Praktikum

7

Perbaikan Citra (Enhancement 3)

1. Tujuan:

1. Mahasiswa dapat membuat program untuk memperjelas citra dengan histogram Equalization

2. Dasar Teori:

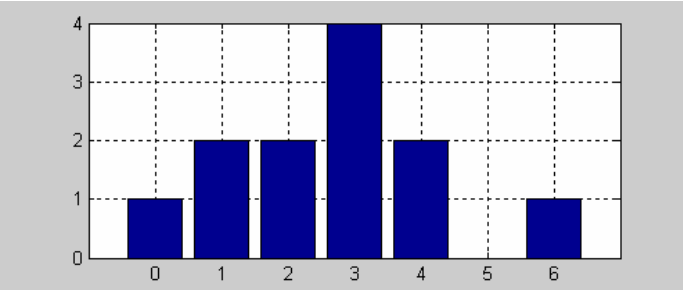
Histogram Equalization

Histogram Equalization adalah suatu proses perataan histogram, dimana distribusi nilai derajat keabuan pada suatu citra dibuat rata. Untuk dapat melakukan histogram equalization ini diperlukan suatu fungsi distribusi kumulatif yang merupakan kumulatif dari histogram.

Misalkan diketahui data sebagai berikut:

2 4 3 1 3 6 4 3 1 0 3 2

Maka histogram dari data di atas adalah:

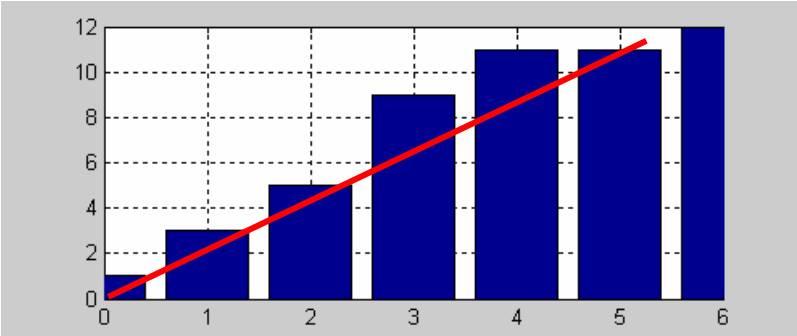


Gambar 7.1 Contoh histogram

Proses perhitungan distribusi kumulatif dapat dijelaskan dengan tabel berikut:

Nilai	Histogram	Dsitribusi kumulatif
0	1	1
1	2	1+2=3
2	2	3+2=5
3	4	5+4=9
4	2	9+2=11
5	0	11+0=11
6	1	11+1=12

Dan diperoleh histogram kumulatif sebagai berikut:



Gambar 7.2 Histogram kumulatif

Histogram equalization (perataan histogram) adalah suatu proses dimana histogram diratakan berdasarkan suatu fungsi linier (garis lurus) seperti terlihat pada gambar 5.2. Teknik perataan histogram adalah sebagai berikut:

Nilai asal	Histogram Kumulatif	Nilai hasil
0	1	½ → 0
1	3	3/2 → 1
2	5	5/2 → 2
3	9	9/2 → 4
4	11	11/2 → 5
5	11	11/2 → 5
6	12	12/2 → 6

Nilai hasil histogram equalization adalah sebagai berikut:

$$w = \frac{c_w.th}{n_x n_y}$$

dimana

w = nilai keabuan hasil histogram equalization

c_w = histogram kumulatif dari w

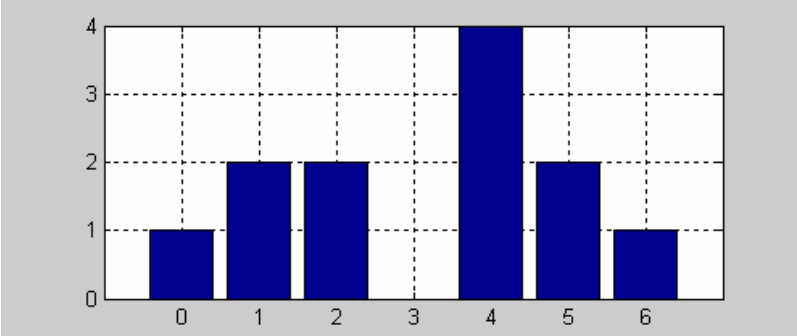
th = threshold derajat keabuan (256)

n_x dan n_y = ukuran gambar

Hasil setelah histogram equalization adalah sebagai berikut:

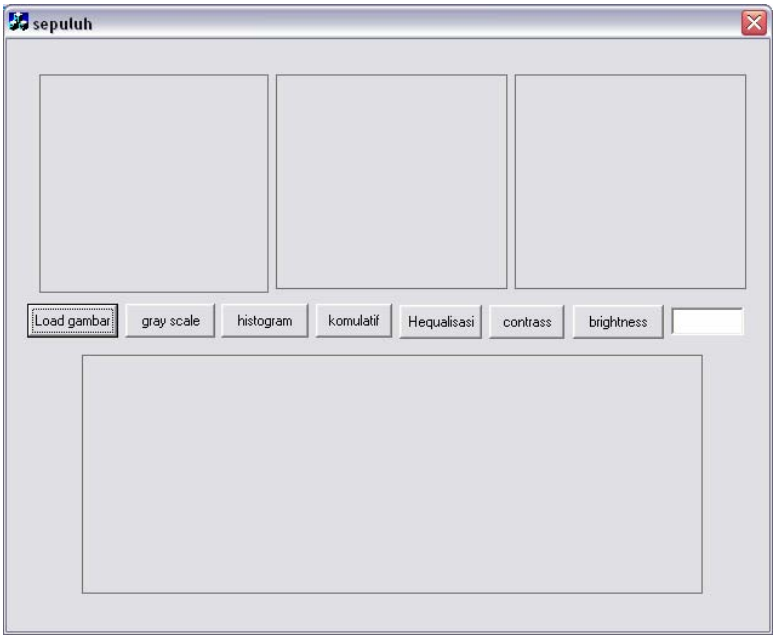
2 5 4 1 4 6 5 4 1 0 4 2

Histogram dari hasil histogram equalization:



Gambar 7.3 Histogram dari hasil histogram equalization

Percobaan:



Gambar 7.4. Disain window aplikasi

Program di button load gambar:

```
     CDC* pDC = m_pic1.GetDC();
     CDC dcMem1;
     CRect rect;
     BITMAP bm;
HBITMAP hBitmap=(HBITMAP)::LoadImage(AfxGetInstanceHandle(),
"satu.bmp",IMAGE_BITMAP, 0, 0, LR_LOADFROMFILE|LR_CREATEDIBSECTION);
     if(hBitmap)
     {
      if(m_bmpBitmap.DeleteObject())
            m_bmpBitmap.Detach();
      m_bmpBitmap.Attach(hBitmap);
     }
     m_pic1.GetClientRect(rect);
     m_bmpBitmap.GetBitmap(&bm);
     dcMem1.CreateCompatibleDC(pDC);
     dcMem1.SelectObject(&m_bmpBitmap);


  pDC->StretchBlt(0,0,rect.Width(),rect.Height(),&dcMem1,
  0,0,bm.bmWidth,bm.bmHeight,SRCCOPY);
```

Program di button gray scale :

```
int i,j,red,green,blue,gray;
     long int warna,warnagray;
     CDC* pDC = m_pic2.GetDC();
     CDC dcMem1;
     CRect rect;
     BITMAP bm;
HBITMAP hBitmap=(HBITMAP)::LoadImage(AfxGetInstanceHandle(),
"satu.bmp",IMAGE_BITMAP, 0, 0, LR_LOADFROMFILE|LR_CREATEDIBSECTION);
     if(hBitmap)
     {
      if(m_bmpBitmap.DeleteObject())
            m_bmpBitmap.Detach();
      m_bmpBitmap.Attach(hBitmap);
     }
     m_pic2.GetClientRect(rect);
     m_bmpBitmap.GetBitmap(&bm);
     dcMem1.CreateCompatibleDC(pDC);
     dcMem1.SelectObject(&m_bmpBitmap);
     for(i=0;i<bm.bmHeight;i++)
      for(j=0;j<bm.bmWidth;j++)
      {
            warna=dcMem1.GetPixel(j,i);
            WarnaToRGB(warna,&red,&green,&blue);
            gray=int(red+green+blue)/3;
            warnagray=RGBToWarna(gray,gray,gray);
            dcMem1.SetPixel(j,i,warnagray);
      }

  pDC->StretchBlt(0,0,rect.Width(),rect.Height(),&dcMem1,
  0,0,bm.bmWidth,bm.bmHeight,SRCCOPY);
```

Program di button contrass

```
     int i,j,red,green,blue,gray;
     float k=float(GetDlgItemInt(IDC_EDIT1))/10;
     long int warna,warnagray;
     CDC* pDC = m_pic4.GetDC();
     CDC dcMem1;
     CRect rect;
     BITMAP bm;
HBITMAP
hBitmap=(HBITMAP)::LoadImage(AfxGetInstanceHandle(),"satu.bmp",IMAGE_BITMAP,
0, 0, LR_LOADFROMFILE|LR_CREATEDIBSECTION);
     if(hBitmap)
     {
      if(m_bmpBitmap.DeleteObject())
            m_bmpBitmap.Detach();
      m_bmpBitmap.Attach(hBitmap);
     }
     m_pic4.GetClientRect(rect);
     m_bmpBitmap.GetBitmap(&bm);
     dcMem1.CreateCompatibleDC(pDC);
     dcMem1.SelectObject(&m_bmpBitmap);
     for(i=0;i<bm.bmHeight;i++)
      for(j=0;j<bm.bmWidth;j++)
      {
            warna=dcMem1.GetPixel(j,i);
            WarnaToRGB(warna,&red,&green,&blue);
            gray=int(red+green+blue)/3;
            gray=int(gray*k);
            if(gray>255)gray=255;
            warnagray=RGBToWarna(gray,gray,gray);
            dcMem1.SetPixel(j,i,warnagray);
      }
  pDC->StretchBlt(0,0,rect.Width(),rect.Height(),&dcMem1,0,0,
bm.bmWidth,bm.bmHeight,SRCCOPY);
}
```

Program di button brigthness

```
     int i,j,red,green,blue,gray;
     int k=GetDlgItemInt(IDC_EDIT1);
     long int warna,warnagray;
     CDC* pDC = m_pic4.GetDC();
     CDC dcMem1;
     CRect rect;
     BITMAP bm;
HBITMAP
hBitmap=(HBITMAP)::LoadImage(AfxGetInstanceHandle(),"satu.bmp",IMAGE_BITMAP,
0, 0, LR_LOADFROMFILE|LR_CREATEDIBSECTION);
     if(hBitmap)
     {
      if(m_bmpBitmap.DeleteObject())
            m_bmpBitmap.Detach();
      m_bmpBitmap.Attach(hBitmap);
     }
     m_pic4.GetClientRect(rect);
     m_bmpBitmap.GetBitmap(&bm);
     dcMem1.CreateCompatibleDC(pDC);
     dcMem1.SelectObject(&m_bmpBitmap);
     for(i=0;i<bm.bmHeight;i++)
      for(j=0;j<bm.bmWidth;j++)
      {
            warna=dcMem1.GetPixel(j,i);
            WarnaToRGB(warna,&red,&green,&blue);
            gray=int(red+green+blue)/3;
            gray=gray+k;
            if(gray>255)gray=255;
            if(gray<0)gray=0;
            warnagray=RGBToWarna(gray,gray,gray);
            dcMem1.SetPixel(j,i,warnagray);
      }
  pDC->StretchBlt(0,0,rect.Width(),rect.Height(),&dcMem1,
  0,0,bm.bmWidth,bm.bmHeight,SRCCOPY);
```

Program di button histogram

```
     CDC* pDC = m_pic3.GetDC();
     CDC dcMem1;
     CRect rect;
     BITMAP bm;
     int i,j;
     int red,green,blue,gray;
     long int warna;
     float h[256];
HBITMAP hBitmap=(HBITMAP)::LoadImage(AfxGetInstanceHandle(),
"satu.bmp",IMAGE_BITMAP, 0, 0, LR_LOADFROMFILE|LR_CREATEDIBSECTION);
     if(hBitmap)
     {
      if(m_bmpBitmap.DeleteObject())
            m_bmpBitmap.Detach();
      m_bmpBitmap.Attach(hBitmap);
     }
     m_pic2.GetClientRect(rect);
     m_bmpBitmap.GetBitmap(&bm);
     dcMem1.CreateCompatibleDC(pDC);
     dcMem1.SelectObject(&m_bmpBitmap);
     for(i=0;i<256;i++)h[i]=0;
     for(i=0;i<bm.bmHeight;i++)
      for(j=0;j<bm.bmWidth;j++)
      {
            warna=dcMem1.GetPixel(j,i);
            WarnaToRGB(warna,&red,&green,&blue);
            gray=int(red+green+blue)/3;
            h[gray]++;

      }
     float hmax=h[0];
     for(i=1;i<256;i++)
      if(h[i]>hmax)hmax=h[i];
     for(i=0;i<256;i++)
      h[i]=190*h[i]/hmax;
     CDC* pDC1 = m_pic3.GetDC();
     pDC1->MoveTo(0,190);
     pDC1->LineTo(470,190);
     pDC1->MoveTo(0,190-(int)h[0]);
     for(i=1;i<256;i++)
      pDC1->LineTo(i*2,190-(int)h[i]);
```

Program di button komulatif

```
     CDC* pDC = m_pic3.GetDC();
     CDC dcMem1;
     CRect rect;
     BITMAP bm;
     int i,j;
     int red,green,blue,gray;
     long int warna;
     float h[256];
HBITMAP hBitmap=(HBITMAP)::LoadImage(AfxGetInstanceHandle(),
"satu.bmp",IMAGE_BITMAP, 0, 0, LR_LOADFROMFILE|LR_CREATEDIBSECTION);
     if(hBitmap)
     {
      if(m_bmpBitmap.DeleteObject())
            m_bmpBitmap.Detach();
      m_bmpBitmap.Attach(hBitmap);
     }
     m_pic2.GetClientRect(rect);
     m_bmpBitmap.GetBitmap(&bm);
     dcMem1.CreateCompatibleDC(pDC);
     dcMem1.SelectObject(&m_bmpBitmap);
     for(i=0;i<256;i++)h[i]=0;
     for(i=0;i<bm.bmHeight;i++)
      for(j=0;j<bm.bmWidth;j++)
      {
            warna=dcMem1.GetPixel(j,i);
            WarnaToRGB(warna,&red,&green,&blue);
            gray=int(red+green+blue)/3;
            h[gray]++;

      }
     float c[256];
     c[0]=h[0];
     for(i=1;i<256;i++)
      c[i]=c[i-1]+h[i];
     float hmax=c[255];
     for(i=0;i<256;i++)
      c[i]=190*c[i]/hmax;
     CDC* pDC1 = m_pic3.GetDC();
     pDC1->MoveTo(0,190);
     pDC1->LineTo(470,190);
     pDC1->MoveTo(0,190-(int)c[0]);
     for(i=1;i<256;i++)
      pDC1->LineTo(i*2,190-(int)c[i]);
```

Program di button Hequalisasi

```
CDC* pDC = m_pic2.GetDC();
     CDC dcMem1;
     CRect rect;
     BITMAP bm;
     int i,j;
     int red,green,blue,gray;
     long int warna;
     float h[256];
HBITMAP hBitmap=(HBITMAP)::LoadImage(AfxGetInstanceHandle(),
"satu.bmp",IMAGE_BITMAP, 0, 0, LR_LOADFROMFILE|LR_CREATEDIBSECTION);
     if(hBitmap)
     {
      if(m_bmpBitmap.DeleteObject())
            m_bmpBitmap.Detach();
      m_bmpBitmap.Attach(hBitmap);
     }
     m_pic2.GetClientRect(rect);
     m_bmpBitmap.GetBitmap(&bm);
     dcMem1.CreateCompatibleDC(pDC);
     dcMem1.SelectObject(&m_bmpBitmap);
     for(i=0;i<256;i++)
      h[i]=0;
     for(i=0;i<bm.bmHeight;i++)
      for(j=0;j<bm.bmWidth;j++)
      {
            warna=dcMem1.GetPixel(j,i);
            WarnaToRGB(warna,&red,&green,&blue);
            gray=int(red+green+blue)/3;
            h[gray]++;
      }
     float c[256];
     c[0]=h[0];
     for(i=1;i<256;i++)
      c[i]=c[i-1]+h[i];
     for(i=0;i<256;i++)
      c[i]=c[i]/bm.bmHeight/bm.bmWidth;
     for(i=0;i<256;i++)h[i]=0;
     for(i=0;i<bm.bmHeight;i++)
      for(j=0;j<bm.bmWidth;j++)
      {
            warna=dcMem1.GetPixel(j,i);
            WarnaToRGB(warna,&red,&green,&blue);
            gray=int(red+green+blue)/3;
            gray=c[gray]*255;
            h[gray]++;
            warna=RGBToWarna(gray,gray,gray);
            dcMem1.SetPixel(j,i,warna);

      }

     pDC-
>StretchBlt(0,0,rect.Width(),rect.Height(),&dcMem1,0,0,bm.bmWidth,bm.bmHeight,SRCCOPY);
```

```
c[0]=h[0];
for(i=1;i<256;i++)
 c[i]=c[i-1]+h[i];
float hmax=c[255];
for(i=0;i<256;i++)
 c[i]=190*c[i]/hmax;
CDC* pDC1 = m_pic3.GetDC();
pDC1->MoveTo(0,190);
pDC1->LineTo(470,190);
pDC1->MoveTo(0,190-(int)c[0]);
for(i=1;i<256;i++)
 pDC1->LineTo(i*2,190-(int)c[i]);
```

Fungsi mengubah warna ke rgb

```
void WarnaToRGB(long int warna,int *Red, int *Green, int *Blue)
{
      *Red = warna & 0x000000FF;
      *Green = (warna & 0x0000FF00) >> 8;
      *Blue = (warna & 0x00FF0000) >> 16;
}
```

Fungsi mengubah rgb ke warna

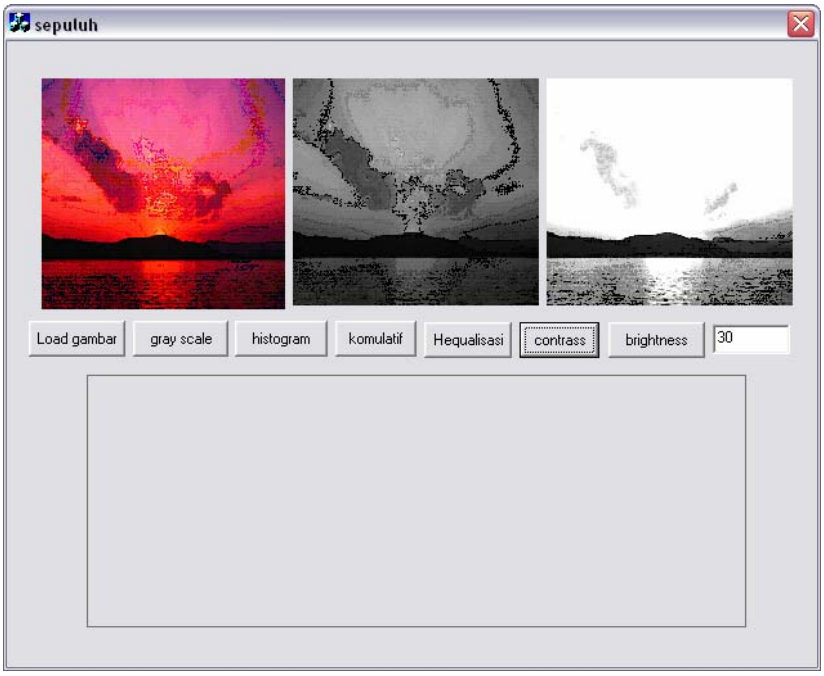
```
long int RGBToWarna(int Red, int Green, int Blue)
{
      return(Red+(Green<<8)+(Blue<<16));
}
```

Tambahan pada header file

```
public:
      Cbitmap m_bmpBitmap;
```

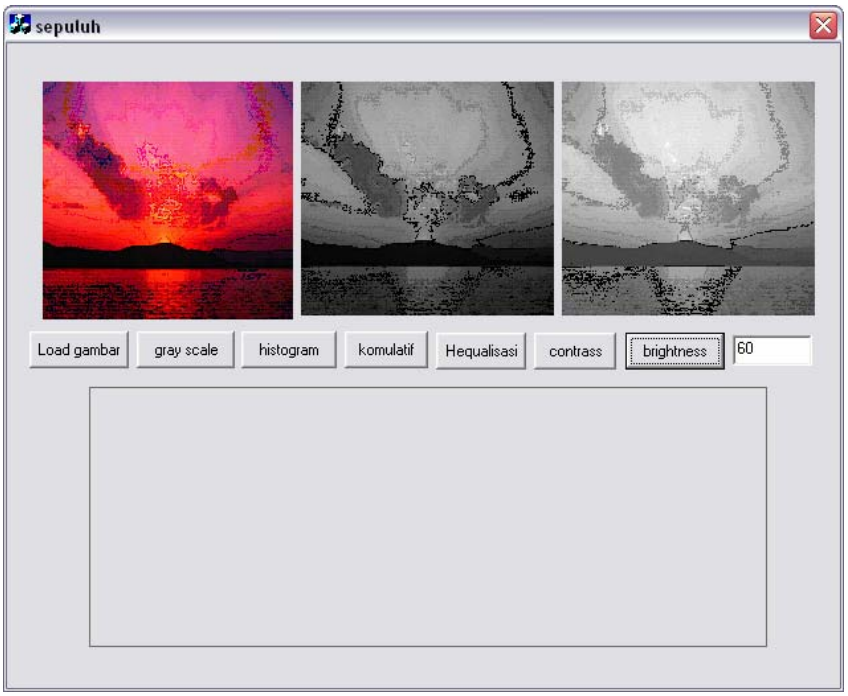
Menjalankan aplikasi

Contrass: nilai dimasukkan pada textbox, gambar hasil pada pic paling kanan



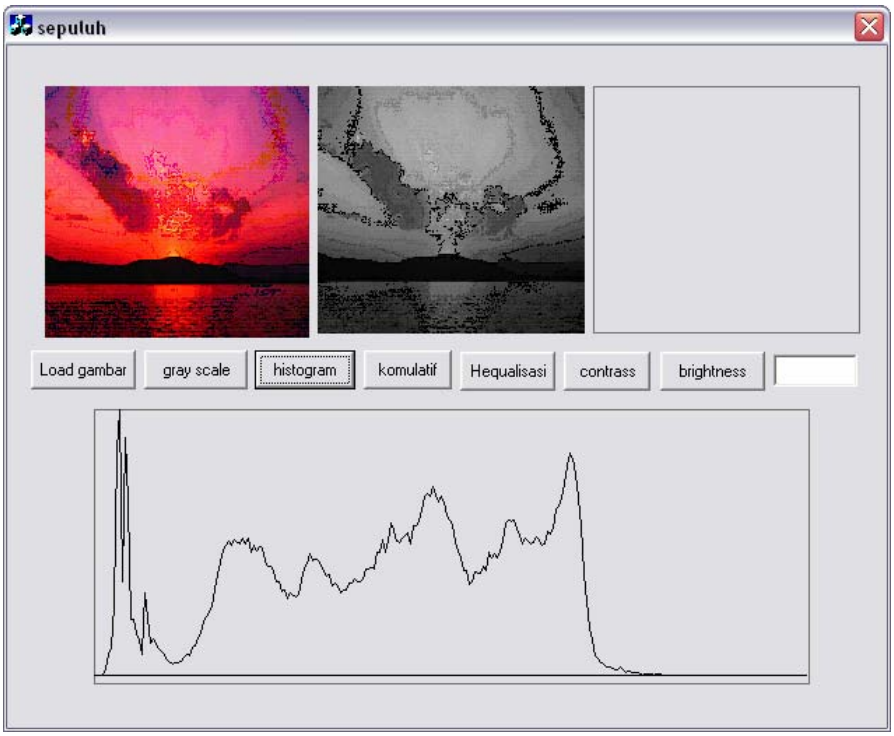
Gambar 7.5. Hasil Contrass dengan brightness 30

Brigthness: nilai dimasukkan pada textbox, gambar hasil pada pic paling kanan



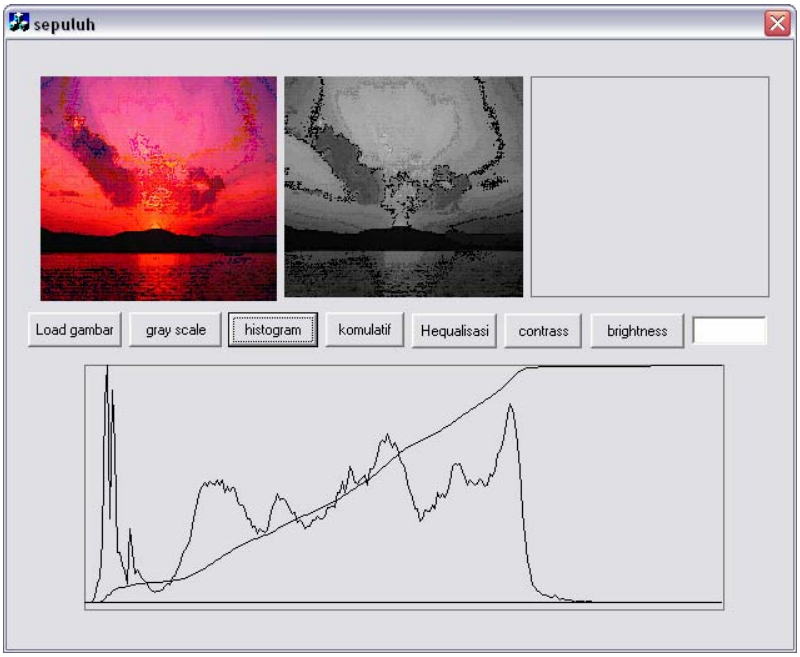
Gambar 7.6. Hasil Contrass dengan brightness 60

Histogram: grafik kemunculan tiap pixel gambar gray scale(tengah) akan digambarkan pada pic bawah



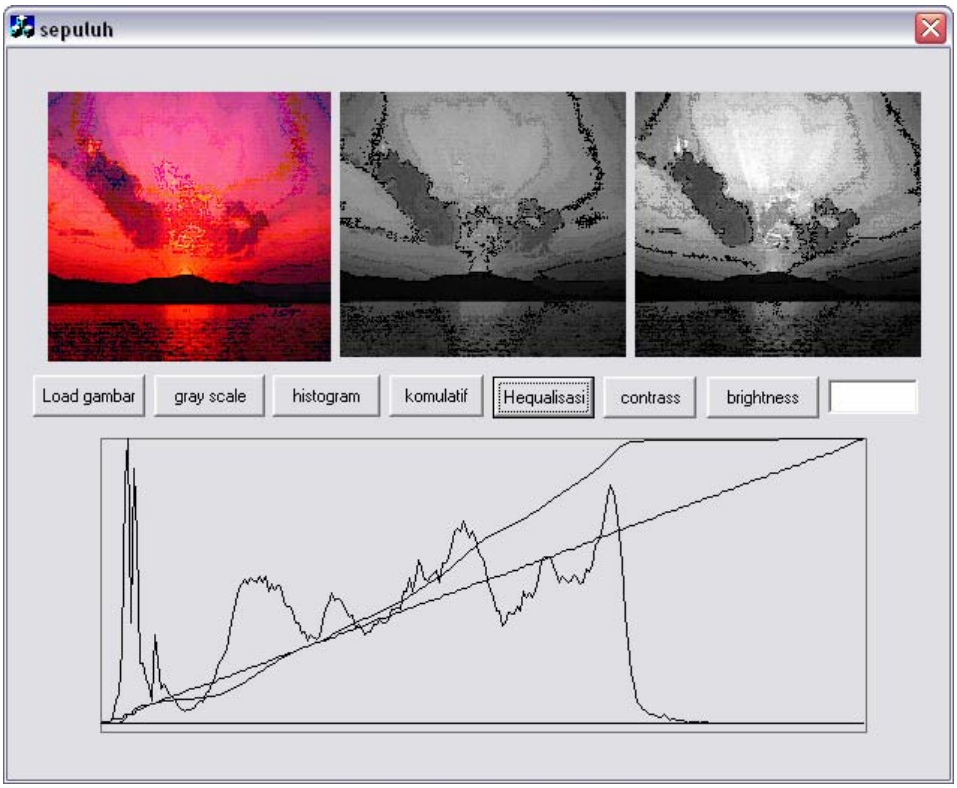
Gambar 7.7. Hasil histogram citra gray scale

Kumulatif: kumulatif dari histogram akan digambarkan



Gambar 7.8. Hasil kumulatif histogram citra gray scale

Hequalisasi: grafik hasil equalisasi akan digambarkan sekaligus citra hasil akan ditampilkan di pic paling kanan



Gambar 7.9. Hasil histogram equalisasi citra gray scale

Tugas :

1. Definisikan kembali :

a. Histogram citra gray scale

b. Kumulatif histogram citra gray scale

c. Histogram equalisasi citra gray scale

2. Cobalah dengan gambar input gelap lakukan kontras amati histogram citra input dan histogram citra setelah penambahan kontras

3. Cobalah dengan gambar input gelap lakukan brightness amati histogram citra input dan histogram citra setelah penambahan brightness

4. Buatlah kesimpulan dari histogram equalisasi yang didapatkan dengan histogram citra input, bagaimana sebarannya

5. Buatlah kesimpulan dari hasil yang didapatkan enhancement citra dengan histogram equalisasi