Praktikum

6

Perbaikan Citra (Enhancement 2)

1. Tujuan:

1. Mahasiswa dapat membuat program untuk transformasi citra yaitu inverse, log, inverse log, nth power, nth root power

2. Dasar Teori:

Transformasi Citra:

Inversi Citra

Inversi citra adalah proses negatif pada citra, misalkan pada photo, dimana setiap nilai citra dibalik dengan acuan threshold yang diberikan. Proses ini banyak digunakan pada citra-citra medis seperti USG dan X-Ray. Untuk citra dengan derajat keabuan 256, proses inversi citra didefinisikan dengan:

$$xn = 255 - x$$

Tranformasi Logaritmik

Tranformasi Logaritmik didefinisikan dengan G = c Log (F +1)

Tranformasi Invers Logaritmik didefinisikan dengan G = c Log (L-F +1)

Dimana G adalah citra hasil, F citra asal, c adalah konstanta yang dipasang sebagai efek perubahan kontras

Transformasi Power Law

Transformasi Power Law ada dua yaitu nth power dan nth root power

Transformasi nth power dedifinisikan dengan $G = C F^{Y}$

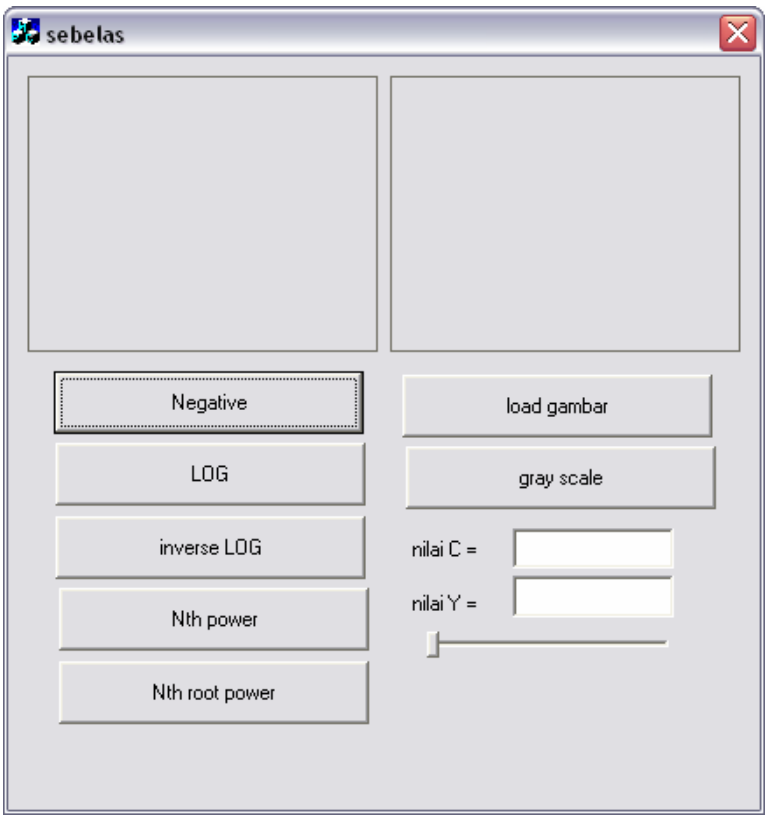
Transformasi nth root power didefinisikan dengan $G = C F^{1/Y}$

Dimana G adalah citra hasil, F adalah citra asal, c dan y adalah konstanta positif

Percobaan:

Transformasi Citra

Desain aplikasi yang dibuat



Gambar 6.1. Disai Window Aplikasi

Program pada button load gambar

```
     static char BASED_CODE szFilter[]="Bitmap Files (*.bmp)|*.bmp||";
     CfileDialog m_ldFile(TRUE,"*.bmp",name,OFN_HIDEREADONLY|  OFN_OVERWRITEPROMPT,
szFilter);
     if(m_ldFile.DoModal()==IDOK)
     {
      name=m_ldFile.GetPathName();
     }
     CDC* pDC = m_pic1.GetDC();
     CDC dcMem1;
     CRect rect;
     BITMAP bm;
     HBITMAP hBitmap=(HBITMAP)::LoadImage(AfxGetInstanceHandle(),
     name ,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 pada 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(),
     name,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 pada button negative

```
    int i,j,red,green,blue,gray;
    long int warna, warnagray,max;
    CDC* pDC = m_pic2.GetDC();
    CDC dcMem1;
    CRect rect;
    BITMAP bm;
    HBITMAP hBitmap=(HBITMAP)::LoadImage(AfxGetInstanceHandle(),
    name,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);
    max=m_slider.GetPos();
    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=max-gray;
           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 pada button log

```
     int i,j,red,green,blue,gray,c;
     long int warna, warnagray;
     CDC* pDC = m_pic2.GetDC();
     CDC dcMem1;
     CRect rect;
     BITMAP bm;
     HBITMAP hBitmap=(HBITMAP)::LoadImage(AfxGetInstanceHandle(),
     name,IMAGE_BITMAP, 0, 0, LR_LOADFROMFILE|LR_CREATEDIBSECTION);
     if(hBitmap)
     {
      if(m_bmpBitmap.DeleteObject())
            m_bmpBitmap.Detach();
      m_bmpBitmap.Attach(hBitmap);
     }

     c=GetDlgItemInt(IDC_EDIT1);
      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;
            gray=int (c*log(gray+1));
            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 pada button inverse log

```
int i,j,red,green,blue,gray,c,max;
     long int warna, warnagray;
     CDC* pDC = m_pic2.GetDC();
     CDC dcMem1;
     CRect rect;
     BITMAP bm;
     HBITMAP hBitmap=(HBITMAP)::LoadImage(AfxGetInstanceHandle(),
     name,IMAGE_BITMAP, 0, 0, LR_LOADFROMFILE|LR_CREATEDIBSECTION);
     if(hBitmap)
     {
      if(m_bmpBitmap.DeleteObject())
            m_bmpBitmap.Detach();
      m_bmpBitmap.Attach(hBitmap);
     }
```

```
    c=GetDlgItemInt(IDC_EDIT1);
    max=m_slider.GetPos();
    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;
           gray=int (c*log(max-(gray+1)));
           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 pada button Nth power

```
    int i,j,red,green,blue,gray,c,y;
    long int warna, warnagray;
    double thc, thy;
    CDC* pDC = m_pic2.GetDC();
    CDC dcMem1;
    CRect rect;
    BITMAP bm;
    HBITMAP hBitmap=(HBITMAP)::LoadImage(AfxGetInstanceHandle(),
    name,IMAGE_BITMAP, 0, 0, LR_LOADFROMFILE|LR_CREATEDIBSECTION);
    if(hBitmap)
    {
     if(m_bmpBitmap.DeleteObject())
           m_bmpBitmap.Detach();
     m_bmpBitmap.Attach(hBitmap);
    }
    c=GetDlgItemInt(IDC_EDIT1);
    y=GetDlgItemInt(IDC_EDIT2);
    thc=(float)c/100.;
    thy=(float)y/100.;

    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;
           gray=int (thc*pow(gray,thy));
```

```
            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 pada button Nth root power

```
    int i,j,red,green,blue,gray,c,y;
    long int warna, warnagray;
    double thc, thy;
    CDC* pDC = m_pic2.GetDC();
    CDC dcMem1;
    CRect rect;
    BITMAP bm;
    HBITMAP hBitmap=(HBITMAP)::LoadImage(AfxGetInstanceHandle(),
    name,IMAGE_BITMAP, 0, 0, LR_LOADFROMFILE|LR_CREATEDIBSECTION);
    if(hBitmap)
    {
     if(m_bmpBitmap.DeleteObject())
           m_bmpBitmap.Detach();
     m_bmpBitmap.Attach(hBitmap);
    }

    c=GetDlgItemInt(IDC_EDIT1);
    y=GetDlgItemInt(IDC_EDIT2);
    thc=(float)c/100.;
    thy=(float)y/100.;
    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;
           gray=int (thc*pow(gray,1./thy));
           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);
```

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

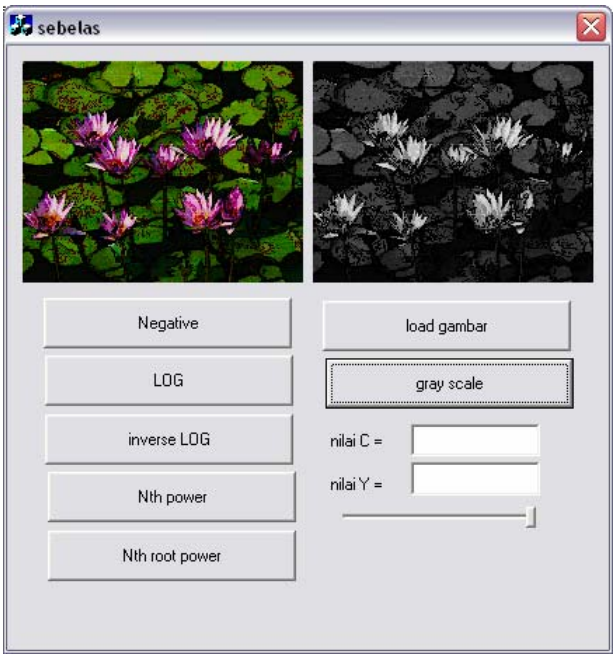
```
long int RGBToWarna(int Red, int Green, int Blue)
{
      return(Red+(Green<<8)+(Blue<<16));
}
```

Tambahan pada header file

```
public:
      Cbitmap m_bmpBitmap;
      CString name;
```

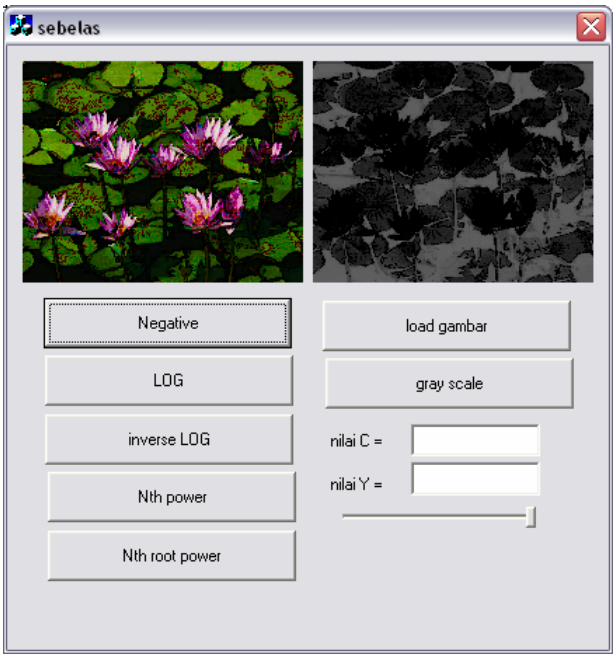
Menjalankan aplikasi

Gray scale



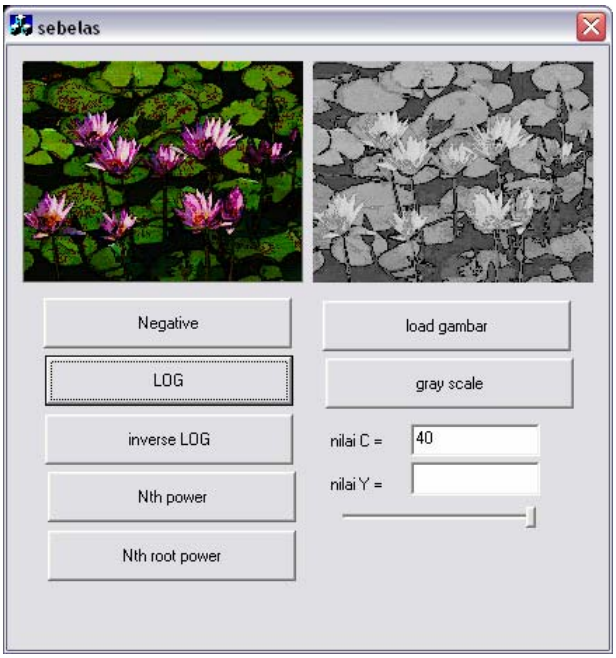
Gambar 6.2. Hasil Percobaan Gray Scale

Negative : nilai maximum diambil dari slider



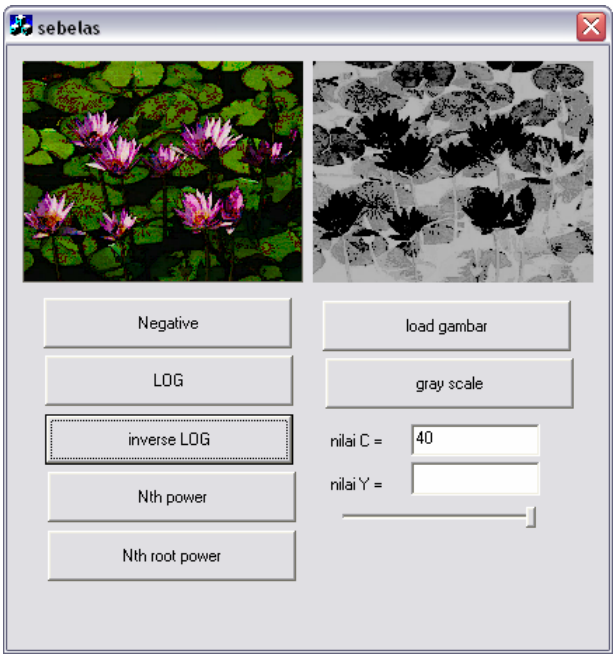
Gambar 6.3. Hasil Percobaan Negative

Log



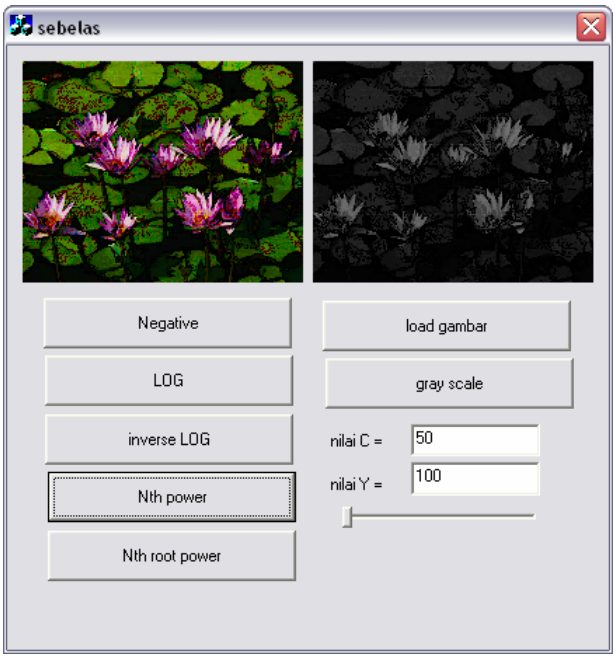
Gambar 6.4. Hasil Percobaan LOG

Inverse log



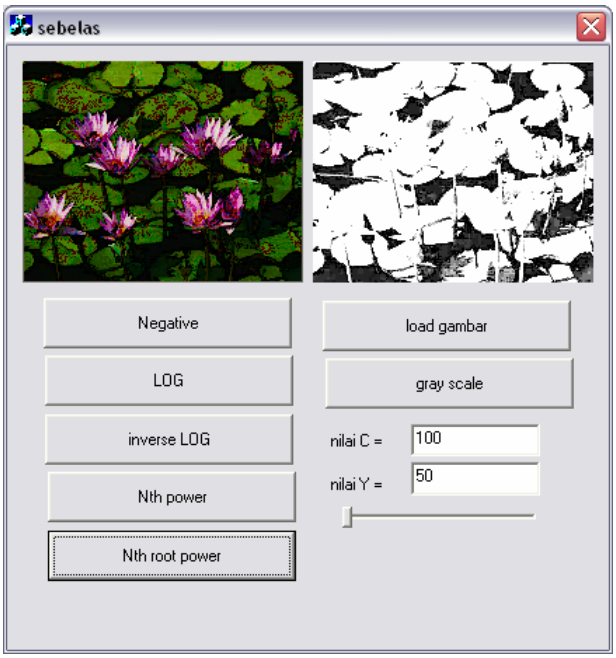
Gambar 6.5. Hasil Percobaan Inverse LOG

Nth power



Gambar 6.6. Hasil Percobaan Nth power

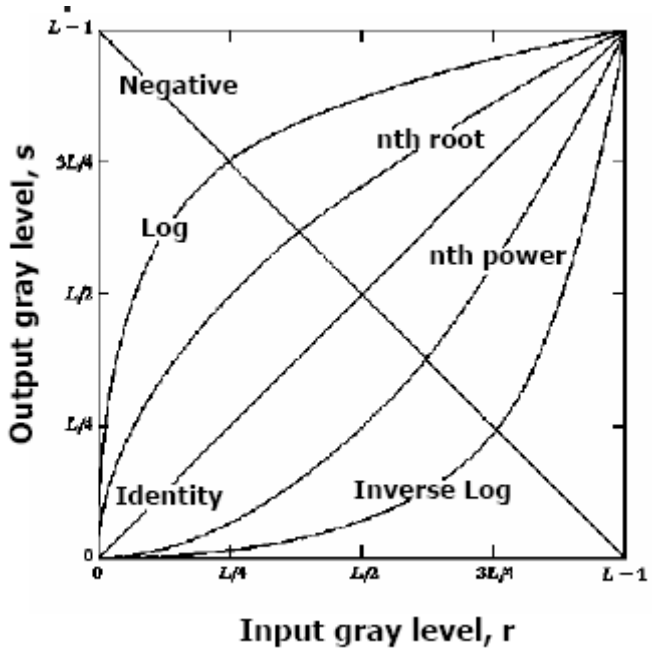
Nth root power



Gambar 6.7. Hasil Percobaan Nth root power

Tugas :

1. Buatlah kesimpulan dari pengamatan terhadap hasil yang di dapatkan pada :
    a. Transformasi citra negative
    b. Transformasi citra dengan fungsi LOG
    c. Transformasi citra dengan fungsi inverse LOG
    d. Transformasi citra Nth power
    e. Transformasi citra Nth root power
2. Buatlah kesimpulan umum tentang transformasi citra dihubungkan dengan grafik input output citra setelah mengalami transformasi pada gambar



Gambar 6.8. Grafik Input Output Citra Hasil Transformasi