah kelompok. Setiap individu atau partikel berperilaku dengan cara menggunakan kecerdasannya sendiri dan juga dipengaruhi perilaku kelompok kolektifnya (Budi Santosa, 2011). Jadi, jika satu partikel atau individu menemukan jalan yang tepat atau efisien menuju ke sumber makanan, sisa kelompok yang lain juga akan dapat segera mengikuti jalan tersebut meskipun lokasi mereka jauh di kelompok tersebut.

Dalam *Particle Swarm Optimization* (PSO) kawanan diasumsikan mempunyai ukuran tertentu dengan setiap partikel posisi awalnya terletak di suatu lokasi yang acak dalam ruang multidimensi. Setiap partikel diasumsikan memiliki dua karakteristik yaitu posisi dan kecepatan. Setiap partikel bergerak dalam ruang tertentu dan mengingat posisi terbaik yang pernah dilalui atau ditemukan terhadap sumber makanan atau nilai fungsi objektif. Setiap partikel menyampaikan informasi atau posisi terbaiknya kepada partikel yang lain dan menyesuaikan posisi dan kecepatan masing-masing berdasarkan informasi yang diterima mengenai posisi tersebut.

Rumus posisi dan kecepatan partikel pada suatu dimensi ruang tertentu :

$$X_i = x_{i1}(t)x_{i2}(t), \dots x_{iN}(t) \quad \text{(II.28)}$$

$$V_i = v_{i1}(t)v_{i2}(t), \dots v_{iN}(t) \quad \text{(II.29)}$$

Dimana

X = posisi partikel;

V = kecepatan partikel;

i = indeks partikel;

t = iterasi ke-t;

N = ukuran dimensi ruang;

Rumus *updating status* pada PSO :

$$V_i(t) = V_i(t-1) + c1r1\left(X_i^L - X_i(t-1)\right) + c2r2\left(X^G - X_i(t-1)\right) \qquad \text{(II.30)}$$

$$X_i = V_i(t) + X_i(t-1) \qquad \text{(II.31)}$$

Dimana :

$X_i^L = X_{i1}^L, X_{i2}^L, \ldots., X_{iN}^L$= *local best* dari partikel ke-i;

$X^G = X_{i1}^G, X_{i2}^G, \ldots., X_{iN}^G$= *global best* dari seluruh kawanan;

c 1 dan c 2 = suatu konstanta yang bernilai positif yang biasanya disebut sebagai learning factor ;

r 1 dan r 2 = suatu bilangan random yang bernilai antara 0 sampai 1;

2.17.1 Modifikasi *Particle Swarm Optimization*

Particle Swarm Opmtimization (PSO) dimodifikasi karena ditemukan bahwa dalam implementasinya *particle standard* dalam PSO diupdate terlalu cepat dan nilai minimum fungsi tujuan seringkali terlewati. Adapun perbaikan yang dilakukan yaitu ditambahkan sebuah term inersia θ untuk mengurangi kecepatan pada formula update kecepatan. Term inersia θ biasanya dibuat

sedemikian sehingga semakin besar iterasi yang dilalui, kecepatan partikel semakin mengecil.

Rumus *updating status* pada modifikasi PSO :

$$V_j(i) = \theta V_j(i-1) + c1r1\left(Pbest, j - X_j(i-1)\right) + c2r2\left(X^G - X_j(i-1)\right) \qquad \text{(II.32)}$$

Bobot inersia ini diusulkan oleh Shi and Eberhart [1998] untuk meredam kecepatan selama iterasi, yang memungkinkan kawanan burung menuju titik target secara lebih akurat dan efisien dibandingkan dengan algoritma aslinya. Nilai bobot inersia yang tinggi menambah porsi pencarian global, sedangkan nilai yang rendah lebih menekankan pencarian lokal. Untuk tidak terlalu menitikberatkan pada salah satu bagian dan tetap mencari area pencarian yang baru dalam ruang berdimensi tertentu, maka perlu dicari nilai bobot inersia (θ) yang secara imbang menjaga pencarian global dan lokal. Untuk mencapai itu dan mempercepat konvergensi, suatu bobot inersia yang mengecil nilainya dengan bertambahnya iterasi digunakan dengan formula

$$\theta(i) = \theta_{max} - \left(\frac{\theta_{max}\theta_{min}}{i_{max}}\right) i \qquad \text{(II.33)}$$

Dimana :

θ_{max} dan θ_{min}= masing-masing nilai awal dan nilai akhir bobot inersia biasanya θ_{max} =0,9 dan θ_{min}(0.4);

i_{max}= jumlah iterasi maksimum yang digunakan ;

i = iterasi yang sekarang;

2.17.2 Kelebihan Algoritma Optimasi PSO

Menurut penelitian yang dilakukan oleh Emad Elbeltagi, Tarek Hegazy dan Donald Grierson (2005) dalam jurnalnya yang berjudul *Comparison among five evolutionary-base optimization algorithms* dimana mereka melakukan penelitian untuk mencari algortima mana yang paling handal antara *genetic algorithm, memetic algorithm, particle swarm, ant colony* dan *shuffled frog leaping.* Mereka melakukan percobaan untuk optimasi kontinyu dan optimasi diskrit dan parameter pembandingnya adalah persentase keberhasilan, nilai rata-rata dari solusi yang diperoleh dari setiap percobaan dan waktu pemrosesan untuk mendapat nilai optimal. Dan dari penelitiannya didapatkan hasilnya bahwa algoritma optimasi PSO memiliki tingkat keberhasilan dan kualitas solusi terbaik dibandingkan algoritma lain dan terbaik kedua dalam hal waktu pemrosesan.

2.18 Defuzzifikasi

Proses defuzzifikasi merupakan proses untuk mengembalikan nilai dari himpunan *fuzzy* menjadi nilai *crisp* (Sri Kusumadewi, 2002). Secara umum proses defuzzifikasi menggunakan kebalikan (*inverse*) dari operator fungsi keanggotaan yang digunakan pada saaat proses fuzzifikasi

Rumus Defuzzifikasi untuk wilayah kurang cerah :

$$v = \mu_u^{-1}[\mu_u'(v)] \qquad \text{(II.34)}$$

Rumus Defuzzifikasi untuk wilayah terlalu cerah :

$$v = \mu_o^{-1}[\mu_o'(v)] \qquad \text{(II.35)}$$

Dimana :

v = derajat keabuan dari citra;

$\mu'_u(v)$= nilai fungsi keanggotaan yang telah di intensifikasi untuk wilayah kurang cerah;

$\mu'_o(v)$= nilai fungsi keanggotaan yang telah di intensifikasi untuk wilayah terlalu cerah;

2.19 Menghitung *Saturation*

Perbaikan pada saturasi warna terkadang membuat citra kehilangan detailnya yang menyebabkan hilangkan informasi penting. Oleh karena itu untuk meningkatkan saturasi tergantung apakah piksel dibuah lebih gelap atau lebih cerah.

Rumus peningkatan saturasi :

$$I'_o(S) = [I_o(S)]^{S_o} \qquad \forall \text{ piksel di wilayah terlalu cerah} \qquad \text{(II.36)}$$

$$I'_u(S) = [I_u(S)]^{S_u} \qquad \forall \text{ piksel di wilayah kurang cerah} \qquad \text{(II.37)}$$

Dimana :

S_o= intensifier dan de-intensifer saturasi terlalu cerah;

S_u= intensifier dan de-intensifer saturasi kurang cerah;

I_u dan I'_u = komponen saturasi asli dan modifikasi dari model HSV untuk wilayah kurang cerah;

I_o dan I'_o = komponen saturasi asli dan modifikasi dari model HSV untuk wilayah terlalu cerah;

2.20 Mengubah Ruang Warna Citra HSV ke RGB

Untuk transformasi kembali dari ruang warna HSV menjadi RGB, Dengan hue H ∈ [0°, 360°], saturation S ∈ [0, 1], dan value V ∈ [0, 1] :

$$C = V\ x\ S \qquad \text{(II.38)}$$

$$H' = \frac{H}{60°} \qquad \text{(II.39)}$$

$$X = C(1 - |(H' mod\ 2) - 1|) \qquad \text{(II.40)}$$

$$(R1, G1, B1) = \begin{cases} (0,0,0), jika\ H\ tidak\ terdefinisi \\ (C,X,0), jika\ 0 \leq H' < 1 \\ (X,C,0), jika\ 1 \leq H' < 2 \\ (0,C,X), jika\ 2 \leq H' < 3 \\ (0,X,C), jika\ 3 \leq H' < 4 \\ (X,0,C), jika\ 4 \leq H' < 5 \\ (C,0,0), jika\ 5 \leq H' < 6 \end{cases} \qquad \text{(II.41)}$$

$$m = V - C \qquad \text{(II.42)}$$

$$(R, G, B) = (R1 + m, G1 + m, B1 + m) \qquad \text{(II.43)}$$

Dimana :

C, X dan m = Variable pendukung

H' = nilai H pada citra setelah dibagi 60°

Karena rumus di atas akan menghasilkan nilai RGB dalam jangkauan [0,1] maka harus kalikan dengan 255 untuk mendapat jangkauan [0,255].

2.21 Kriteria Mengukur Kebenaran Hasil Peningkatan Kualitas

Ada dua jenis kriteria yang digunakan untuk mengukur kebenaran hasil peningkatan kualitas citra, yaitu kriteria subjektif dan kriteria objektif.

a. Kriteria Kebenaran Subjektif

Kriteria kebenaran subjektif dilakukan dengan menggunakan *Mean Opinion Score* (MOS) yaitu menanyakan langsung kepada orang-orang mengenai penilaian mereka terhadap kualitas citra hasil peningkatan kecerahan. Penilaian ini dilakukan dengan sistem penglihatan manusia. Kemudian citra asli dan citra hasil dibandingkan, lalu dibuat skala penilaian yang berkaitan dengan kualitas citra. Misalnya, dibuat skala penilaian (0, 1, 2, 3, 4, 5) yang berturut-turut mewakili kualitas (lebih buruk, agak buruk, sama, agak baik, lebih baik). Penilaian dilakukan dengan ukuran sampel tertentu, lalu hasil penilaian setiap sampel dirata-ratakan.

b. Kriteria Kebenaran Objektif

Kriteria kebenaran objektif merupakan pengukuran kebenaran hasil peningkatan kualitas citra dengan pendekatan matematika. Pada penelitian ini digunakan pendekatan menggunakan PSNR(*Peak Signal to noise ratio*) untuk mengukur kebenaran hasil kompresi citra yang dalam hal ini adalah kualitas citra hasil peningkatan. Semakin besar nilai PSNR maka citra hasil semakin mendekati citra aslinya, dengan kata lain semakin sedikit perubahan yang terjadi pada citra tersebut. Sebaliknya, semakin kecil nilai PSNR, semakin baik kualitas citra hasil karena terdapat banyak perubahan yang terjadi. Rumus untuk menghitung PSNR

adalah sebagai berikut.

$$PSNR = 10\ \log_{10} \frac{(MAX_f)^2}{MSE} \quad \text{(II.44)}$$

MAX_f adalah nilai intensitas terbesar yang terdapat pada citra, yaitu 255. Di mana nilai MSE (*Mean Squared Error*) dihitung dengan persamaan berikut:

$$MSE = \frac{1}{MN} \sum_{i-1}^{M} \sum_{j-1}^{N} (f(i,j) - g(i,j))^2 \quad \text{(II.45)}$$

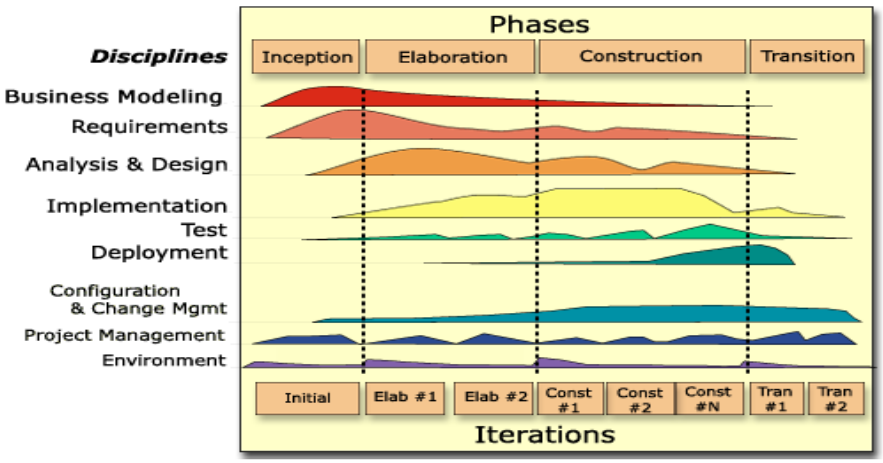
f dan g masing masing adalah representasi dari data matriks citra asli dan citra hasil kompresi, M adalah representasi jumlah baris piksel dari citra dan i adalah indeks baris, dan N adalah representasi jumlah kolom piksel dari citra dan j merupakan indeks kolom. PSNR diukur dengan satuan decibel (dB).

2.22 *Rational Unified Process* (RUP)

Metode pengembangan perangkat lunak yang digunakan pada penelitian ini adalah metode RUP (*Rational Unified Process*). RUP menganut konsep *object-oriented.* Objek digambarkan dengan mengilustrasikan benda, orang, tempat dan sebagainya memiliki atribut dan metode (Kruchten, 2000). Aktivitas pengembangan perangkat lunak berfokus pada pengembangan model dengan menggunakan *Unified Modeling Language* (UML). UML merupakan bahasa standar yang digunakan untuk memvisualisasikan, mendeskripsikan, membangun dan mendokumentasikan perangkat lunak. UML berfungsi menstandarisasi notasi paradigma berorientasi objek.

1. Struktur Proses RUP

Arsitektur keseluruhan dari proses RUP yang mempunyai struktur dua dimensi, arsitektur ini terdapat pada Gambar II.6.



Gambar II.11. Arsitektur *Rational Unified Process* (Kruchten, 2000)

Berdasarkan Gambar II.11 struktur proses dalam arsitektur RUP secara keseluruhan terdiri atas dua dimensi, yaitu:

a. Dimensi horizontal, dimensi ini menjelaskan aspek dinamis dari proses pengembangan perangkat lunak yang meliputi: siklus, fase, iterasi, dan *milestones*. Dimensi ini terdiri atas fase *inception, elaboration, construction, transition*.

b. Dimensi vertikal, dimensi ini menjelaskan aspek statis dari metode RUP yang meliputi; aktivitas, alur kerja, dan pekerja yang bertanggung jawab terhadap suatu proses. Alur kerja pada dimensi kedua ini terdiri atas; *business modeling*, *requirement*, *analysis and*

design, *implementation*, *test*, *deployment*, *configuration and change management*, *project management* dan *environment*.

2. Fase-Fase RUP

Berdasarkan Gambar II.6 terdapat empat fase dari model proses RUP yaitu:

a. Fase Insepsi *(Inception)*, fase ini berfokus pada pemodelan bisnis. Pemodelan bisnis tersebut antara lain; mendefinisikan masalah, membatasi ruang lingkup proyek, dan membangun *business case*;

b. Fase Elaborasi *(Elaboration),* menjelaskan aktifitas yang paling dominan pada analisis kebutuhan perangkat lunak, perancangan, dan menentukan resiko-resiko yang mungkin terjadi selama masa pengembangan perangkat lunak berlangsung;

c. Fase Konstruksi *(Construction)*, merupakan tahapan yang mengimplementasikan kebutuhan perangkat lunak ke dalam modul-modul dalam bentuk kode program. Selain itu, aktivitas pada fase ini juga berfokus terhadap pengujian perangkat lunak. Sehingga, perangkat lunak yang dihasilkan berkualitas dan sesuai dengan kebutuhan;

d. Fase Transisi (*Transition*), pada fase ini akan merilis versi beta dari perangkat lunak, melakukan distribusi perangkat lunak, mempersiapkan lingkungan operasi perangkat lunak, membuat buku panduan perangkat lunak dan melakukan pelatihan kepada *user*.