

**APLIKASI PENINGKATAN KONTRAS CITRA GRAYSCALE  
DENGAN ADAPTIVE FUZZY CONTRAST ENHANCEMENT  
ALGORITHM WITH DETAILS PRESERVING**

**TUGAS AKHIR**

Oleh:

**JUANDY HARTANTO**

NIM. 121110669

**KELVIN**

NIM. 121110651



**PROGRAM STUDI TEKNIK INFORMATIKA  
SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER  
MIKROSKIL  
MEDAN  
2016**

**GRAYSCALE IMAGE CONTRAST ENHANCEMENT  
USING ADAPTIVE CONTRAST ENHANCEMENT  
ALGORITHM WITH DETAILS PRESERVING**

**FINAL RESEARCH**

By:

**JUANDY HARTANTO**

ID. 121110669

**KELVIN**

ID. 121110651



**STUDY PROGRAM OF INFORMATICS ENGINEERING  
SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER  
MIKROSKIL  
MEDAN  
2016**

**LEMBARAN PENGESAHAN**

**APLIKASI PENINGKATAN KONTRAS CITRA GRAYSCALE  
DENGAN ADAPTIVE FUZZY CONTRAST ENHANCEMENT  
ALGORITHM WITH DETAILS PRESERVING**

**TUGAS AKHIR**

Diajukan untuk Melengkapi Persyaratan Guna  
Mendapatkan Gelar Sarjana Strata Satu  
Program Studi Teknik Informatika

Oleh:

**JUANDY HARTANTO**  
ID. 121110669

**KELVIN**  
ID. 121110651

Disetujui Oleh:

Dosen Pembimbing I,

Dosen Pembimbing II,

Syanti Irviantina, S.Kom., M.Kom.

Irpan Adiputra Pardosi, S.Kom., M.TI

Medan, ..... 20...  
Diketahui dan Disahkan Oleh:

Ketua Program Studi,  
Teknik Informatika,

Hardy, S.Kom., M.Sc.

## ABSTRAK

Kontras yang rendah pada suatu citra memberikan kesan sulitnya memahami informasi dan objek yang dimiliki oleh citra tersebut. Namun peningkatan kontras yang berlebihan dapat berakibat banyak informasi yang hilang dari citra awal. Untuk itu dilakukan pengolahan citra dengan meningkatkan kontras pada citra disertai dengan pelestarian informasi.

Salah satu metode yang diimplementasikan untuk meningkatkan kualitas kontras citra adalah *Adaptive Fuzzy Contrast Enhancement with Details Preserving*. Dengan menggunakan metode ini, kontras akan ditingkatkan dengan melakukan perataan histogram terhadap citra (*Histogram Equalization* / HE). Namun sebelum ditingkatkan, dilakukan pemotongan histogram (*Clipped Histogram*) agar mengurangi peningkatan kontras yang berlebihan. Nilai clipping limit didapatkan dengan fungsi *plateau level* yang disesuaikan pada tiap tingkat keabuan citra, baik itu rendah, menengah, dan tinggi. Dengan adanya elemen fuzzy, dapat mengoptimalkan penggunaan fungsi *plateau level* yang akan digunakan. Pengujian dilakukan dengan memproses citra asli dengan algoritma yang dibahas, dan membandingkan citra hasil algoritma dengan citra asli menggunakan *Shannon Entropy* dan *Contrast Improvement Evaluation*.

Pada Tugas akhir ini, aplikasi dapat meningkatkan kontras citra sekaligus melestarikan informasi citra. Berdasarkan pengujian yang dilakukan, nilai konstanta  $c_1$  paling optimal adalah -0.015 dan nilai konstanta  $c_2$  paling optimal adalah 0.005 hingga 0.007. Selain itu, algoritma *Adaptive Fuzzy Contrast Enhancement with Details Preserving* lebih baik dalam meningkatkan kontras dan melestarikan informasi dibandingkan dengan algoritma *Adaptive Contrast Enhancement with Details Preserving* dilihat dari nilai *Shannon Entropy* dan *Contrast Improvement Evaluation*.

Kata Kunci : Peningkatan Kontras Citra, *Clipped Histogram Equalization*, *Details Preserving*, *Fuzzy Logic*, *Shannon Entropy*

## KATA PENGANTAR

Puji syukur penulis panjatkan kepada Tuhan Yang Maha Esa karena berkat Rahmat dan Karunia-Nya penulis dapat menyelesaikan Tugas Akhir yang berjudul “Aplikasi Peningkatan Kontras Citra Grayscale dengan Adaptive Contrast Enhancement Algorithm with Details Preserving”.

Pengerjaan Tugas Akhir ini dilakukan untuk meningkatkan tingkat kontras dan melestarikan detail citra grayscale.

Dalam kesempatan ini, penulis menyampaikan ucapan terima kasih kepada:

1. Ibu Syanti Irviantina, S.Kom., M.Kom., selaku Dosen Pembimbing I yang telah membimbing selama pengerjaan Tugas Akhir ini.
2. Bapak Irpan Adiputra Pardosi, S.Kom., M.TI, selaku Dosen Pembimbing II yang telah membimbing selama pengerjaan Tugas Akhir ini.
3. Bapak Dr. Mimpin Ginting, M.S., selaku Ketua STMIK Mikroskil Medan.
4. Bapak Djoni, S.Kom., M.T.I., selaku Wakil Ketua I STMIK Mikroskil Medan.
5. Bapak Hardy, S.Kom., M.Sc., selaku Ketua Program Studi Teknik Informatika STMIK Mikroskil Medan.
6. Bapak dan ibu Dosen yang telah mendidik dan membimbing dalam mengerjakan Tugas Akhir ini.
7. Kepada orang tua dan keluarga yang telah memberikan dukungan doa, material, dan motivasi selama mengikuti pendidikan hingga selesainya Tugas Akhir ini.
8. Kepada sahabat-sahabat yang ikut memberikan bantuan dan dukungan dalam pengerjaan Tugas Akhir ini.
9. Semua pihak lain yang telah membantu dalam penyelesaian Tugas Akhir ini.

Tugas Akhir ini dibuat untuk melengkapi persyaratan guna mendapatkan gelar sarjana strata satu program studi Teknik Informatika STMIK Mikroskil Medan.

Penulis mengucapkan banyak terima kasih dan mengharapkan saran dan kritik yang bersifat membangun demi kesempurnaan Tugas Akhir ini.

## DAFTAR ISI

Abstrak .....	i
Kata Pengantar .....	ii
Daftar Isi.....	iii
Daftar Gambar.....	vi
Daftar Tabel .....	viii
Daftar Lampiran .....	x
BAB I Pendahuluan.....	1
1.1 Latar Belakang .....	1
1.2 Rumusan Masalah .....	2
1.3 Ruang Lingkup.....	3
1.4 Tujuan .....	3
1.5 Manfaat .....	3
1.6 Metodologi Pengembangan Sistem.....	4
BAB II Tinjauan Pustaka.....	6
2.1 Citra.....	6
2.1.1 Citra Analog dan Citra Digital .....	6
2.1.2 Proses Akuisisi Citra .....	6
2.1.3 Representasi Citra Digital .....	7
2.1.4 Jenis Citra .....	8
2.1.5 Format File Citra .....	9
2.1.6 Contrast, Low Contrast dan High Contrast. ....	11
2.2 Logika Fuzzy.....	11
2.2.1 Himpunan Fuzzy .....	12
2.2.2 Fungsi Keanggotaan .....	13
2.3 Pengolahan Citra .....	15
2.3.1 Konversi Citra ke Grayscale .....	16
2.3.2 Perbaikan Kualitas Citra.....	16
2.3.3 Peregangan Kontras ( <i>Contrast Stretching</i> ) .....	16
2.3.4 Histogram Equalization.....	18

2.4	Adaptive Histogram Equalization (AHE) on Image Gray Level Mapping .....	19
2.5	Adaptive Contrast Enhancement Algorithm with Details Preserving (ACEDP).....	21
2.6	Adaptive Fuzzy Contrast Enhancement Algorithm with Details Preserving (AFCEDP) .....	24
2.7	Perbandingan Citra.....	27
2.7.1	Shannon Entropy .....	28
2.7.2	Contrast Improvement Evaluation .....	28
BAB III Metodologi Penelitian .....		29
3.1	Analisis.....	29
3.1.1	Analisis Proses .....	29
3.1.1.1	Citra Asli.....	30
3.1.1.2	Analisis Determinasi Fungsi Keanggotaan dan Perhitungan Derajat Keanggotaan.....	31
3.1.1.3	Analisis Perhitungan Nilai Intensitas Referensi	33
3.1.1.4	Analisis Mendefinisikan Tiga Fungsi <i>Plateau</i> dan Melakukan Komputasi <i>Clipping Limit</i> .....	34
3.1.1.5	Analisis <i>Clipping</i> dan Ekualisasi Histogram ....	35
3.1.1.6	Analisis Perbandingan Citra Hasil dengan Input.	39
3.1.2	Analisis Kebutuhan Fungsional .....	45
3.2	Perancangan .....	53
3.2.1	Form Utama.....	53
3.2.2	Form Pengujian .....	54
3.2.3	Form Zoom.....	55
3.2.4	Form Log.....	56
3.2.5	Form About Us.....	57
BAB IV Hasil dan Pengujian .....		58
4.1	Hasil .....	58
4.2	Pengujian.....	64

BAB V Kesimpulan dan Saran .....	79
5.1 Kesimpulan .....	79
5.2 Saran.....	79
DAFTAR PUSTAKA .....	80



## DAFTAR GAMBAR

Gambar 2.1 Proses akuisisi citra digital .....	7
Gambar 2.2 Sistem Koordinasi Citra digital .....	7
Gambar 2.3 Gambar <i>low contrast</i> dan <i>high contrast</i> dengan histogramnya.....	11
Gambar 2.4 Kurva Linear Naik.....	13
Gambar 2.5 Kurva Linear Turun.....	14
Gambar 2.6 Kurva Segitiga.....	14
Gambar 2.7 Kurva Trapesium.....	15
Gambar 2.8 Peregangan Kontras .....	17
Gambar 2.9 Hasil <i>histogram equalization</i> dan histogram.....	18
Gambar 2.10 Relasi antara <i>entropy</i> dengan $\beta$ .....	20
Gambar 2.11 Klasifikasi Jenis Citra.....	22
Gambar 2.12 Determinasi dari <i>plateau level</i> berdasarkan jenis citra.....	22
Gambar 2.13 Distribusi derajat keanggotaan berbentuk trapesium .....	25
Gambar 2.14 Fungsi <i>plateau</i> .....	26
Gambar 3.1 <i>Flowchart</i> Proses Peningkatan kontras citra pada aplikasi menggunakan AFCEDP .....	30
Gambar 3.2 Contoh Citra Asli .....	30
Gambar 3.3 Fungsi keanggotaan untuk $\mu_{low}$ (a), $\mu_{mid}$ (b), $\mu_{high}$ (c).....	32
Gambar 3.4 Citra Akhir setelah dilakukan ekualisasi histogram.....	39
Gambar 3.5 Histogram Citra Awal dan Citra Akhir .....	39
Gambar 3.6 Citra Awal (gambar 3.3) dan Citra Akhir (gambar 3.5).....	40
Gambar 3.7 Citra Awal dan Citra Akhir.....	43
Gambar 3.8 Diagram <i>Use Case</i> dari Aplikasi.....	46
Gambar 3.9 Rancangan Tampilan <i>Form</i> Utama .....	53
Gambar 3.10 Rancangan <i>form</i> Pengujian.....	55
Gambar 3.11 Rancangan Tampilan <i>form</i> Zoom.....	56
Gambar 3.12 Rancangan <i>Form Log</i> .....	56
Gambar 3.13 Rancangan Tampilan <i>form</i> About .....	57
Gambar 4.1 Tampilan <i>Form</i> Utama .....	58

Gambar 4.2 Kotak Dialog <i>Open</i> .....	59
Gambar 4.3 Tampilan citra, nilai <i>contrast</i> , nilai <i>entropy</i> pada <i>Form</i> Utama.....	59
Gambar 4.4 Tampilan Hasil Peningkatan Kontras Citra pada <i>Form</i> Utama.....	60
Gambar 4.5 Kotak Dialog <i>Save As</i> .....	61
Gambar 4.6 Tampilan <i>Form</i> Tampil Citra .....	61
Gambar 4.7 Tampilan <i>Form</i> Tampil Citra ( <i>Zoom In</i> 200%).....	62
Gambar 4.8 Tampilan <i>Form</i> Tampil Citra ( <i>Zoom Out</i> 50%).....	62
Gambar 4.9 Tampilan <i>Form</i> Log .....	63
Gambar 4.10 Tampilan <i>Form</i> Percobaan Algoritma.....	63
Gambar 4.11 Tampilan <i>Form</i> About Us.....	64
Gambar 4.12 Hasil Pengujian Citra dengan Range HE 100 – 255 .....	76
Gambar 4.13 Hasil Pengujian Citra dengan Range HE 0 – 175 .....	76

## DAFTAR TABEL

Tabel 3.1 Tabel Hasil Perhitungan $H(k)$ .....	34
Tabel 3.2 Tabel Hasil Perhitungan $p(k)$ .....	34
Tabel 3.3 Tabel Hasil Perhitungan $new\_p(k)$ .....	36
Tabel 3.4 Tabel Hasil Perhitungan $c(k)$ .....	36
Tabel 3.5 Tabel Hasil Perhitungan $new\_p(k)'$ .....	37
Tabel 3.6 Tabel Hasil Perhitungan $c(k)$ .....	37
Tabel 3.7 Tabel Hasil Perhitungan Nilai keabuan baru. ....	38
Tabel 3.8 Probabilitas kemunculan nilai keabuan ke- $i$ .....	41
Tabel 3.9 Hasil Perhitungan $E$ .....	42
Tabel 3.10 Hasil perhitungan $g^2(u,v)$ .....	43
Tabel 3.11 Deskripsi proses “Pilih Citra” .....	47
Tabel 3.12 Deskripsi Proses “Buka Citra Asli” .....	47
Tabel 3.13 Deskripsi Proses “Gunakan Citra Asli sebagai Citra Awal” .....	47
Tabel 3.14 Deskripsi Proses “Hitung <i>Entropy</i> dan <i>Contrast</i> Citra Awal” .....	48
Tabel 3.15 Deskripsi Proses “Peningkatan Kontras Citra” .....	48
Tabel 3.16 Deskripsi Proses “Hitung <i>Entropy</i> dan <i>Contrast</i> semua Citra” .....	49
Tabel 3.17 Deskripsi Proses “Tulis log yang terjadi saat proses peningkatan” ....	49
Tabel 3.18 Deskripsi Proses “Simpan” .....	50
Tabel 3.19 Deskripsi Proses “Tampilkan Citra” .....	50
Tabel 3.20 Deskripsi Proses “Tampilkan Histogram Citra” .....	50
Tabel 3.21 Deskripsi Proses “Zoom Citra” .....	51
Tabel 3.22 Deskripsi Proses “Pengujian Beberapa Citra” .....	51
Tabel 3.23 Deskripsi Proses “Pilih lokasi beberapa citra yang akan diuji” .....	51
Tabel 3.24 Deskripsi Proses “Hitung <i>Entropy</i> dan <i>Contrast</i> untuk semua citra” .	52
Tabel 3.25 Deskripsi Proses “Hitung rata-rata <i>Entropy</i> dan <i>Contrast</i> dari Hasil Pengujian” .....	52
Tabel 4.1 Pengujian beberapa nilai $cI$ .....	66
Tabel 4.2 Pengujian nilai $c2$ dengan nilai $cI = -0,015$ .....	73
Tabel 4.3 Pengujian Batas Bawah dan Batas Atas HE .....	75

Tabel 4.4 Pengujian <i>Entropy</i> dan <i>Contrast</i> untuk AFCEDP dan ACEDP.....	77
--	----

## **DAFTAR LAMPIRAN**

Lampiran 1.	Listing Program .....	81
Lampiran 2.	Daftar Riwayat Hidup.....	115

# BAB I

## PENDAHULUAN

### 1.1 Latar Belakang

Peningkatan kontras dengan pelestarian detail pada citra memainkan peran yang sangat penting di berbagai bidang termasuk bidang medis, militer dan pertanian. Tujuan dari peningkatan kontras adalah untuk membentuk citra dengan kualitas visual yang lebih baik dengan memanipulasi intensitas piksel pada citra. (Tang & Mat Isa, 2014). Akan tetapi, saat proses peningkatan kontras tersebut dilakukan akan terjadi perubahan detail yang terkandung pada citra. Hal tersebut dapat berdampak pada tahap proses citra lanjutan seperti pengenalan pola, yang menyebabkan kurang efektifnya dalam mengenali citra yang disebabkan oleh hilangnya beberapa detail pada citra ataupun bertambahnya objek-objek yang tidak diinginkan pada citra.

Algoritma *Histogram Equalization (HE)* merupakan salah satu algoritma yang dipakai secara luas untuk menyelesaikan masalah peningkatan kontras citra. Kekurangan algoritma HE adalah terbentuknya objek-objek yang tidak diinginkan pada citra sehingga memungkinkan terjadinya penurunan detail citra. (Tang & Mat Isa, 2014). Algoritma *Adaptive Histogram Equalization (AHE)* merupakan algoritma yang digunakan untuk menyelesaikan permasalahan pada *Histogram Equalization*. Algoritma AHE menerapkan fungsi *Entropy* untuk memperkecil kehilangan informasi yang terjadi saat proses peningkatan kontras dilakukan. (Zhu & Huang, 2012). Kekurangan algoritma AHE adalah peningkatan kontras yang dilakukan menjadi tidak maksimal karena sangat terfokus dalam pelestarian detail citra.

Metode *Adaptive Contrast Enhancement Algorithm with Details Preserving (ACEDP)* dikembangkan untuk meningkatkan kontras lebih baik dari AHE, namun tetap melestarikan detail citra lebih baik dari HE. Kelemahan ACEDP adalah penentuan kategori citra yang tidak efektif, yang mengabaikan kemungkinan dari kategori citra yang dianalisa sehingga citra yang dihasilkan kurang optimal. Metode

*Adaptive Fuzzy Contrast Enhancement Algorithm with Details Preserving* (AFCEDP) merupakan perkembangan dari ACEDP, dimana algoritma AFCEDP ini menerapkan unsur *fuzzy* dalam penentuan kategori citra sehingga penentuan kategori citra menjadi lebih efektif. Awalnya, citra yang ingin diuji diproses terlebih dahulu untuk menentukan derajat keanggotaannya. Setelah itu, lakukan perhitungan untuk mendapatkan *Clipping Limit* dan melakukan perataan histogram yang menggunakan fungsi transformasi histogram dimana fungsi tersebut diyakini lebih baik dari fungsi transformasi konvensional. (Ooi, et al., 2009). Tes numerik menunjukkan bahwa algoritma tersebut mampu meningkatkan kontras dan melestarikan detail dari citra. Pengujian dilakukan dengan menggunakan *Shannon Entropy* dan *Contrast Improvement Evaluation* (CIE). (Tang & Mat Isa, 2014). Pengujian juga akan dilakukan dengan membandingkan hasil dari ACEDP dengan AFCEDP guna untuk membuktikan tingkat efektivitas dari unsur *fuzzy* pada AFCEDP.

Berdasarkan uraian diatas maka penulis mengambil Tugas Akhir yang berjudul “**Aplikasi Peningkatan Kontras Citra Grayscale dengan *Adaptive Fuzzy Contrast Enhancement with Details Preserving***”.

## 1.2 Rumusan Masalah

Berdasarkan latar belakang pada uraian sebelumnya, maka yang menjadi rumusan masalah dalam penelitian ini adalah :

1. Apakah algoritma *Adaptive Fuzzy Contrast Enhancement with Details Preserving* mampu meningkatkan kontras dan mampu menjaga kelestarian sebuah citra *grayscale* secara perhitungan numerik dengan metode *Shannon Entropy* dan *Contrast Improvement Evaluation*.
2. Bagaimana parameter  $c1$  dan  $c2$  serta batas bawah dan batas atas HE mempengaruhi hasil citra output.
3. Bagaimana membandingkan citra keluaran yang menggunakan algoritma *Adaptive Fuzzy Contrast Enhancement with Details Preserving* dengan *Adaptive Contrast Enhancement with Details Preserving*.

### 1.3 Ruang Lingkup

Agar pembahasan masalah lebih fokus, maka dilakukan beberapa pembatasan masalah yakni :

1. *Input* berupa *file* berformat citra \*.bmp, \*.jpg, \*.png, \*.tiff, \*.gif.
2. Citra yang digunakan bersifat statis atau tidak bergerak.
3. Citra warna akan otomatis diubah menjadi citra *grayscale* dengan menggunakan algoritma *luma*.
4. Citra input bersumber dari *CVG-UGR-Database* (Group, 2002) yang merupakan citra grayscale 8 bit.
5. Fungsi keanggotaan yang digunakan untuk memetakan himpunan *fuzzy* adalah fungsi keanggotaan dengan representasi kurva trapesium.
6. Batas bawah dan batas atas HE adalah 0 sampai 255.
7. Konstanta landaian *c1* dan *c2* yang terdapat pada fungsi clipping limit berjarak antara [-0.015, -0.005] dan [0.005, 0.007] secara berurutan.

### 1.4 Tujuan

Tujuan dari tugas akhir ini adalah untuk membuat sebuah aplikasi yang dapat meningkatkan kontras dan melestarikan detail citra grayscale dengan menggunakan metode *Adaptive Fuzzy Contrast Enhancement with Details Preserving*.

### 1.5 Manfaat

Manfaat yang diperoleh dari tugas akhir ini adalah:

1. Aplikasi tersebut dapat digunakan untuk meningkatkan kontras dan menjaga kelestarian informasi sebuah citra *grayscale*.
2. Laporan dari tugas akhir ini dapat digunakan sebagai referensi untuk pembuatan tugas akhir yang bertopik peningkatan kontras citra, khususnya pada citra grayscale.



## 1.6 Metodologi Pengembangan Sistem

Metodologi yang digunakan untuk menyelesaikan tugas akhir ini adalah *waterfall* model dengan susunan langkah-langkah sebagai berikut.

### 1. Persiapan

Persiapan dilakukan dengan mengumpulkan data, bahan dan materi serta mempelajari materi berdasarkan tugas akhir yang dibuat.

### 2. Dalam pembuatan perangkat lunak, dilakukan hal berikut.

#### a. Desain Sistem

Tahap dimana dilakukan perancangan sistem dan pemodelan sistem sebelum memulai proses *coding*. Proses ini berfokus pada perancangan arsitektur perangkat lunak, cara kerja sistem dan rancangan tampilan utama perangkat lunak.

#### b. Penulisan Kode Program (*Coding*)

Penulisan Kode Program merupakan suatu tahapan penerjemahan design dan rancangan sistem perangkat lunak ke dalam bahasa yang dimengerti oleh komputer. Tahapan ini merupakan tahapan secara nyata dalam pengerjaan aplikasi pada tugas akhir ini. Proses *coding* menggunakan bahasa pemrograman Microsoft Visual C#.Net dengan menggunakan aplikasi Microsoft Visual Studio 2013.

#### c. Pengujian

Pengujian akan dilakukan dengan sampel citra dari *CVG-UGR-Database* (Group, 2002) dengan syarat ukuran citra 512 x 512 piksel, jenis citra berupa *grayscale* dan format citra berupa GIF. Kemudian citra tersebut akan diolah dengan menggunakan algoritma AFCEDP sehingga menghasilkan citra baru yang telah mengalami peningkatan kontras citra. Selanjutnya dilakukan pengujian citra keluaran dengan metode *Shannon Entropy* dan *Contrast Improvement Evaluation* untuk membuktikan secara numerik bahwa citra hasil algoritma telah mengalami peningkatan. Parameter  $c1$ ,  $c2$ , batas bawah HE dan batas atas HE juga akan diuji menggunakan sampel citra dari *CVG-UGR-Database* (Group, 2002) serta dilakukan perhitungan secara numerik yakni menggunakan metode *Shannon Entropy* dan *Contrast*

*Improvement Evaluation* untuk mencari nilai  $c1$  dan  $c2$  yang paling optimal. Pengujian secara numerik juga akan dilakukan terhadap algoritma ACEDP (tanpa *fuzzy*) dengan menggunakan citra sampel dari *CVG-UGR-Database* (Group, 2002), guna untuk membandingkan hasil yang diperoleh dari algoritma ACEDP dengan algoritma AFCEDP.

d. Kesimpulan

Memberikan kesimpulan terhadap pengujian yang telah dilakukan.

3. Tahap akhir adalah pembuatan laporan sesuai dengan panduan yang telah diberikan guna memenuhi persyaratan tugas akhir.

## **BAB II**

### **TINJAUAN PUSTAKA**

#### **2.1 Citra**

Citra adalah suatu representasi (gambaran), kemiripan, atau imitasi dari suatu objek. Citra sebagai keluaran suatu sistem perekaman data dapat bersifat optik berupa foto, bersifat analog berupa sinyal – sinyal video seperti gambar pada monitor televisi, atau bersifat digital yang dapat langsung disimpan pada suatu media penyimpanan. (Sutoyo, et al., 2009, p. 9)

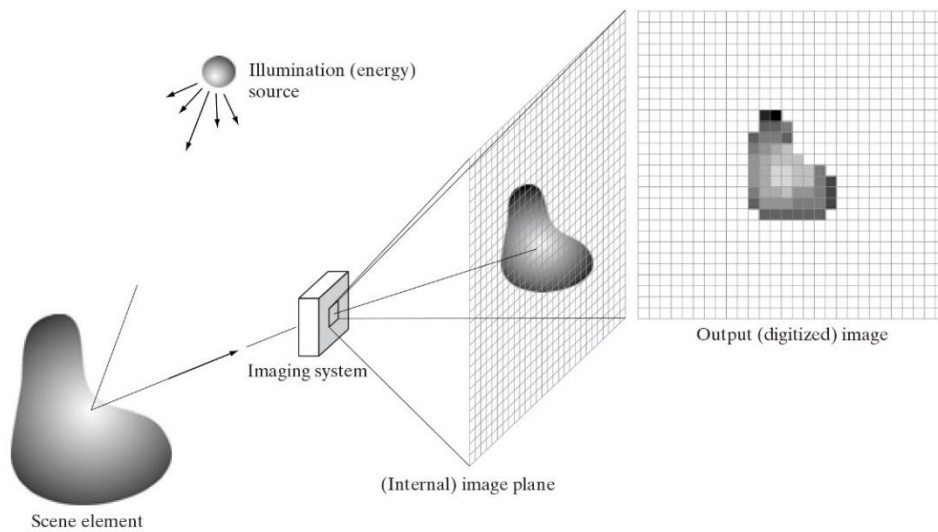
##### **2.1.1 Citra Analog dan Citra Digital**

Citra analog adalah citra yang bersifat kontinu, seperti gambar pada monitor televisi, foto sinar X, foto yang tercetak dikertas foto, lukisan, pemandangan alam, hasil CT scan dan lain sebagainya. Citra analog tidak dapat dipresentasikan dalam komputer sehingga tidak bisa diproses di komputer secara langsung. Oleh sebab itu, agar citra ini dapat diproses di komputer, proses konversi analog ke digital harus dilakukan terlebih dahulu. Citra analog dihasilkan dari alat-alat analog diantaranya adalah video kamera analog, kamera foto analog dan CT scan.

Sedangkan, citra digital adalah sebuah larik (*array*) yang berisi nilai - nilai real maupun kompleks yang direpresentasikan dengan deretan bit tertentu. (Putra, 2010, p. 19).

##### **2.1.2 Proses Akuisisi Citra**

Proses akuisisi citra adalah pemetaan suatu pandangan (*scene*) menjadi citra kontinu dengan menggunakan sensor. Ada beberapa macam sensor untuk akuisisi citra, yaitu sensor tunggal (*single sensor*), sensor garis (*sensor strip*), dan sensor larik (*sensor array*). Proses akuisisi citra dapat dilihat pada gambar 2.1. (Putra, 2010, p. 28).

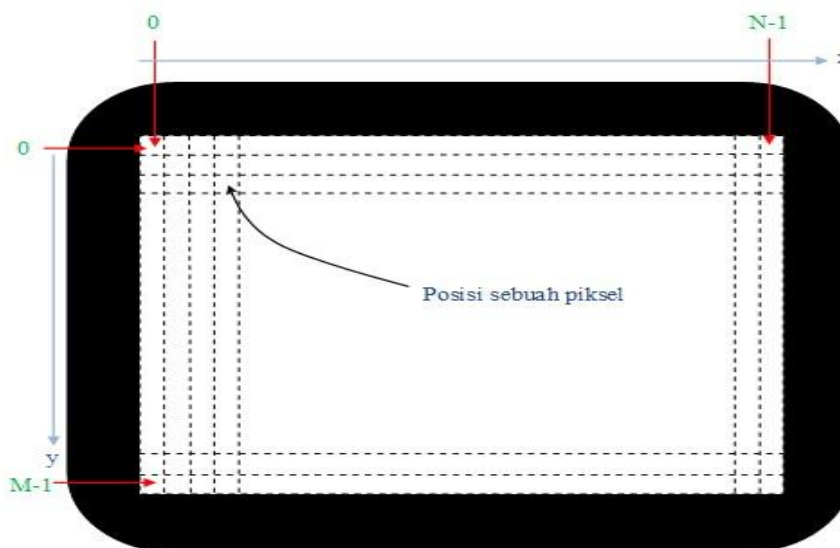


**Gambar 2.1** Proses akuisisi citra digital

(Sumber: Gonzalez & Woods, 2002, p. 20).

### 2.1.3 Representasi Citra Digital

Citra digital dibentuk oleh kumpulan titik yang dinamakan piksel (*pixel* atau “picture element”). Setiap piksel digambarkan sebagai satu kotak kecil. Setiap piksel mempunyai koordinat posisi. Sistem koordinat yang dipakai untuk menyatakan citra digital ditunjukkan di Gambar 2.2.



**Gambar 2.2** Sistem Koordinasi Citra digital

(Sumber: Kadir & Susanto, 2013, p. 10)

Citra digital yang berukuran  $M \times N$  dapat ditulis dalam bentuk matrik sebagai berikut.

$$f(x, y) = \begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,N-1) \\ f(1,0) & f(1,1) & \cdots & f(1,N-1) \\ \vdots & \vdots & \vdots & \vdots \\ f(M-1,0) & f(M-1,1) & \cdots & f(M-1,N-1) \end{bmatrix} \quad \dots (2.1)$$

Nilai pada suatu irisan antara baris dan kolom pada posisi  $(x, y)$  disebut dengan *picture elements*, *image elements*, *pels*, atau *pixels*. Istilah terakhir (pixel) paling sering digunakan pada citra digital. (Putra, 2010, p. 20).

#### 2.1.4 Jenis Citra

Nilai suatu piksel memiliki nilai dalam rentang tertentu dari nilai minimum sampai nilai maksimum. Jangkauan yang digunakan berbeda-beda tergantung jenis warnanya. Namun secara umum jangkauannya adalah 0 – 255. Citra dengan penggambaran seperti ini digolongkan ke dalam citra integer. Berikut adalah jenis-jenis citra berdasarkan nilai pixelnya.

##### 1. Citra Biner

Citra biner adalah citra digital yang hanya memiliki dua kemungkinan nilai pixel yaitu hitam dan putih. Citra biner juga disebut sebagai citra monokrom atau B&W (*black and white*). Hanya dibutuhkan 1 bit untuk mewakili nilai tiap pixel dari citra biner.

##### 2. Citra Grayscale

Citra *grayscale* merupakan citra digital yang hanya memiliki satu kanal oada setiap pixelnya, dengan kata lain nilai bagian RED = GREEN = BLUE. Nilai tersebut digunakan untuk menunjukkan tingkat intensitas. Warna yang dimiliki adalah warna dari hitam , keabuan dan putih.

##### 3. Citra Warrna (8 bit)

Setiap piksel dari citra wanra (8 bit) hanya diwakili oleh 8 bit dengan jumlah warna maksimum yang dapat digunakan adalah 256 warna.

4. Citra Warna (16 bit)

Citra warna 16 bit (biasanya disebut sebagai citra *highcolor*) dengan setiap pikselnya diwakili dengan 2 *byte memory* (16 bit). Warna 16 bit memiliki 65.536 warna. Dalam formasi bitnya, nilai merah dan biru mengambil tempat di 5 bit di kanan dan kiri. Komponen hijau memiliki 5 bit ditambah 1 bit ekstra. Pemilihan komponen hijau dengan deret 6 bit dikarenakan penglihatan manusia lebih sensitif terhadap warna hijau.

5. Citra Warna (24 bit)

Setiap pixel dari citra warna 24 bit diwakili dengan 24 bit sehingga total 16.777.216 variasi warna. Variasi ini sudah lebih dari cukup untuk memvisualkan seluruh warna yang dapat dilihat penglihatan manusia. (Putra, 2010, pp. 39-44).

### 2.1.5 Format File Citra

Format *file* citra standar yang digunakan saat ini terdiri dari beberapa jenis. Format-format ini digunakan dalam menyimpan citra dalam sebuah *file*. Setiap format memiliki karakteristik masing-masing. Berikut adalah penjelasan beberapa format citra yang umum digunakan saat ini:

1. Bitmap (\*.bmp)

Format .bmp adalah format penyimpanan standar tanpa kompresi yang umum dapat digunakan untuk menyimpan citra biner hingga citra warna. Format ini terdiri dari beberapa jenis yang setiap jenisnya ditentukan dengan jumlah bit yang digunakan untuk menyimpan sebuah nilai piksel.

2. JPEG (\*.jpg)

Format .jpg adalah format yang sangat umum digunakan saat ini, khususnya untuk transmisi citra. Format ini digunakan untuk menyimpan citra hasil kompresi dengan metode JPEG.

3. *Graphics Interchange Format* (\*.gif)

Format ini dapat digunakan pada citra warna dengan palet 8 bit. Pada umumnya digunakan pada aplikasi web. Kualitas yang rendah

menyebabkan format ini tidak terlalu populer di kalangan peneliti pengolahan citra digital.

4. *Tagged Image Format* (\*.tif, \*.tiff)

Format ini merupakan format citra yang dapat digunakan untuk menyimpan citra *bitmap* hingga citra dengan warna palet terkompresi. Format ini dapat digunakan untuk menyimpan citra yang tidak terkompresi dan juga citra terkompresi.

5. *Portable Network Graphics* (\*.png)

Format .png adalah format penyimpanan citra terkompresi. Format ini dapat digunakan pada citra *grayscale*, citra dengan palte warna dan juga citra *full color*. Format .png juga mampu menyimpan informasi hingga kanal *alpha* dengan penyimpanan sebesar 1 hingga 16 bit per kanal.

6. MPEG (\*.mpg)

Format ini digunakan di dunia *internet* dan diperuntukkan sebagai format penyimpanan citra bergerak (*video*).

7. RGB (\*.rgb)

Format ini merupakan format penyimpanan citra yang dibuat oleh *silicon graphics* untuk menyimpan citra berwarna.

8. RAS (\*.ras)

Format .ras digunakan untuk menyimpan citra RGB tanpa kompresi.

9. *Postscript* (\*.pas, \*.epas, \*.epfs)

Format ini diperkenalkan sebagai format untuk menyimpan citra buku elektronik. Dalam format ini, citra direpresentasikan ke dalam deret nilai desimal atau heksidesimal yang dikodekan ke dalam ASCII.

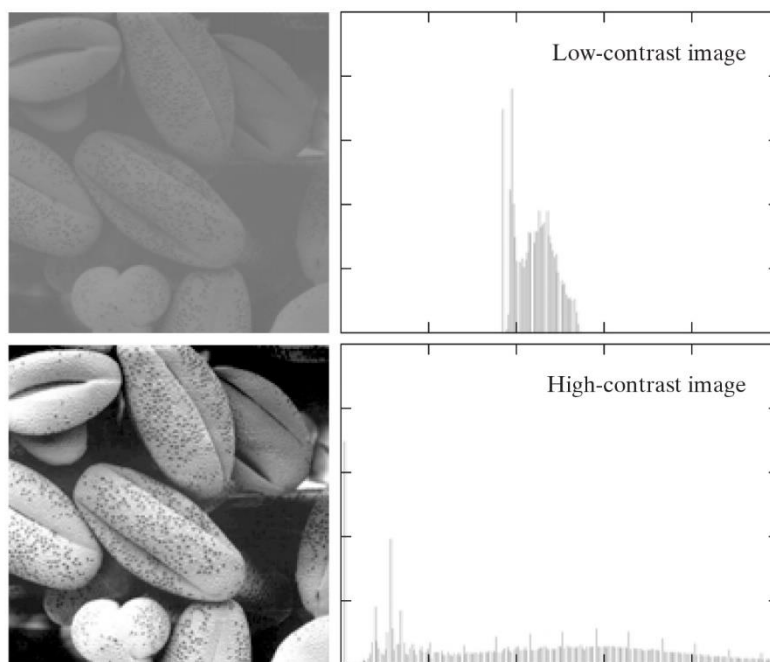
10. *Portable Image File Format*

Format ini memiliki beberapa bagian, di antaranya adalah *portable bitmap*, *portable graymap*, *portable pixmap*, dan *portable network map*, dengan format berturut-turut adalah .pbm, .pgm, .ppm, dan .pnm. (Putra, 2010)

### 2.1.6 Contrast, Low Contrast dan High Contrast.

Kontras suatu citra adalah distribusi atau tingkat penyebaran piksel-piksel ke dalam intensitas warna. Sebuah citra *grayscale* dengan kontras rendah maka akan terlihat terlalu gelap, terlalu terang, atau terlalu abu-abu. Histogram citra dengan kontras rendah, semua *pixel* akan terkonsentrasi pada sisi kiri, sisi kanan atau ditengah (Gambar 2.3). Semua *pixel* akan terkelompok secara rapat pada suatu sisi tertentu dan menggunakan sebagian kecil dari semua kemungkinan nilai *pixel*. (Putra, 2010)

Citra dengan kontras tinggi memiliki daerah gelap dan tereang yang luas. Histogram citra dengan kontras tinggi memiliki perataan yang merata di semua bagian histogram (Gambar 2.3).



**Gambar 2.3** Gambar *low contrast* dan *high contrast* dengan histogramnya.

(Sumber : Gonzalez & Woods, 2002, p. 107)

## 2.2 Logika Fuzzy

Logika *fuzzy* adalah suatu cara yang tepat untuk memetakan suatu ruang input ke dalam suatu ruang output. Ada beberapa alasan mengapa orang menggunakan logika *fuzzy*, antara lain:



1. Konsep logika *fuzzy* mudah dimengerti. Konsep matematis yang mendasari penalaran *fuzzy* sangat sederhana dan mudah dimengerti.
2. Logika *fuzzy* sangat fleksibel.
3. Logika *fuzzy* memiliki toleransi terhadap data-data yang tidak tepat.
4. Logika *fuzzy* mampu memodelkan fungsi-fungsi nonlinear yang sangat kompleks.
5. Logika *fuzzy* dapat membangun dan mengaplikasikan pengalaman-pengalaman para pakar secara langsung tanpa harus melalui proses pelatihan.
6. Logika *fuzzy* dapat bekerjasama dengan teknik-teknik kendali secara konvensional.
7. Logika *fuzzy* didasarkan pada bahasa alami.

### 2.2.1 Himpunan Fuzzy

Pada himpunan tegas ( *crisp* ), nilai keanggotaan suatu item  $x$  dalam suatu himpunan  $A$ , yang sering ditulis dengan  $\mu_A[x]$ , memiliki dua kemungkinan, yaitu:

- Satu (1), yang berarti bahwa suatu item menjadi anggota dalam suatu himpunan, atau
- Nol (0), yang berarti bahwa suatu item tidak menjadi anggota dalam suatu himpunan.

Sedangkan pada himpunan *fuzzy* nilai keanggotaannya terletak pada rentang 0 sampai 1. Apabila  $x$  memiliki nilai keanggotaan *fuzzy*  $\mu_A[x] = 0$  berarti  $x$  tidak menjadi anggota himpunan  $A$ , demikian pula apabila  $x$  memiliki nilai keanggotaan *fuzzy*  $\mu_A[x] = 1$  berarti  $x$  menjadi anggota penuh pada himpunan  $A$ . Himpunan *fuzzy* memiliki 2 atribut, yaitu:

- a. Linguistik, yaitu penamaan suatu grup yang mewakili suatu keadaan atau kondisi tertentu dengan menggunakan bahasa alami, seperti MUDA, PAROBAYA, TUA.
- b. Numeris, yaitu suatu nilai (angka) yang menunjukkan ukuran dari suatu variabel seperti 40, 25, 50, dsb.

Ada beberapa hal yang perlu diketahui dalam memahami sistem *fuzzy*, yaitu:

- a. Variabel *fuzzy*, merupakan variabel yang hendak dibahas dalam suatu sistem *fuzzy*.
- b. Himpunan *fuzzy*, merupakan suatu grup yang mewakili suatu kondisi atau keadaan tertentu dalam suatu variabel *fuzzy*.
- c. Semesta Pembicaraan, adalah 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.
- d. Domain, merupakan keseluruhan nilai yang diijinkan dalam semesta pembicaraan dan boleh dioperasikan dalam suatu himpunan *fuzzy*.

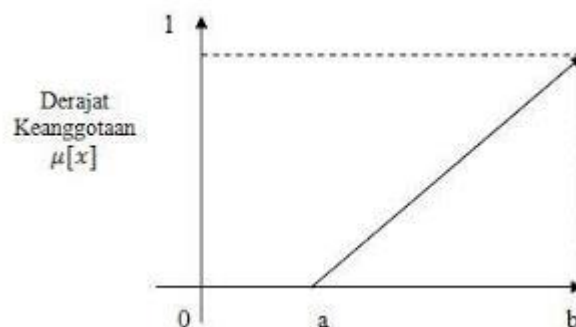
### 2.2.2 Fungsi Keanggotaan

Fungsi keanggotaan (*membership function*) adalah suatu kurva yang menunjukkan pemetaan titik-titik input data ke dalam nilai keanggotaannya (sering juga disebut dengan derajat keanggotaan) yang memiliki interval antara 0 sampai 1. Ada beberapa fungsi yang bisa digunakan.

#### 1. Representasi Kurva Linear

Pada representasi kurva linear, pemetaan input ke derajat keanggotaannya digambarkan sebagai suatu garis lurus. Terdapat 2 jenis keadaan pada representasi tersebut yakni

- a. Linear naik, himpunan yang dimulai dari domain dengan nilai keanggotaan 0 ke nilai domain yang memiliki derajat keanggotaan lebih tinggi.

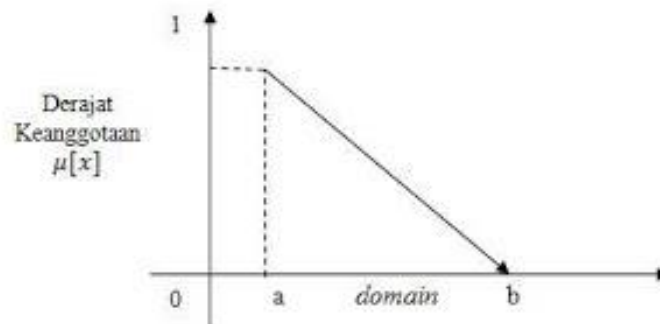


**Gambar 2.4** Kurva Linear Naik

Dengan fungsi keanggotaan sebagai berikut.

$$\mu [x] = \begin{cases} 0; & x \leq a \\ \frac{x-a}{b-a}; & a \leq x \leq b \\ 1; & x \geq b \end{cases} \quad \dots(2.2)$$

- b. Linear turun, himpunan yang dimulai dari domain dengan nilai keanggotaan 1 ke nilai domain yang memiliki derajat keanggotaan 0.



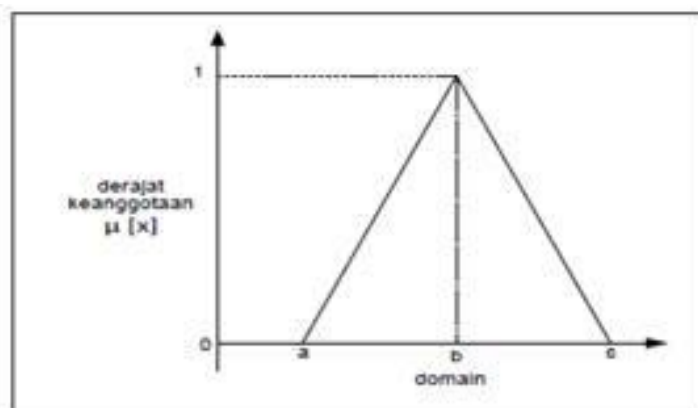
**Gambar 2.5** Kurva Linear Turun

Dengan fungsi keanggotaan sebagai berikut.

$$\mu [x] = \begin{cases} \frac{b-x}{b-a}; & a \leq x \leq b \\ 0; & x \geq b \end{cases} \quad \dots(2.3)$$

## 2. Representasi Kurva Segitiga

Kurva segitiga pada dasarnya merupakan gabungan antara 2 garis (linear) seperti yang terlihat pada gambar 2.6.



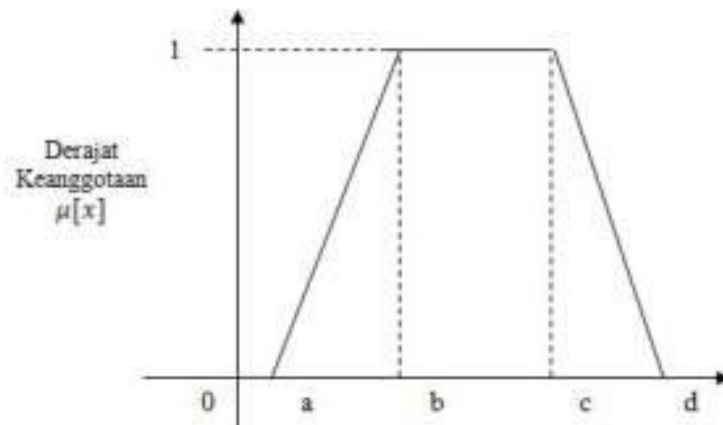
**Gambar 2.6** Kurva Segitiga

Dengan fungsi keanggotaan sebagai berikut.

$$\mu[x] = \begin{cases} 0; & x \leq a \text{ atau } x \geq c \\ (x - a)/(b - a); & a \leq x \leq b \\ (b - x)/(c - b); & b \leq x \leq c \end{cases} \quad \dots(2.4)$$

### 3. Representasi Kurva Trapesium

Kurva trapesium pada dasarnya seperti bentuk segitiga, hanya saja ada beberapa titik yang memiliki nilai keanggotaan 1 (gambar 2.7)



**Gambar 2.7** Kurva Trapesium

Dengan fungsi keanggotaan sebagai berikut. (Kusumadewi & Purnomo, 2004)

$$\mu[x] = \begin{cases} 0; & x \leq a \text{ atau } x \geq d \\ (x - a)/(b - a); & a \leq x \leq b \\ 1; & b \leq x \leq c \\ (d - x)/(d - c); & x \geq d \end{cases} \quad \dots(2.5)$$

### 2.3 Pengolahan Citra

Meskipun sebuah citra kaya informasi, namun seringkali citra mengalami penurunan mutu (degradasi), misalnya mengandung cacat atau derau (*noise*), warnanya terlalu kontras, kurang tajam, kabur (*blurring*), dan sebagainya. Tentu saja citra semacam ini menjadi lebih sulit diinterpretasi karena informasi yang

disampaikan oleh citra tersebut menjadi berkurang. Agar citra yang mengalami gangguan mudah diinterpretasi (baik oleh manusia maupun mesin), maka citra tersebut perlu dimanipulasi menjadi citra lain yang kualitasnya lebih baik. Bidang studi yang menyangkut hal ini adalah pengolahan citra (*image processing*). (Munir , 2004, p. 3).

### 2.3.1 Konversi Citra ke Grayscale

Konversi citra warna ke citra *grayscale* dapat dilakukan dengan berbagai cara. Beberapa diantaranya yaitu : konversi warna citra ke *grayscale* menggunakan cara klasik dan konversi citra warna ke *grayscale* menggunakan teknik *luma*.

Konversi citra ke *grayscale* dengan menggunakan teknik klasik dilakukan dengan rumus berikut.

$$Gray = (R + G + B) / 3 \quad \dots(2.6)$$

*Gray* menunjukkan nilai *gray* yang baru, *R* adalah nilai *Red* pada citra warna asli, *G* adalah nilai *Green* pada citra warna asli, dan *B* adalah nilai *Blue* pada citra warna asli. Sedangkan konversi citra ke *grayscale* dengan menggunakan teknik *luma* dapat dilakukan dengan menggunakan rumus berikut. (Tanner, 2011)

$$Gray = (R * 0.3) + (G * 0.59) + (B * 0.11) \quad \dots(2.7)$$

### 2.3.2 Perbaikan Kualitas Citra

Perbaikan citra bertujuan untuk meningkatkan kualitas tampilan citra untuk pandangan manusia atau untuk mengkonversi suatu citra agar memiliki format yang lebih baik sehingga citra tersebut menjadi lebih mudah diolah dengan mesin (komputer). (Putra, 2010, p. 119).

### 2.3.3 Peregangan Kontras (*Contrast Stretching*)

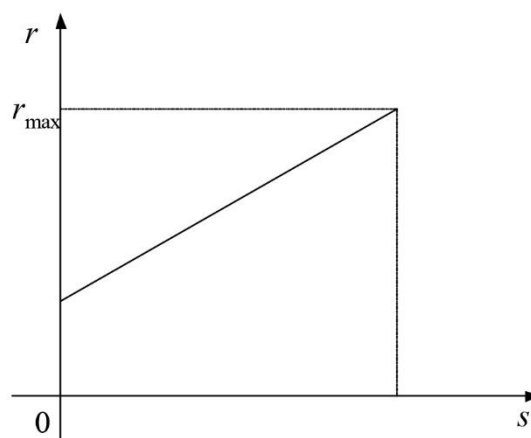
Peregangan kontras adalah teknik yang digunakan untuk memperbaiki kontras citra terutama citra yang memiliki kontras rendah. Melalui operasi ini, nilai-nilai keabuan *pixel* akan merentang dari 0 sampai 255 (pada citra 8-bit), dengan kata lain seluruh nilai keabuan *pixel* terpakai secara merata.

Algoritma peregangan kontras adalah sebagai berikut.

1. Cari batas bawah pengelompokan *pixel* dengan cara memindai (*scan*) histogram dari nilai keabuan terkecil ke nilai keabuan terbesar (0 sampai 255) untuk menemukan *pixel* pertama yang melebihi nilai ambang pertama yang telah dispesifikasikan.
2. Cari batas atas pengelompokan *pixel* dengan cara memindai histogram dari nilai keabuan tertinggi ke nilai keabuan terendah (255 sampai 0) untuk menemukan nilai *pixel* pertama yang lebih kecil dari nilai ambang kedua yang dispesifikasikan.
3. *Pixel-pixel* yang berada dibawah nilai ambang pertama di-*set* sama dengan 0, sedangkan *pixel-pixel* yang berada di atas nilai ambang kedua di-*set* sama dengan 255.
4. *Pixel-pixel* yang berada diantara nilai ambang pertama dan nilai ambang kedua dipetakan (diskalakan) untuk memenuhi rentang nilai-nilai keabuan yang lengkap (0 sampai 255) dengan persamaan:

$$s = \frac{r - r_{\min}}{r_{\max} - r_{\min}} \times 255 \quad \text{.....(2.8)}$$

yang dalam hal ini,  $r$  adalah nilai keabuan dalam citra semula,  $s$  adalah nilai keabuan yang baru,  $r_{\min}$  adalah nilai keabuan terendah dari kelompok *pixel*, dan  $r_{\max}$  adalah nilai keabuan tertinggi dari kelompok *pixel*. (Gambar 2.8) (Munir , 2004, pp. 94-96).



**Gambar 2.8** Peregangan Kontras

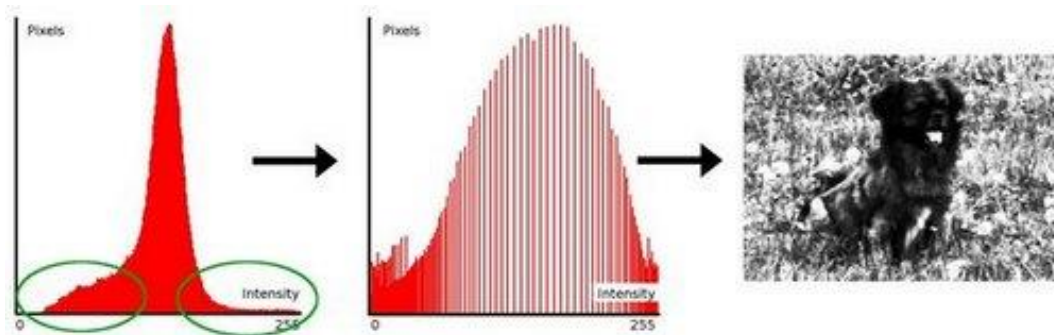
(Sumber : Munir , 2004, p. 96)

### 2.3.4 Histogram Equalization

Histogram citra merupakan diagram yang menggambarkan frekuensi setiap nilai intensitas yang muncul di seluruh piksel citra. Nilai besar menyatakan bahwa piksel-piksel yang mempunyai intensitas tersebut sangat banyak. (Kadir & Susanto, 2013, p. 36).

Ekualisasi Histogram adalah suatu proses untuk meratakan histogram agar derajat keabuan dari yang paling rendah (0) sampai dengan yang paling tinggi (255) mempunyai kemunculan yang rata. Dengan *histogram equalization*, hasil gambar yang memiliki histogram yang tidak merata atau distribusi kumulatif yang banyak loncatan gradiasinya akan menjadi gambar yang lebih jelas karena derajat keabuannya tidak dominan gelap atau dominan terang. Proses *histogram equalization* ini menggunakan distribusi kumulatif, karena pada proses ini dilakukan perataan *gradient* dari distribusi kumulatifnya. Tujuan dari HE adalah untuk memperoleh penyebaran histogram yang merata sehingga setiap derajat keabuan memiliki jumlah piksel yang relatif sama. (Sutoyo, et al., 2009, p. 46)

Dengan menggunakan *histogram equalization*, maka histogram hasil ekualisasi akan disebarakan (*spreading*). Hasil *histogram equalization* dan histogramnya dapat dilihat pada gambar 2.9



**Gambar 2.9** Hasil *histogram equalization* dan histogram

(Sumber : OpenCV Documentation 2.4.11.0)

Langkah awal untuk melakukan *histogram equalization* adalah penentuan *probability density function* (PDF) dari citra dengan menggunakan rumus :

$$p(k) = \frac{H(k)}{N}, \text{ for } k = 0, 1, \dots, L-1 \quad \dots(2.9)$$

Dimana  $N$  adalah total *pixel* yang berada di dalam citra,  $H(k)$  menyatakan nilai intensitas  $k$  didalam citra dan  $L$  adalah total tingkat keabuan yang terdapat pada citra. Langkah selanjutnya adalah penentuan fungsi kumulatif dengan menggunakan *cumulative density function* (CDF), yang didefinisikan sebagai berikut:

$$c(k) = \sum_{i=0}^k p(i), \text{ for } k = 0, 1, \dots, L-1 \quad \dots(2.10)$$

Langkah terakhir adalah melakukan pemetaan tingkat keabuan kembali dengan menggunakan rumus transformasi yang berikut.

$$f(k) = X_0 + (X_{L-1} - X_0) \cdot c(k) \quad \dots(2.11)$$

Dimana  $X_0$  dan  $X_{L-1}$  menyatakan tingkat keabuan terendah dan tertinggi secara berurutan. HE melakukan pemetaan ulang citra awal ke seluruh rentang nilai intensitas  $[X_0, X_{L-1}]$ .

## 2.4 Adaptive Histogram Equalization (AHE) on Image Gray Level Mapping

AHE on image gray level mapping tersebut memiliki ide yakni  $f_i$  adalah nilai keabuan dari tingkat keabuan ke- $i$  yang terdapat pada citra asli. Posisi  $j$  dari tingkat keabuan yang telah dipetakan  $g_j$  ditentukan dari rasio dari  $\frac{\sum_{k=0}^{i-1} p_k}{\sum_{k=i+1}^{m-1} p_k}$  dan  $\frac{\sum_{k=0}^{i-1} p_k}{\sum_{k=0}^{m-1} p_k}$ . Untuk mencapai distribusi seragam atau distribusi seragam lokal, algoritma melakukan perbandingan  $i$  dengan  $j$  : jika  $j < i$ , maka dilakukan pemetaan secara *ascending*. Jika  $j > i$ , maka dilakukan pemetaan secara *descending*.

$$j = (m-1) \frac{\sum_{k=0}^{i-1} p_k}{\sum_{k=0}^{i-1} p_k + \sum_{k=i+1}^{m-1} p_k} \quad \dots(2.12)$$

Dimana,  $\sum_{k=0}^{m-1} p_k = 1$ ,  $p_k = \frac{q_k}{Q}$ .

Pada proses pemetaan, tingkat keabuan dengan jumlah frekuensi kemunculan yang kecil akan tertutup oleh tingkat keabuan dengan jumlah frekuensi yang besar. Hal ini yang menyebabkan terjadinya kehilangan informasi. Untuk mencegahnya, maka di perkenalkan sebuah parameter adaptif  $\beta$  di dalam proses pemetaan keabuan. Untuk mendapatkan efek visual yang lebih baik, maka digunakan fungsi *entropy* sebagai fungsi objektif untuk memilih  $\beta$  secara adaptif



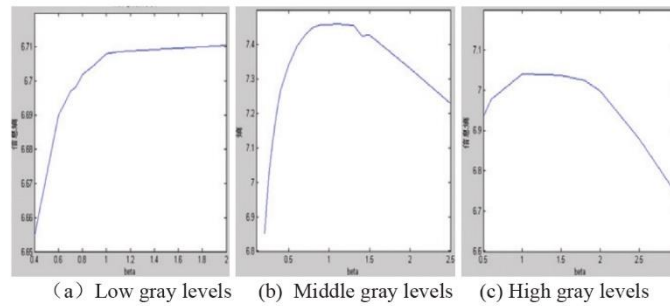
berdasarkan distribusi keabuan yang terdapat pada citra awal. Hubungan pemetaannya adalah

$$q_k = \log(q_k + 1) \quad \dots(2.13)$$

$$j = (m - 1) \frac{\sum_{k=0}^{i-1} p_k}{\sum_{k=0}^{i-1} p_k + \beta \sum_{k=i+1}^{m-1} p_k}, \quad \beta \in (0, +\infty) \quad \dots(2.14)$$

Penyeleksian parameter adaptif  $\beta$ :

Dari rumus (2.14), telah jelas bahwa  $j$  merupakan fungsi penurunan monoton dari  $\beta$ . Jika sebuah citra cenderung gelap, maka tingkat keabuan akan berkumpul secara berlebihan di sisi kiri dari histogram. Untuk mendapatkan efek visual yang lebih baik, maka  $j$  harus ditambahkan dan  $\beta$  harus lebih kecil dari 1. Pada gambar 2.10(a), nilai  $\beta$  yang tepat adalah 0,8. Jika tingkat kecerahan citra adalah sedang, maka tingkat keabuan berkumpul di bagian tengah dari histogram. Dari gambar 2.10(b), nilai  $\beta$  yang tepat adalah 1,1. Jika sebuah citra cenderung terang, maka tingkat keabuannya berkumpul di sisi kanan dari histogram, nilai  $j$  harus dikurangi dan  $\beta$  harus lebih besar dari 1. Dari gambar 2.10(c), nilai  $\beta$  yang tept adalah 1,5. Berdasarkan pernyataan sebelumnya, maka dapat disimpulkan bahwa nilai  $\beta$  berhubungan dengan tingkat keabuan yang terdapat pada citra awal. Relasi antara *entropy* dengan  $\beta$  di tunjukkan pada gambar 2.6 berikut.



**Gambar 2.10** Relasi antara *entropy* dengan  $\beta$

(Sumber : Zhu & Huang, 2012)

Definisi tingkat keabuan:

Misalkan sebuah citra mempunyai 256 tingkat keabuan. Itu dapat dibagi menjadi 3 jenis : tingkat keabuan rendah (*low gray levels*), tingkat keabuan menengah (*middle gray levels*), dan tingkat keabuan tinggi (*high gray levels*). Ditetapkan *threshold*  $TL=85$ ,  $TH = 170$ . Jika nilai keabuan berada dibawah 85, maka dapat diklasifikasikan sebagai tingkat keabuan rendah; jika nilai keabuan berada diantara 85 dan 170, maka dapat diklasifikasikan sebagai tingkat keabuan menengah; jika nilai keabuan berada diatas 170, maka dapat diklasifikasikan sebagai tingkat keabuan tinggi. Di saat yang bersamaan, jumlah *pixel* untuk tingkat keabuan rendah, menengah, dan tinggi dihitung masing-masing dan disimpan sebagai *num\_low*, *num\_mid*, *num\_high*. Nilai tertinggi dari ketiganya akan menentukan jenis citra. Jika *num\_low* merupakan yang terbesar, maka citra tersebut merupakan citra yang sangat gelap. (Zhu & Huang, 2012)

## **2.5 Adaptive Contrast Enhancement Algorithm with Details Preserving (ACEDP)**

ACEDP memperkenalkan teknik yang telah di modifikasi untuk melakukan peningkatan kontras sebuah citra sambil mempertahankan detail dari citra. ACEDP terdiri dari beberapa langkah yaitu:

1. Klasifikasikan jenis citra berdasarkan jumlah terbanyak dari nilai intensitasnya *pixel*.

Pertama, histogram dari citra awal dibentuk. Dua nilai *threshold* dibentuk, yang dinamakan *upper threshold* dan *lower threshold* dimana nilainya adalah 85 dan 170 secara berurutan. Kedua *threshold* tersebut akan membagi histogram menjadi 3 bagian yang sama besar. (Zhu & Huang, 2012). Citra diklasifikasikan menjadi citra dengan tingkat keabuan rendah,

menengah, dan tinggi berdasarkan jumlah terbanyak dari intensitas *pixel*.(Gambar 2. 11)

1:	IF <i>maximum_no_of_pixels_intensities</i> < 85
2:	THEN <i>image_type</i> =low gray level
3:	ELSE IF <i>maximum_no_of_pixels_intensities</i> >170
4:	THEN <i>image_type</i> =high gray level
5:	ELSE <i>image_type</i> =middle gray level

**Gambar 2.11** Klasifikasi Jenis Citra

(Sumber : Tang & Mat Isa, 2014)

## 2. Menentukan *Plateau Levels*

Pada ACEDP telah menetapkan beberapa fungsi untuk *histogram clipping* berdasarkan jenis citranya. Jika citra yang relatif gelap dengan jumlah terbanyak dari intensitas *pixel* lebih kecil dari 85, maka akan menggunakan rumus (2.15). Hal yang sama juga dilakukan untuk citra dengan tingkat keabuan menengah dan tinggi, maka fungsi *plateau level* yang digunakan adalah rumus (2.16) dan (2.17) secara berurutan.(Gambar 2.12). Konstanta *c1* dan *c2* yang digunakan memiliki rentang [-0.015,-0.005] dan [0.005,0.007] secara berurutan.

1:	IF <i>image_type</i> =low_gray_level	
2:	THEN $level = c_1 + \max(pdf)$	(2.15)
3:	IF <i>image_type</i> =middle_gray_level	
4:	THEN $level = \text{mean}(pdf)$	(2.16)
5:	IF <i>image_type</i> =high_gray_level	
6:	THEN $level = c_2 + \text{mean}(pdf)$	(2.17)

**Gambar 2.12** Determinasi dari *plateau level* berdasarkan jenis citra

(Sumber : Tang & Mat Isa, 2014)

## 3. *Histogram Clipping* dan *Equalization*

Dengan *plateau level* yang didapatkan pada langkah sebelumnya, maka *histogram clipping* dijalankan.

Tentukan terlebih dahulu histogram untuk intensitas  $k$ ,  $P(k)$  dinyatakan dengan rumus berikut:

$$P(k) = n_k, \text{ for } k = 0, 1, \dots, L-1 \quad \dots(2.18)$$

Dimana  $n_k$  adalah jumlah kemunculan intensitas  $k$  di dalam citra dan  $L$  adalah total tingkat keabuan yang terdapat pada citra. *Probability density function* (PDF) dari citra,  $r(k)$  dinyatakan dengan rumus :

$$r(k) = \frac{P(k)}{N}, \text{ for } k = 0, 1, \dots, L-1 \quad \dots(2.19)$$

Dimana  $N$  adalah jumlah *pixel* didalam citra. Penjumlahan dari seluruh  $r(k)$  sama dengan 1 yang terlihat pada rumus (2.20).

$$\sum_{i=0}^{L-1} r(i) = 1 \quad \dots(2.20)$$

*Cumulative density function* (CDF),  $c_k$  dinyatakan dengan rumus berikut:

$$c(k) = \sum_{i=0}^k r(i), \text{ for } k = 0, 1, \dots, L-1 \quad \dots(2.21)$$

Proses *clipping* pada histogram dilakukan dengan menggunakan rumus berikut.

$$P_{clip} = \begin{cases} P(k), & \text{for } P(k) \leq level(k) \\ level(k), & \text{for } P(k) > level(k) \end{cases} \quad \dots(2.22)$$

Setelah proses *clipping*, terapkan fungsi transformasi HE dengan beberapa perubahan seperti yang tertera pada rumus (2.17).

$$f(k) = X_0 + (X_{L-1} - X_0) \cdot \sum_{k=0}^{L-1} P_{clip}(k) \quad \dots(2.23)$$

Dimana  $X_0$  dan  $X_{L-1}$  merepresentasikan tingkat keabuan terendah dan tertinggi secara berurutan.

## 2.6 Adaptive Fuzzy Contrast Enhancement Algorithm with Details Preserving (AFCEDP)

AFCEDP mengintegrasikan teknik histogram clipping sebelum melakukan perataan histogram. Dalam penentuan fungsi *clipping limit*, digunakan element *fuzzy* untuk menentukan kategori kontras citra. Ini disebabkan perlakuan untuk tiap fungsi *plateau level* pada tiap kategori kontras citra berbeda-beda. Langkah-langkah yang terdapat pada metode ini adalah sebagai berikut

1. Penentuan fungsi keanggotaan untuk tiap intensitas.

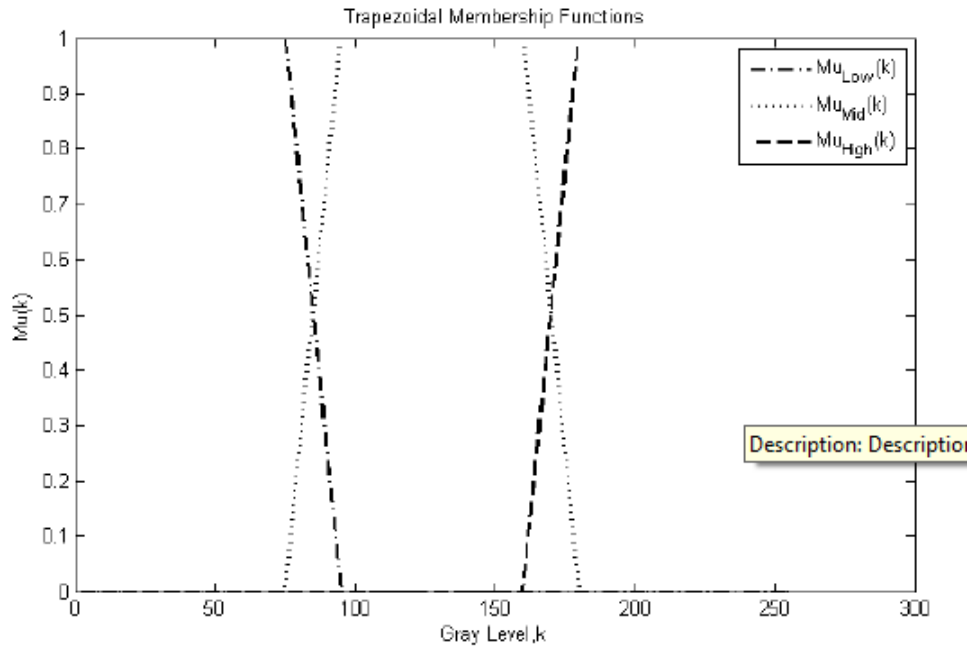
Fungsi keanggotaan yang digunakan pada metode ini adalah fungsi keanggotaan trapesium yang dikategorikan untuk 3 kategori kontras citra, yakni rendah, sedang, dan tinggi (*low*, *mid*, dan *high*) yang didefinisikan sebagai berikut.

$$\mu_{low}(k) = \begin{cases} 0 & , k > 95 \\ \frac{95-k}{20} & , 75 \leq k \leq 95 \\ 1 & , k < 75 \end{cases} \quad \dots (2.24)$$

$$\mu_{mid}(k) = \begin{cases} 0 & , (k < 75) \cup (k > 180) \\ \frac{k-75}{20} & , 75 \leq k \leq 95 \\ 1 & , 95 \leq k \leq 160 \\ \frac{180-k}{20} & , 160 \leq k \leq 180 \end{cases} \quad \dots (2.25)$$

$$\mu_{high}(k) = \begin{cases} 0 & , k < 160 \\ \frac{k-160}{20} & , 160 \leq k \leq 180 \\ 1 & , k > 180 \end{cases} \quad \dots (2.26)$$

dimana  $k$  merupakan intensitas pixel pada citra. Bentuk distribusi keanggotaan pada fungsi keanggotaan terdapat pada Gambar 2.13



**Gambar 2.13** Distribusi derajat keanggotaan berbentuk trapesium

(Sumber : Tang & Mat Isa, 2014)

2. Perhitungan derajat keanggotaan dan nilai intensitas referensi

Untuk mendapatkan nilai intensitas referensi dari fungsi keanggotaan trapesium, derajat dari citra yang termasuk diantara tiga kategori menggunakan partisi dihitung. Nilai intensitas referensi akan digunakan pada tahapan selanjutnya. Perhitungan intensitas referensi dilakukan sebagai berikut.

$$\lambda = (\text{low\_part} \times 43) + (\text{mid\_part} \times 128) + (\text{high\_part} \times 213) \dots (2.27)$$

3. Mendefinisikan 3 Fungsi *Plateau* dan melakukan komputasi *Clipping Limit*.

Teknik AFCEDP menggunakan fungsi *clipping* yang sama dengan teknik ACEDP. Sesuai yang dijelaskan di ACEDP, jarak yang diterima untuk *slopes c1* dan *c2* adalah  $[-0.015, -0.005]$  dan  $[0.005, 0.007]$  secara berurutan. Pada AFCEDP menggunakan nilai *c1* dan *c2* yang sama dengan ACEDP. Nilai untuk *c1* dan *c2* yang digunakan adalah -0.01 dan 0.007 secara berurutan. Perhitungan untuk fungsi *clipping*,  $\sigma(k)$ , sebagai berikut.

$$\sigma(k) = [\mu_{low}(\lambda) \times level_{low}(k)] + [\mu_{mid}(\lambda) \times level_{mid}(k)] + [\mu_{high}(\lambda) \times level_{high}(k)] \quad \dots(2.28)$$

Fungsi *plateau* yang diterapkan adalah sebagai berikut.

$$\begin{aligned} level_{low} &= c_1 + \max(pdf) \\ level_{mid} &= \text{mean}(pdf) \\ level_{high} &= c_2 + \text{mean}(pdf) \end{aligned}$$

**Gambar 2.14** Fungsi *plateau*

(Sumber : Tang & Mat Isa, 2014)

4. Lakukan *Clipping* dan Ekualisasi *histogram*.

Fungsi *clipping*  $\sigma(k)$  menyediakan pembatasan untuk tiap tingkat keabuan. Anggap bentuk citra masukan berupa citra *grayscale*, maka *histogram* dari citra,  $H(k)$  didefinisikan sebagai berikut.

$$H(k) = n_k, \text{ for } k = 0, 1, \dots, L-1 \quad \dots(2.29)$$

Dimana  $n_k$  adalah jumlah kemunculan dari intensitas  $k$  di citra dan  $L$  adalah total tingkat keabuan didalam citra. Fungsi probabilitas densitas dari citra didefinisikan sebagai berikut.

$$p(k) = \frac{H(k)}{N}, \text{ for } k = 0, 1, \dots, L-1 \quad \dots(2.30)$$

Dimana  $N$  adalah total piksel didalam citra.

Fungsi kumulatif densitas,  $c(k)$  didefinisikan sebagai berikut.

$$c(k) = \sum_{i=0}^k p(i), \text{ for } k = 0, 1, \dots, L-1 \quad \dots(2.31)$$

HE menggunakan fungsi transformasi untuk memetakan tingkat keabuan masukan menjadi tingkat keabuan yang baru, fungsi transformasi  $f(k)$  didefinisikan sebagai berikut.

$$f(k) = X_0 + (X_{L-1} - X_0) \cdot c(k) \quad \dots(2.32)$$

dimana  $X_0$  dan  $X_{L-1}$  merepresentasikan batas bawah dan batas atas dari histogram secara berurutan.

Fungsi transformasi baru  $new\_f(k)$ , menawarkan peningkatan ketajaman citra sesuai dengan rumus (2.33) berikut. Fungsi tersebut akan menggantikan fungsi transformasi sebelumnya. (Ooi, et al., 2009).

$$new\_f(k) = X_0 + (X_{L-1} - X_0) \cdot (c(k) - \frac{1}{2} p(k)) \quad \dots(2.33)$$

Beberapa fungsi baru lain yang diterapkan di algoritma tersebut adalah fungsi probabilitas densitas dan fungsi kumulatif densitas yang terlihat pada rumus (2.34) dan (2.35) secara berurutan.

$$new\_p(k) = \min(p(k), \sigma(k)), \text{ for } k = 0, L-1 \quad \dots(2.34)$$

$$c(k) = \sum_{i=0}^k new\_p(i), \text{ for } k = 0, 1, \dots, L-1 \quad \dots(2.35)$$

Lakukan pencarian nilai  $p(k)$  baru dengan rumus (2.34), setelah itu lakukan perhitungan  $c(k)$  yang baru dengan rumus (2.35).

Karena telah dilakukan *clipping* maka nilai kumulatif dari pdf tidak akan menjadi 1. Untuk mendapatkan nilai kumulatif 1, maka dilakukan rumus dibawah ini untuk mendapatkan nilai kumulatif = 1.

$$new\_p(k)' = new\_p(k) / \sum pdf \quad \dots(2.36)$$

Setelah mendapatkan nilai pdf baru maka lakukan kembali rumus (2.35) dengan menggantikan nilai  $new\_p(k)$  dengan nilai  $new\_p(k)$  yang didapat dari rumus (2.36) untuk mendapatkan nilai  $new\_c(k)$ , Langkah terakhir adalah lakukan fungsi transformasi dengan menggunakan fungsi transformasi baru pada rumus (2.35), dengan nilai  $new\_p(k)'$  yang baru hasil dari rumus (2.36).

## 2.7 Perbandingan Citra

Perbandingan citra digunakan untuk membandingkan kemiripan antar citra secara matematis. Metode yang digunakan untuk melakukan perbandingan citra adalah *Shannon Entropy*, *Standard Deviation*, dan *Contrast Improvement Evaluation*.



### 2.7.1 Shannon Entropy

*Shannon Entropy* merupakan rumus matematika yang secara luas digunakan untuk menghitung kekayaan informasi. Semakin tinggi nilai *Entropy* maka semakin tinggi pula detail dan informasi yang dimiliki oleh citra tersebut. Rumus *Shannon Entropy* dinyatakan pada rumus (2.37).

$$E = - \sum_{i=0}^N r(i) \log_2 r(i) \quad \dots(2.37)$$

Dimana  $r(i)$  merupakan probabilitas kemunculan nilai keabuan,  $N$  adalah nilai keabuan tertinggi.

### 2.7.2 Contrast Improvement Evaluation

Untuk mengetahui perbedaan atau peningkatan nilai kontras pada dua citra yang sama, maka rumus *Contrast Improvement Evaluation* sering dimanfaatkan untuk alat ukur peningkatan kontras pada dua citra yang sama. Rumus *Contrast Improvement Evaluation* dapat dilihat pada rumus (2.38).

$$C = 10 \log_{10} \left[ \frac{1}{WH} \sum_{u=1}^W \sum_{v=1}^H g^2(u,v) - \left| \frac{1}{WH} \sum_{u=1}^W \sum_{v=1}^H g(u,v) \right|^2 \right] \quad \dots(2.38)$$

Dimana  $W$  dan  $H$  adalah *width* dan *height* ( panjang dan tinggi ) dari citra,  $g(u,v)$  adalah intensitas dari piksel di posisi 2 dimensi  $(u,v)$ . Semakin besar nilai *Contrast Improvement Evaluation* untuk citra hasil, maka berarti semakin bagus juga peningkatan kontras yang terjadi. (Tang & Mat Isa, 2014).

## **BAB III**

### **METODOLOGI PENELITIAN**

#### **3.1 Analisis**

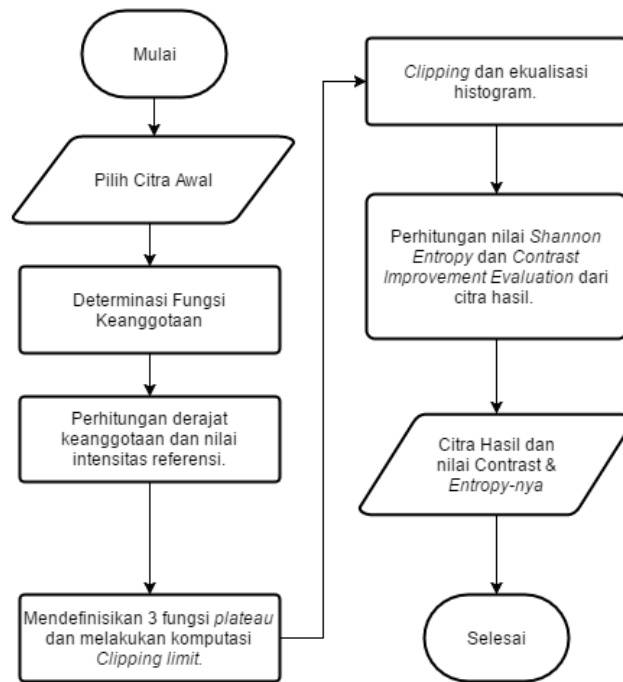
Pada Sub bab Analisis dibagi menjadi analisis proses dan analisis kebutuhan fungsional.

##### **3.1.1 Analisis Proses**

Pada aplikasi, proses awal yang akan dilakukan adalah memasukkan citra asli. Citra asli berupa citra *grayscale*. Citra tersebut akan diproses dengan algoritma *Adaptive Fuzzy Contrast Enhancement algorithms with Details Preserving*. Algoritma tersebut terdiri dari 4 tahap pengerjaan yakni:

1. Penentuan fungsi keanggotaan setiap entitas.
2. Perhitungan derajat keanggotaan dan nilai intensitas referensi.
3. Mendefinisikan 3 fungsi *plateau* dan melakukan komputasi *Clipping limit*.
4. Lakukan *Clipping* dan ekualisasi histogram.

Hasil citra dari algoritma AFCEDP akan dibandingkan dengan citra awal dengan menggunakan metode *Shannon Entropy* dan metode *Contrast Improvement Evaluation* hal ini untuk memperlihatkan peningkatan kontras dan pelestarian detail citra yang terjadi setelah dilakukan perbaikan pada citra awal. Sebagai tambahan citra juga akan diproses dengan algoritma *Adaptive Contrast Enhancement algorithms with Details Preserving* guna untuk melakukan perbandingan antara citra hasil dari algoritma AFCEDP dengan ACEDP.



**Gambar 3.1** *Flowchart* Proses Peningkatan kontras citra pada aplikasi menggunakan AFCEDP

#### 3.1.1.1 Citra Asli

Sebagai contoh, misalkan citra asli yang akan diproses berukuran 4 x 4 piksel seperti terlihat pada gambar 3.2 berikut.

97	100	103	79
108	86	97	103
144	135	121	145
169	85	153	50

**Gambar 3.2** Contoh Citra Asli

### 3.1.1.2 Analisis Determinasi Fungsi Keanggotaan dan Perhitungan Derajat Keanggotaan

Citra awal dari sebelumnya akan dicari derajat keanggotaannya untuk tiap pixel pada citra. Teknik yang digunakan untuk melakukan pembagian fungsi keanggotaan untuk tiap piksel pada citra adalah sebagai berikut.

$$\mu_{low}(k) = \begin{cases} 0 & , k > 95 \\ \frac{95-k}{20} & , 75 \leq k \leq 95 \\ 1 & , k < 75 \end{cases}$$

$$\mu_{mid}(k) = \begin{cases} 0 & , (k < 75) \cup (k > 180) \\ \frac{k-75}{20} & , 75 \leq k \leq 95 \\ 1 & , 95 \leq k \leq 160 \\ \frac{180-k}{20} & , 160 \leq k \leq 180 \end{cases}$$

$$\mu_{high}(k) = \begin{cases} 0 & , k < 160 \\ \frac{k-160}{20} & , 160 \leq k \leq 180 \\ 1 & , k > 180 \end{cases}$$

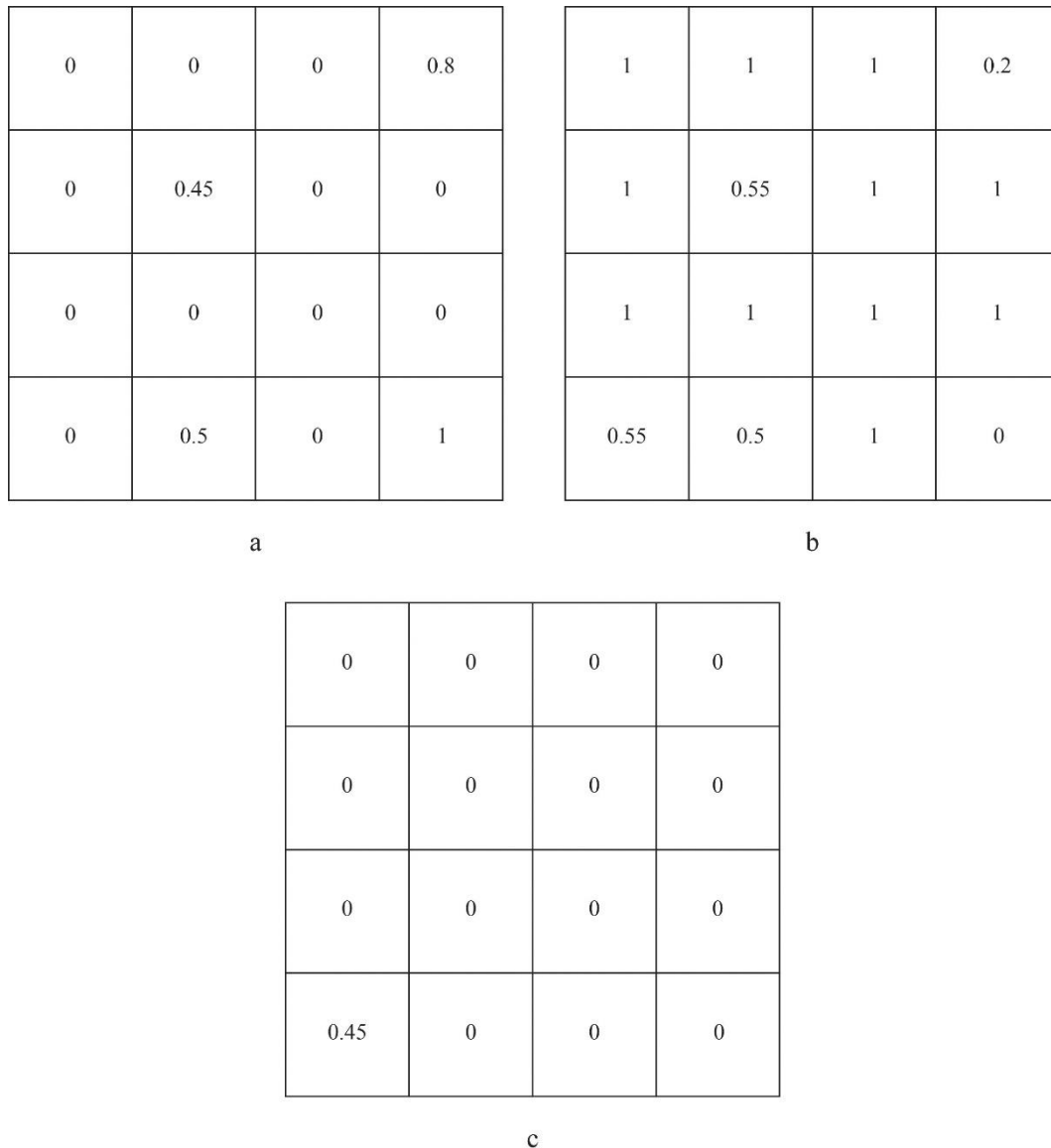
Hasil akhir dari proses tersebut akan membentuk 3 fungsi keanggotaan yakni  $\mu_{low}$ ,  $\mu_{mid}$ , dan  $\mu_{high}$ . Proses penggelempokkan sebagai berikut.

1. Piksel-1 (Gray= 97), maka  
 $\mu_{low}$  untuk piksel-1 adalah 0 (Gray > 95),  
 $\mu_{mid}$  untuk piksel-1 adalah 1 (95 ≤ Gray ≤ 160),  
 $\mu_{high}$  untuk piksel-1 adalah 0 (Gray < 160)
2. Piksel-2 (Gray = 100), maka  
 $\mu_{low}$  untuk piksel-2 adalah 0 (Gray > 95),  
 $\mu_{mid}$  untuk piksel-2 adalah 1 (95 ≤ Gray ≤ 160),  
 $\mu_{high}$  untuk piksel-2 adalah 0 (Gray < 160)
3. Piksel-3 (Gray = 103), maka  
 $\mu_{low}$  untuk piksel-3 adalah 0 (Gray > 95),  
 $\mu_{mid}$  untuk piksel-3 adalah 1 (95 ≤ Gray ≤ 160),

$\mu_{high}$  untuk piksel-3 adalah 0 ( $Gray < 160$ )

4. Perhitungan yang sama dilakukan hingga ke piksel terakhir atau piksel-16.

Hasil proses pengelompokkan piksel piksel *gray* tersebut dapat dilihat pada gambar 3.3 berikut ini.



**Gambar 3.3** Fungsi keanggotaan untuk  $\mu_{low}$  (a),  $\mu_{mid}$  (b),  $\mu_{high}$  (c)

### 3.1.1.3 Analisis Perhitungan Nilai Intensitas Referensi

Nilai intensitas referensi diperoleh dari rumus berikut.

$$\lambda = (\text{low\_part} \times 43) + (\text{mid\_part} \times 128) + (\text{high\_part} \times 213)$$

*Low part* diperoleh dari nilai rata-rata fungsi keanggotaan untuk  $\mu_{low}$ . Begitu juga halnya *mid part* dan *high part* yang didapat melalui nilai rata-rata fungsi keanggotaan untuk  $\mu_{mid}$  dan  $\mu_{high}$  secara berurutan. Untuk contoh diatas maka didapatkan nilai *low part*, *mid part*, *high part* sebagai berikut.

$$\begin{aligned} \text{Low part} &= \text{Total nilai pada fungsi keanggotaan } \mu_{low} / \text{Total piksel} \\ &= (0+0+0+0.8+0+0.45+0+0+0+0+0+0+0+0.5+0+1)/16 \\ &= 0.171875 \end{aligned}$$

$$\begin{aligned} \text{Mid part} &= \text{Total nilai pada fungsi keanggotaan } \mu_{mid} / \text{Total piksel} \\ &= (1+1+1+0.2+1+0.55+1+1+1+1+1+1+0.55+0.5+1+0)/16 \\ &= 0.8 \end{aligned}$$

$$\begin{aligned} \text{High part} &= \text{Total nilai pada fungsi keanggotaan } \mu_{mid} / \text{Total piksel} \\ &= (0+0+0+0+0+0+0+0+0+0+0+0+0.45+0+0+0)/16 \\ &= 0.028125 \end{aligned}$$

Dengan diperolehnya nilai *low part*, *mid part*, *high part*, maka nilai intensitas referensinya adalah:

$$\begin{aligned} \lambda &= (0.171875 \times 43) + (0.8 \times 128) + (0.028125 \times 213) \\ &= 115.78125 \text{ ( dibulatkan menjadi 115 )} \end{aligned}$$

### 3.1.1.4 Analisis Mendefinisikan Tiga Fungsi *Plateau* dan Melakukan Komputasi *Clipping Limit*

Tiga fungsi plateau yang di maksud adalah  $level_{low}$ ,  $level_{mid}$ ,  $level_{high}$ . Ketiga fungsi *plateau* tersebut akan digunakan untuk mencari nilai *clipping limit* yang nantinya akan digunakan saat perataan histogram. Pencarian ketiga fungsi *plateau* dapat menggunakan rumus berikut.

$$\begin{aligned} level_{low} &= c_1 + \max(pdf) \\ level_{mid} &= \text{mean}(pdf) \\ level_{high} &= c_2 + \text{mean}(pdf) \end{aligned}$$

Pencarian ketiga fungsi *plateau* dimulai dari pencarian *pdf* (*probability density function*) yang di dapat menggunakan rumus.

$$p(k) = \frac{H(k)}{N}, \text{ for } k = 0, 1, \dots, L-1$$

Dimana nilai  $H(k)$  berarti banyak kemunculan dari piksel dengan keabuan  $k$  dapat di lihat pada tabel berikut.

**Tabel 3.1** Tabel Hasil Perhitungan  $H(k)$

Nilai keabuan	50	79	85	86	97	100	103	108	121	135	144	145	153	169
$H(k)$	1	1	1	1	2	1	2	1	1	1	1	1	1	1

Maka, nilai *probability density function* dapat dilihat pada tabel dibawah ini.

**Tabel 3.2** Tabel Hasil Perhitungan  $p(k)$

Nilai Keabuan	$p(k)$	Nilai Keabuan	$p(k)$
50	0.0625	108	0.0625
79	0.0625	121	0.0625
85	0.0625	135	0.0625
86	0.0625	144	0.0625
97	0.125	145	0.0625
100	0.0625	153	0.0625
103	0.125	169	0.0625

Nilai  $c_1$  dan  $c_2$  yang digunakan pada algoritma tersebut berkisar antara  $[-0.015, -0.005]$  dan  $[0.005, 0.007]$  secara berurutan. Pada konteks ini nilai  $c_1$  dan  $c_2$  yang digunakan adalah  $-0.01$  dan  $0.007$  secara berurutan. Maka, nilai  $level_{low}$ ,  $level_{mid}$ ,  $level_{high}$  adalah.

$$\begin{aligned} level_{low} &= -0.01 + 0.125 \text{ (nilai tertinggi dari pdf)} \\ &= 0.115 \end{aligned}$$

$$\begin{aligned} level_{mid} &= 1 / 14 \text{ (nilai rata-rata dari pdf)} \\ &= 0.0714 \end{aligned}$$

$$\begin{aligned} level_{high} &= 0.007 + 0.0714 \\ &= 0.0784 \end{aligned}$$

Dengan diperolehnya nilai  $level_{low}$ ,  $level_{mid}$ ,  $level_{high}$  maka nilai *clipping limit* nya adalah

$$\sigma(k) = [\mu_{low}(\lambda) \times level_{low}(k)] + [\mu_{mid}(\lambda) \times level_{mid}(k)] + [\mu_{high}(\lambda) \times level_{high}(k)]$$

$\mu_{low}$ ,  $\mu_{mid}$ ,  $\mu_{high}$  untuk piksel dengan nilai keabuan 115 ( $\lambda$ ) didapatkan dengan menggunakan cara yang serupa dengan yang sebelumnya yakni :

$\mu_{low}$  untuk keabuan 115 adalah 0 ( $Gray > 95$ ),

$\mu_{mid}$  untuk keabuan 115 adalah 1 ( $95 \leq Gray \leq 160$ ),

$\mu_{high}$  untuk keabuan 115 adalah 0 ( $Gray < 160$ )

maka hasil akhir untuk *clipping limit* adalah sebagai berikut:

$$\begin{aligned} \sigma &= [0 \times 0.115] + [1 \times 0.0714] + [0 \times 0.0784] \\ &= 0.0714 \end{aligned}$$

### 3.1.1.5 Analisis *Clipping* dan Ekualisasi Histogram

Pada tahap ini, akan dilakukan ekualisasi pada histogram dengan menggunakan rumus yang sudah ada. Untuk nilai  $H(k)$  dan  $pdf$  sudah di dapatkan sebelumnya pada tabel 3.1 dan tabel 3.2 secara berurutan. Langkah selanjutnya yakni pencarian nilai  $pdf$  baru dengan menggunakan rumus berikut.

$$new\_p(k) = \min(p(k), \sigma(k)), \text{ for } k = 0, L-1$$



1. Keabuan = 50

$$new\_p(50) = \text{MIN} ( 0.0625 , 0.0714 ) = 0.0625$$

2. Keabuan = 79

$$new\_p(79) = \text{MIN} ( 0.0625 , 0.0714 ) = 0.0625$$

Perhitungan yang sama dilakukan hingga nilai keabuan terakhir pada citra.

Hasil dari perhitungan tersebut dapat dilihat pada tabel berikut.

**Tabel 3.3** Tabel Hasil Perhitungan  $new\_p(k)$

Nilai Keabuan	$new\_p(k)$	Nilai Keabuan	$new\_p(k)$
50	0.0625	108	0.0625
79	0.0625	121	0.0625
85	0.0625	135	0.0625
86	0.0625	144	0.0625
97	0.0714	145	0.0625
100	0.0625	153	0.0625
103	0.0714	169	0.0625

Perhitungan dilanjutkan dengan mencari nilai  $c(k)$  dengan menggunakan rumus :

$$c(k) = \sum_{i=0}^k new\_p(i), \text{ for } k = 0, 1, \dots, L-1$$

Hasil Perhitungan untuk mendapatkan nilai  $c(k)$  dapat dilihat pada tabel berikut.

**Tabel 3.4** Tabel Hasil Perhitungan  $c(k)$

Nilai Keabuan	$c(k)$	Nilai Keabuan	$c(k)$
50	0.0625	108	0.5178
79	0.125	121	0.5803
85	0.1875	135	0.6428
86	0.25	144	0.7053
97	0.3214	145	0.7678
100	0.3839	153	0.8303
103	0.4553	169	0.8928

Untuk mendapatkan nilai kumulatif dari  $new\_p(k) = 1$ , maka dilakukan rumus berikut.

$$new\_p(k)' = new\_p(k) / \sum pdf$$

1. Nilai keabuan = 50

$$new\_p(k)' = 0.0625 / 0.8928 = 0.07$$

2. Nilai keabuan = 79

$$new\_p(k)' = 0.0625 / 0.8928 = 0.07$$

Perhitungan yang sama dilakukan kembali hingga ke nilai keabuan terakhir pada citra. Hasil perhitungan  $new\_p(k)'$  dapat dilihat pada tabel berikut.

**Tabel 3.5** Tabel Hasil Perhitungan  $new\_p(k)'$

Nilai Keabuan	$new\_p(k)'$	Nilai Keabuan	$new\_p(k)'$
50	0.07	108	0.07
79	0.07	121	0.07
85	0.07	135	0.07
86	0.07	144	0.07
97	0.08	145	0.07
100	0.07	153	0.07
103	0.08	169	0.07

Perhitungan dilanjutkan dengan mencari nilai  $c(k)$  baru dengan menggunakan rumus  $c(k)$  sebelumnya, dengan catatan nilai  $new\_p(k)$  diganti dengan nilai  $new\_p(k)'$ . Hasil perhitungan  $c(k)$  dapat dilihat pada tabel berikut.

**Tabel 3.6** Tabel Hasil Perhitungan  $c(k)$

Nilai Keabuan	$c(k)$	Nilai Keabuan	$c(k)$
50	0.07	108	0.58
79	0.14	121	0.65
85	0.21	135	0.72
86	0.28	144	0.79
97	0.36	145	0.86
100	0.43	153	0.93
103	0.51	169	1.00

Dilakukan perataan histogram dengan menggunakan fungsi transformasi berikut guna untuk mendapatkan nilai intensitas baru.

$$new\_f(k) = X_0 + (X_{L-1} - X_0) \cdot (c(k) - \frac{1}{2} p(k))$$

$X_0$  dan  $X_{L-1}$  merujuk pada batas bawah dan batas atas pada histogram. Nilai  $X_0$  dan  $X_{L-1}$  yang digunakan pada contoh tersebut adalah 0 dan 255.

1. Nilai keabuan = 50

$$\begin{aligned} new\_f(k) &= 0 + (255 - 0) \cdot (0.07 - (0.5 \cdot 0.07)) \\ &= 8.925 \text{ (dibulatkan menjadi 8)} \end{aligned}$$

2. Nilai keabuan = 79

$$\begin{aligned} new\_f(k) &= 0 + (255 - 0) \cdot (0.14 - (0.5 \cdot 0.07)) \\ &= 26.775 \text{ (dibulatkan menjadi 26)} \end{aligned}$$

3. Perhitungan yang sama dilakukan hingga nilai keabuan terakhir.

Hasil Perataan histogram dapat dilihat pada tabel berikut.

**Tabel 3.7** Tabel Hasil Perhitungan Nilai keabuan baru.

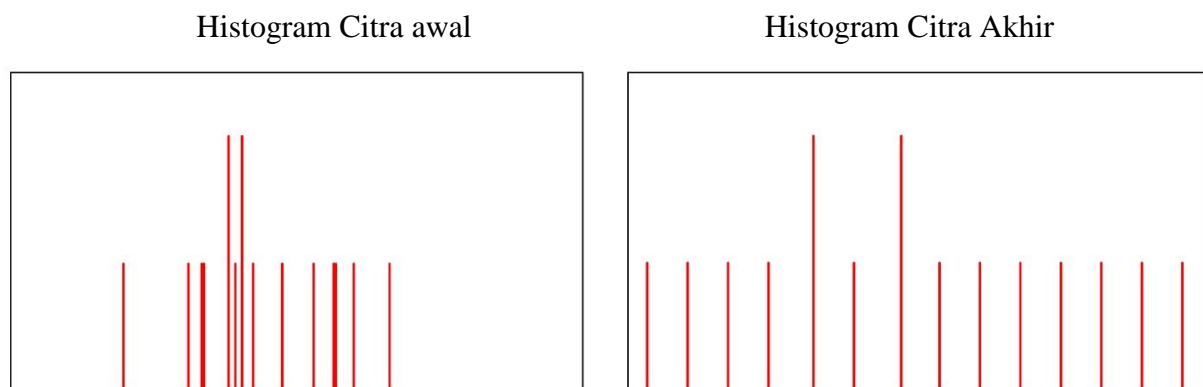
Nilai Keabuan	Nilai keabuan baru	Nilai Keabuan	Nilai keabuan baru
50	8	108	138
79	26	121	156
85	44	135	174
86	62	144	192
97	82	145	210
100	100	153	228
103	121	169	246

Nilai keabuan baru yang didapat tersebut akan menggantikan nilai keabuan yang sebelumnya sehingga membentuk sebuah citra baru yang telah di ekualisasi histogram. Gambar citra akhir dapat dilihat pada gambar 3.4.

82	100	121	26
138	62	82	121
192	174	156	210
246	44	228	8

**Gambar 3.4** Citra Akhir setelah dilakukan ekualisasi histogram

Perbandingan histogram citra awal (gambar 3.3) dengan histogram hasil ekualisasi dapat dilihat pada gambar 3.5 berikut.



**Gambar 3.5** Histogram Citra Awal dan Citra Akhir

#### 3.1.1.6 Analisis Perbandingan Citra Hasil dengan Input.

Perbandingan citra hasil dengan citra awal terdiri dari metode *Shannon Entropy* dan metode *Contrast Improvement Evaluation*.

A. Metode *Shannon Entropy*

Sebagai contoh, akan dihitung persentase informasi yang dilestarikan dengan cara membandingkan hasil *shannon entropy* dari citra awal dengan citra hasil.(gambar 3.2 dan gambar 3.4) seperti terlihat pada gambar 3.6 berikut.

Citra Awal				Citra Akhir			
97	100	103	79	82	100	121	26
108	86	97	103	138	62	82	121
144	135	121	145	192	174	156	210
169	85	153	50	246	44	228	8

**Gambar 3.6** Citra Awal (gambar 3.3) dan Citra Akhir (gambar 3.5)

Rumus untuk menghitung *shannon entropy* adalah sebagai berikut.

$$E = - \sum_{i=0}^N r(i) \log_2 r(i)$$

Dengan catatan,  $r(i)$  merepresentasikan nilai probabilitas kemunculan nilai keabuan  $i$  dan  $N$  adalah nilai keabuan tertinggi. Berikut perhitungan *shannon entropy* untuk citra awal:

1. Daftar probabilitas kemunculan nilai keabuan pada gambar 3.6 (Citra awal) dapat dilihat pada tabel berikut.

**Tabel 3.8** Probabilitas kemunculan nilai keabuan ke- $i$

Nilai Keabuan	Probabilitas kemunculan	Nilai keabuan	Probabilitas kemunculan
50	0.0625	108	0.0625
79	0.0625	121	0.0625
85	0.0625	135	0.0625
86	0.0625	144	0.0625
97	0.125	145	0.0625
100	0.0625	153	0.0625
103	0.125	169	0.0625

2. Lakukan perhitunga nilai  $E$ .

$i = 50$ , maka

$$\begin{aligned} E(i) &= - ( 0.0625 * \log_2 (0.0625)) \\ &= - ( 0.0625 * -4 ) \\ &= - ( - 0.25 ) = 0.25 \end{aligned}$$

$i = 79$ , maka

$$\begin{aligned} E(i) &= - ( 0.0625 * \log_2 (0.0625)) \\ &= - ( 0.0625 * -4 ) \\ &= - ( -0.25 ) = 0.25 \end{aligned}$$

Perhitungan yang sama dilakukan hingga nilai keabuan terakhir. Hasil perhitungan dapat dilihat pada tabel berikut.

**Tabel 3.9** Hasil Perhitungan E

Nilai Keabuan	$E(i)$	Nilai keabuan	$E(i)$
50	0.25	108	0.25
79	0.25	121	0.25
85	0.25	135	0.25
86	0.25	144	0.25
97	0.375	145	0.25
100	0.25	153	0.25
103	0.375	169	0.25

Setelah didapatkan nilai  $E(i)$ , maka Nilai  $E$  adalah :

$$E = E(50) + E(79) + E(85) + E(86) + E(97) + E(100) + E(103) + E(108) \\ + E(121) + E(135) + E(144) + E(145) + E(153) + E(169)$$

$$E = 0.25 + 0.25 + 0.25 + 0.25 + 0.375 + 0.25 + 0.375 + 0.25 + 0.25 + 0.25 \\ + 0.25 + 0.25 + 0.25 + 0.25$$

$$E = 3.75$$

Dengan demikian hasil *Shannon Entropy* untuk citra awal adalah 3.75. Perhitungan yang serupa dilakukan kepada citra hasil dan didapatkan nilai *Shannon Entropy* sebesar 3.75. Untuk mendapatkan persentase informasi citra yang dilestarikan maka akan digunakan rumus (  $Entropy\ hasil / Entropy\ awal * 100\%$  ). Persentase pelestarian informasi untuk contoh tersebut adalah  $(3.75/3.75 * 100 \%) = 100\%$ .

#### B. Metode *Contrast Improvement Evaluation*

Sebagai contoh, akan digunakan citra awal dan citra hasil (gambar 3.2 dengan gambar 3.4) yang dapat dilihat pada gambar 3.7 berikut.

Citra Awal				Citra Akhir			
97	100	103	79	82	100	121	26
108	86	97	103	138	62	82	121
144	135	121	145	192	174	156	210
169	85	153	50	246	44	228	8

**Gambar 3.7** Citra Awal dan Citra Akhir

Rumus untuk menghitung *Contrast Improvement Evaluation* (CIE) adalah sebagai berikut.

$$C = 10 \log_{10} \left[ \frac{1}{WH} \sum_{u=1}^W \sum_{v=1}^H g^2(u,v) - \left| \frac{1}{WH} \sum_{u=1}^W \sum_{v=1}^H g(u,v) \right|^2 \right]$$

W adalah lebar dari citra dan H adalah tinggi dari citra yang ingin dicari nilai C,  $g(u,v)$  adalah intensitas citra pada piksel dengan posisi  $(u,v)$ . Berikut perhitungan CIE yang dilakukan pada citra awal (gambar 3.7).

1. Hitung nilai  $\sum_{u=1}^W \sum_{v=1}^H g^2(u,v)$ .

$u = 1, v = 1$ . Piksel pada posisi ke  $(1,1) = 97$ , maka nilai  $g^2(u,v)$  adalah  $97^2 = 9409$ . Perhitungan yang sama dilakukan hingga posisi piksel terakhir dari citra. Hasil perhitungan dapat dilihat pada tabel 3.10.

**Tabel 3.10** Hasil perhitungan  $g^2(u,v)$

Nilai Keabuan	$g^2(u,v)$	Nilai Keabuan	$g^2(u,v)$
97	9409	144	20736



100	10000	135	18225
103	10609	121	14641
79	6241	145	21025
108	11664	169	28561
86	7396	85	7225
97	9409	153	23409
103	10609	50	2500

Maka nilai  $\sum_{u=1}^W \sum_{v=1}^H g^2(u,v)$  adalah 211659.

2. Hitung nilai  $\sum_{u=1}^W \sum_{v=1}^H g(u,v)$

$u = 1, v = 1$ . Piksel pada posisi ke  $(1,1) = 97$ , maka nilai  $g(u,v)$  adalah 97.

Sehingga nilai  $\sum_{u=1}^W \sum_{v=1}^H g(u,v)$  adalah 1775.

3. Hitung nilai  $\frac{1}{WH} \sum_{u=1}^W \sum_{v=1}^H g^2(u,v)$

$W = 4$  dan  $H = 4$ , maka nilai  $WH = 4*4 = 16$ . Dan nilai dari  $\frac{1}{WH} \sum_{u=1}^W \sum_{v=1}^H g^2(u,v)$  adalah  $211659 / 16 = 13228,6875$ .

4. Hitung nilai  $\left| \frac{1}{WH} \sum_{u=1}^W \sum_{v=1}^H g(u,v) \right|^2$

$W = 4$  dan  $H = 4$ , maka nilai  $WH = 4*4 = 16$ . Dan nilai dari  $\frac{1}{WH} \sum_{u=1}^W \sum_{v=1}^H g(u,v)$  adalah  $1775 / 16 = 110,9375$ . Nilai untuk  $\left| \frac{1}{WH} \sum_{u=1}^W \sum_{v=1}^H g(u,v) \right|^2$  adalah  $(110,9375)^2 = 12307,12890625$ .

5. Hitung nilai  $C$

Dengan didapatnya nilai  $\frac{1}{WH} \sum_{u=1}^W \sum_{v=1}^H g^2(u,v)$  dan  $\left| \frac{1}{WH} \sum_{u=1}^W \sum_{v=1}^H g(u,v) \right|^2$  maka nilai  $C$  dapat dicari dengan cara berikut.

$$C = 10 \log_{10} \left[ \frac{1}{WH} \sum_{u=1}^W \sum_{v=1}^H g^2(u,v) - \left| \frac{1}{WH} \sum_{u=1}^W \sum_{v=1}^H g(u,v) \right|^2 \right]$$

$$\begin{aligned} C &= 10 \log_{10} (13228,6875 - 12307,12890625) \\ &= 10 \log_{10} (921.55859375) \end{aligned}$$

$$= 10 * 2,9645229533731166$$

$$= 29,645229533731166$$

Perhitungan yang sama juga dilakukan terhadap citra hasil (gambar 3.8).

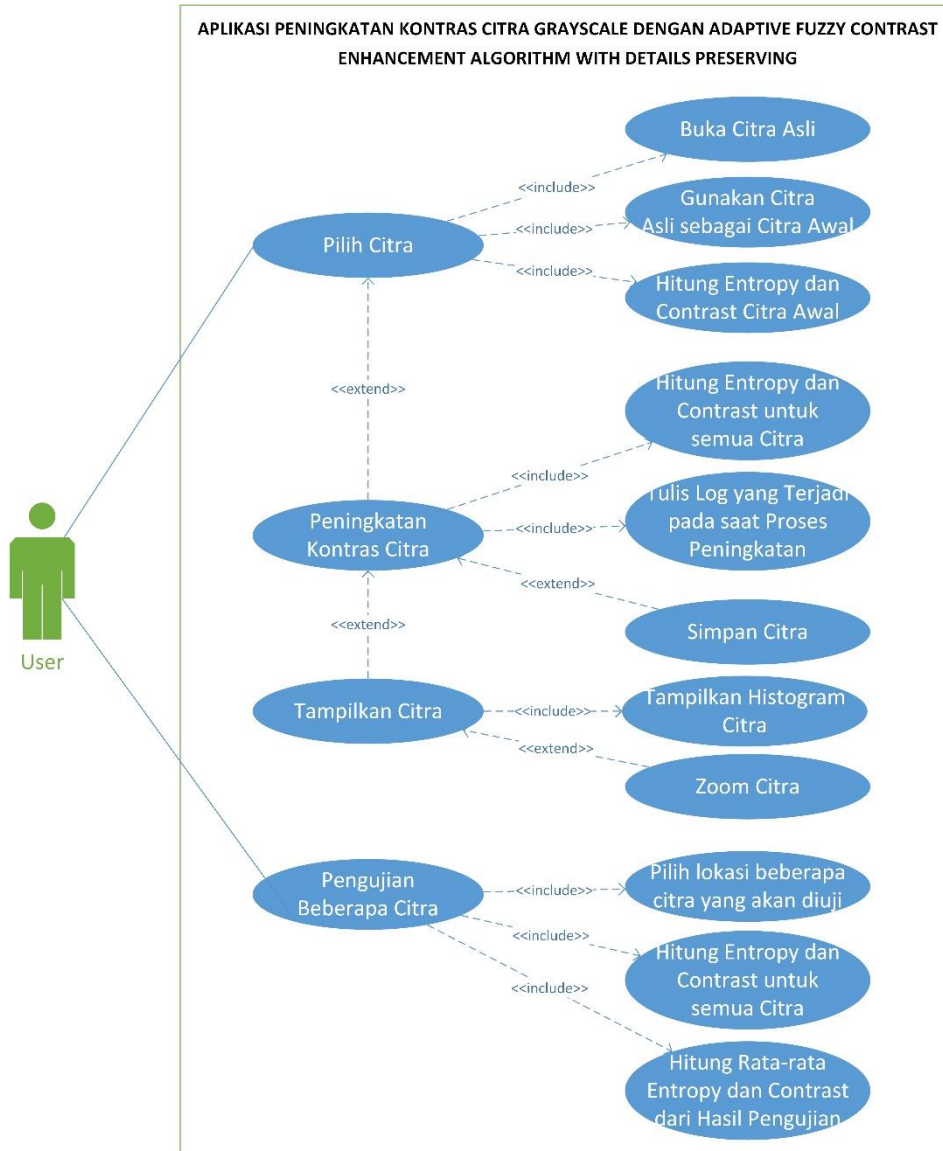
Hasil perhitungan terhadap citra hasil adalah 36,92757441432866.

### 3.1.2 Analisis Kebutuhan Fungsional

Analisis kebutuhan sistem merupakan proses identifikasi dan evaluasi permasalahan yang terdapat di dalam suatu sistem, sehingga sistem yang dibangun sesuai dengan kriteria yang diharapkan. Kebutuhan fungsional adalah kebutuhan yang berisi proses-proses apa saja yang nantinya harus dapat dilakukan oleh sistem. Aplikasi peningkatan kontras citra *grayscale* harus memenuhi fungsi sebagai berikut.

1. Aplikasi harus dapat menerima input citra berupa grayscale yang akan digunakan sebagai citra awal.
2. Aplikasi harus dapat meningkatkan kontras citra dengan menggunakan algoritma *Adaptive Fuzzy Contrast Enhancement algorithm with Details Preserving* dan algoritma *Adaptive Contrast Enhancement algorithm with Details Preserving*.
3. Aplikasi harus dapat menyimpan citra hasil proses.
4. Aplikasi harus dapat melakukan proses pengujian dengan menghitung nilai *Entropy* dan *Contrast Improvement Evaluation*.
5. Aplikasi harus dapat melakukan proses perbandingan hasil citra yang menggunakan algoritma *Adaptive Fuzzy Contrast Enhancement algorithm with Details Preserving* dengan algoritma *Adaptive Contrast Enhancement algorithm with Details Preserving*.
6. Aplikasi harus dapat mengubah nilai parameter batas atas dan batas bawah HE serta nilai *c1* dan *c2* yang akan digunakan.
7. Aplikasi harus dapat menampilkan histogram dari masing masing citra hasil pemrosesan, sehingga *user* dapat melihat histogram citra, serta memperbesar dan memperkecil citra hasil proses.

Untuk memenuhi kebutuhan fungsional, sistem dimodelkan dengan menggunakan *Use Case* seperti terlihat pada gambar 3.8



**Gambar 3.8** Diagram *Use Case* dari Aplikasi

Pada gambar 3.8, *use case diagram* menunjukkan interaksi antara penngguna dan sistem di dalam diagram *use case*. Hubungan *include* menggambarkan bahwa suatu *use case* seluruhnya meliputi fungsionalitas dari *use case* lainnya. Hubungan *extend* antar *use case* berarti bahwa suatu *use case* merupakan tambahan fungsionalitas dari *use case* yang lain. Berikut merupakan tabel deskripsi tiap-tiap proses yang terdapat pada *use case diagram*.

**Tabel 3.11** Deskripsi proses “Pilih Citra”

Use Case	Pilih Citra	
Actor	User	
Description	User membuka menu untuk memilih citra	
Pre-condition	User telah membuka aplikasi dan ingin melakukan peningkatan kontras citra	
Post-condition	User telah membuka kotak dialog Open	
	User	System
	1. Memilih menu File	2. Drop-down menu
	3. Memilih sub-menu Open	4. Buka dialog Open

**Tabel 3.12** Deskripsi Proses “Buka Citra Asli”

Use Case	Buka Citra Asli	
Actor	User	
Description	User memilih citra yang akan ditingkatkan	
Pre-condition	User telah membuka kotak dialog Open	
Post-condition	User telah membuka citra yang akan ditingkatkan kontrasnya	
	User	System
	1. Memilih citra pada kotak dialog Open	
	2. Memilih tombol OK	3. Baca filename citra

**Tabel 3.13** Deskripsi Proses “Gunakan Citra Asli sebagai Citra Awal”

Use Case	Gunakan Citra Asli sebagai Citra Awal	
Actor	System	
Description	System menjadikan citra asli pilihan user menjadi citra awal pada aplikasi	
Pre-condition	User telah membuka citra yang akan ditingkatkan kontrasnya	
Post-condition	System telah menampilkan citra pada aplikasi	
	User	System
		1. Rekam bitmap dari lokasi filename
		2. Tampilkan citra yang disematkan pada kotak gambar

**Tabel 3.14** Deskripsi Proses “Hitung *Entropy* dan *Contrast* Citra Awal”

Use Case	Hitung Entropy dan Contrast Citra Awal	
Actor	System	
Description	System menghitung nilai Entropy dan Contrast dari citra awal	
Pre-condition	System telah menampilkan citra pada aplikasi	
Post-condition	System telah menampilkan nilai Entropy dan Contrast citra awal	
	User	System
		1. Hitung histogram dari citra awal
		2. Hitung probabilitas kemunculan dari histogram yang telah didapatkan
		3. Hitung nilai Entropy dan Contrast
		4. Tampilkan tiap nilai pada lokasi yang disediakan di aplikasi

**Tabel 3.15** Deskripsi Proses “Peningkatan Kontras Citra”

Use Case	Peningkatan Kontras Citra	
Actor	User	
Description	User meningkatkan kontras dari citra awal dan nilai parameter pilihan user	
Pre-condition	User telah membuka citra yang akan ditingkatkan kontrasnya	
Post-condition	User telah meningkatkan kontras dan system menunjukkan hasil citra	
	User	System
	1. Menentukan nilai parameter yang digunakan	
	2. Tekan tombol Proses	3. Tentukan nilai clipping limit AFCEDP untuk bitmap dari citra awal
		4. Lakukan perataan histogram AFCEDP
		5. Tentukan nilai clipping limit ACEDP untuk bitmap dari citra awal

	6. Lakukan perataan histogram ACEDP
	7. Rekam tiap proses pada log
	8. Tampilkan dua hasil citra pada tiap kotak gambar

**Tabel 3.16** Deskripsi Proses “Hitung *Entropy* dan *Contrast* semua Citra”

Use Case	Hitung Entropy dan Contrast Semua Citra	
Actor	System	
Description	System menghitung nilai Entropy dan Contrast untuk dua citra yang dihasilkan	
Pre-condition	User telah meningkatkan kontras dan system menunjukkan hasil citra	
Post-condition	System telah menampilkan nilai Entropy dan Contrast untuk hasil citra	
	User	System
		1. Hitung nilai histogram untuk hasil AFCEDP dan ACEDP
		2. Menghitung nilai probabilitas kemunculan dari nilai histogram AFCEDP dan ACEDP
		3. Hitung nilai Entropy dan Contrast
		4. Tampilkan tiap nilai pada lokasi yang disediakan di aplikasi

**Tabel 3.17** Deskripsi Proses “Tulis log yang terjadi saat proses peningkatan”

Use Case	Tulis Log yang Terjadi pada saat Proses Peningkatan	
Actor	System	
Description	System menulis log untuk pada lokasi textbox yang disediakan	
Pre-condition	System telah menampilkan nilai Entropy dan Contrast untuk hasil citra	
Post-condition	System telah menampilkan log pada aplikasi	
	User	System
		1. Tampilkan tiap proses yang telah direkam pada textbox di aplikasi

**Tabel 3.18** Deskripsi Proses “Simpan”

Use Case	Simpan	
Actor	User	
Description	User menyimpan citra hasil peningkatan kontras dengan AFCEDP	
Pre-condition	User telah meningkatkan kontras dan system menunjukkan hasil citra	
Post-condition	System telah menyimpan citra dengan filename yang diinginkan user	
	User	System
	1. Memilih menu File	2. Drop-down menu
	3. Memilih sub-menu Save As	4. Buka dialog Save As
	5. Memilih lokasi, nama file dan ekstensi untuk disimpan	
	6. Memilih tombol OK	7. Ekspor bitmap citra hasil AFCEDP ke bentuk file yang diinginkan user

**Tabel 3.19** Deskripsi Proses “Tampilkan Citra”

Use Case	Tampilkan Citra	
Actor	User	
Description	User membuka citra dalam ukuran besar	
Pre-condition	User telah meningkatkan kontras dan system menunjukkan hasil citra	
Post-condition	System telah menampilkan form Tampil Citra dan citra pada kotak gambar yang disediakan	
	User	System
	1. Click pada salah satu kotak gambar yang tersedia	2. Membuka form Tampil Citra baru pada aplikasi
		3. Menampilkan citra pada kotak gambar yang disediakan

**Tabel 3.20** Deskripsi Proses “Tampilkan Histogram Citra”

Use Case	Tampilkan Histogram Citra	
Actor	System	
Description	System menampilkan gambar histogram dari citra	
Pre-condition	System telah menampilkan citra pada kotak gambar yang disediakan	
Post-condition	System telah menampilkan histogram citra pada form Tampil Citra	
	User	System

		1. Pindahkan nilai histogram citra yang telah direkam ke form Tampil Citra
		2. Bentuk histogram dari nilai histogram
		3. Tampilkan nilai lainnya yang berkaitan dengan citra

**Tabel 3.21** Deskripsi Proses “Zoom Citra”

Use Case	Zoom Citra	
Actor	User	
Description	User mengatur besar tampilan citra pada kotak gambar sesuai keinginan	
Pre-condition	System telah menampilkan form Tampil Citra dan citra pada kotak gambar yang disediakan	
Post-condition	System telah menampilkan citra dengan besar tampilan yang diinginkan user pada kotak gambar yang disediakan	
	User	System
	1. Memilih pilihan zoom yang ingin digunakan	2. Mengatur citra sesuai ratio dari zoom yang diinginkan user

**Tabel 3.22** Deskripsi Proses “Pengujian Beberapa Citra”

Use Case	Pengujian Beberapa Citra	
Actor	User	
Description	User menguji beberapa citra secara sekaligus	
Pre-condition	User telah membuka aplikasi dan ingin melakukan pengujian beberapa citra	
Post-condition	System telah menampilkan form Pengujian	
	User	System
	1. Memilih menu Pengujian	2. Tampilkan form Pengujian

**Tabel 3.23** Deskripsi Proses “Pilih lokasi beberapa citra yang akan diuji”

Use Case	Pilih lokasi beberapa citra yang akan diuji	
Actor	User	
Description	User memilih lokasi dari beberapa citra yang akan diuji sekaligus	
Pre-condition	System telah menampilkan form Pengujian dan User telah mengisi parameter	
Post-condition	User telah memilih lokasi beberapa citra yang ingin diuji	
	User	System



1. Mengatur parameter yang diinginkan	2. Tampilkan form Pengujian
3. Click tombol Browse	4. Tampilkan dialog Browse
5. Pilih lokasi dari beberapa citra yang ingin diuji	
6. Memilih tombol OK	7. Rekam lokasi dan seluruh file citra didalamnya

**Tabel 3.24** Deskripsi Proses “Hitung *Entropy* dan *Contrast* untuk semua citra”

Use Case	Hitung Entropy dan Contrast untuk semua Citra	
Actor	System	
Description	System menghitung nilai Entropy dan Contrast dari lokasi citra yang direkam	
Pre-condition	User telah memilih lokasi beberapa citra yang ingin diuji	
Post-condition	System telah menunjukkan hasil perhitungan pada tabel di aplikasi	
	User	System
		1. Menghitung nilai histogram untuk tiap citra
		2. Menghitung probabilitas kemunculan tiap pixel untuk tiap citra
		3. Menghitung hasil untuk AFCEDP dan ACEDP dari range konstanta c1 dan c2
		4. Menghitung nilai Entropy dan Contrast dari semua hasil citra
		5. Tampilkan tiap hasil pada tabel

**Tabel 3.25** Deskripsi Proses “Hitung rata-rata *Entropy* dan *Contrast* dari Hasil Pengujian”

Use Case	Hitung Rata-rata Entropy dan Contrast dari Hasil Pengujian	
Actor	System	
Description	System menghitung rata-rata nilai Entropy dan Contrast dari seluruh citra awal, hasil AFCEDP, dan ACEDP	
Pre-condition	System telah menunjukkan hasil perhitungan pada tabel di aplikasi	
Post-condition	System telah menampilkan rata-rata Entropy dan Contrast untuk hasil pengujian	

	User	System
		1. Menghitung rata-rata untuk seluruh hasil pengujian
		2. Menampilkan hasil rata-rata pada aplikasi

### 3.2 Perancangan

Pengembangan aplikasi dilakukan dengan menggunakan bahasa pemrograman Microsoft C# .Net 2013. Aplikasi memiliki 4 buah tampilan, yaitu *form* Utama, *form* Pengujian, *form* Zoom, *form* About Us.

#### 3.2.1 Form Utama

*Form* Utama merupakan tampilan utama dari aplikasi. Pada *form* ini, ditampilkan citra awal, citra hasil peningkatan kontras citra dengan menggunakan algoritma AFCEDP dan ACEDP. *Form* ini juga memiliki beberapa menu yang berisi fungsi dari aplikasi. Rancangan tampilan dari *form* Utama dapat dilihat pada gambar berikut.

**Gambar 3.9** Rancangan Tampilan *Form* Utama

Keterangan :

- 1 : Menu *File*, untuk menampilkan menu tambahan seperti menu *Open* yang berguna untuk membuka citra, menu *Save* yang berguna untuk menyimpan citra hasil proses AFCEDP, menu *Save As* yang berguna untuk menyimpan

citra hasil proses AFCEDP dengan lokasi yang berbeda, dan menu *Exit*

- 2 : Menu *Pengujian*, untuk membuka *form* Pengujian.
- 3 : Menu *About*, untuk membuka *form* About Us.
- 4 : *picturebox*, untuk menampilkan citra awal.
- 5 : *picturebox*, untuk menampilkan citra hasil proses algoritma AFCEDP.
- 6 : *picturebox*, untuk menampilkan citra hasil proses algoritma ACEDP.
- 7 : *textbox*, untuk menampilkan nilai *Contrast* citra awal.
- 8 : *textbox*, untuk menampilkan nilai *Entropy* citra awal.
- 9 : *textbox*, untuk menampilkan nilai *Contrast* citra hasil proses AFCEDP.
- 10 : *textbox*, untuk menampilkan nilai *Entropy* citra hasil proses AFCEDP.
- 11 : *textbox*, untuk menampilkan nilai *Contrast* citra hasil proses ACEDP.
- 12 : *textbox*, untuk menampilkan nilai *Entropy* citra hasil proses ACEDP.
- 13 : *numeric up down*, untuk mengatur nilai konstanta c1.
- 14 : *numeric up down*, untuk mengatur nilai konstanta c2.
- 15 : *textbox*, untuk mengatur nilai batas bawah HE.
- 16 : *textbox*, untuk mengatur nilai batas atas HE.
- 17 : tombol “Proses”, untuk memulai proses peningkatan kontras citra.

### 3.2.2 Form Pengujian

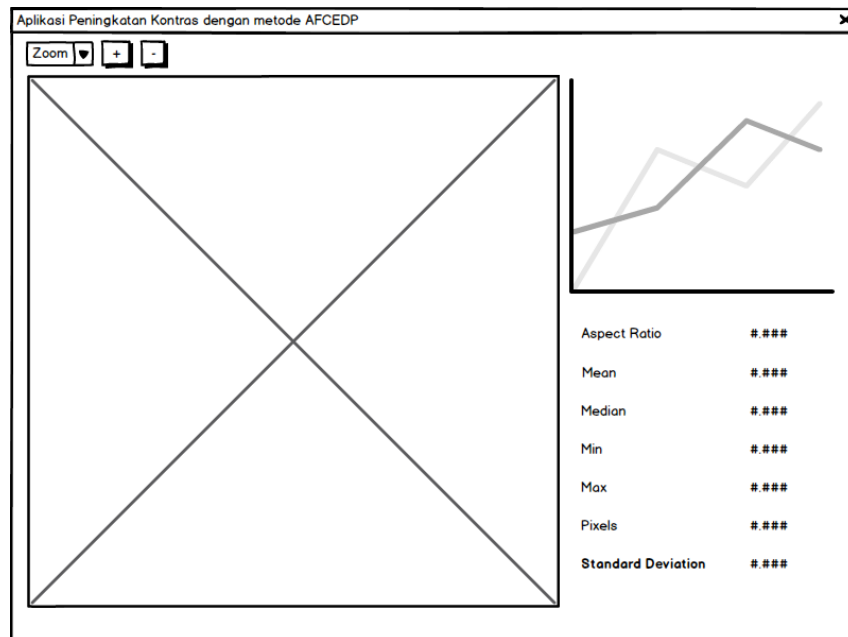
*Form* ini berfungsi untuk melakukan pengujian sekaligus dengan mengambil sebuah lokasi *folder* sebagai sumber beberapa sampel citra yang akan diuji dan menampilkan nilai pengujian pada tabel. Kemudian menunjukkan nilai pengujian rata-rata yang didapatkan dari sampel citra pengujian. Rancangan Tampilan *form* pengujian dapat dilihat pada gambar 3.10 berikut.

[illegible]

**Gambar 3.10** Rancangan *form* Pengujian.

### 3.2.3 Form Zoom

*Form* ini berfungsi untuk memperbesar (*zoom in*) atau memperkecil (*zoom out*) citra atau citra hasil proses dari algoritma. *Level / Tingkatan zoom* dapat diatur dari 25% hingga 200%. Pada *form* ini juga akan menampilkan histogram dari citra yang dilihat serta informasi tambahan lainnya seperti *Aspect Ratio*, *Mean*, *Median*, *Min*, *Max*, *Pixels*, *Standard Deviation*. Rancangan tampilan dari *form zoom* dapat dilihat pada gambar 3.11 berikut.



**Gambar 3.11** Rancangan Tampilan *form* Zoom.

### 3.2.4 Form Log

*Form Log* berisi informasi singkat data fisik dari citra awal dan juga citra hasil. Rancangan *form* dapat dilihat pada gambar 3.12 berikut.

Log					
Show All Data					
Pixel Citra Awal	Jumlah Kemunculan Pixel	Pixel AFCEDP	Jumlah Kemunculan AFCEDP	Pixel ACEDP	Jumlah Kemunculan ACEDP
0	###	###	###	###	###
1	###	###	###	###	###
2	###	###	###	###	###
3	###	###	###	###	###

**Gambar 3.12** Rancangan *Form Log*

### 3.2.5 Form About Us

*Form About* berisi informasi singkat mengenai aplikasi dan identitas mahasiswa yang mengembangkan aplikasi. Rancangan *form* dapat dilihat pada gambar 3.13 berikut.

The diagram shows a rectangular window titled "About". Inside the window, there is a header section labeled "NAMA APLIKASI". Below this header is a large rectangular area labeled "Informasi Singkat Mengenai Aplikasi". At the bottom left of the window is a smaller rectangular area labeled "Identitas Mahasiswa". At the bottom right of the window is a button labeled "OK".

**Gambar 3.13** Rancangan Tampilan *form* About

## BAB IV

### HASIL DAN PENGUJIAN

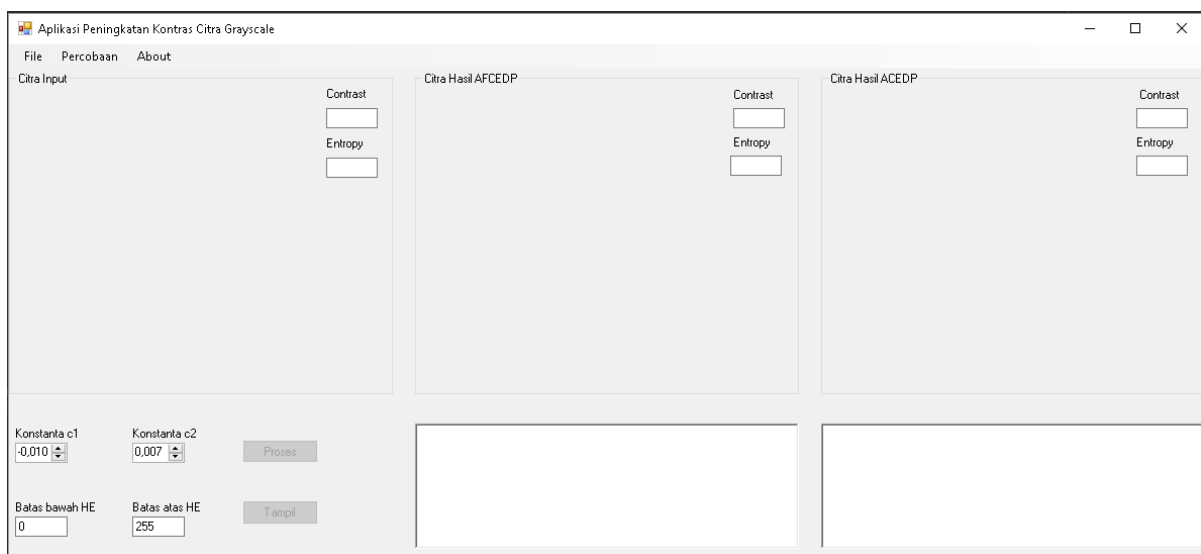
#### 4.1 Hasil

Untuk menjalankan aplikasi, perangkat lunak ( *software* ) yang dibutuhkan adalah sebagai berikut:

1. Sistem Operasi *Windows XP / 7 / 8 / 10*.
2. *Microsoft Visual Studio 2013* atau cukup meng-*install .Net Framework* versi 3.5 ke atas.

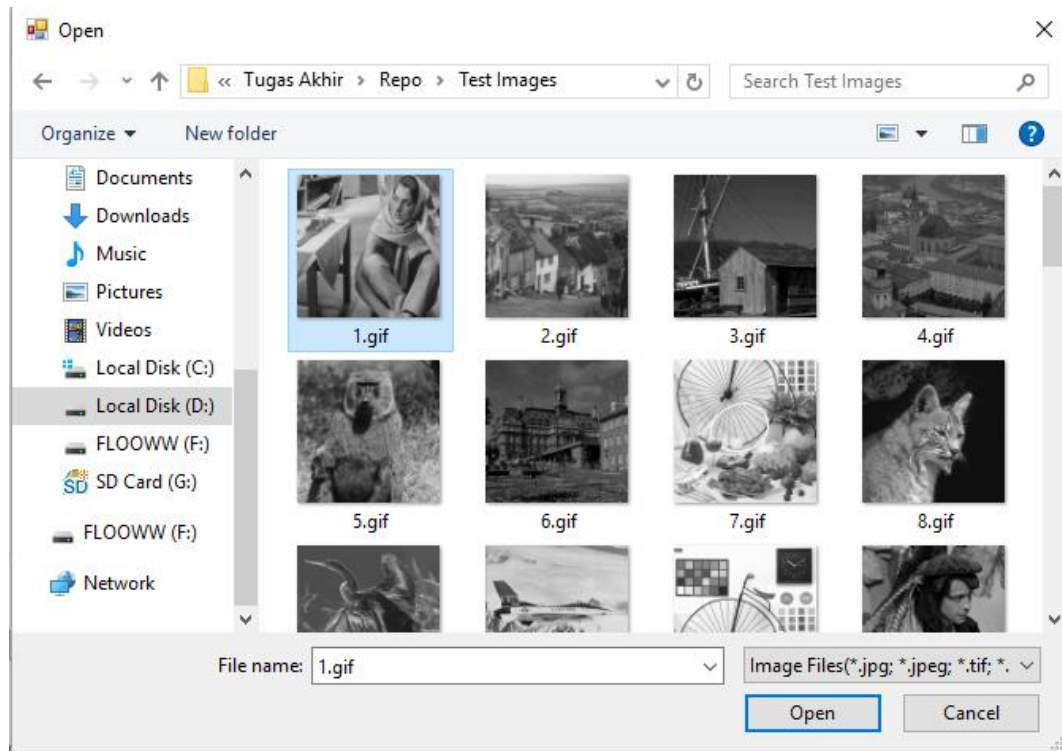
Berikut akan dijelaskan implementasi hasil dari aplikasi peningkatan kontras citra:

1. Saat aplikasi dijalankan, *form* Utama akan muncul seperti terlihat pada gambar 4.1. *Form* Utama berisi tampilan citra awal dan citra hasil proses dari algoritma peningkatan kontras citra, serta nilai *Entropy* dan *Contrast* citra awal maupun citra hasil.



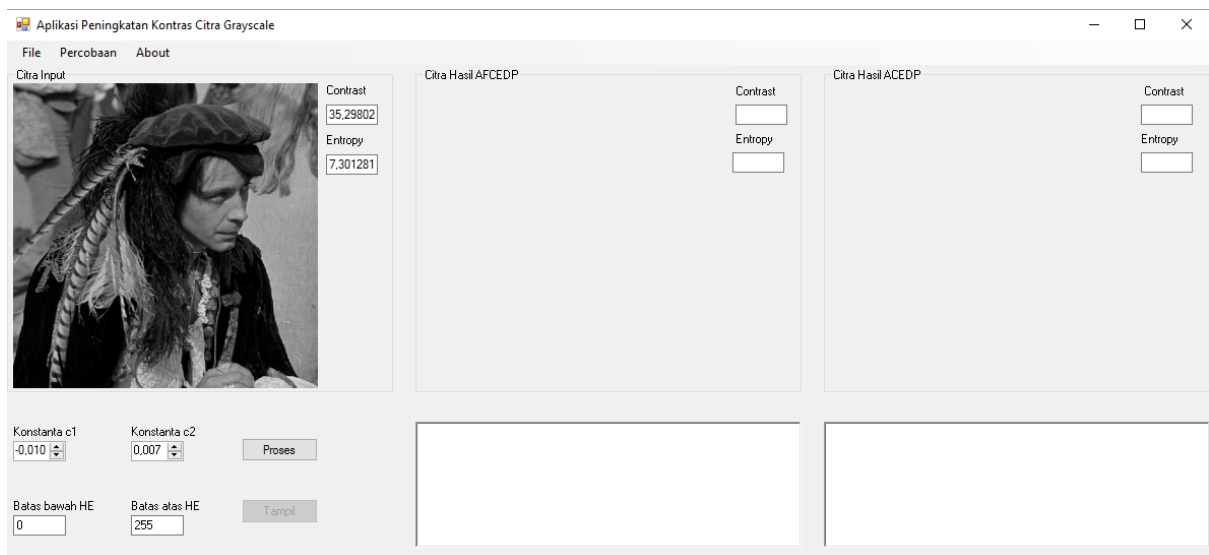
**Gambar 4.1** Tampilan *Form* Utama

2. Untuk memilih citra yang ingin diproses , klik pada menu [File] – [Open], maka akan muncul kotak dialog untuk pemilihan citra input seperti terlihat pada gambar 4.2 berikut.



**Gambar 4.2** Kotak Dialog *Open*

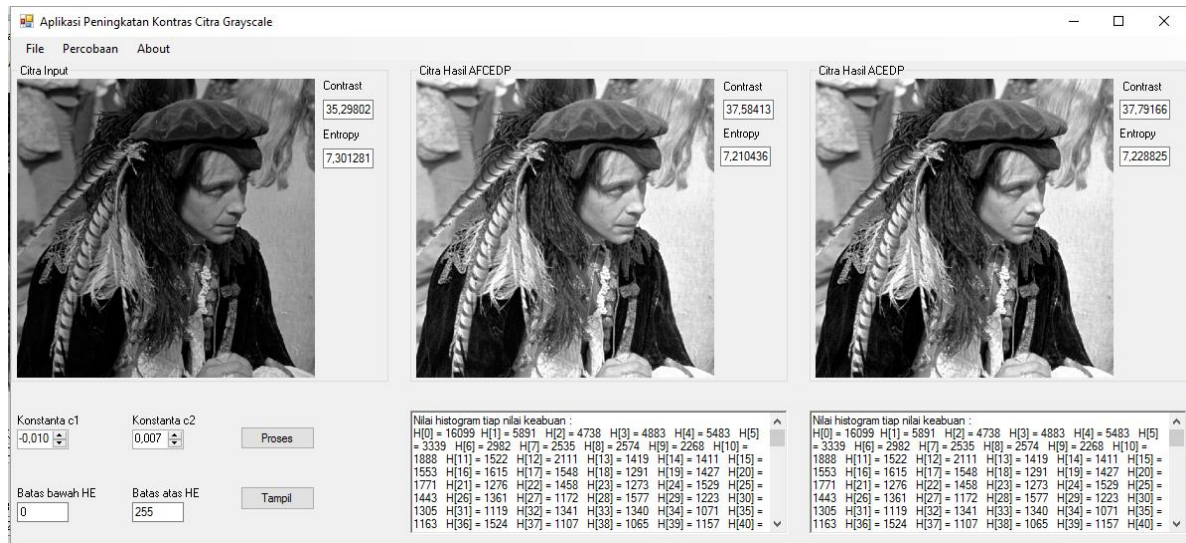
Pilih *file* citra asli, kemudian citra asli tersebut akan ditampilkan pada *form* Utama beserta nilai *Entropy* dan *Contrast* citra asli, seperti yang terlihat pada gambar 4.3.



**Gambar 4.3** Tampilan citra, nilai *contrast*, nilai *entropy* pada *Form* Utama

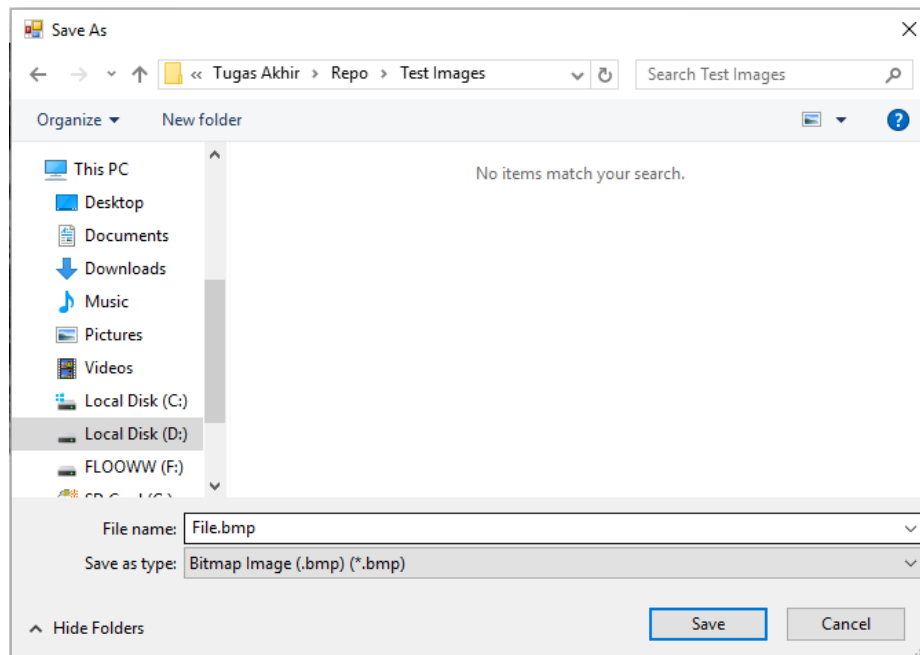


3. Nilai konstanta  $c1$ ,  $c2$ , batas bawah dan batas atas HE dapat diisi sesuai dengan *range* yang ada. Klik pada tombol [Proses] untuk memulai proses peningkatan kontras terhadap citra dengan menggunakan nilai konstanta yang telah ditentukan sebelumnya. Hasil peningkatan kontras citra beserta nilai *entropy* dan nilai *contrast* dapat dilihat pada gambar 4.4 berikut.



**Gambar 4.4** Tampilan Hasil Peningkatan Kontras Citra pada *Form* Utama

4. Fungsi untuk menyimpan citra hasil algoritma dapat dipilih pada menu [File] – [Save], citra akan langsung disimpan ditempat citra itu dibuka. Jika ingin citra hasil disimpan di tempat yang berbeda, maka pilih menu [File] – [Save As]. Citra hasil yang disimpan adalah citra hasil proses algoritma AFCEDP. Kotak dialog *save as* akan muncul seperti terlihat pada gambar 4.5.



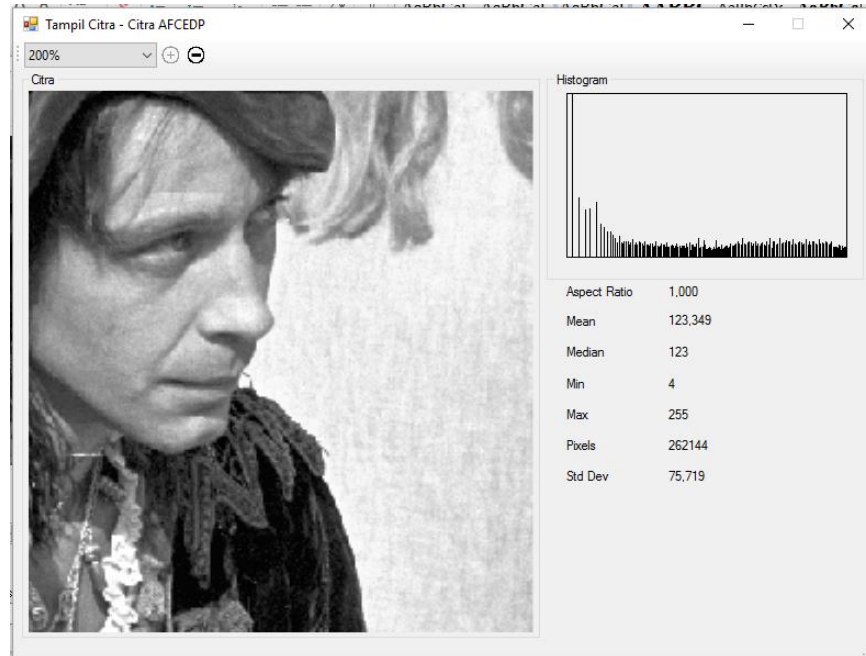
**Gambar 4.5** Kotak Dialog *Save As*

5. Untuk memperbesar dan memperkecil citra serta melihat data lain-lainnya seperti histogram pada citra dapat diakses dengan cara mengklik kiri sekali pada citra yang ingin dilihat. *Form* Tampil Citra akan muncul seperti terlihat pada gambar 4.6 berikut.



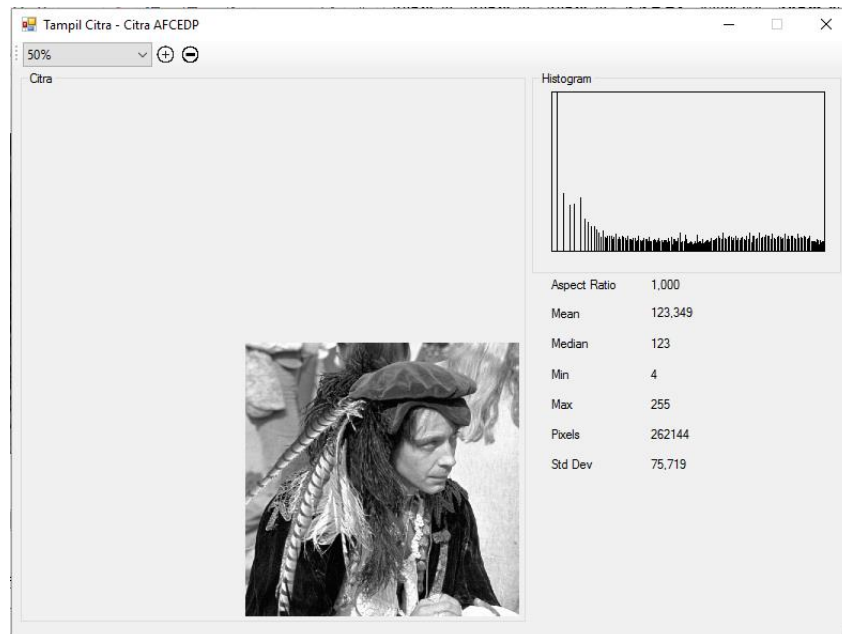
**Gambar 4.6** Tampilan *Form* Tampil Citra

Klik tombol “+” pada *form* Tampil Citra untuk memperbesar (*zoom in*) citra dari ukuran semula, seperti yang terlihat pada gambar 4.7 berikut.



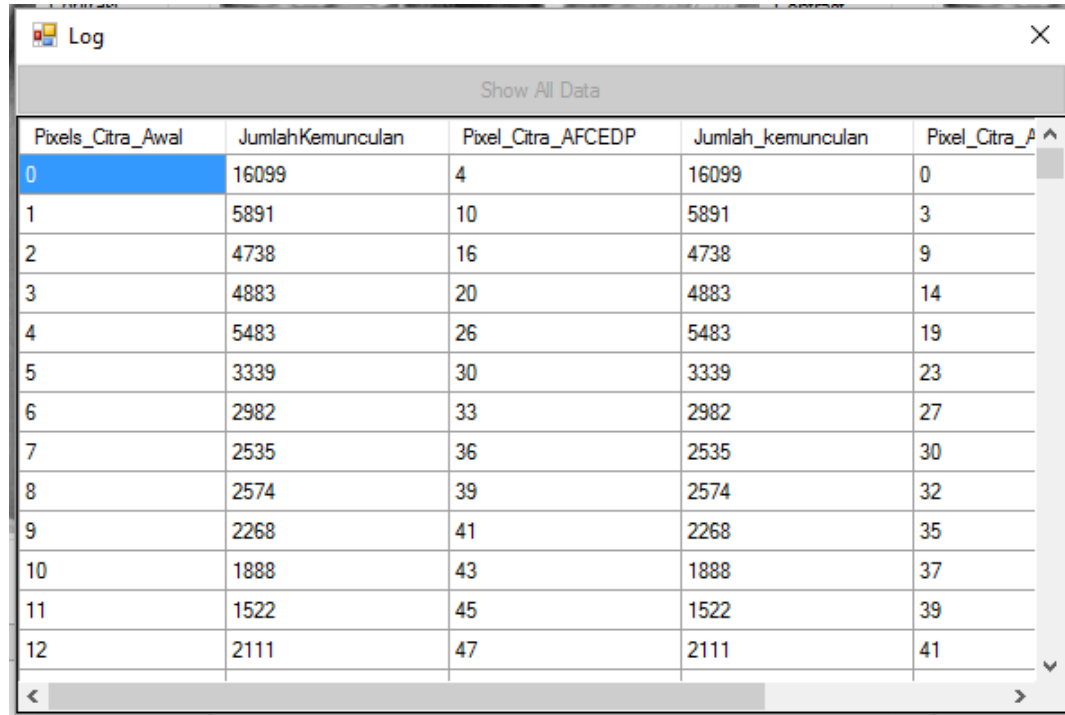
**Gambar 4.7** Tampilan *Form* Tampil Citra (*Zoom In* 200%)

Klik tombol “-” pada *form* Tampil Citra untuk memperkecil (*zoom out*) citra dari ukuran semula, seperti terlihat pada gambar 4.8 berikut.



**Gambar 4.8** Tampilan *Form* Tampil Citra (*Zoom Out* 50%)

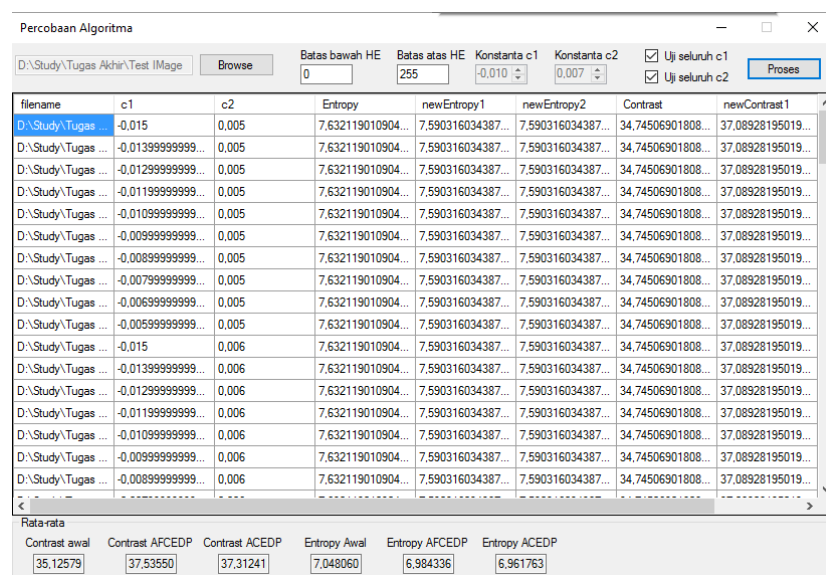
6. Klik pada tombol [Tampil] untuk menampilkan *form log* yang berisi data fisik citra awal dan citra hasil, seperti terlihat pada gambar 4.9 berikut.



Pixels_Citra_Awal	JumlahKemunculan	Pixel_Citra_AFCEDP	Jumlah_kemunculan	Pixel_Citra_A
0	16099	4	16099	0
1	5891	10	5891	3
2	4738	16	4738	9
3	4883	20	4883	14
4	5483	26	5483	19
5	3339	30	3339	23
6	2982	33	2982	27
7	2535	36	2535	30
8	2574	39	2574	32
9	2268	41	2268	35
10	1888	43	1888	37
11	1522	45	1522	39
12	2111	47	2111	41

**Gambar 4.9** Tampilan *Form Log*

7. Klik menu [Percobaan] akan menampilkan *form* Percobaan yang digunakan untuk menghitung nilai *entropy* dan *contrast* dari semua citra yang berada pada folder yang dipilih. *Form* Percobaan dapat dilihat pada gambar 4.10 berikut.



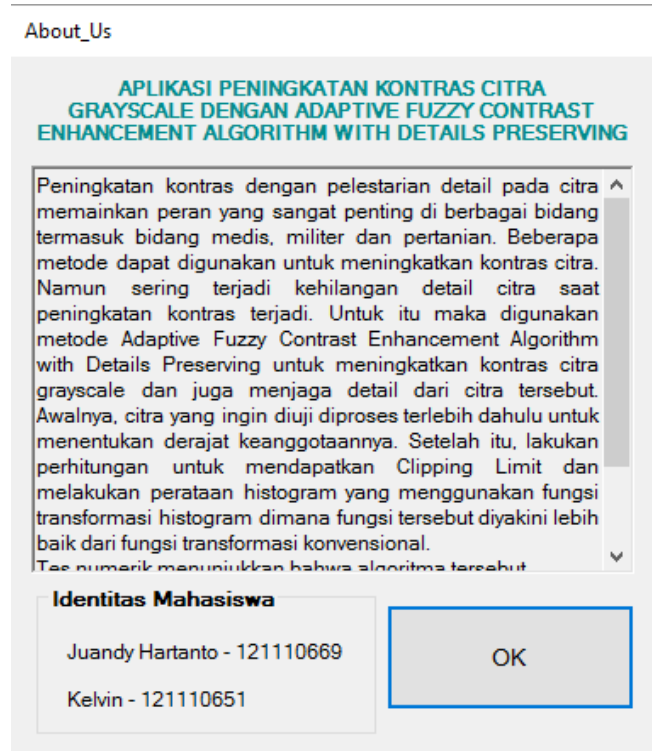
filename	c1	c2	Entropy	newEntropy1	newEntropy2	Contrast	newContrast1
D:\Study\Tugas ...	-0.015	0.005	7.632119010904...	7.590316034387...	7.590316034387...	34.74506901808...	37.08928195019...
D:\Study\Tugas ...	-0.01399999999...	0.005	7.632119010904...	7.590316034387...	7.590316034387...	34.74506901808...	37.08928195019...
D:\Study\Tugas ...	-0.01299999999...	0.005	7.632119010904...	7.590316034387...	7.590316034387...	34.74506901808...	37.08928195019...
D:\Study\Tugas ...	-0.01199999999...	0.005	7.632119010904...	7.590316034387...	7.590316034387...	34.74506901808...	37.08928195019...
D:\Study\Tugas ...	-0.01099999999...	0.005	7.632119010904...	7.590316034387...	7.590316034387...	34.74506901808...	37.08928195019...
D:\Study\Tugas ...	-0.00999999999...	0.005	7.632119010904...	7.590316034387...	7.590316034387...	34.74506901808...	37.08928195019...
D:\Study\Tugas ...	-0.00899999999...	0.005	7.632119010904...	7.590316034387...	7.590316034387...	34.74506901808...	37.08928195019...
D:\Study\Tugas ...	-0.00799999999...	0.005	7.632119010904...	7.590316034387...	7.590316034387...	34.74506901808...	37.08928195019...
D:\Study\Tugas ...	-0.00699999999...	0.005	7.632119010904...	7.590316034387...	7.590316034387...	34.74506901808...	37.08928195019...
D:\Study\Tugas ...	-0.00599999999...	0.005	7.632119010904...	7.590316034387...	7.590316034387...	34.74506901808...	37.08928195019...
D:\Study\Tugas ...	-0.015	0.006	7.632119010904...	7.590316034387...	7.590316034387...	34.74506901808...	37.08928195019...
D:\Study\Tugas ...	-0.01399999999...	0.006	7.632119010904...	7.590316034387...	7.590316034387...	34.74506901808...	37.08928195019...
D:\Study\Tugas ...	-0.01299999999...	0.006	7.632119010904...	7.590316034387...	7.590316034387...	34.74506901808...	37.08928195019...
D:\Study\Tugas ...	-0.01199999999...	0.006	7.632119010904...	7.590316034387...	7.590316034387...	34.74506901808...	37.08928195019...
D:\Study\Tugas ...	-0.01099999999...	0.006	7.632119010904...	7.590316034387...	7.590316034387...	34.74506901808...	37.08928195019...
D:\Study\Tugas ...	-0.00999999999...	0.006	7.632119010904...	7.590316034387...	7.590316034387...	34.74506901808...	37.08928195019...
D:\Study\Tugas ...	-0.00899999999...	0.006	7.632119010904...	7.590316034387...	7.590316034387...	34.74506901808...	37.08928195019...

Rata-rata					
Contrast awal	Contrast AFCEDP	Contrast ACEDP	Entropy Awal	Entropy AFCEDP	Entropy ACEDP
35,12579	37,53550	37,31241	7,048060	6,984336	6,961763

**Gambar 4.10** Tampilan *Form Percobaan Algoritma*

8. Klik menu [About] akan menampilkan *form About Us*. *Form About Us* akan muncul seperti terlihat pada gambar 4.11 berikut.



The screenshot shows a window titled "About\_Us". Inside, the title is "APLIKASI PENINGKATAN KONTRAS CITRA GRAYSCALE DENGAN ADAPTIVE FUZZY CONTRAST ENHANCEMENT ALGORITHM WITH DETAILS PRESERVING". The main text describes the application's purpose: to enhance contrast while preserving details in grayscale images using the Adaptive Fuzzy Contrast Enhancement Algorithm with Details Preserving. It mentions that this is important in fields like medicine, military, and agriculture, where detail loss is a common problem. The text also states that the algorithm was tested on 49 images and that its performance was evaluated using Contrast Improvement Evaluation (CIE). At the bottom, there is a section titled "Identitas Mahasiswa" (Student Identity) with two entries: "Juandy Hartanto - 121110669" and "Kelvin - 121110651". An "OK" button is located to the right of this section.

**Gambar 4.11** Tampilan *Form About Us*

## 4.2 Pengujian

Pengujian terhadap aplikasi dilakukan dengan menggunakan beberapa skenario sebagai berikut:

1. Menguji aplikasi untuk mengetahui apakah algoritma AFCEDP mampu meningkatkan kontras citra awal. Pengujian dilakukan dengan menggunakan nilai  $c1 = -0,01$ , nilai  $c2 = 0,007$ , batas bawah HE = 0, dan batas atas HE = 255 terhadap 49 buah citra. Nilai “% Entropy” didapatkan dari perbandingan nilai *entropy* AFCEDP dengan citra awal, kemudian dikalikan dengan 100. Untuk nilai kontras didapatkan dari Contrast Improvement Evaluation (CIE) yang dihitung dari citra awal dan citra hasil AFCEDP. Hasil pengujian dapat dilihat pada tabel 4.1 berikut.

**Tabel 4.1** Pengujian Peningkatan kontras AFCEDP

File Name	c1	c2	Entropy Awal	Entropy AFCEDP	% Entropy	Nilai CIE awal	Nilai CIE AFCEDP
1.gif	-0,01	0,007	7,6321	7,5903	99,4523	34,7451	37,0893
10.gif	-0,01	0,007	6,6776	6,5340	97,8484	33,0867	36,9885
11.gif	-0,01	0,007	4,1535	4,1535	100,0000	37,3299	37,6326
12.gif	-0,01	0,007	7,3013	7,2104	98,7558	35,2980	37,5841
13.gif	-0,01	0,007	7,2416	7,1395	98,5900	33,6614	37,0695
14.gif	-0,01	0,007	6,7166	6,6959	99,6916	32,9939	36,5427
15.gif	-0,01	0,007	6,4342	6,3988	99,4501	31,8387	37,3106
16.gif	-0,01	0,007	5,7538	5,7159	99,3429	36,0499	37,5502
17.gif	-0,01	0,007	6,6501	6,5766	98,8949	35,9049	37,6268
18.gif	-0,01	0,007	6,4678	6,4344	99,4835	36,3197	37,4289
19.gif	-0,01	0,007	6,4190	6,3595	99,0745	33,7226	37,3942
2.gif	-0,01	0,007	7,4778	7,3777	98,6611	33,8440	36,4798
20.gif	-0,01	0,007	6,6721	6,6355	99,4514	33,5413	37,3405
21.gif	-0,01	0,007	5,4720	5,4378	99,3738	31,3369	37,3743
22.gif	-0,01	0,007	6,3322	6,3180	99,7759	31,6017	36,1474
23.gif	-0,01	0,007	5,8211	5,7610	98,9668	31,3773	37,2888
24.gif	-0,01	0,007	5,8108	5,7671	99,2493	30,4114	36,5977
25.gif	-0,01	0,007	5,7922	5,7621	99,4811	28,0088	37,1777
26.gif	-0,01	0,007	6,2175	6,1747	99,3125	28,5541	36,9206
27.gif	-0,01	0,007	5,8484	5,7827	98,8777	32,6972	37,3657
28.gif	-0,01	0,007	6,3651	6,3537	99,8217	32,2622	36,8417
29.gif	-0,01	0,007	5,7148	5,6494	98,8548	33,7684	37,4419
3.gif	-0,01	0,007	6,2786	6,2503	99,5492	31,6492	37,2664
30.gif	-0,01	0,007	6,4227	6,3793	99,3245	35,0465	37,4057
31.gif	-0,01	0,007	6,1061	6,0198	98,5860	34,2057	37,3991
32.gif	-0,01	0,007	6,0172	5,9891	99,5325	36,7986	38,4096
33.gif	-0,01	0,007	6,6286	6,5817	99,2926	33,4393	37,0685
34.gif	-0,01	0,007	6,5449	6,5314	99,7934	33,9010	37,2586
35.gif	-0,01	0,007	6,6144	6,6096	99,9268	32,5474	37,1786
36.gif	-0,01	0,007	6,1162	6,0461	98,8540	32,7056	37,3977
37.gif	-0,01	0,007	6,1286	6,0904	99,3775	29,7351	37,2346
38.gif	-0,01	0,007	6,1808	6,1641	99,7295	31,6764	37,7015
39.gif	-0,01	0,007	5,8038	5,7938	99,8269	33,9829	38,9743
4.gif	-0,01	0,007	6,1391	6,0880	99,1678	28,5668	37,1970
40.gif	-0,01	0,007	6,7560	6,7099	99,3181	28,5879	36,1753
41.gif	-0,01	0,007	7,0572	6,9984	99,1672	32,9390	35,8808
42.gif	-0,01	0,007	7,0195	6,9055	98,3747	32,0454	36,1626

43.gif	-0,01	0,007	6,7893	6,5445	96,3946	34,0525	37,3345
44.gif	-0,01	0,007	3,8595	3,8593	99,9949	34,4296	36,2219
45.gif	-0,01	0,007	7,4570	7,3866	99,0565	36,0897	37,6275
46.gif	-0,01	0,007	5,7056	5,7046	99,9834	34,7636	36,5074
47.gif	-0,01	0,007	7,3577	7,2927	99,1154	32,5269	36,5619
48.gif	-0,01	0,007	7,1238	7,0533	99,0113	34,3738	36,6110
49.gif	-0,01	0,007	6,4326	6,3270	98,3585	37,1441	38,2160
5.gif	-0,01	0,007	4,4619	4,4609	99,9778	32,8277	36,3284
6.gif	-0,01	0,007	6,2649	6,2127	99,1666	30,5447	37,2607
7.gif	-0,01	0,007	4,3691	4,3691	100,0000	36,0664	37,2583
8.gif	-0,01	0,007	5,1924	5,1334	98,8644	35,2080	37,2403
9.gif	-0,01	0,007	6,1546	6,1453	99,8498	33,9478	36,5748

Pada tabel 4.1 terlihat bahwa algoritma AFCEDP ( *Adaptive Fuzzy Contrast Enhancement with Details Preserving* ) mampu meningkatkan kontras citra awal dan mampu melestarikan detail citra. Hal tersebut dibuktikan dengan perolehan nilai CIE untuk AFCEDP lebih tinggi dibandingkan dengan nilai CIE citra awal dan perbedaan nilai *entropy* AFCEDP dengan citra awal cukup kecil.

2. Menguji aplikasi untuk mengetahui nilai *c1* yang terbaik. Pengujian dilakukan dengan nilai *c1* dari -0.015 hingga -0.005, nilai *c2* = 0.007, batas bawah HE = 0, dan batas atas HE = 255 terhadap 22 buah citra. Hasil pengujian dapat dilihat pada tabel 4.2 berikut.

**Tabel 4.2** Pengujian beberapa nilai *c1*

Nama File	c1	c2	Entropy AFCEDP	Nilai CIE AFCEDP	Nilai c1 terbaik untuk Entropy	Nilai c1 terbaik untuk CIE
15.gif	-0,015	0,007	6,39607	37,290528	-0,01	-0,005
15.gif	-0,014	0,007	6,391791	37,292058		
15.gif	-0,013	0,007	6,386824	37,294246		
15.gif	-0,012	0,007	6,387432	37,307696		
15.gif	-0,011	0,007	6,387432	37,311233		
15.gif	-0,01	0,007	6,398777	37,310582		
15.gif	-0,009	0,007	6,397622	37,31538		
15.gif	-0,008	0,007	6,397034	37,320076		
15.gif	-0,007	0,007	6,397068	37,318441		
15.gif	-0,006	0,007	6,392413	37,320207		



15.gif	-0,005	0,007	6,392413	37,32535		
16.gif	-0,015	0,007	5,719107	37,538855	-0,015	-0,01
16.gif	-0,014	0,007	5,718046	37,534875		
16.gif	-0,013	0,007	5,712378	37,537889		
16.gif	-0,012	0,007	5,717333	37,541377		
16.gif	-0,011	0,007	5,718825	37,542674		
16.gif	-0,01	0,007	5,715945	37,550177		
16.gif	-0,009	0,007	5,709686	37,540305		
16.gif	-0,008	0,007	5,714299	37,536047		
16.gif	-0,007	0,007	5,715706	37,535091		
16.gif	-0,006	0,007	5,712347	37,536679		
16.gif	-0,005	0,007	5,712284	37,533167		
17.gif	-0,015	0,007	6,584568	37,639102	-0,013	-0,015
17.gif	-0,014	0,007	6,588393	37,635658		
17.gif	-0,013	0,007	6,595446	37,631742		
17.gif	-0,012	0,007	6,594098	37,63051		
17.gif	-0,011	0,007	6,586596	37,627432		
17.gif	-0,01	0,007	6,57659	37,626776		
17.gif	-0,009	0,007	6,578325	37,623571		
17.gif	-0,008	0,007	6,584399	37,61745		
17.gif	-0,007	0,007	6,577814	37,6158		
17.gif	-0,006	0,007	6,582812	37,61464		
17.gif	-0,005	0,007	6,591589	37,610174		
18.gif	-0,015	0,007	6,447309	37,470296	-0,013	-0,015
18.gif	-0,014	0,007	6,451266	37,467149		
18.gif	-0,013	0,007	6,451789	37,462524		
18.gif	-0,012	0,007	6,44832	37,445424		
18.gif	-0,011	0,007	6,440251	37,436023		
18.gif	-0,01	0,007	6,434431	37,428885		
18.gif	-0,009	0,007	6,448394	37,425973		
18.gif	-0,008	0,007	6,451736	37,427472		
18.gif	-0,007	0,007	6,445022	37,422512		
18.gif	-0,006	0,007	6,439979	37,414553		
18.gif	-0,005	0,007	6,434354	37,402435		
19.gif	-0,015	0,007	6,354135	37,435426	-0,005	-0,015
19.gif	-0,014	0,007	6,362857	37,430614		
19.gif	-0,013	0,007	6,365107	37,429237		
19.gif	-0,012	0,007	6,361878	37,421048		
19.gif	-0,011	0,007	6,356594	37,411055		
19.gif	-0,01	0,007	6,359549	37,394226		
19.gif	-0,009	0,007	6,363199	37,392676		



19.gif	-0,008	0,007	6,365618	37,388598		
19.gif	-0,007	0,007	6,367479	37,387004		
19.gif	-0,006	0,007	6,37316	37,387231		
19.gif	-0,005	0,007	6,376753	37,374823		
20.gif	-0,015	0,007	6,648551	37,342613	-0,014	-0,012
20.gif	-0,014	0,007	6,653353	37,345461		
20.gif	-0,013	0,007	6,644411	37,345534		
20.gif	-0,012	0,007	6,643782	37,351474		
20.gif	-0,011	0,007	6,63753	37,336593		
20.gif	-0,01	0,007	6,635496	37,340527		
20.gif	-0,009	0,007	6,64243	37,332951		
20.gif	-0,008	0,007	6,640134	37,321173		
20.gif	-0,007	0,007	6,647032	37,320715		
20.gif	-0,006	0,007	6,650381	37,308363		
20.gif	-0,005	0,007	6,638463	37,30869		
21.gif	-0,015	0,007	5,447903	37,387368	-0,015	-0,012
21.gif	-0,014	0,007	5,438442	37,388817		
21.gif	-0,013	0,007	5,438442	37,388367		
21.gif	-0,012	0,007	5,435314	37,393598		
21.gif	-0,011	0,007	5,438744	37,392788		
21.gif	-0,01	0,007	5,43778	37,374311		
21.gif	-0,009	0,007	5,441293	37,373953		
21.gif	-0,008	0,007	5,434105	37,37278		
21.gif	-0,007	0,007	5,429419	37,372246		
21.gif	-0,006	0,007	5,432597	37,37417		
21.gif	-0,005	0,007	5,436923	37,372303		
23.gif	-0,015	0,007	5,757065	37,304442	-0,005	-0,005
23.gif	-0,014	0,007	5,757241	37,298088		
23.gif	-0,013	0,007	5,759342	37,289116		
23.gif	-0,012	0,007	5,751897	37,293454		
23.gif	-0,011	0,007	5,761129	37,291024		
23.gif	-0,01	0,007	5,760974	37,288833		
23.gif	-0,009	0,007	5,760974	37,289365		
23.gif	-0,008	0,007	5,760983	37,294345		
23.gif	-0,007	0,007	5,761101	37,29994		
23.gif	-0,006	0,007	5,753486	37,302719		
23.gif	-0,005	0,007	5,761164	37,312336		
25.gif	-0,015	0,007	5,767457	37,10948	-0,012	-0,005
25.gif	-0,014	0,007	5,767171	37,12576		
25.gif	-0,013	0,007	5,763005	37,138182		
25.gif	-0,012	0,007	5,772429	37,148885		

25.gif	-0,011	0,007	5,767008	37,160379		
25.gif	-0,01	0,007	5,762126	37,177701		
25.gif	-0,009	0,007	5,758334	37,197396		
25.gif	-0,008	0,007	5,75802	37,203675		
25.gif	-0,007	0,007	5,764953	37,213898		
25.gif	-0,006	0,007	5,763749	37,219501		
25.gif	-0,005	0,007	5,760873	37,229881		
26.gif	-0,015	0,007	6,180468	36,833483		
26.gif	-0,014	0,007	6,173537	36,86121		
26.gif	-0,013	0,007	6,17307	36,86929		
26.gif	-0,012	0,007	6,174734	36,890624		
26.gif	-0,011	0,007	6,169526	36,90778		
26.gif	-0,01	0,007	6,174737	36,920646	-0,015	-0,005
26.gif	-0,009	0,007	6,16822	36,943716		
26.gif	-0,008	0,007	6,168562	36,957057		
26.gif	-0,007	0,007	6,17052	36,971782		
26.gif	-0,006	0,007	6,16772	36,982426		
26.gif	-0,005	0,007	6,164906	36,997929		
27.gif	-0,015	0,007	5,796438	37,370427		
27.gif	-0,014	0,007	5,790214	37,364719		
27.gif	-0,013	0,007	5,795392	37,368455		
27.gif	-0,012	0,007	5,798374	37,367654		
27.gif	-0,011	0,007	5,788311	37,367334		
27.gif	-0,01	0,007	5,782731	37,365708	-0,012	-0,015
27.gif	-0,009	0,007	5,772422	37,359564		
27.gif	-0,008	0,007	5,78558	37,351231		
27.gif	-0,007	0,007	5,784758	37,347518		
27.gif	-0,006	0,007	5,780679	37,349737		
27.gif	-0,005	0,007	5,776687	37,3459		
29.gif	-0,015	0,007	5,639171	37,453335		
29.gif	-0,014	0,007	5,649826	37,446016		
29.gif	-0,013	0,007	5,655495	37,445682		
29.gif	-0,012	0,007	5,655907	37,439484		
29.gif	-0,011	0,007	5,645192	37,440455		
29.gif	-0,01	0,007	5,649354	37,441912	-0,012	-0,015
29.gif	-0,009	0,007	5,642924	37,430192		
29.gif	-0,008	0,007	5,653554	37,426127		
29.gif	-0,007	0,007	5,64983	37,42845		
29.gif	-0,006	0,007	5,649586	37,411141		
29.gif	-0,005	0,007	5,653081	37,415544		
3.gif	-0,015	0,007	6,245375	37,215549	-0,01	-0,005

3.gif	-0,014	0,007	6,249801	37,239218		
3.gif	-0,013	0,007	6,248464	37,247235		
3.gif	-0,012	0,007	6,24082	37,254448		
3.gif	-0,011	0,007	6,243225	37,260611		
3.gif	-0,01	0,007	6,25032	37,266385		
3.gif	-0,009	0,007	6,247152	37,278459		
3.gif	-0,008	0,007	6,242017	37,282085		
3.gif	-0,007	0,007	6,235967	37,285565		
3.gif	-0,006	0,007	6,240253	37,293647		
3.gif	-0,005	0,007	6,249817	37,300449		
30.gif	-0,015	0,007	6,390897	37,346691	-0,008	-0,005
30.gif	-0,014	0,007	6,390943	37,356704		
30.gif	-0,013	0,007	6,383138	37,370001		
30.gif	-0,012	0,007	6,388303	37,38201		
30.gif	-0,011	0,007	6,389275	37,390398		
30.gif	-0,01	0,007	6,379317	37,405739		
30.gif	-0,009	0,007	6,387983	37,42123		
30.gif	-0,008	0,007	6,398727	37,430702		
30.gif	-0,007	0,007	6,393278	37,430546		
30.gif	-0,006	0,007	6,386315	37,438946		
30.gif	-0,005	0,007	6,378347	37,452253	-0,012	-0,015
31.gif	-0,015	0,007	6,025275	37,431681		
31.gif	-0,014	0,007	6,029164	37,423272		
31.gif	-0,013	0,007	6,027348	37,417376		
31.gif	-0,012	0,007	6,030523	37,403608		
31.gif	-0,011	0,007	6,022033	37,399602		
31.gif	-0,01	0,007	6,019768	37,399091		
31.gif	-0,009	0,007	6,022046	37,393642		
31.gif	-0,008	0,007	6,022539	37,389448		
31.gif	-0,007	0,007	6,029956	37,379114		
31.gif	-0,006	0,007	6,024458	37,367346	-0,013	-0,005
31.gif	-0,005	0,007	6,024888	37,361833		
34.gif	-0,015	0,007	6,533492	37,217261		
34.gif	-0,014	0,007	6,520945	37,225218		
34.gif	-0,013	0,007	6,53898	37,245209		
34.gif	-0,012	0,007	6,535559	37,254421		
34.gif	-0,011	0,007	6,530437	37,253841		
34.gif	-0,01	0,007	6,531416	37,258578		
34.gif	-0,009	0,007	6,525483	37,269068		
34.gif	-0,008	0,007	6,52359	37,28136		
34.gif	-0,007	0,007	6,521364	37,284355		

34.gif	-0,006	0,007	6,525832	37,289979		
34.gif	-0,005	0,007	6,511253	37,296338		
36.gif	-0,015	0,007	6,062295	37,417027	-0,013	-0,015
36.gif	-0,014	0,007	6,063221	37,409908		
36.gif	-0,013	0,007	6,06354	37,402944		
36.gif	-0,012	0,007	6,062152	37,398009		
36.gif	-0,011	0,007	6,057073	37,395124		
36.gif	-0,01	0,007	6,046101	37,397676		
36.gif	-0,009	0,007	6,046752	37,379456		
36.gif	-0,008	0,007	6,04098	37,374287		
36.gif	-0,007	0,007	6,056557	37,37099		
36.gif	-0,006	0,007	6,05175	37,36732		
36.gif	-0,005	0,007	6,047714	37,365071		
37.gif	-0,015	0,007	6,111306	37,111116	-0,015	-0,005
37.gif	-0,014	0,007	6,100968	37,146619		
37.gif	-0,013	0,007	6,108266	37,161943		
37.gif	-0,012	0,007	6,094774	37,185251		
37.gif	-0,011	0,007	6,108266	37,202302		
37.gif	-0,01	0,007	6,090431	37,234593		
37.gif	-0,009	0,007	6,089119	37,258412		
37.gif	-0,008	0,007	6,104057	37,273304		
37.gif	-0,007	0,007	6,103136	37,289273		
37.gif	-0,006	0,007	6,097834	37,294804		
37.gif	-0,005	0,007	6,090536	37,317641		
4.gif	-0,015	0,007	6,100591	37,121399	-0,014	-0,005
4.gif	-0,014	0,007	6,103781	37,138749		
4.gif	-0,013	0,007	6,083655	37,161095		
4.gif	-0,012	0,007	6,096407	37,170019		
4.gif	-0,011	0,007	6,089057	37,184107		
4.gif	-0,01	0,007	6,087993	37,19703		
4.gif	-0,009	0,007	6,098634	37,207214		
4.gif	-0,008	0,007	6,101248	37,220692		
4.gif	-0,007	0,007	6,097269	37,229184		
4.gif	-0,006	0,007	6,097063	37,241731		
4.gif	-0,005	0,007	6,089016	37,247844		
43.gif	-0,015	0,007	6,564166	37,338918	-0,015	-0,015
43.gif	-0,014	0,007	6,555623	37,333879		
43.gif	-0,013	0,007	6,551294	37,333311		
43.gif	-0,012	0,007	6,544556	37,337053		
43.gif	-0,011	0,007	6,545732	37,335373		
43.gif	-0,01	0,007	6,544481	37,334534		

43.gif	-0,009	0,007	6,546376	37,337375		
43.gif	-0,008	0,007	6,53908	37,334433		
43.gif	-0,007	0,007	6,527542	37,338535		
43.gif	-0,006	0,007	6,541249	37,336295		
43.gif	-0,005	0,007	6,536658	37,338091		
6.gif	-0,015	0,007	6,200958	37,220221	-0,01	-0,005
6.gif	-0,014	0,007	6,204296	37,235741		
6.gif	-0,013	0,007	6,205601	37,244669		
6.gif	-0,012	0,007	6,200736	37,253724		
6.gif	-0,011	0,007	6,207319	37,257486		
6.gif	-0,01	0,007	6,21273	37,260746		
6.gif	-0,009	0,007	6,211741	37,267158		
6.gif	-0,008	0,007	6,20168	37,282934		
6.gif	-0,007	0,007	6,207952	37,29125		
6.gif	-0,006	0,007	6,206088	37,30213		
6.gif	-0,005	0,007	6,200227	37,306327		
8.gif	-0,015	0,007	5,14575	37,281402	-0,009	-0,015
8.gif	-0,014	0,007	5,140521	37,237166		
8.gif	-0,013	0,007	5,143692	37,238885		
8.gif	-0,012	0,007	5,141354	37,237147		
8.gif	-0,011	0,007	5,141547	37,237832		
8.gif	-0,01	0,007	5,133406	37,240322		
8.gif	-0,009	0,007	5,150574	37,24332		
8.gif	-0,008	0,007	5,149003	37,24406		
8.gif	-0,007	0,007	5,142878	37,245219		
8.gif	-0,006	0,007	5,133512	37,248042		
8.gif	-0,005	0,007	5,141329	37,248388		

Pada tabel 4.2, terlihat bahwa nilai  $c1$  terbaik untuk penjagaan detail citra adalah -0.015 ( sebanyak 5 citra) yang diikuti oleh -0.012 dan -0.013 yakni sebanyak 4 citra untuk masing masing nilai  $c1$ . Sedangkan nilai  $c1$  untuk peningkatan kontras adalah -0.005 yakni sebanyak 10 buah citra dan -0.015 sebanyak 9 buah citra. Maka nilai  $c1$  yang paling optimal dalam peningkatan kontras serta pelestarian detail adalah -0.015.

3. Menguji aplikasi untuk mengetahui nilai  $c2$  yang terbaik. Pengujian dilakukan dengan nilai  $c1 = -0,015$  (paling optimal) , nilai  $c2$  berkisar antara 0,005 hingga

0,007, batas bawah HE = 0, dan batas atas HE = 255 terhadap 22 buah citra.

Hasil pengujian dapat dilihat pada tabel 4.3 berikut.

**Tabel 4.3** Pengujian nilai  $c_2$  dengan nilai  $c_1 = -0,015$

Nama File	$c_1$	$c_2$	Entropy AFCEDP	Nilai CIE AFCEDP	Nilai $c_2$ untuk Entropy AFCEDP	Nilai $c_2$ untuk CIE AFCEDP
15.gif	-0,015	0,005	6,39607	37,29053	Sama Besar	Sama Besar
15.gif	-0,015	0,006	6,39607	37,29053		
15.gif	-0,015	0,007	6,39607	37,29053		
16.gif	-0,015	0,005	5,719107	37,53885	Sama Besar	Sama Besar
16.gif	-0,015	0,006	5,719107	37,53885		
16.gif	-0,015	0,007	5,719107	37,53885		
17.gif	-0,015	0,005	6,584568	37,6391	Sama Besar	Sama Besar
17.gif	-0,015	0,006	6,584568	37,6391		
17.gif	-0,015	0,007	6,584568	37,6391		
18.gif	-0,015	0,005	6,447309	37,4703	Sama Besar	Sama Besar
18.gif	-0,015	0,006	6,447309	37,4703		
18.gif	-0,015	0,007	6,447309	37,4703		
19.gif	-0,015	0,005	6,354135	37,43543	Sama Besar	Sama Besar
19.gif	-0,015	0,006	6,354135	37,43543		
19.gif	-0,015	0,007	6,354135	37,43543		
20.gif	-0,015	0,005	6,648551	37,34261	Sama Besar	Sama Besar
20.gif	-0,015	0,006	6,648551	37,34261		
20.gif	-0,015	0,007	6,648551	37,34261		
21.gif	-0,015	0,005	5,447903	37,38737	Sama Besar	Sama Besar
21.gif	-0,015	0,006	5,447903	37,38737		
21.gif	-0,015	0,007	5,447903	37,38737		
23.gif	-0,015	0,005	5,757065	37,30444	Sama Besar	Sama Besar
23.gif	-0,015	0,006	5,757065	37,30444		
23.gif	-0,015	0,007	5,757065	37,30444		
25.gif	-0,015	0,005	5,767457	37,10948	Sama Besar	Sama Besar
25.gif	-0,015	0,006	5,767457	37,10948		
25.gif	-0,015	0,007	5,767457	37,10948		
26.gif	-0,015	0,005	6,180468	36,83348	Sama Besar	Sama Besar
26.gif	-0,015	0,006	6,180468	36,83348		
26.gif	-0,015	0,007	6,180468	36,83348		
27.gif	-0,015	0,005	5,796438	37,37043	Sama Besar	Sama Besar
27.gif	-0,015	0,006	5,796438	37,37043		
27.gif	-0,015	0,007	5,796438	37,37043		
29.gif	-0,015	0,005	5,639171	37,45333	Sama Besar	Sama Besar

29.gif	-0,015	0,006	5,639171	37,45333		
29.gif	-0,015	0,007	5,639171	37,45333		
3.gif	-0,015	0,005	6,245375	37,21555	Sama Besar	Sama Besar
3.gif	-0,015	0,006	6,245375	37,21555		
3.gif	-0,015	0,007	6,245375	37,21555		
30.gif	-0,015	0,005	6,390897	37,34669	Sama Besar	Sama Besar
30.gif	-0,015	0,006	6,390897	37,34669		
30.gif	-0,015	0,007	6,390897	37,34669		
31.gif	-0,015	0,005	6,025275	37,43168	Sama Besar	Sama Besar
31.gif	-0,015	0,006	6,025275	37,43168		
31.gif	-0,015	0,007	6,025275	37,43168		
34.gif	-0,015	0,005	6,533492	37,21726	Sama Besar	Sama Besar
34.gif	-0,015	0,006	6,533492	37,21726		
34.gif	-0,015	0,007	6,533492	37,21726		
36.gif	-0,015	0,005	6,062295	37,41703	Sama Besar	Sama Besar
36.gif	-0,015	0,006	6,062295	37,41703		
36.gif	-0,015	0,007	6,062295	37,41703		
37.gif	-0,015	0,005	6,111306	37,11112	Sama Besar	Sama Besar
37.gif	-0,015	0,006	6,111306	37,11112		
37.gif	-0,015	0,007	6,111306	37,11112		
4.gif	-0,015	0,005	6,100591	37,1214	Sama Besar	Sama Besar
4.gif	-0,015	0,006	6,100591	37,1214		
4.gif	-0,015	0,007	6,100591	37,1214		
43.gif	-0,015	0,005	6,564166	37,33892	Sama Besar	Sama Besar
43.gif	-0,015	0,006	6,564166	37,33892		
43.gif	-0,015	0,007	6,564166	37,33892		
6.gif	-0,015	0,005	6,200958	37,22022	Sama Besar	Sama Besar
6.gif	-0,015	0,006	6,200958	37,22022		
6.gif	-0,015	0,007	6,200958	37,22022		
8.gif	-0,015	0,005	5,14575	37,2814	Sama Besar	Sama Besar
8.gif	-0,015	0,006	5,14575	37,2814		
8.gif	-0,015	0,007	5,14575	37,2814		

Pada Tabel 4.3, terlihat bahwa semua nilai  $c_2$  yang diuji memberikan nilai yang sama baik itu nilai *entropy* yang dihasilkan maupun nilai *contrast*.

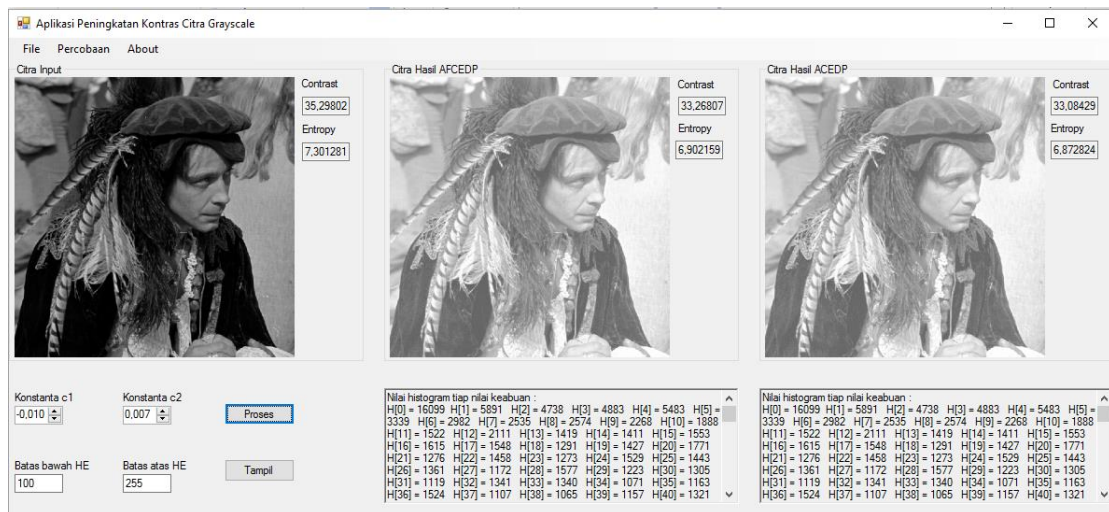
- Menguji hasil citra bila *range* (batas bawah atau batas atas) HE diubah dari *range default* 0-255. Pengujian dilakukan terhadap citra “12.gif” dan hasilnya ditampilkan pada tabel 4.4 berikut.

**Tabel 4.4** Pengujian Batas Bawah dan Batas Atas HE

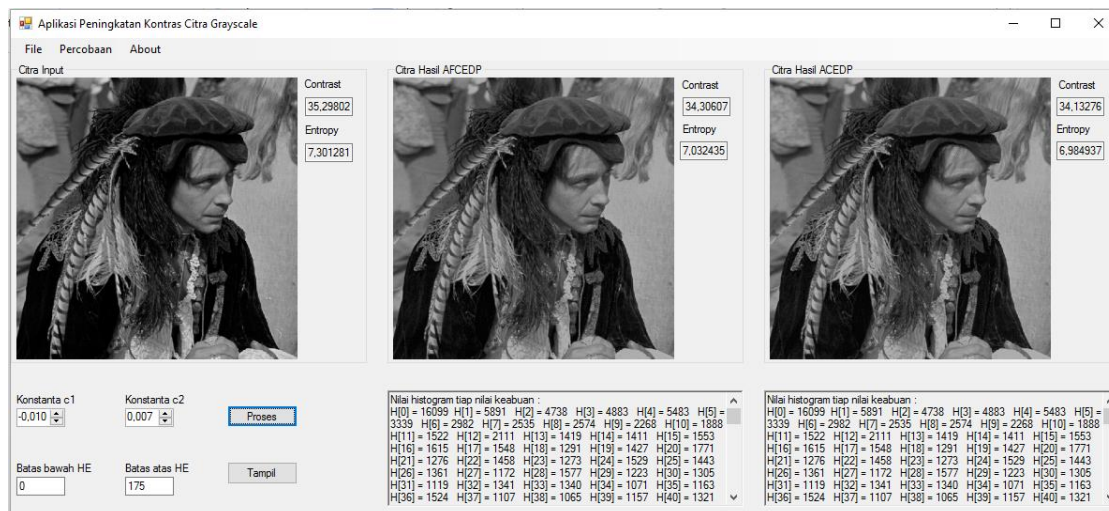
Range (batas bawah, batas atas)	Entropy AFCEDP	Nilai CIE AFCEDP
0,255	7,210436234	37,58413293
<b>(Batas bawah dinaikkan)</b>		
50,255	7,144610973	35,68622497
75,255	7,035796564	34,55620057
100,255	6,902159334	33,2680739
125,255	6,695468216	31,73802283
150,255	6,45992951	29,86972318
175,255	6,166487296	27,53336105
200,255	5,697207692	24,26395102
225,255	4,922618394	19,0578334
255,255	0	-∞
<b>(Batas atas diturunkan)</b>		
0,255	7,161653732	36,50051725
0,200	7,130223897	35,47535638
0,175	7,032435689	34,30607774
0,150	6,848834376	32,98325774
0,125	6,652650459	31,3841769
0,100	6,411615173	29,46588307
0,75	6,071607153	26,95255719
0,50	5,585668101	23,4164322
0,25	4,674021661	17,43754075
0,0	0	-∞

Pada tabel 4.4 memperlihatkan bahwa perubahan nilai batas atas maupun batas bawah HE akan mengurangi nilai *entropy* serta kontras citra yang dihasilkan. Hal ini terbukti dengan turunnya nilai *entropy* dan nilai CIE dari citra hasil saat dinaikkan batas bawah HE maupun diturunkan batas atas HE. Beberapa citra hasil pengujian dengan mengubah nilai batas bawah dan batas atas HE dapat dilihat pada gambar 4.12 dan 4.13.





**Gambar 4.12** Hasil Pengujian Citra dengan Range HE 100 – 255



**Gambar 4.13** Hasil Pengujian Citra dengan Range HE 0 – 175

5. Menguji aplikasi untuk mengetahui algoritma paling optimal antara *Adaptive Fuzzy Contrast Enhancement with Details Preserving* (AFCEDP) atau *Adaptive Contrast Enhancement with Details Preserving* (ACEDP). Pengujian dilakukan dengan nilai  $c1 = -0,015$ , nilai  $c2 = 0,007$ , batas bawah HE = 0, dan batas atas HE = 255 terhadap 49 buah citra. Hasil pengujian dapat dilihat pada tabel 4.4 berikut.

**Tabel 4.5** Pengujian *Entropy* dan *Contrast* untuk AFCEDP dan ACEDP

nama file	c1	c2	entropy AFCEDP	entropy ACEDP	contrast AFCEDP	contrast ACEDP	entropy terbaik	contrast terbaik
1.gif	-0,015	0,007	7,59032	7,59032	37,0893	37,0893	Sama Besar	Sama Besar
10.gif	-0,015	0,007	6,53397	6,53397	36,9885	36,9885	Sama Besar	Sama Besar
11.gif	-0,015	0,007	4,15349	4,15349	37,6326	37,5861	Sama Besar	AFCEDP
12.gif	-0,015	0,007	7,22432	7,21028	37,6065	37,4472	AFCEDP	AFCEDP
13.gif	-0,015	0,007	7,13951	7,06845	37,0695	37,3964	AFCEDP	ACEDP
14.gif	-0,015	0,007	6,69587	6,69587	36,5427	36,5427	Sama Besar	Sama Besar
15.gif	-0,015	0,007	6,39607	6,39703	37,2905	37,3209	ACEDP	ACEDP
16.gif	-0,015	0,007	5,71911	5,6786	37,5389	37,2656	AFCEDP	AFCEDP
17.gif	-0,015	0,007	6,58457	6,59292	37,6391	37,431	ACEDP	AFCEDP
18.gif	-0,015	0,007	6,44731	6,44832	37,4703	37,4454	ACEDP	AFCEDP
19.gif	-0,015	0,007	6,35414	6,35414	37,4354	37,4354	Sama Besar	Sama Besar
2.gif	-0,015	0,007	7,37766	7,37766	36,4798	36,4798	Sama Besar	Sama Besar
20.gif	-0,015	0,007	6,64855	6,64684	37,3426	37,3288	AFCEDP	AFCEDP
21.gif	-0,015	0,007	5,4479	5,43193	37,3874	37,3015	AFCEDP	AFCEDP
22.gif	-0,015	0,007	6,318	6,318	36,1474	36,1474	Sama Besar	Sama Besar
23.gif	-0,015	0,007	5,75707	5,75707	37,3044	37,3044	Sama Besar	Sama Besar
24.gif	-0,015	0,007	5,76516	5,79441	36,5833	35,7426	ACEDP	AFCEDP
25.gif	-0,015	0,007	5,76746	5,76087	37,1095	37,2299	AFCEDP	ACEDP
26.gif	-0,015	0,007	6,18047	6,16043	36,8335	37,2096	AFCEDP	ACEDP
27.gif	-0,015	0,007	5,79644	5,79644	37,3704	37,3704	Sama Besar	Sama Besar
28.gif	-0,015	0,007	6,35371	6,35371	36,8417	36,8417	Sama Besar	Sama Besar
29.gif	-0,015	0,007	5,63917	5,65199	37,4533	37,3852	ACEDP	AFCEDP
3.gif	-0,015	0,007	6,24538	6,24538	37,2155	37,2155	Sama Besar	Sama Besar
30.gif	-0,015	0,007	6,3909	6,3721	37,3467	37,4745	AFCEDP	ACEDP
31.gif	-0,015	0,007	6,02528	6,02205	37,4317	37,3915	AFCEDP	AFCEDP
32.gif	-0,015	0,007	5,9891	5,9671	38,4096	37,3747	AFCEDP	AFCEDP
33.gif	-0,015	0,007	6,59282	6,57194	36,9815	37,2398	AFCEDP	ACEDP
34.gif	-0,015	0,007	6,53349	6,5151	37,2173	37,2953	AFCEDP	ACEDP
35.gif	-0,015	0,007	6,6096	6,6096	37,1786	37,1786	Sama Besar	Sama Besar
36.gif	-0,015	0,007	6,0623	6,0623	37,417	37,417	Sama Besar	Sama Besar
37.gif	-0,015	0,007	6,11131	6,10827	37,1111	37,2026	AFCEDP	ACEDP
38.gif	-0,015	0,007	6,1641	6,1641	37,7015	37,7015	Sama Besar	Sama Besar
39.gif	-0,015	0,007	5,7938	5,76676	38,9743	37,4286	AFCEDP	AFCEDP
4.gif	-0,015	0,007	6,10059	6,10125	37,1214	37,2147	ACEDP	ACEDP
40.gif	-0,015	0,007	6,70988	6,70988	36,1753	36,1753	Sama Besar	Sama Besar
41.gif	-0,015	0,007	6,99844	6,99844	35,8808	35,8808	Sama Besar	Sama Besar
42.gif	-0,015	0,007	6,90545	6,90545	36,1626	36,1626	Sama Besar	Sama Besar

43.gif	-0,015	0,007	6,56417	6,54742	37,3389	37,3375	AFCEDP	AFCEDP
44.gif	-0,015	0,007	3,85927	3,85927	36,2219	36,2219	Sama Besar	Sama Besar
45.gif	-0,015	0,007	7,38661	7,45696	37,6275	36,0897	ACEDP	AFCEDP
46.gif	-0,015	0,007	5,70461	5,70461	36,5074	36,5074	Sama Besar	Sama Besar
47.gif	-0,015	0,007	7,29266	7,29266	36,5619	36,5619	Sama Besar	Sama Besar
48.gif	-0,015	0,007	7,05333	7,05333	36,611	36,611	Sama Besar	Sama Besar
49.gif	-0,015	0,007	6,32704	6,27594	38,216	38,253	AFCEDP	ACEDP
5.gif	-0,015	0,007	4,46089	4,46089	36,3284	36,3284	Sama Besar	Sama Besar
6.gif	-0,015	0,007	6,20096	6,20096	37,2202	37,2202	Sama Besar	Sama Besar
7.gif	-0,015	0,007	4,36911	4,36911	37,2583	37,2276	Sama Besar	AFCEDP
8.gif	-0,015	0,007	5,14575	5,12539	37,2814	37,1407	AFCEDP	AFCEDP
9.gif	-0,015	0,007	6,14533	6,12069	36,5748	37,2851	AFCEDP	ACEDP

Pada tabel 4.4, terlihat bahwa algoritma AFCEDP lebih optimal dibandingkan dengan algoritma ACEDP baik itu dalam hal pelestarian detail citra maupun dalam hal peningkatan kontras citra. Rata-rata nilai *entropy* yang didapatkan AFCEDP yaitu sebesar 6,19658 dan untuk ACEDP sebesar 6,19101. Rata-rata nilai *contrast* yang didapatkan dengan menggunakan AFCEDP sebesar 37,12712 dan untuk ACEDP sebesar 37,04953.

## **BAB V**

### **KESIMPULAN DAN SARAN**

#### **5.1 Kesimpulan**

Setelah menyelesaikan pengembangan aplikasi peningkatan kontras citra dengan *Adaptive Fuzzy Contrast Enhancement with Details Preserving*, dapat ditarik kesimpulan sebagai berikut:

1. Algoritma *Adaptive Fuzzy Contrast Enhancement with Details Preserving* dapat digunakan untuk meningkatkan kontras citra *grayscale* dan melestarikan detailnya dengan nilai  $c1$  yang paling optimal adalah -0,015 dan nilai  $c2$  berupa semua nilai yang berkisar 0,005 hingga 0,007.
2. Perubahan nilai batas atas dan batas bawah HE mampu mengurangi tingkat kontras dan mengurangi detail yang terkandung pada citra.
3. Algoritma *Adaptive Fuzzy Contrast Enhancement with Details Preserving* (AFCEDP) lebih unggul dibandingkan algoritma *Adaptive Contrast Enhancement with Details Preserving* (ACEDEP) baik dalam hal pelestarian detail citra maupun peningkatan kontras citra.

#### **3.2 Saran**

Adapun beberapa saran yang dapat diberikan dan mungkin akan membantu dalam pengembangan aplikasi ini lebih lanjut adalah:

1. Aplikasi dapat dikembangkan dengan menggunakan fungsi keanggotaan yang lain, dimana fungsi keanggotaan tersebut digunakan untuk mengelompokkan himpunan *fuzzy* pada algoritma seperti fungsi keanggotaan dengan representasi kurva segitiga.
2. Aplikasi dapat dikembangkan sehingga dapat menerima dan memproses *input* citra berupa citra warna 16 bit, 24 bit, 32 bit ataupun citra transparan.
3. Perlu dilakukan penelitian tentang pengaruh algoritma tersebut terhadap faktor pencahayaan ( *brightness* ) dari citra hasil.

## DAFTAR PUSTAKA

- Gonzalez, R. C. & Woods, R. E., 2002. *Digital Image Processing*. 2nd ed. New Jersey: Prentice-Hall.
- Group, C. V., 2002. *CVG-UGR-Database*. [Online], tersedia pada <http://decsai.ugr.es/cvg/CG/base.htm>, tanggal akses 14 Juli 2016.
- Kadir, A. & Susanto, A., 2013. *Teori dan Aplikasi Pengolahan Citra*. Yogyakarta: Penerbit Andi.
- Kusumadewi, S. & Purnomo, H., 2004. *Aplikasi Logika Fuzzy Untuk Pendukung Keputusan*. 1st ed. Yogyakarta: Penerbit Graha Ilmu.
- Munir, R., 2004. *Pengolahan Citra Digital*. Bandung: Penertit Informatika.
- Ooi, C. H., Ibrahim, H. & Sia, N. P. K., 2009. IEEE Transactions on Consumer Electronics. *Bi-Histogram Equalization with a Plateau Limit for Digital Image Enhancement*, Volume 55, pp. 2072-2080.
- Putra, D., 2010. *Pengolahan Citra Digital*. Yogyakarta: Penerbit Andi.
- Shannon, C. E., 1948. The Bell System Technical Journal. *A Mathematical Theory of Communication*, Volume 27, pp. 379 - 423, 623 - 656.
- Sutoyo, T. et al., 2009. *Teori Pengolahan Citra Digital*. Semarang: Penerbit Andi.
- Tang, J. R. & Mat Isa, N. A., 2014. J. ICT Res. Appl.. *An Adaptive Fuzzy Contrast Enhancement Algorithm with Details Preserving*, Volume 8, pp. 126-140.
- Tang, J. R. & Mat Isa, N. A., 2014. Proceeding of International Conference on Electrical Engineering, Computer Science and Informatics (EECSI 2014). *An Adaptive Contrast Enhancement Algorithm with Details Preserving*.
- Tanner, 2011. *Seven grayscale conversion algorithms (with pseudocode and VB6 source code)*. Tersedia pada <http://www.tannerhelland.com>, tanggal akses 28 Juni 2016.
- Zhu, Y. & Huang, C., 2012. An Adaptive Histogram Equalization Algorithm on the Image Gray Level Mapping. *Physics Procedia* 25, pp. 601 - 608.

## LISTING PROGRAM

### FORM Menu Utama

```

using System;
using
System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Drawing.Imaging;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace ProgramTA
{
    public partial class Home
    : Form
    {
        string filename = "";
        string asfilename =
"File";
        double c1;
        double c2;
        int[] f1,f2;
        int HElowbound;
        int HEhibound;
        int[] H;
        int[] newH2;
        int[] newH1;
        double[] low, mid,
high;
        double[] pdf;
        int totallow,
totalmid, totalhigh;
        public Home()
        {
            InitializeComponent();
            low = new
double[256];
            mid = new
double[256];
            high = new
double[256];
            for(int i = 0; i <
low.Length; i++)
            {
                if (i > 95)
                {
                    low[i] =
0;
                }
            }
            else if (i <
75)
            {
                low[i] =
1;
            }
            else
            {
                low[i] =
(95 - i) / 20.00;
                mid[i] =
(i - 75) / 20.00;
            }
            if (i < 75 ||
i > 180)
            {
                mid[i] =
0;
            }
            else if (i >=
95 && i <= 160)
            {
                mid[i] =
1;
            }
            if (i < 160)
            {
                high[i] =
0;
            }
            else if (i >
180)
            {
                high[i] =
1;
            }
            else
            {
                mid[i] =
(180 - i) / 20.00;
                high[i] =
(i - 160) / 20.00;
            }
        }
        public double
means(double[] p)
        {
            double rerata = 0;
            for (int i = 0; i
< p.Length; i++)
            {

```

```

        if (p[i] > 0)
            rerata++;
    }
    return
    (double)p.Sum() /
    (double)rerata;
}
private void
openToolStripMenuItem_Click(object sender, EventArgs e)
{
    openFileDialog1.FileName =
    "File";

    openFileDialog1.Filter =
    "Image Files(*.jpg; *.jpeg;
    *.tif; *.tiff; *.bmp; *.png;
    *.gif)|*.jpg; *.jpeg; *.tif;
    *.tiff; *.bmp; *.png; *.gif;";
    if
    (openFileDialog1.ShowDialog()
    == DialogResult.OK ||
    openFileDialog1.FileName !=
    "File")
    {
        button1.Enabled = true;
        totalallow = 0;
        totalmid = 0; totalhigh = 0;

        pictureBox1.Image = new
        Bitmap(openFileDialog1.FileName);
        Bitmap test =
        new
        Bitmap(openFileDialog1.FileName);
        filename =
        openFileDialog1.FileName;
        long totalgr1
        = 0; long totalgr2 = 0;
        H = new
        int[256];
        pdf = new
        double[256];
        double
        Contrast = 0;
        Bitmap bmp =
        new Bitmap(pictureBox1.Image);
        for (int x =
        0; x < bmp.Width; x++)
        {
            for (int y
            = 0; y < bmp.Height; y++)
            {
                if (p[i] > 0)
                {
                    Color
                    c = bmp.GetPixel(x, y);
                    int
                    grayscale = (int)((c.R * 0.3f)
                    + (c.G * 0.59f) + (c.B *
                    0.11f));
                    Color
                    g = Color.FromArgb(grayscale,
                    grayscale, grayscale);
                    bmp.SetPixel(x, y, g);

                    H[grayscale]++;

                    totalgr1 += (grayscale *
                    grayscale);
                    totalgr2 += grayscale;
                    if
                    (grayscale < 85)
                    totalallow++;
                    else
                    if (grayscale > 170)
                    totalhigh++;
                    else
                    totalmid++;
                }
            }
        }
        Contrast =
        (double)10f *
        Math.Log10(((double)(totalgr1 / (double)bmp.Width
        / (double)bmp.Height) -
        Math.Pow((double)totalgr2 /
        (double)bmp.Width /
        (double)bmp.Height, 2)));
        double Entropy
        = 0;
        for(int i = 0;
        i < H.Length; i++)
        {
            pdf[i] =
            (double)H[i] /
            (double)H.Sum();
            if (pdf[i]
            > 0)
            Entropy += ((double)(-1) *
            (double)pdf[i] *
            (double)Math.Log(pdf[i], 2));
        }
        pictureBox1.Image = bmp;
    }
}

```

```

        textBox3.Text
= Entropy.ToString();
        textBox6.Text
= Contrast.ToString();
    }
}

private void
saveToolStripMenuItem_Click(ob
ject sender, EventArgs e)
{
    Bitmap bmp = new
Bitmap(pictureBox2.Image);

bmp.Save(filename);
}

private void
saveAsToolStripMenuItem_Click(
object sender, EventArgs e)
{
    saveFileDialog1.FileName =
asfilename;

saveFileDialog1.Filter =
"Bitmap Image (.bmp)|*.bmp|Gif
Image (.gif)|*.gif|JPEG Image
(.jpeg)|*.jpeg|Png Image
(.png)|*.png|Tiff Image
(.tiff)|*.tiff";
    if
(saveFileDialog1.ShowDialog()
== DialogResult.OK)
    {
        pictureBox2.Image.Save(saveFil
eDialog1.FileName);
        asfilename =
saveFileDialog1.FileName;
    }
}

private void
exitToolStripMenuItem_Click(ob
ject sender, EventArgs e)
{
    Application.Exit();
}

private void
percobaanToolStripMenuItem_Cli
ck(object sender, EventArgs e)
{

```

```

        Percobaan
percobaan = new Percobaan(low,
mid, high);

percobaan.ShowDialog();
}

private void
aboutToolStripMenuItem_Click(o
bject sender, EventArgs e)
{
}

private double
percen(double current,double
max)
{
    return current /
max * 100;
}

private void
button1_Click(object sender,
EventArgs e)
{
    if
(Convert.ToInt32(textBox1.Text
) >= 0 &&
Convert.ToInt32(textBox1.Text)
<= 255 &&
Convert.ToInt32(textBox2.Text)
>= 0 &&
Convert.ToInt32(textBox2.Text)
<= 255 &&
Convert.ToInt32(textBox1.Text)
<=
Convert.ToInt32(textBox2.Text)
)
    {
        Cursor.Current
= Cursors.WaitCursor;

Application.DoEvents();

progressBar1.Visible = true;

lbl_persen.Visible = true;

progressBar1.Maximum =
H.Length * 12;

progressBar1.Value = 1;

lbl_persen.Text =

```



```

percen(progressBar1.Value,
progressBar1.Maximum).ToString
() + " %";
lbl_persen.Refresh();

progressBar1.Step = 1;

saveToolStripMenuItem.Enabled
= true;

saveAsToolStripMenuItem.Enable
d = true;

richTextBox1.Text = "";
richTextBox2.Text = "";
c1 =
(double)numericUpDown1.Value;
c2 =
(double)numericUpDown2.Value;
HElowbound =
Int32.Parse(textBox1.Text);
HEhibound =
Int32.Parse(textBox2.Text);
Bitmap bmp1 =
new Bitmap(pictureBox1.Image);
Bitmap bmp2 =
new Bitmap(pictureBox1.Image);
long totalgr1
= 0; long totalgr2 = 0;
double tlow =
0, tmid = 0, thigh = 0;
int intref;
int deflow =
43; int defmid = 128; int
defhigh = 213;
double
newEntropy1 = 0; double
newEntropy2 = 0; double
newContrast1 = 0; double
newContrast2 = 0;

richTextBox1.Text += "Nilai
histogram tiap nilai keabuan
:\n";

for (int i =
0; i < H.Length; i++)
{

progressBar1.PerformStep();
progressBar1.Refresh();

lbl_persen.Text =
percen(progressBar1.Value,
progressBar1.Maximum).ToString
("F2") + " %";
progressBar1.Refresh();

richTextBox1.Text += "Pdf[" +
i + "]" = " +
pdf[i].ToString("0.000").PadRi
ght(6);
}

richTextBox2.Text =
richTextBox1.Text;

richTextBox1.Text +=
"\n\nDerajat keanggotaan untuk
tiap nilai keabuan (μ) :\n";
for (int i =
0; i < H.Length; i++)
{

progressBar1.PerformStep();
progressBar1.Refresh();

lbl_persen.Text =
percen(progressBar1.Value,
progressBar1.Maximum).ToString
("F2") + " %";
progressBar1.Refresh();

richTextBox1.Text += "μLow[" +
i + "]" = " +
low[i].ToString().PadRight(10)
+ "μMid[" + i + "]" = " +
mid[i].ToString().PadRight(10)
+ "\t" + "μHigh[" + i + "]" = "

```

```

+
high[i].ToString().PadRight(10
) + "\n";
                tlow +=
low[i] * H[i]; tmid += mid[i]
* H[i]; thigh += high[i] *
H[i];
                }

richTextBox1.Text +=
"\nLow_part = " + (tlow /
(tlow + tmid +
thigh)).ToString("0.000") +
"\tMid_part = " + (tmid /
(tlow + tmid +
thigh)).ToString("0.000") +
"\tHigh_Part = " + (thigh /
(tlow + tmid +
thigh)).ToString("0.000") +
"\nNilai intensitas referensi
citra \n= low_part * 43 +
mid_part * 128 + high_part *
213\n=";

richTextBox1.Text += (tlow /
(tlow + tmid +
thigh)).ToString("0.000") + "
* 43 + " + (tmid / (tlow +
tmid +
thigh)).ToString("0.000") + "
* 128 + " + (thigh / (tlow +
tmid +
thigh)).ToString("0.000") + "
* 213 = ";
                intref =
Convert.ToInt32(Math.Round((tl
ow / (tlow + tmid + thigh) *
deflow) + (tmid / (tlow + tmid
+ thigh) * defmid) + (thigh /
(tlow + tmid + thigh) *
defhigh)));

richTextBox1.Text += intref +
"\n\n";
                newH1 = new
int[256]; double[] newHpdf1 =
new double[256];
                double[] npdf1
= new double[256];
                double[] ncdf1
= new double[256];
                double lvlow,
lvmid, lvhigh;
                double
cliplimit = 0;

                f1 = new
int[256];
                lvlow = 0;
lvmid = 0; lvhigh = 0;

richTextBox1.Text +=
"Perhitungan nilai clipping
limit untuk tiap intensitas
:\n";
                for (int i =
0; i < H.Length; i++)
                {
progressBar1.PerformStep();
progressBar1.Refresh();

                lbl_persen.Text =
percen(progressBar1.Value,
progressBar1.Maximum).ToString
("F2") + " %";
                lbl_persen.Refresh();
                if (i ==
0)
                {
                        lvlow
= (c1) + pdf.Max();
                        lvhigh
= (c2) + means(pdf);
                        lvmid
= means(pdf);

                cliplimit = (low[intref] *
lvlow) + (mid[intref] * lvmid)
+ (high[intref] * lvhigh);

                richTextBox1.Text += "Levello
w = c1 + Max(Pdf)\n\t=" + c1 + "
+ " +
pdf.Max().ToString("0.000") +
" = " +
lvlow.ToString("0.00000") +
"\n";

                richTextBox1.Text += "Levelmi
d = Mean(Pdf)\n\t=" +
lvmid.ToString("0.00000") +
"\n";

                richTextBox1.Text +=
"Levelhigh = c2 +
Mean(Pdf)\n\t=" + c1 + " + " +
lvmid.ToString("0.000") + " =
" + lvhigh.ToString("0.00000")
+ "\n";

                richTextBox1.Text += "Clipping

```

```

Limit (CL) = (μLow[" + intref
+ "]" * Levellow) + (μMid[" +
intref + "]" * Levelmid) +
(μHigh[" + intref + "]" *
Levelhigh\n";

richTextBox1.Text += "CL = ("
+ low[intref] + " * " +
lvlow.ToString("0.00000") + "
+ (" + mid[intref] + " * " +
lvmid.ToString("0.00000") + "
+ (" + high[intref] + " * " +
lvhigh.ToString("0.00000") +
") = " +
cliplimit.ToString("0.00000")
+ "\n\n";

richTextBox1.Text += "Proses
Clipping Histogram :\n";
    }

richTextBox1.Text += "Pdf[" +
i + "]" = " +
pdf[i].ToString("0.00000") +
"\n";
        if (pdf[i]
> cliplimit)
        {

richTextBox1.Text += "Pdf[" +
i + "]" > CL dimana " +
pdf[i].ToString("0.00000") + "
> " +
cliplimit.ToString("0.00000")
+ ", Maka nPdf[" + i + "]" = "
+
cliplimit.ToString("0.00000")
+ "\n";

npdf1[i] = cliplimit;
        }
        else
        {

richTextBox1.Text += "Pdf[" +
i + "]" < CL dimana " +
pdf[i].ToString("0.00000") + "
< " +
cliplimit.ToString("0.00000")
+ ", Maka nPdf[" + i + "]" = "
+ pdf[i].ToString("0.00000") +
"\n";

npdf1[i] = pdf[i];
        }
        if (i > 0)

{
ncdf1[i] = ncdf1[i - 1] +
npdf1[i];

richTextBox1.Text += "nCdf[" +
i + "]" = nCdf[" + (i - 1) + "]"
+ nPdf[" + i + "]" = " +
ncdf1[i].ToString("0.00000") +
"\n\n";
        }
        else
        {

ncdf1[i] = npdf1[i];

richTextBox1.Text += "nCdf[" +
i + "]" = nPdf[0] = " +
ncdf1[i].ToString("0.00000") +
"\n\n";
        }
    }

richTextBox1.Text += "\nUntuk
mendapatkan nilai kumulatif
terakhir nCdf[255] = 1, Maka
:\n";
        for (int i =
0; i < H.Length; i++)
        {

progressBar1.PerformStep();
progressBar1.Refresh();

lbl_persen.Text =
persen(progressBar1.Value,
progressBar1.Maximum).ToString
("F2") + " %";
lbl_persen.Refresh();

richTextBox1.Text += "nPdf[" +
i + "]" = " +
npdf1[i].ToString("0.00000") +
"/" +
ncdf1[255].ToString("0.00000")
+ " = ";

npdf1[i]
/= ncdf1[255];

richTextBox1.Text +=
npdf1[i].ToString("0.00000") +
"\n";
        if (i > 0)
        {

```

```

ncdf1[i] = ncdf1[i - 1] +
npdf1[i];

richTextBox1.Text += "nCdf[" +
i + "] = nCdf[" + (i - 1) + "]
+ nPdf[" + i + "] = " +
ncdf1[i].ToString("0.00000") +
"\n\n";
    }
    else
    {

ncdf1[i] = npdf1[i];

richTextBox1.Text += "nCdf[" +
i + "] = nPdf[0] = " +
ncdf1[i].ToString("0.00000") +
"\n\n";
        }
        f1[i] =
Convert.ToInt32(Math.Round(HEl
owbound + ((HEhibound -
HElowbound) * (ncdf1[i] -
(npdf1[i] / 2)))));
    }

richTextBox1.Text +=
"\nPerataan histogram dengan
fungsi transformasi dilakukan
untuk mendapatkan nilai
intensitas baru:\n";

richTextBox1.Text += "Nilai
X(0) = " + HElowbound + " dan
Nilai X(L-1) = " + HEhibound +
"\n";
        for (int i =
0; i < H.Length; i++)
        {

progressBar1.PerformStep();
progressBar1.Refresh();

lbl_persen.Text =
percen(progressBar1.Value,
progressBar1.Maximum).ToString
("F2") + " %";
lbl_persen.Refresh();

richTextBox1.Text += "new_f["
+ i + "] = X(0) + (X(L-1) -
X(0)) x (nCdf[" + i + "] - 1/2
nPdf[" + i + "]) = " + f1[i] +
"\n";

richTextBox1.Text += "Nilai
intensitas " + i +
"ditransformasi menjadi
intensitas baru " + f1[i] +
"\n";
        }
        for (int x =
0; x < bmp1.Width; x++)
        {
            for (int y
= 0; y < bmp1.Height; y++)
            {
                Color
c = bmp1.GetPixel(x, y);
                int gr
= f1[(int)c.R];
                Color
g = Color.FromArgb(gr, gr,
gr);
                bmp1.SetPixel(x, y, g);

newH1[gr]++;

totalgr1 += (gr * gr);

totalgr2 += gr;
            }
        }

richTextBox1.Text +=
"\nSehingga tiap pixel dari
citra awal ditransformasi
membentuk citra baru dengan
nilai Histogram:\n";
        for (int i =
0; i < H.Length; i++)
        {

progressBar1.PerformStep();
progressBar1.Refresh();

lbl_persen.Text =
percen(progressBar1.Value,
progressBar1.Maximum).ToString
("F2") + " %";
lbl_persen.Refresh();

richTextBox1.Text += "newH["
+ i + "] = " + newH1[i] + "\n";
        }
        newContrast1 =
(double)10f *
Math.Log10((double)((double)to
talgr1 / (double)(bmp1.Width *

```

```

bmp1.Height)) -
Math.Pow((double)totalgr2 /
(double)(bmp1.Width *
bmp1.Height), 2));

pictureBox2.Image = bmp1;
        newH2 = new
int[256]; double[] newHpdf2 =
new double[256];
        double[] npdf2
= new double[256];
        double[] ncdf2
= new double[256];
        f2 = new
int[256];
        cliplimit = 0;

richTextBox2.Text +=
"\n\nPerhitungan nilai
clipping limit untuk tiap
intensitas :\n";
        for (int i =
0; i < H.Length; i++)
        {

progressBar1.PerformStep();
progressBar1.Refresh();

lbl_persen.Text =
percen(progressBar1.Value,
progressBar1.Maximum).ToString
("F2") + " %";
lbl_persen.Refresh();
        if (i ==
0)
        {
            if
(totallow > totalmid &&
totalhigh > totalhigh)
            {

cliplimit = (c1) + pdf.Max();

richTextBox2.Text += "Clipping
Limit (CL) = c1 +
Max(Pdf)\n\t=" + c1 + " + " +
pdf.Max().ToString("0.00000")
+ " = " +
cliplimit.ToString("0.00000")
+ "\n";
            }
            else
if (totalhigh > totallow &&
totalhigh > totalmid)
            {
                cliplimit = (c2) + means(pdf);

richTextBox2.Text += "Clipping
Limit (CL) = c2 +
Mean(Pdf)\n\t=" + c2 + " + " +
means(pdf).ToString("0.00000")
+ " = " +
cliplimit.ToString("0.00000")
+ "\n";
            }
            else
            {
                cliplimit = means(pdf);

richTextBox2.Text += "Clipping
Limit (CL) = Mean(Pdf)\n\t=" +
cliplimit + "\n";
            }

richTextBox1.Text += "Pdf[" +
i + "] = " +
pdf[i].ToString("0.00000") +
"\n";
            if (pdf[i]
> cliplimit)
            {

richTextBox2.Text += "Pdf[" +
i + "] > CL dimana " +
pdf[i].ToString("0.00000") + "
> " +
cliplimit.ToString("0.00000")
+ ", Maka nPdf[" + i + "] = "
+
cliplimit.ToString("0.00000")
+ "\n";

npdf2[i] = cliplimit;
            }
            else
            {

richTextBox2.Text += "Pdf[" +
i + "] < CL dimana " +
pdf[i].ToString("0.00000") + "
< " +
cliplimit.ToString("0.00000")
+ ", Maka nPdf[" + i + "] = "
+ pdf[i].ToString("0.00000") +
"\n";

npdf2[i] = pdf[i];
            }

```

```

        if (i > 0)
        {
            ncdf2[i] = ncdf2[i - 1] +
            npdf2[i];

            richTextBox2.Text += "nCdf[" +
            i + "] = nCdf[" + (i - 1) + "]
            + nPdf[" + i + "] = " +
            ncdf2[i].ToString("0.00000") +
            "\n\n";
        }
        else
        {
            ncdf2[i] = npdf2[i];

            richTextBox2.Text += "nCdf[" +
            i + "] = nPdf[0] = " +
            ncdf2[i].ToString("0.00000") +
            "\n\n";
        }
    }

    richTextBox2.Text += "\nUntuk
mendapatkan nilai kumulatif
terakhir nCdf[255] = 1, Maka
:\n";

    for (int i =
0; i < H.Length; i++)
    {

        progressBar1.PerformStep();
        progressBar1.Refresh();

        lbl_persen.Text =
        persen(progressBar1.Value,
        progressBar1.Maximum).ToString
        ("F2") + " %";
        lbl_persen.Refresh();

        richTextBox2.Text += "nPdf[" +
        i + "] = " +
        npdf2[i].ToString("0.00000") +
        "/" +
        ncdf2[255].ToString("0.00000")
        + " = ";

        npdf2[i]
        /= ncdf2[255];

        richTextBox2.Text +=
        npdf2[i].ToString("0.00000") +
        "\n";

        if (i > 0)
        {
            ncdf2[i] = ncdf2[i - 1] +
            npdf2[i];

            richTextBox2.Text += "nCdf[" +
            i + "] = nCdf[" + (i - 1) + "]
            + nPdf[" + i + "] = " +
            ncdf2[i].ToString("0.00000") +
            "\n\n";
        }
        else
        {
            ncdf2[i] = npdf2[i];

            richTextBox2.Text += "nCdf[" +
            i + "] = nPdf[0] = " +
            ncdf2[i].ToString("0.00000") +
            "\n\n";
        }
        f2[i] =
        Convert.ToInt32(Math.Round(HEl
        owbound + ((HEhibound -
        HElowbound) * (ncdf2[i] -
        (npdf2[i] / 2)))));
    }

    richTextBox2.Text +=
    "\nPerataan histogram dengan
    fungsi transformasi dilakukan
    untuk mendapatkan nilai
    intensitas baru:\n";

    richTextBox2.Text += "Nilai
    X(0) = " + HElowbound + " dan
    Nilai X(L-1) = " + HEhibound +
    "\n";

    for (int i =
    0; i < H.Length; i++)
    {

        progressBar1.PerformStep();
        progressBar1.Refresh();

        lbl_persen.Text =
        persen(progressBar1.Value,
        progressBar1.Maximum).ToString
        ("F2") + " %";
        lbl_persen.Refresh();

        richTextBox2.Text += "new_f["
        + i + "] = X(0) + (X(L-1) -
        X(0)) x (nCdf[" + i + "] - 1/2
        nPdf[" + i + "]) = " + f2[i] +
        "\n";
    }

```

```

richTextBox2.Text += "Nilai
intensitas " + i +
"ditransformasi menjadi
intensitas baru " + f2[i] +
"\n";
    }
    totalgr1 = 0;
totalgr2 = 0;
    for (int x =
0; x < bmp2.Width; x++)
    {
        for (int y
= 0; y < bmp2.Height; y++)
        {
            Color
c = bmp2.GetPixel(x, y);
            int gr
= f2[(int)c.R];
            Color
g = Color.FromArgb(gr, gr,
gr);
            bmp2.SetPixel(x, y, g);

newH2[gr]++;

totalgr1 += (gr * gr);

totalgr2 += gr;
        }
    }

richTextBox2.Text +=
"\nSehingga tiap pixel dari
citra awal ditransformasi
membentuk citra baru dengan
nilai Histogram:\n";
    for (int i =
0; i < H.Length; i++)
    {

progressBar1.PerformStep();
progressBar1.Refresh();

lbl_persen.Text =
percen(progressBar1.Value,
progressBar1.Maximum).ToString
("F2") + " %";
lbl_persen.Refresh();

richTextBox2.Text += "newH[" +
i + "]" = " + newH2[i] + "\n";
    }
    newContrast2 =
(double)10f *

Math.Log10((double)((double)to
talgr1 / (double)(bmp2.Width *
bmp2.Height)) -
Math.Pow((double)totalgr2 /
(double)(bmp2.Width *
bmp2.Height), 2));

pictureBox3.Image = bmp2;
    for (int i =
0; i < H.Length; i++)
    {

progressBar1.PerformStep();
progressBar1.Refresh();

lbl_persen.Text =
percen(progressBar1.Value,
progressBar1.Maximum).ToString
() + " %";
lbl_persen.Refresh();

newHpdf1[i] = (double)newH1[i]
/ (double)newH1.Sum();

newHpdf2[i] = (double)newH2[i]
/ (double)newH2.Sum();
        if
(newHpdf1[i] > 0)

newEntropy1 += ((double)(-1) *
(double)newHpdf1[i] *
(double)Math.Log(newHpdf1[i],
2));
        if
(newHpdf2[i] > 0)

newEntropy2 += ((double)(-1) *
(double)newHpdf2[i] *
(double)Math.Log(newHpdf2[i],
2));
    }
    textBox4.Text
= newEntropy1.ToString();
    textBox5.Text
= newEntropy2.ToString();
    textBox7.Text
= newContrast1.ToString();
    textBox8.Text
= newContrast2.ToString();

btn_Show.Enabled = true;
    Cursor.Current
= Cursors.Default;

progressBar1.Visible = false;

```

```

lbl_persen.Visible = false;

this.UseWaitCursor = false;
    }
    else

MessageBox.Show("Batas bawah
dan atas HE harus memiliki
nilai didalam range 0 hingga
255\nBatas bawah HE tidak
boleh lebih besar dari batas
atas HE");
    }

    private void
pictureBox1_Click(object
sender, EventArgs e)
    {
        if
        (pictureBox1.Image != null)
        {
            Tampil_Citra

            tmp1 = new
            Tampil_Citra((Bitmap)pictureBo
            x1.Image, "Citra Asli");

            tmp1.ShowDialog();
        }
    }

    private void
Home_Load(object sender,
EventArgs e)
    {
        button1.Enabled =
false;
    }

    private void
pictureBox2_Click(object
sender, EventArgs e)
    {
        if
        (pictureBox2.Image != null)
        {
            Tampil_Citra

            tmp1 = new
            Tampil_Citra((Bitmap)pictureBo
            x2.Image, "Citra AFCEDP");

            tmp1.ShowDialog();
        }
    }

```

```

        private void
pictureBox3_Click(object
sender, EventArgs e)
        {
            if
            (pictureBox3.Image != null)
            {
                Tampil_Citra

                tmp1 = new
                Tampil_Citra((Bitmap)pictureBo
                x3.Image, "Citra ACEDP");

                tmp1.ShowDialog();
            }
        }

        private void
btn_Show_Click(object sender,
EventArgs e)
        {
            Log log = new
            Log(H,newH1,newH2,f1,f2);
            log.ShowDialog();
        }

        private void
aboutToolStripMenuItem1_Click(
object sender, EventArgs e)
        {
            About_Us abu = new
            About_Us();
            abu.ShowDialog();
        }
    }
}

```

#### FORM LOG

```

using System;
using
System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace ProgramTA
{
    public partial class Log :
    Form
    {
        public class data

```



```

        {
            public int
Pixels_Citra_Awal { set; get;
}
            public int
JumlahKemunculan { set; get; }
            public int
Pixel_Citra_AFCEDP { set; get;
}
            public int
Jumlah_kemunculan { set; get;
}
            public int
Pixel_Citra_ACEDP { set; get;
}
            public int
jumlah_kemunculan_ { set; get;
}
            public data(int
pixels,int h1,int f1,int
h2,int f2,int h3)
            {

Pixels_Citra_Awal = pixels;

JumlahKemunculan = h1;

Pixel_Citra_AFCEDP = f1;

Jumlah_kemunculan = h2;

Pixel_Citra_ACEDP = f2;

jumlah_kemunculan_ = h3;
            }
            List<data>
datatoShow1;
            int[] h1;
            int[] h2;
            int[] h3;
            int[] f1;
            int[] f2;
            public Log(int[] H,
int[] H2, int[] H3,int[]
F1,int[] F2)
            {
                this.h1 = H;
                this.h2 = H2;
                this.h3 = H3;
                f1 = F1; f2 = F2;
                datatoShow1 = new
List<data>();

InitializeComponent();
            }

            private void
Log_Load(object sender,
EventArgs e)
            {

dataGridView1.Visible = false;
//dataGridView2.Visible =
false; dataGridView3.Visible =
false;
            }

            private void
button1_Click(object sender,
EventArgs e)
            {

dataGridView1.Visible = true;
                for (int i = 0; i
< h1.Length; i++)
                {

datatoShow1.Add(new
data(i,h1[i],f1[i],h2[f1[i]],f
2[i],h3[f2[i]]));
                }
                var bind1 = new
Library.Forms.SortableBindingL
ist<data>(datatoShow1);
                //var bind = new
BindingList<data>(datatoShow);

dataGridView1.DataSource =
bind1;
                for (int i = 0; i
< dataGridView1.Columns.Count;
i++)
                {

dataGridView1.Columns[i].AutoS
izeMode =
DataGridViewAutoSizeColumnMode
.DisplayedCells;
                }
                button1.Enabled =
false;
            }

            private void
dataGridView1_ColumnHeaderMous
eClick(object sender,
DataGridViewCellEventArgs
e)
            {

```

```

        DataGridView dtg =
sender as DataGridView;
        DataGridViewColumn
newColumn =
dtg.Columns[e.ColumnIndex];
        DataGridViewColumn
oldColumn = dtg.SortedColumn;
        ListSortDirection
direction;

        // If oldColumn is
null, then the DataGridView is
not sorted.
        if (oldColumn !=
null)
        {
            // Sort the
same column again, reversing
the SortOrder.
            if (oldColumn
== newColumn &&
dtg.SortOrder ==
SortOrder.Ascending)
            {
                direction
=
ListSortDirection.Descending;
            }
            else
            {
                // Sort a
new column and remove the old
SortGlyph.
                direction
= ListSortDirection.Ascending;

oldColumn.HeaderCell.SortGlyph
Direction = SortOrder.None;
            }
            else
            {
                direction =
ListSortDirection.Ascending;
            }

            // Sort the
selected column.

dataGridView1.Sort(newColumn,
direction);
            //if(dtg ==
dataGridView1)
            //{

```

```

//
dataGridView1.Sort(newColumn,
direction);
        //}
        //else if(dtg ==
dataGridView2)
        //{
        //
dataGridView2.Sort(newColumn,
direction);
        //}
        //else
        //{
        //
dataGridView3.Sort(newColumn,
direction);
        //}

newColumn.HeaderCell.SortGlyph
Direction =
        direction ==
ListSortDirection.Ascending ?
SortOrder.Ascending :
SortOrder.Descending;

//this.dataGridView1.Sort(this
.dataGridView1.Columns[dtg_Col
umn.Name],
ListSortDirection.Ascending)
    }

    private void
dataGridView1_DataBindingCompl
ete(object sender,

DataGridViewBindingCompleteEve
ntArgs e)
    {
        // Put each of the
columns into programmatic sort
mode.

        foreach
(DataGridViewColumn column in
dataGridView1.Columns)
        {

column.SortMode =
DataGridViewColumnSortMode.Pro
grammatic;
        }
    }
}

```

**FORM PERCOBAAN**

```
using System;
using System.IO;
using
System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace ProgramTA
{
    public partial class
Percobaan : Form
    {
        string folderpath =
"";
        double[] low, mid,
high;
        List<CitraCoba> coba;
        double conrerata,
conrerata1, conrerata2,
enrerata, enrerata1,
enrerata2;
        public
Percobaan(double[] low,
double[]mid,double[]high)
        {

InitializeComponent();

folderBrowserDialog1.RootFolde
r =
Environment.SpecialFolder.Desk
top;
            this.low = low;
this.mid = mid; this.high =
high;
        }

        private void
button1_Click(object sender,
EventArgs e)
        {
            button2.Enabled =
true;
            if (folderpath !=
"")
                folderBrowserDialog1.SelectedP
ath = folderpath;
```

```
        if
(folderBrowserDialog1.ShowDial
og() == DialogResult.OK)
        {
            folderpath =
folderBrowserDialog1.SelectedP
ath;
            textBox1.Text
= folderpath;
            coba = new
List<CitraCoba>();
        }

        private void
button2_Click(object sender,
EventArgs e)
        {
            coba.Clear();
            Cursor.Current =
Cursors.WaitCursor;

Application.DoEvents();
            label11.Visible =
true;

progressBar1.Visible = true;
            if
(checkBox1.Checked &&
checkBox2.Checked)
            {

progressBar1.Maximum =
System.IO.Directory.GetFiles(f
olderpath).Length * 33;
            }
            else if
(checkBox1.Checked)
            {

progressBar1.Maximum =
System.IO.Directory.GetFiles(f
olderpath).Length * 11;
            }
            else
            if (checkBox2.Checked)
            {

progressBar1.Maximum =
System.IO.Directory.GetFiles(f
olderpath).Length * 3;
            }
            else
            {

progressBar1.Maximum =
```

```

System.IO.Directory.GetFiles(f
olderpath).Length;
    }
    progressBar1.Value
= 1; progressBar1.Step = 1;
    label11.Text =
(((double)progressBar1.Value /
progressBar1.Maximum) *
100).ToString("F2") + " %";
label11.Refresh();
    if
(Convert.ToInt32(textBox3.Text
) >= 0 &&
Convert.ToInt32(textBox3.Text)
<= 255 &&
Convert.ToInt32(textBox2.Text)
>= 0 &&
Convert.ToInt32(textBox2.Text)
<= 255 &&
Convert.ToInt32(textBox3.Text)
<=
Convert.ToInt32(textBox2.Text)
)
    {
        conrerata = 0;
conreratal = 0; conrerata2 =
0; enrerata = 0; enreratal =
0; enrerata2 = 0;
        foreach
(string s in
System.IO.Directory.GetFiles(f
olderpath))
        {
            double c1
= -0.015; double c2 = 0.005;
            do
            {
                if
(checkBox1.Checked == false)
                    c1
=
(double)numericUpDown1.Value;
                if
(checkBox2.Checked == false)
                    c2
=
(double)numericUpDown2.Value;
            }
            do
            {
                label11.Text =
(((double)progressBar1.Value /
progressBar1.Maximum) *
100).ToString("F2") + " %";
                label11.Refresh();
                coba.Add(new CitraCoba(s, low,
mid, high,
Convert.ToInt32(textBox2.Text)
,
Convert.ToInt32(textBox3.Text)
, c1, c2, false));
                conrerata += coba[coba.Count -
1].Contrast;
                conreratal += coba[coba.Count
- 1].Contrast_AFCEDP;
                conrerata2 += coba[coba.Count
- 1].Contrast_ACEDP;
                enrerata += coba[coba.Count -
1].Entropy;
                enreratal += coba[coba.Count -
1].Entropy_AFCEDP;
                enrerata2 += coba[coba.Count -
1].Entropy_ACEDP;
                c1
= Convert.ToDouble((decimal)c1
+ 0.001M);
            }
            while (c1 <= -0.005 &&
checkBox1.Checked == true);
            c1 = -
0.015;
            c2 +=
0.001;
        } while
(c2 <= 0.007 &&
checkBox2.Checked == true);
    }
    List<TabelCoba> tabel = new
List<TabelCoba>();
    foreach
(CitraCoba c in coba)
    {
        tabel.Add(new
TabelCoba(Path.GetFileName(c.f

```

```

ilename), c.c1, c.c2,
c.Entropy, c.Entropy_AFCEDP,
c.Entropy_ACEDP, c.Contrast,
c.Contrast_AFCEDP,
c.Contrast_ACEDP));
    }
    var bind = new
Library.Forms.SortableBindingL
ist<TabelCoba>(tabel);
    textBox6.Text
= ((double)conrerata /
(double)coba.Count).ToString()
;
    textBox4.Text
= ((double)conrerata1 /
(double)coba.Count).ToString()
;
    textBox5.Text
= ((double)conrerata2 /
(double)coba.Count).ToString()
;
    textBox7.Text
= ((double)enrerata /
(double)coba.Count).ToString()
;
    textBox8.Text
= ((double)enrerata1 /
(double)coba.Count).ToString()
;
    textBox9.Text
= ((double)enrerata2 /
(double)coba.Count).ToString()
;
dataGridView2.DataSource =
bind;
progressBar1.Visible = false;
label11.Visible = false;
    Cursor.Current
= Cursors.Default;
    }
    else
    {
        MessageBox.Show("Batas bawah
dan atas HE harus memiliki
nilai didalam range 0 hingga
255\nBatas bawah HE tidak
boleh lebih besar dari batas
atas HE");
    }

    private void
dataGridView2_ColumnHeaderMous
eClick(object sender,

```

```

DataGridViewCellMouseEventArgs
e)
    {
        DataGridView dtg =
sender as DataGridView;
        DataGridViewColumn
newColumn =
dtg.Columns[e.ColumnIndex];
        DataGridViewColumn
oldColumn = dtg.SortedColumn;
        ListSortDirection
direction;

        // If oldColumn is
null, then the DataGridView is
not sorted.
        if (oldColumn !=
null)
        {
            // Sort the
same column again, reversing
the SortOrder.
            if (oldColumn
== newColumn &&
dtg.SortOrder ==
SortOrder.Ascending)
            {
                direction
=
ListSortDirection.Descending;
            }
            else
            {
                // Sort a
new column and remove the old
SortGlyph.
                direction
= ListSortDirection.Ascending;
                oldColumn.HeaderCell.SortGlyph
Direction = SortOrder.None;
            }
        }
        else
        {
            direction =
ListSortDirection.Ascending;
        }

        // Sort the
selected column.

dataGridView2.Sort(newColumn,
direction);

```

```

newColumn.HeaderCell.SortGlyph
Direction =
    direction ==
ListSortDirection.Ascending ?

SortOrder.Ascending :
SortOrder.Descending;

//this.dataGridView1.Sort(this
.dataGridView1.Columns[dtg_Col
umn.Name],
ListSortDirection.Ascending)
    }

    private void
checkBox2_CheckedChanged(object
t sender, EventArgs e)
    {
        if
(checkBox2.Checked == false)

numericUpDown2.Enabled = true;
        else
numericUpDown2.Enabled =
false;
    }

    private void
checkBox1_CheckedChanged(object
t sender, EventArgs e)
    {
        if
(checkBox1.Checked == false)

numericUpDown1.Enabled = true;
        else
numericUpDown1.Enabled =
false;
    }

    private void
Percobaan_FormClosed(object
sender, FormClosedEventArgs e)
    {
        this.Dispose();
    }
}

```

#### **FORM TAMPIL CITRA**

```

using System;
using
System.Collections.Generic;
using System.ComponentModel;

```

```

using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using AForge;
using AForge.Imaging;
using AForge.Imaging.Filters;
using AForge.Math;
using
System.Drawing.Drawing2D;

namespace ProgramTA
{
    public partial class
Tampil_Citra : Form
    {
        public
Tampil_Citra(System.Drawing.Bi
tmap btmap, string name)
        {
            Citra = btmap;
            this.name = name;

InitializeComponent();
        }

        private Bitmap Citra {
set; get; }
        private string name {
set; get; }
        private float
aspectratio { set; get; }

        private
System.Drawing.Point
startingPoint =
System.Drawing.Point.Empty;
        private
System.Drawing.Point
movingPoint =
System.Drawing.Point.Empty;

        private void
Tampil_Citra_Load(object
sender, EventArgs e)
        {

//pictureBox1.Image = Citra;

//pictureBox1.Visible = false;
        ddPanBox1.Image =
Citra;

            this.Text =
"Tampil Citra - " + name;

```

```

        ImageStatistics
stat = new
ImageStatistics(Citra);
        Histogram hist =
stat.Red;
        histogram1.Color =
Color.Black;
        histogram1.Values
= hist.Values;

ComboboxZoom.SelectedIndex =
2;

        aspectratio =
Convert.ToSingle((float)Citra.
Width / (float)Citra.Height);
        nilaiaspect.Text =
aspectratio.ToString("F3");
        nilaimean.Text =
stat.Red.Mean.ToString("F3");
        nilaimedian.Text =
stat.Red.Median.ToString();
        nilaimin.Text =
stat.Red.Min.ToString();
        nilaimax.Text =
stat.Red.Max.ToString();
        bnykpixel.Text =
stat.Red.TotalCount.ToString()
;
        double stddev =
Statistics.StdDev(hist.Values)
;
        nilaistd.Text =
stddev.ToString("F3");
    }

    private void
toolStripButton1_Click(object
sender, EventArgs e)
    {

if (ComboboxZoom.SelectedIndex
== 1)
    {

ComboboxZoom.SelectedIndex =
0;

toolStripButton1.Enabled =
false;

        int n_height =
Citra.Height * 2;
        int n_width =
Convert.ToInt32(aspectratio *
n_height);

//pictureBox1.Image = new
Bitmap(Citra, n_width,
n_height);

        ddPanBox1.Image = new
Bitmap(Citra, n_width,
n_height);
    }
    else
    if (ComboboxZoom.SelectedIndex
== 2)
    {

ComboboxZoom.SelectedIndex =
1;

        int n_height =
Convert.ToInt32(Citra.Height *
1.5f);
        int n_width =
Convert.ToInt32(aspectratio *
n_height);

//pictureBox1.Image = new
Bitmap(Citra, n_width,
n_height);

        ddPanBox1.Image = new
Bitmap(Citra, n_width,
n_height);
    }
    else if
(ComboboxZoom.SelectedIndex ==
3)
    {

ComboboxZoom.SelectedIndex =
2;

//pictureBox1.Image = Citra;

ddPanBox1.Image = Citra;
    }
    else if
(ComboboxZoom.SelectedIndex ==
4)
    {

ComboboxZoom.SelectedIndex =
3;

        int n_height =
Convert.ToInt32(Citra.Height *
0.75f);

```

```

        int n_width =
Convert.ToInt32 (aspectratio *
n_height);

//pictureBox1.Image = new
Bitmap(Citra, n_width,
n_height);

ddPanBox1.Image = new
Bitmap(Citra, n_width,
n_height);
    }
    else if
(ComboboxZoom.SelectedIndex ==
5)
    {

ComboboxZoom.SelectedIndex =
4;

toolStripButton2.Enabled =
true;

        int n_height =
Convert.ToInt32 (Citra.Height *
0.5f);

        int n_width =
Convert.ToInt32 (aspectratio *
n_height);

//pictureBox1.Image = new
Bitmap(Citra, n_width,
n_height);

ddPanBox1.Image = new
Bitmap(Citra, n_width,
n_height);
    }
}

private void
toolStripButton2_Click(object
sender, EventArgs e)
{
    if
(ComboboxZoom.SelectedIndex ==
0)
    {

toolStripButton1.Enabled =
true;

ComboboxZoom.SelectedIndex =
1;

```

```

        int n_height =
Convert.ToInt32 (Citra.Height *
1.5f);

        int n_width =
Convert.ToInt32 (aspectratio *
n_height);

//pictureBox1.Image = new
Bitmap(Citra, n_width,
n_height);

ddPanBox1.Image = new
Bitmap(Citra, n_width,
n_height);
    }
    else if
(ComboboxZoom.SelectedIndex ==
1)
    {

ComboboxZoom.SelectedIndex =
2;

//pictureBox1.Image = Citra;

ddPanBox1.Image = Citra;
    }
    else if
(ComboboxZoom.SelectedIndex ==
2)
    {

ComboboxZoom.SelectedIndex =
3;

        int n_height =
Convert.ToInt32 (Citra.Height *
0.75f);

        int n_width =
Convert.ToInt32 (aspectratio *
n_height);

//pictureBox1.Image = new
Bitmap(Citra, n_width,
n_height);

ddPanBox1.Image = new
Bitmap(Citra, n_width,
n_height);
    }
    else if
(ComboboxZoom.SelectedIndex ==
3)
    {

ComboboxZoom.SelectedIndex =
4;

```



```

        int n_height =
Convert.ToInt32(Citra.Height *
0.5f);
        int n_width =
Convert.ToInt32(aspectratio *
n_height);

//pictureBox1.Image = new
Bitmap(Citra, n_width,
n_height);

ddPanBox1.Image = new
Bitmap(Citra, n_width,
n_height);
    }
    else if
(ComboboxZoom.SelectedIndex ==
4)
    {
ComboboxZoom.SelectedIndex =
5;

toolStripButton2.Enabled =
false;

        int n_height =
Convert.ToInt32(Citra.Height *
0.25f);

        int n_width =
Convert.ToInt32(aspectratio *
n_height);

//pictureBox1.Image = new
Bitmap(Citra, n_width,
n_height);

ddPanBox1.Image = new
Bitmap(Citra, n_width,
n_height);
    }

    private void
ComboboxZoom_SelectedIndexChanged(
object sender, EventArgs
e)
    {

if(ComboboxZoom.SelectedIndex
== 0)
    {

toolStripButton1.Enabled =
false;

```

```

toolStripButton2.Enabled =
true;

        int n_height =
Citra.Height * 2;
        int n_width =
Convert.ToInt32(aspectratio *
n_height);

//pictureBox1.Image = new
Bitmap(Citra, n_width,
n_height);

ddPanBox1.Image = new
Bitmap(Citra, n_width,
n_height);
    }
    else
if(ComboboxZoom.SelectedIndex
== 1)
    {

toolStripButton1.Enabled =
true;

toolStripButton2.Enabled =
true;

        int n_height =
Convert.ToInt32(Citra.Height *
1.5f);

        int n_width =
Convert.ToInt32(aspectratio *
n_height);

//pictureBox1.Image = new
Bitmap(Citra, n_width,
n_height);

ddPanBox1.Image = new
Bitmap(Citra, n_width,
n_height);
    }
    else
if(ComboboxZoom.SelectedIndex
== 2)
    {

toolStripButton1.Enabled =
true;

toolStripButton2.Enabled =
true;

//pictureBox1.Image = Citra;

ddPanBox1.Image = Citra;

```

```

        }
        else
        if (ComboboxZoom.SelectedIndex
        == 3)
        {
            toolStripButton1.Enabled =
            true;

            toolStripButton2.Enabled =
            true;

            int n_height =
            Convert.ToInt32 (Citra.Height *
            0.75f);

            int n_width =
            Convert.ToInt32 (aspectratio *
            n_height);

            //pictureBox1.Image = new
            Bitmap(Citra, n_width,
            n_height);

            ddPanBox1.Image = new
            Bitmap(Citra, n_width,
            n_height);
        }
        else
        if (ComboboxZoom.SelectedIndex
        == 4)
        {

            toolStripButton1.Enabled =
            true;

            toolStripButton2.Enabled =
            true;

            int n_height =
            Convert.ToInt32 (Citra.Height *
            0.5f);

            int n_width =
            Convert.ToInt32 (aspectratio *
            n_height);

            //pictureBox1.Image = new
            Bitmap(Citra, n_width,
            n_height);

            ddPanBox1.Image = new
            Bitmap(Citra, n_width,
            n_height);
        }
        else
        if (ComboboxZoom.SelectedIndex
        == 5)
        {

            toolStripButton1.Enabled =
            true;

            toolStripButton2.Enabled =
            false;

            int n_height =
            Convert.ToInt32 (Citra.Height *
            0.25f);

            int n_width =
            Convert.ToInt32 (aspectratio *
            n_height);

            //pictureBox1.Image = new
            Bitmap(Citra, n_width,
            n_height);

            ddPanBox1.Image = new
            Bitmap(Citra, n_width,
            n_height);
        }
        }

        private void
        mouse_enter(object
        sender, EventArgs e)
        {
            if (ComboboxZoom.SelectedIndex
            < 2)
            {

                //pictureBox1.Cursor =
                Cursors.Hand;
            }
            else
            {
                //
                pictureBox1.Cursor =
                Cursors.Default;
            }
        }
    }

    }

FORM ABOUT US
using System;
using
System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

```

```

namespace ProgramTA
{
    public partial class
About_Us : Form
    {
        public About_Us()
        {
            InitializeComponent();

            private void
About_Us_Load(object sender,
EventArgs e)
            {
                advRichTextBox1.SelectionAlign
ment = TextAlign.Justify;

                advRichTextBox1.SelectAll();
            }
        }
    }
}

KELAS DDPANBOX.CS
using System;
using
System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Drawing;
using System.Drawing.Imaging;

namespace DDPanBox
{
    class DDPanBox : Control
    {
        public DDPanBox()
        {
            //Set up double
buffering and a little extra.

            this.SetStyle(ControlStyles.UserPaint |
ControlStyles.OptimizedDoubleB
uffer |

ControlStyles.AllPaintingInWmP
aint |
ControlStyles.SupportsTranspar
entBackColor,
            true);
        }

        //set the part of
the source image to be drawn.
        //DrawRect =
DeflateRect(ClientRectangle,
Padding);
        DrawRect =
ClientRectangle;

        //Subscribe to our
event handlers.
        this.MouseDown +=
new
MouseEventHandler(panBox_Mouse
Down);
        this.MouseMove +=
new
MouseEventHandler(panBox_Mouse
Move);
        this.MouseUp += new
MouseEventHandler(panBox_Mouse
Up);
        this.Resize += new
EventHandler(panBox_Resize);
    }
    //if true were at 2:1
false then 1:1
    //bool zoom = false;

    bool dragging = false;
    //Tells us if our image has
been clicked on.

    Point start = new
Point(); //Keep initial click
for accurate panning.

    void
panBox_MouseDown(object
sender, MouseEventArgs e)
    {
        if (e.Button ==
MouseButtons.Left)
        {
            dragging =
true;

            //offset new
point by original one so we
know where in the image we
are.
            start = new
Point(e.Location.X +
DrawRect.Location.X,
e.Location.Y +
DrawRect.Location.Y);
        }
    }
}

```

```

        Cursor =
Cursors.SizeAll; //just for
looks.
    }
    //else if
    (e.Button ==
MouseButtons.Right)
    //{
    //    zoom =
!zoom;
    //    if (zoom)
    //    {
    //        //
modify the drawrect to a
smaller rectangle to zoom, and
center it in previous drawrect
    //        DrawRect
= new Rectangle(DrawRect.X +
ClientRectangle.Width / 4,
DrawRect.Y +
ClientRectangle.Height / 4,
DrawRect.Width / 2,
DrawRect.Height / 2);
    //    }
    //    else
    //    {
    //        //Do the
reverse of above
    //        DrawRect
= new Rectangle(DrawRect.X -
ClientRectangle.Width/4,
DrawRect.Y -
ClientRectangle.Height/4,
ClientRectangle.Width,
ClientRectangle.Height);
    //    }
    //    //Calculate
Draw Rectangle by calling the
resize event.
    //
    panBox_Resize(null,
EventArgs.Empty);
    //}
}

void
panBox_MouseMove(object
sender, MouseEventArgs e)
{
    if (dragging)
    {
        DrawRect.Location = new
Point(start.X - e.Location.X,
start.Y - e.Location.Y);

```

```

        if
(DrawRect.Location.X < 0 -
Padding.Left)

        DrawRect.Location = new
Point(0 - Padding.Left,
DrawRect.Location.Y);

        if
(DrawRect.Location.Y < 0 -
Padding.Top)

        DrawRect.Location = new
Point(DrawRect.Location.X, 0 -
Padding.Top);

        if
(DrawRect.Location.X >
_Image.Width - DrawRect.Width
+ Padding.Right)

        DrawRect.Location = new
Point(_Image.Width -
DrawRect.Width +
Padding.Right,
DrawRect.Location.Y);

        if
(DrawRect.Location.Y >
_Image.Height -
DrawRect.Height +
Padding.Bottom)

        DrawRect.Location = new
Point(DrawRect.Location.X,
_Image.Height -
DrawRect.Height +
Padding.Bottom);

        this.Refresh();
    }
}

void
panBox_MouseUp(object sender,
MouseEventArgs e)
{
    dragging = false;
    Cursor =
Cursors.Default;
}

void
panBox_Resize(object sender,
EventArgs e)

```

```

        {
            if (_Image !=
null)
            {
                //if (zoom)
                //{
                //    DrawRect
= new
Rectangle(DrawRect.Location.X,
DrawRect.Location.Y,
ClientRectangle.Width / 2,
ClientRectangle.Height / 2);
                //}
                //else
                DrawRect =
new
Rectangle(DrawRect.Location.X,
DrawRect.Location.Y,
ClientRectangle.Width,
ClientRectangle.Height);

                if
(DrawRect.Location.X < 0 -
Padding.Left)

DrawRect.Location = new
Point(0 - Padding.Left,
DrawRect.Location.Y);

                if
(DrawRect.Location.Y < 0 -
Padding.Top)

DrawRect.Location = new
Point(DrawRect.Location.X, 0 -
Padding.Top);

                if
(DrawRect.Location.X >
_Image.Width - DrawRect.Width
+ Padding.Right)

DrawRect.Location = new
Point(_Image.Width -
DrawRect.Width +
Padding.Right,
DrawRect.Location.Y);

                if
(DrawRect.Location.Y >
_Image.Height -
DrawRect.Height +
Padding.Bottom)

DrawRect.Location = new
Point(DrawRect.Location.X,

_Image.Height -
DrawRect.Height +
Padding.Bottom);

this.Refresh();
        }

        private Image _Image;

        public Image Image
        {
            get
            {
                return _Image;
            }
            set
            {
                _Image =
value;
                //Calculate
Draw Rectangle by calling the
resize event.

                panBox_Resize(this,
EventArgs.Empty);
            }
        }

        private Rectangle
DrawRect;

        protected override
void OnPaint(PaintEventArgs e)
        {
            if (_Image !=
null)
            {

e.Graphics.InterpolationMode =
System.Drawing.Drawing2D.Inter
polationMode.HighQualityBicubi
c;

e.Graphics.SmoothingMode =
System.Drawing.Drawing2D.Smoot
hingMode.HighQuality;

e.Graphics.DrawImage(_Image,
ClientRectangle, DrawRect,
GraphicsUnit.Pixel);

                //if(zoom)

```

```

        //
        e.Graphics.DrawString("Zoom
2:1", this.Font,
Brushes.White, new PointF(15F,
15F));
        //else
        //
        e.Graphics.DrawString("Zoom
1:1", this.Font,
Brushes.White, new PointF(15F,
15F));
    }

    base.OnPaint(e);
}

    public static
Rectangle
DeflateRect(Rectangle rect,
Padding padding)
    {
        rect.X +=
padding.Left;
        rect.Y +=
padding.Top;
        rect.Width -=
padding.Horizontal;
        rect.Height -=
padding.Vertical;
        return rect;
    }
}

```

#### **KELAS TABEL COBA.CS**

```

using System;
using
System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ProgramTA
{
    class TabelCoba
    {
        public string filename
{ get; set; }
        public double c1 {
get; set; }
        public double c2 {
get; set; }
        public double Entropy
{get;set;} public double
Entropy_AFCEDP {get;set;}
    }
}

```

```

public double
Entropy_ACEDP{get;set;} public
double Contrast {get;set;}
public double
Contrast_AFCEDP{get;set;}
public double
Contrast_ACEDP{get;set;}
    public
TabelCoba(string filename,
double c1, double c2, double
Entropy, double newEntropy1,
double newEntropy2, double
Contrast, double newContrast1,
double newContrast2)
    {
        this.filename =
filename;
        this.c1 = c1;
        this.c2 = c2;
        this.Entropy =
Entropy;
        this.Contrast =
Contrast;

        this.Contrast_AFCEDP =
newContrast1;

        this.Contrast_ACEDP =
newContrast2;

        this.Entropy_AFCEDP =
newEntropy1;
        this.Entropy_ACEDP
= newEntropy2;
    }
}

```

#### **KELAS CITRA COBA.CS**

```

using System;
using
System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Drawing;
using System.Threading.Tasks;

namespace ProgramTA
{
    class CitraCoba
    {
        public string filename
= "";
        public Bitmap
CitraInput, CitraOutput1,
CitraOutput2;
    }
}

```

```

        private int[] H = new
int[256];
        private double[] low,
mid, high;
        public double Entropy,
Entropy_AFCEDP, Entropy_ACEDP,
Contrast, Contrast_AFCEDP,
Contrast_ACEDP;
        public double c1, c2;
        public
CitraCoba(string filename,
double[] low, double[] mid,
double[] high, int HElowbound,
int HEhibound, double c1,
double c2, bool aktifK)
        {
            Image img = new
Bitmap(filename);
            CitraInput = new
Bitmap(img);
            this.filename =
filename;
            this.c1 = c1;
            this.c2 = c2;
            long totalgr1 = 0;
            long totalgr2 = 0;
            for (int x = 0; x
< CitraInput.Width; x++)
            {
                for (int y =
0; y < CitraInput.Height; y++)
                {
                    Color c =
CitraInput.GetPixel(x, y);
                    int
grayscale = (int)((c.R * 0.3f)
+ (c.G * 0.59f) + (c.B *
0.11f));
                    Color g =
Color.FromArgb(grayscale,
grayscale, grayscale);
                    CitraInput.SetPixel(x, y, g);
                    H[grayscale]++;
                    totalgr1
+= grayscale * grayscale;
                    totalgr2
+= grayscale;
                }
            }
            CitraOutput1 = new
Bitmap(CitraInput);
            CitraOutput2 = new
Bitmap(CitraInput);

```

```

            Contrast =
(double)10f *
Math.Log10((double)((double)to
talgr1 /
(double)(CitraInput.Width *
CitraInput.Height)) -
Math.Pow((double)totalgr2 /
(double)(CitraInput.Width *
CitraInput.Height), 2));
            this.low = low;
            this.mid = mid; this.high =
high;
            int intref;
            int deflow = 43;
            int defmid = 128; int defhigh
= 213;
            double tlow = 0,
tmid = 0, thigh = 0;
            for (int i = 0; i
< H.Length; i++)
            {
                tlow += low[i]
* H[i]; tmid += mid[i] * H[i];
                thigh += high[i] * H[i];
            }
            intref =
Convert.ToInt32(Math.Round((tl
ow / (tlow + tmid + thigh) *
deflow) + (tmid / (tlow + tmid
+ thigh) * defmid) + (thigh /
(tlow + tmid + thigh) *
defhigh)));
            int[] newH1 = new
int[256]; double[] newHpdl1 =
new double[256];
            double[] pdf = new
double[256];
            double[] npdl1 =
new double[256];
            double[] ncdf1 =
new double[256];
            double lvlow,
lvmid, lvhigh;
            double cliplimit;
            int[] fl = new
int[256];
            lvlow = 0; lvmid =
0; lvhigh = 0;
            for (int i = 0; i
< H.Length; i++)
            {
                pdf[i] =
(double)H[i] /
(double)H.Sum();
            }

```

```

        for (int i = 0; i
< H.Length; i++)
        {
            if (aktifK ==
false)
            {
                lvlow =
(c1) + pdf.Max();
                lvhigh =
(c2) + means(pdf);
                lvmid =
means(pdf);
            }
            else
            {
                lvlow =
(c1) * i + pdf.Max();
                lvhigh =
(c2) * i + means(pdf);
                lvmid =
means(pdf);
            }
            cliplimit =
(low[intref] * lvlow) +
(mid[intref] * lvmid) +
(high[intref] * lvhigh);
            if (pdf[i] >
cliplimit)
            {
                npdf1[i] =
cliplimit;
            }
            else npdf1[i]
= pdf[i];
            if (i > 0)
                ncdf1[i] =
ncdf1[i - 1] + npdf1[i];
            else ncdf1[i]
= npdf1[i];
        }
        for (int i = 0; i
< H.Length; i++)
        {
            npdf1[i] /=
ncdf1[255];
            if (i > 0)
                ncdf1[i] =
ncdf1[i - 1] + npdf1[i];
            else ncdf1[i]
= npdf1[i];
            f1[i] =
Convert.ToInt32(Math.Round(HEl
owbound + ((HEhibound -
HElowbound) * (ncdf1[i] -
(npdf1[i] / 2)))));
        }

        totalgr1 = 0;
        totalgr2 = 0;
        for (int x = 0; x
< CitraOutput1.Width; x++)
        {
            for (int y =
0; y < CitraOutput1.Height;
y++)
            {
                Color c =
CitraOutput1.GetPixel(x, y);
                int gr =
f1[(int)c.R];
                Color g =
Color.FromArgb(gr, gr, gr);
                CitraOutput1.SetPixel(x, y,
g);
                newH1[gr]++;
                totalgr1
+= (gr * gr);
                totalgr2
+= gr;
            }
        }
        Contrast_AFCEDP =
(double)10f *
Math.Log10((double)((double)to
talgr1 /
(double)(CitraOutput1.Width *
CitraOutput1.Height)) -
Math.Pow((double)totalgr2 /
(double)(CitraOutput1.Width *
CitraOutput1.Height), 2));
        int[] newH2 = new
int[256]; double[] newHpdp2 =
new double[256];
        double[] npdf2 =
new double[256];
        double[] ncdf2 =
new double[256];
        int[] f2 = new
int[256];
        cliplimit = 0;
        for (int i = 0; i
< H.Length; i++)
        {
            if (aktifK ==
false)
            {
                if (tlow >
tmid && tlow > thigh)
                cliplimit = (c1) + pdf.Max();
            }
        }
    }

```



```

        else if
(thigh > tlow && thigh > tmid)
        cliplimit = (c2) + means(pdf);
        else
        cliplimit = means(pdf);
    }
    else
    {
        if (tlow >
tmid && tlow > thigh)
        cliplimit = (c1) * i +
pdf.Max();
        else if
(thigh > tlow && thigh > tmid)
        cliplimit = (c2) * i +
means(pdf);
        else
        cliplimit = means(pdf);
    }
    if (pdf[i] >
cliplimit)
    {
        npdf2[i] =
cliplimit;
    }
    else npdf2[i]
    = pdf[i];
    if (i > 0)
        ncdf2[i] =
ncdf2[i - 1] + npdf2[i];
    else ncdf2[i]
    = npdf2[i];
}
for (int i = 0; i
< H.Length; i++)
{
    npdf2[i] /=
ncdf2[255];
    if (i > 0)
        ncdf2[i] =
ncdf2[i - 1] + npdf2[i];
    else ncdf2[i]
    = npdf2[i];
    f2[i] =
Convert.ToInt32(Math.Round(HEl
owbound + ((HEhibound -
HElowbound) * (ncdf2[i] -
(npdf2[i] / 2)))));
}
    totalgr1 = 0;
totalgr2 = 0;

    for (int x = 0; x
< CitraOutput2.Width; x++)
    {
        for (int y =
0; y < CitraOutput2.Height;
y++)
        {
            Color c =
CitraOutput2.GetPixel(x, y);
            int gr =
f2[(int)c.R];
            Color g =
Color.FromArgb(gr, gr, gr);
CitraOutput2.SetPixel(x, y,
g);
            newH2[gr]++;
            totalgr1
+= (gr * gr);
            totalgr2
+= gr;
        }
    }
    Contrast_ACEDP =
(double)10f *
Math.Log10(((double)((double)to
talgr1 /
(double)(CitraOutput2.Width *
CitraOutput2.Height)) -
Math.Pow((double)totalgr2 /
(double)(CitraOutput2.Width *
CitraOutput2.Height), 2)));
    for (int i = 0; i
< H.Length; i++)
    {
        newHpdf1[i] =
(double)newH1[i] /
(double)newH1.Sum();
        newHpdf2[i] =
(double)newH2[i] /
(double)newH2.Sum();
        if (pdf[i] >
0)
            Entropy +=
((double)(-1) * (double)pdf[i]
* (double)Math.Log(pdf[i],
2));
        if
(newHpdf1[i] > 0)
            Entropy_AFCEDP += ((double)(-
1) * (double)newHpdf1[i] *
(double)Math.Log(newHpdf1[i],
2));
    }

```

```

        if
(newHpdf2[i] > 0)

Entropy_ACEDP += ((double) (-1)
* (double) newHpdf2[i] *
(double) Math.Log(newHpdf2[i],
2));
    }
}
private double
means(double[] p)
{
    double rerata = 0;
    for (int i = 0; i
< p.Length; i++)
    {
        if (p[i] > 0)
rerata++;
    }
    return
(double) p.Sum() /
(double) rerata;
}
}
}

```

#### **KELAS ADVRICHTEXTBOX.CS**

```

using System;
using System.Windows.Forms;
using
System.Runtime.InteropServices
;

/// <summary>
/// Represents a standard <see
cref="RichTextBox"/> with some
/// minor added functionality.
/// </summary>
/// <remarks>
/// AdvRichTextBox provides
methods to maintain
performance
/// while it is being updated.
Additional formatting features
/// have also been added.
/// </remarks>
public class AdvRichTextBox :
RichTextBox
{
    /// <summary>
    /// Maintains performance
while updating.
    /// </summary>
    /// <remarks>
    /// <para>

```

```

    /// It is recommended to
call this method before doing
    /// any major updates that
you do not wish the user to
    /// see. Remember to call
EndUpdate when you are
finished
    /// with the update.
Nested calls are supported.
    /// </para>
    /// <para>
    /// Calling this method
will prevent redrawing. It
will
    /// also setup the event
mask of the underlying
richedit
    /// control so that no
events are sent.
    /// </para>
    /// </remarks>
public void BeginUpdate()
{
    // Deal with nested
calls.
    ++updating;

    if (updating > 1)
        return;

    // Prevent the control
from raising any events.
    oldEventMask =
SendMessage(new
HandleRef(this, Handle),
EM_SETEVENTMASK, 0, 0);

    // Prevent the control
from redrawing itself.
    SendMessage(new
HandleRef(this, Handle),
WM_SETREDRAW, 0, 0);
}

    /// <summary>
    /// Resumes drawing and
event handling.
    /// </summary>
    /// <remarks>
    /// This method should be
called every time a call is
made

```

```

        /// made to BeginUpdate.
        It resets the event mask to
        it's
        /// original value and
        enables redrawing of the
        control.
        /// </remarks>
        public void EndUpdate()
        {
            // Deal with nested
calls.
            --updating;

            if (updating > 0)
                return;

            // Allow the control
            to redraw itself.
            SendMessage(new
            HandleRef(this, Handle),
            WM_SETREDRAW, 1, 0);

            // Allow the control
            to raise event messages.
            SendMessage(new
            HandleRef(this, Handle),
            EM_SETEVENTMASK, 0,
            oldEventMask);
        }

        /// <summary>
        /// Gets or sets the
        alignment to apply to the
        current
        /// selection or insertion
        point.
        /// </summary>
        /// <remarks>
        /// Replaces the
        SelectionAlignment from
        /// <see
        cref="RichTextBox"/>.
        /// </remarks>
        public new TextAlign
        SelectionAlignment
        {
            get
            {
                PARAFORMAT fmt =
                new PARAFORMAT();
                fmt.cbSize =
                Marshal.SizeOf(fmt);

```

```

        // Get the
        alignment.
        SendMessage(new
        HandleRef(this, Handle),
        EM_GETPARAFORMAT,
        SCF_SELECTION, ref fmt);

        // Default to Left
        align.
        if ((fmt.dwMask &
        PFM_ALIGNMENT) == 0)
            return
            TextAlign.Left;

        return
        (TextAlign)fmt.wAlignment;
    }

    set
    {
        PARAFORMAT fmt =
        new PARAFORMAT();
        fmt.cbSize =
        Marshal.SizeOf(fmt);
        fmt.dwMask =
        PFM_ALIGNMENT;
        fmt.wAlignment =
        (short)value;

        // Set the
        alignment.
        SendMessage(new
        HandleRef(this, Handle),
        EM_SETPARAFORMAT,
        SCF_SELECTION, ref fmt);
    }

    /// <summary>
    /// This member overrides
    /// <see
    cref="Control"/>.OnHandleCreat
    ed.
    /// </summary>
    protected override void
    OnHandleCreated(EventArgs e)
    {
        base.OnHandleCreated(e);

        // Enable support for
        justification.

```

```

        SendMessage(new
HandleRef(this, Handle),

EM_SETTYPOGRAPHYOPTIONS,

TO_ADVANCEDTYPOGRAPHY,

TO_ADVANCEDTYPOGRAPHY);
    }

    private int updating = 0;
    private int oldEventMask =
0;

    // Constants from the
Platform SDK.
    private const int
EM_SETEVENTMASK = 1073;
    private const int
EM_GETPARAFORMAT = 1085;
    private const int
EM_SETPARAFORMAT = 1095;
    private const int
EM_SETTYPOGRAPHYOPTIONS =
1226;
    private const int
WM_SETREDRAW = 11;
    private const int
TO_ADVANCEDTYPOGRAPHY = 1;
    private const int
PFM_ALIGNMENT = 8;
    private const int
SCF_SELECTION = 1;

    // It makes no difference
if we use PARAFORMAT or
    // PARAFORMAT2 here, so I
have opted for PARAFORMAT2.

[StructLayout(LayoutKind.Seque
ntial)]
    private struct PARAFORMAT
    {
        public int cbSize;
        public uint dwMask;
        public short
wNumbering;
        public short
wReserved;
        public int
dxStartIndent;
        public int
dxRightIndent;
        public int dxOffset;
        public short
wAlignment;

        public short
cTabCount;

        [MarshalAs(UnmanagedType.ByVal
Array, SizeConst = 32)]
        public int[] rgxTabs;

        // PARAFORMAT2 from
here onwards.
        public int
dySpaceBefore;
        public int
dySpaceAfter;
        public int
dyLineSpacing;
        public short sStyle;
        public byte
bLineSpacingRule;
        public byte
bOutlineLevel;
        public short
wShadingWeight;
        public short
wShadingStyle;
        public short
wNumberingStart;
        public short
wNumberingStyle;
        public short
wNumberingTab;
        public short
wBorderSpace;
        public short
wBorderWidth;
        public short wBorders;
    }

    [DllImport("user32",
CharSet = CharSet.Auto)]
    private static extern int
SendMessage(HandleRef hWnd,

int msg,

int wParam,

int lParam);

    [DllImport("user32",
CharSet = CharSet.Auto)]
    private static extern int
SendMessage(HandleRef hWnd,

int msg,

int wParam,

```

```

ref PARAFORMAT lp);

    private void
InitializeComponent()
    {

this.SuspendLayout();

this.ResumeLayout(false);

    }
}

/// <summary>
/// Specifies how text in a
<see cref="AdvRichTextBox"/>
is
/// horizontally aligned.
/// </summary>
public enum TextAlign
{
    /// <summary>
    /// The text is aligned to
the left.
    /// </summary>
    Left = 1,

    /// <summary>
    /// The text is aligned to
the right.
    /// </summary>
    Right = 2,

    /// <summary>
    /// The text is aligned in
the center.
    /// </summary>
    Center = 3,

    /// <summary>
    /// The text is justified.
    /// </summary>
    Justify = 4
}

KELAS SORTABLEBINDINGLIST.CS
using System;
using
System.Collections.Generic;
using System.ComponentModel;

namespace Library.Forms
{
    /// <summary>
        /// Provides a generic
        collection that supports data
        binding and additionally
        supports sorting.
        /// See
        http://msdn.microsoft.com/en-
        us/library/ms993236.aspx
        /// If the elements are
        IComparable it uses that;
        otherwise compares the
        ToString()
        /// </summary>
        /// <typeparam
        name="T">The type of elements
        in the list.</typeparam>
        public class
        SortableBindingList<T> :
        BindingList<T> where T : class
        {
            private bool
            _isSorted;
            private
            ListSortDirection
            _sortDirection =
            ListSortDirection.Ascending;
            private
            PropertyDescriptor
            _sortProperty;

            /// <summary>
            /// Initializes a new
            instance of the <see
            cref="SortableBindingList{T}"/>
            > class.
            /// </summary>
            public
            SortableBindingList()
            {
            }

            /// <summary>
            /// Initializes a new
            instance of the <see
            cref="SortableBindingList{T}"/>
            > class.
            /// </summary>
            /// <param
            name="list">An <see
            cref="T:System.Collections.Gen
            eric.IList`1" /> of items to
            be contained in the <see
            cref="T:System.ComponentModel.
            BindingList`1" />.</param>
            public
            SortableBindingList(IList<T>
            list)

```

```

        : base(list)
    {
        /// <summary>
        /// Gets a value
        indicating whether the list
        supports sorting.
        /// </summary>
        protected override
        bool SupportsSortingCore
        {
            get { return true;
        }

        }

        /// <summary>
        /// Gets a value
        indicating whether the list is
        sorted.
        /// </summary>
        protected override
        bool IsSortedCore
        {
            get { return
            _isSorted; }
        }

        /// <summary>
        /// Gets the direction
        the list is sorted.
        /// </summary>
        protected override
        ListSortDirection
        SortDirectionCore
        {
            get { return
            _sortDirection; }
        }

        /// <summary>
        /// Gets the property
        descriptor that is used for
        sorting the list if sorting is
        implemented in a derived
        class; otherwise, returns null
        /// </summary>
        protected override
        PropertyDescriptor
        SortPropertyCore
        {
            get { return
            _sortProperty; }
        }

        /// <summary>
        /// Removes any sort
        applied with ApplySortCore if
        sorting is implemented
        /// </summary>
        protected override
        void RemoveSortCore()
        {
            _sortDirection =
            ListSortDirection.Ascending;
            _sortProperty =
            null;
            _isSorted = false;
            //thanks Luca
        }

        /// <summary>
        /// Sorts the items if
        overridden in a derived class
        /// </summary>
        /// <param
        name="prop"></param>
        /// <param
        name="direction"></param>
        protected override
        void
        ApplySortCore(PropertyDescript
        or prop, ListSortDirection
        direction)
        {
            _sortProperty =
            prop;
            _sortDirection =
            direction;

            List<T> list =
            Items as List<T>;
            if (list == null)
            return;

            list.Sort(Compare);

            _isSorted = true;
            //fire an event
            that the list has been
            changed.
            OnListChanged(new
            ListChangedEventArgs(ListChang
            edType.Reset, -1));
        }

        private int Compare(T
        lhs, T rhs)
        {

```

```

        var result =
OnComparison(lhs, rhs);
        //invert if
descending
        if (_sortDirection
==
ListSortDirection.Descending)
            result = -
result;
        return result;
    }

    private int
OnComparison(T lhs, T rhs)
    {
        object lhsValue =
lhs == null ? null :
_sortProperty.GetValue(lhs);
        object rhsValue =
rhs == null ? null :
_sortProperty.GetValue(rhs);
        if (lhsValue ==
null)
        {
            return
(rhsValue == null) ? 0 : -1;
//nulls are equal
        }

        if (rhsValue ==
null)
        {
            return 1;
//first has value, second
doesn't
        }
        if (lhsValue is
IComparable)
        {
            return
((IComparable)lhsValue).Compar
eTo(rhsValue);
        }
        if
(lhsValue.Equals(rhsValue))
        {
            return 0;
//both are the same
        }
        //not comparable,
compare ToString
        return
lhsValue.ToString().CompareTo(
rhsValue.ToString());
    }
}

```

## **DAFTAR RIWAYAT HIDUP**

Nama : Juandy Hartanto  
Umur : 21 tahun  
Tempat / Tanggal Lahir : Medan / 12 Desember 1994  
Jenis Kelamin : Pria  
Agama : Buddha  
Tempat Tinggal : Jl. Thamrin No. 82 H, Medan

**Pendidikan :**

- |                                  |            |
|----------------------------------|------------|
| 1. Tamatan TK Tri Ratna Sibolga  | tahun 2000 |
| 2. Tamatan SD Tri Ratna Sibolga  | tahun 2006 |
| 3. Tamatan SMP Tri Ratna Sibolga | tahun 2009 |
| 4. Tamatan SMA Tri Ratna Sibolga | tahun 2012 |

Demikian daftar riwayat hidup ini saya perbuat dengan sesungguhnya.

Hormat saya,

Juandy Hartanto



## DAFTAR RIWAYAT HIDUP

Nama : Kelvin  
Umur : 21 tahun  
Tempat / Tanggal Lahir : Sibolga / 15 Desember 1994  
Jenis Kelamin : Pria  
Agama : Buddha  
Tempat Tinggal : Jl. Thamrin No. 82 H, Medan

### Pendidikan :

- |                                  |            |
|----------------------------------|------------|
| 5. Tamatan TK Tri Ratna Sibolga  | tahun 2000 |
| 6. Tamatan SD Tri Ratna Sibolga  | tahun 2006 |
| 7. Tamatan SMP Tri Ratna Sibolga | tahun 2009 |
| 8. Tamatan SMA Tri Ratna Sibolga | tahun 2012 |

Demikian daftar riwayat hidup ini saya perbuat dengan sesungguhnya.

Hormat saya,

Kelvin