Teori Pengolahan Citra
PJJ-4

Hero Yudo Martono

28 April 2016

Thresholding

- *Thresholding* adalah proses mengubah citra berderajat keabuan menjadi citra biner atau hitam putih sehingga dapat diketahui daerah mana yang termasuk obyek dan *background* dari citra secara jelas. Citra hasil *thresholding* biasanya digunakan lebih lanjut untuk proses pengenalan obyek serta ekstraksi fitur.
  Metode *thresholding* secara umum dibagi menjadi dua, yaitu :

Thresholding global

- *Thresholding* dilakukan dengan mempartisi histogram dengan menggunakan sebuah*threshold* (batas ambang) global T, yang berlaku untuk seluruh bagian pada citra.

Thresholding adaptif

- *Thesholding* dilakukan dengan membagi citra menggunakan beberapa sub citra. Lalu pada setiap sub citra, segmentasi dilakukan dengan menggunakan *threshold* yang berbeda.

2.2. Thresholding

Thresholding digunakan untuk mengatur jumlah derajat keabuan yang ada pada citra. Dengan menggunakan thresholding maka derajat keabuan bisa diubah sesuai keinginan, misalkan diinginkan menggunakan derajat keabuan 16, maka tinggal membagi nilai derajat keabuan dengan 16. Proses thresholding ini pada dasarnya adalah proses pengubahan kuantisasi pada citra, sehingga untuk melakukan thresholding dengan derajat keabuan dapat digunakan rumus:

$$x = b.\text{int}\left({}^{w}\!/_{b}\right)$$

dimana :

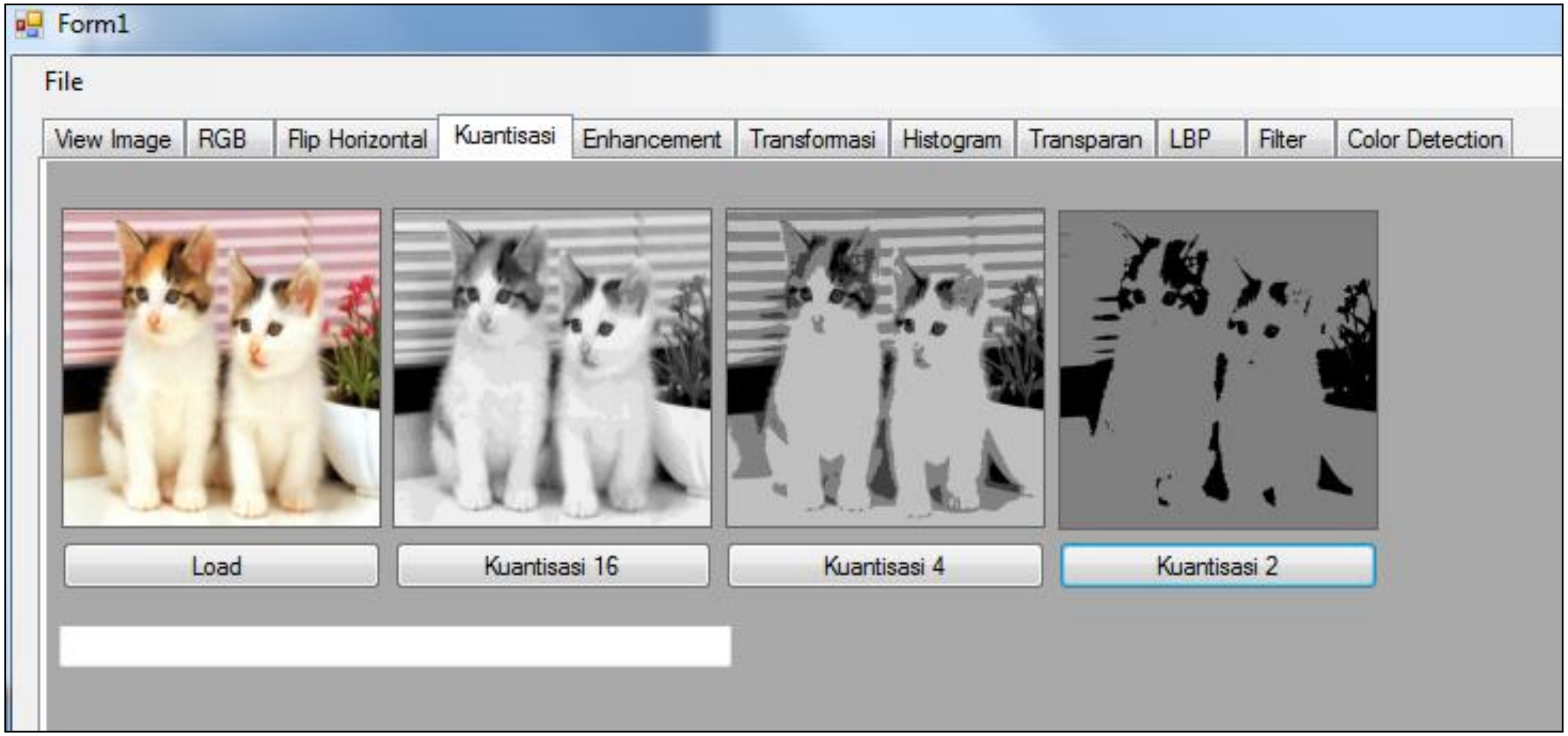
w adalah nilai derajat keabuan sebelum thresholding

x adalah nilai derajat keabuan setelah thresholding

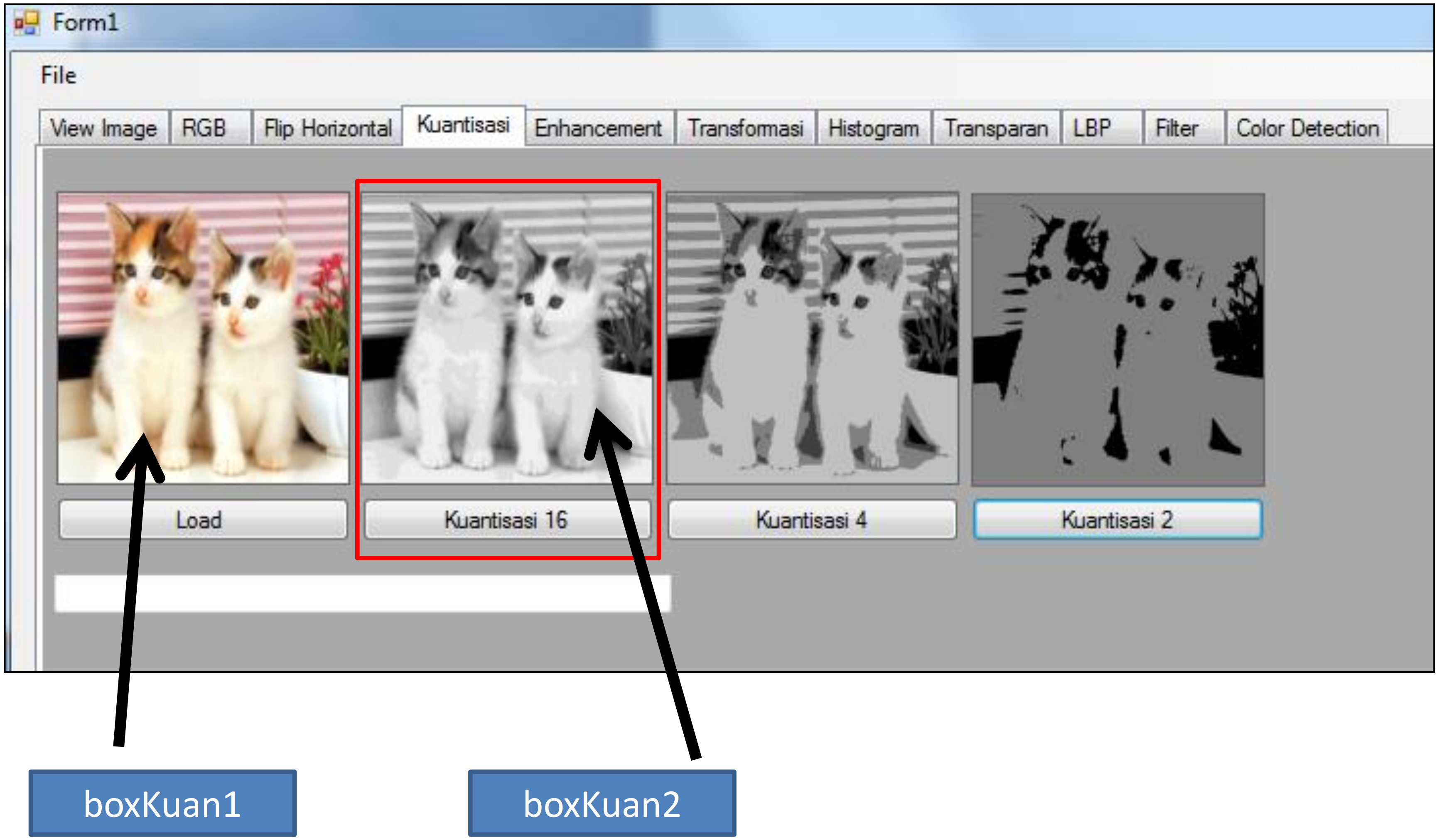
$$b = \text{int}\left({}^{256}\!/_{a}\right)$$

Berikut ini contoh thresholding mulai di 256, 16, 4 dan 2.

Kuantisasi

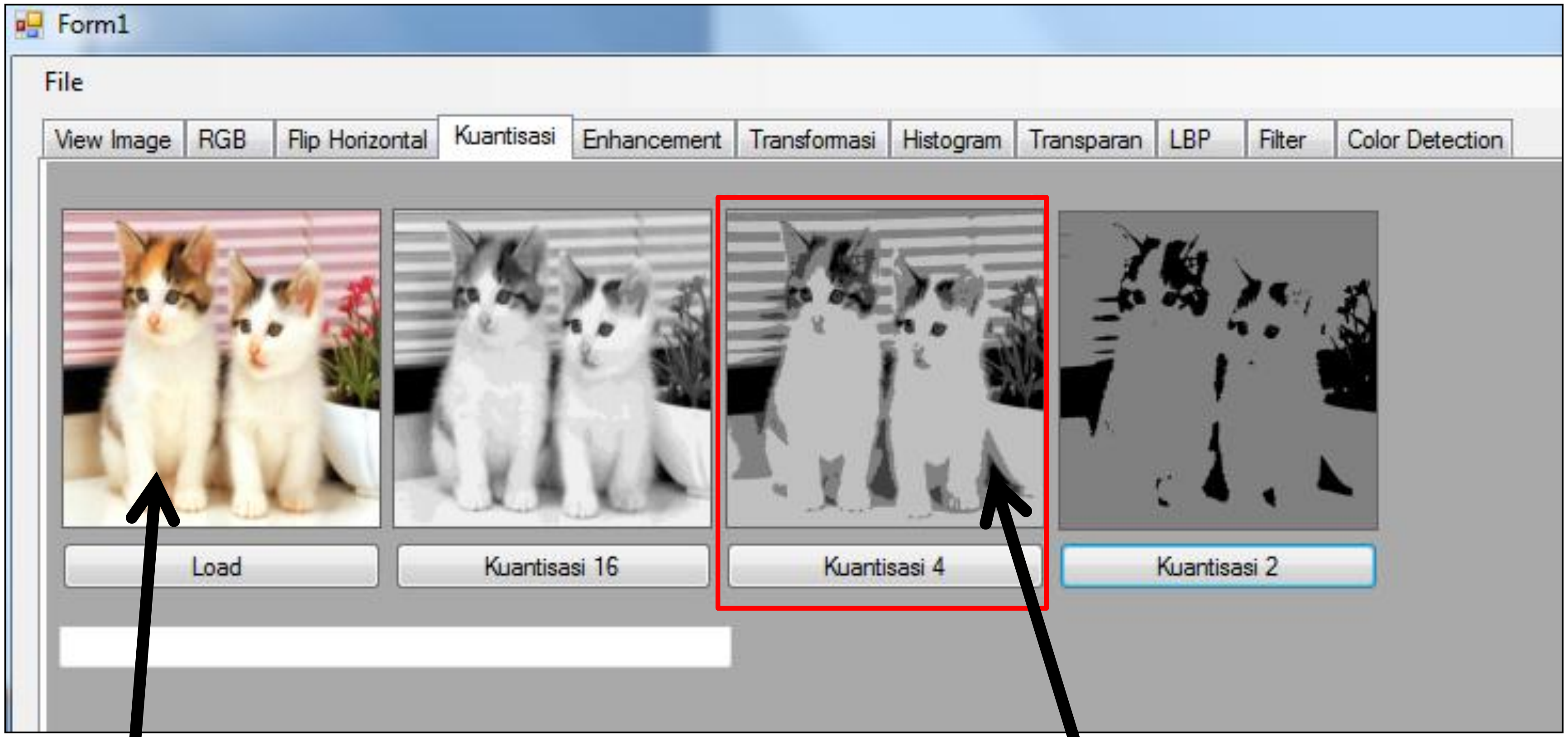

Form1
File
View Image
RGB
Flip Horizontal
Kuantisasi
Enhancement
Transformasi
Histogram
Transparan
LBP
Filter
Color Detection
Load
Kuantisasi 16
Kuantisasi 4
Kuantisasi 2

Kuantisasi 16



```
Bitmap bmp1 = (Bitmap)boxKuan1.Image;
Color pixelColor;
int K=16;
int th = (int) 256 / K;
for (int y = 0; y < bmp1.Height; y++)
{
    for (int x = 0; x < bmp1.Width; x++)
    {
        pixelColor = bmp1.GetPixel(x, y);
        int red = pixelColor.R;
        int green = pixelColor.G;
        int blue = pixelColor.B;
        int rata = (int)(red + green + blue) / 3;
        int kuantisasi = (int)(rata / th);
        int result = (int)th * kuantisasi;
        bmp1.SetPixel(x,y,Color.FromArgb(result,result,result));
    }
}
boxKuan2.Image = new Bitmap(boxKuan2.Width, boxKuan2.Height);
boxKuan2.SizeMode = PictureBoxSizeMode.StretchImage;
boxKuan2.Image = bmp1;
```

Kuantisasi 4

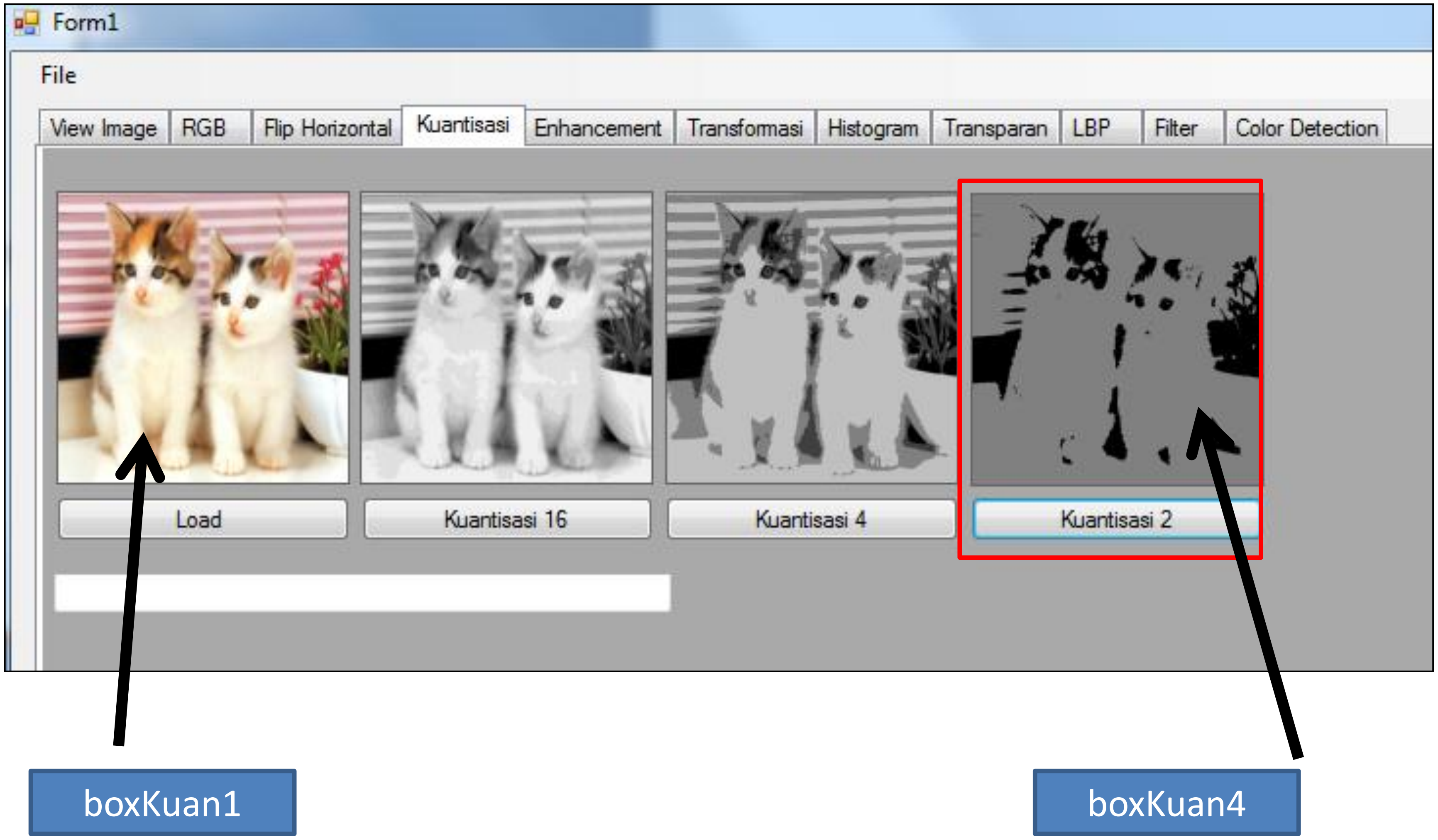


boxKuan1

boxKuan3

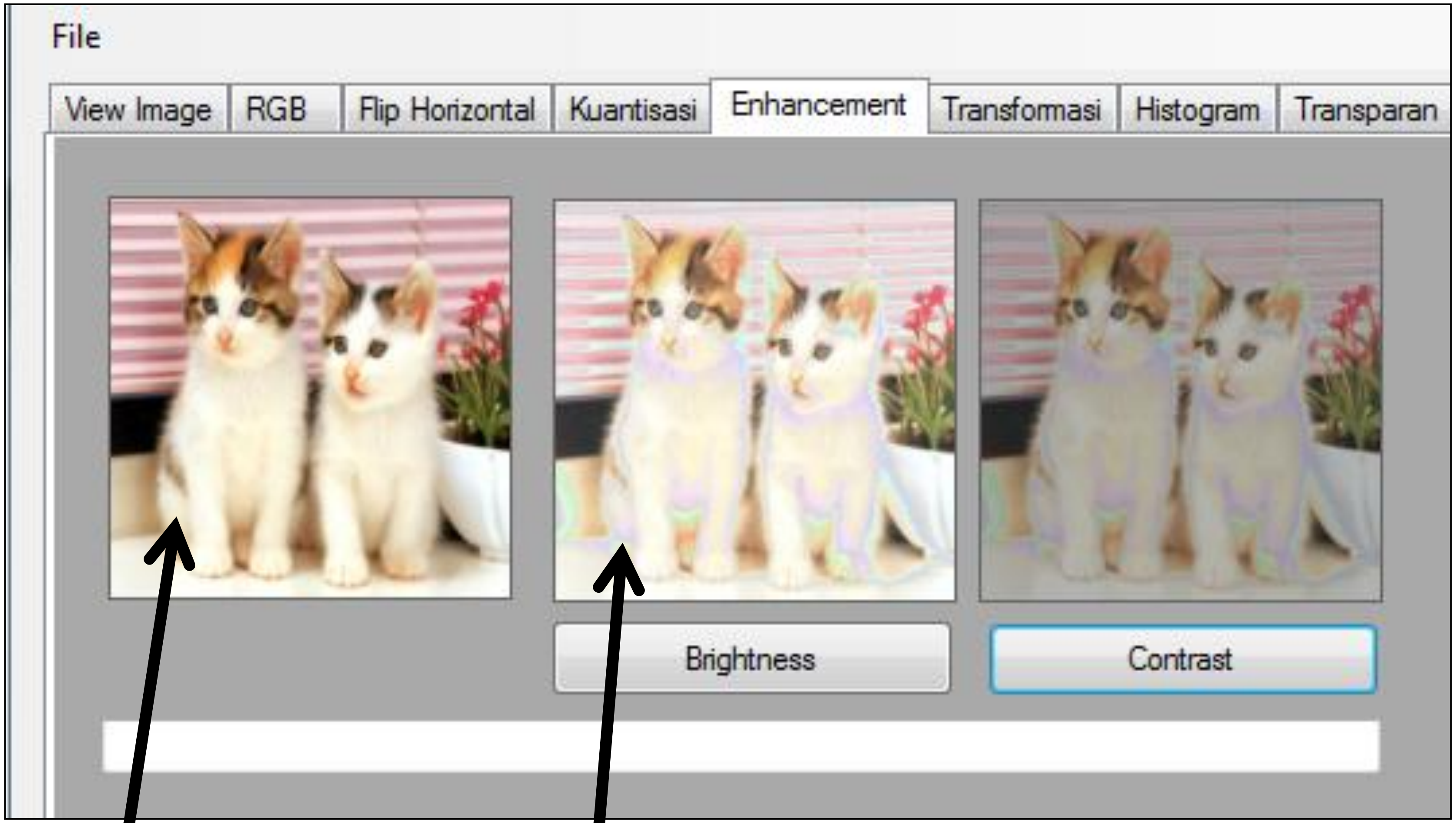
```
Bitmap bmp1 = (Bitmap)boxKuan1.Image;
Color pixelColor;
int K = 4;
int th = (int)256 / K;
for (int y = 0; y < bmp1.Height; y++)
{
    for (int x = 0; x < bmp1.Width; x++)
    {
        pixelColor = bmp1.GetPixel(x, y);
        int red = pixelColor.R;
        int green = pixelColor.G;
        int blue = pixelColor.B;
        int rata = (int)(red + green + blue) / 3;
        int kuantisasi = (int)(rata / th);
        int result = (int)th * kuantisasi;
        bmp1.SetPixel(x, y, Color.FromArgb(result, result, result));
    }
}
boxKuan3.Image = new Bitmap(boxKuan3.Width, boxKuan3.Height);
boxKuan3.SizeMode = PictureBoxSizeMode.StretchImage;
boxKuan3.Image = bmp1;
```

Kuantisasi 2



```
Bitmap bmp1 = (Bitmap)boxKuan1.Image;
Color pixelColor;
int K = 2;
int th = (int)256 / K;
for (int y = 0; y < bmp1.Height; y++)
{
    for (int x = 0; x < bmp1.Width; x++)
    {
        pixelColor = bmp1.GetPixel(x, y);
        int red = pixelColor.R;
        int green = pixelColor.G;
        int blue = pixelColor.B;
        int rata = (int)(red + green + blue) / 3;
        int kuantisasi = (int)(rata / th);
        int result = (int)th * kuantisasi;
        bmp1.SetPixel(x, y, Color.FromArgb(result, result, result));
    }
}
boxKuan4.Image = new Bitmap(boxKuan4.Width, boxKuan4.Height);
boxKuan4.SizeMode = PictureBoxSizeMode.StretchImage;
boxKuan4.Image = bmp1;
```

Enhancement

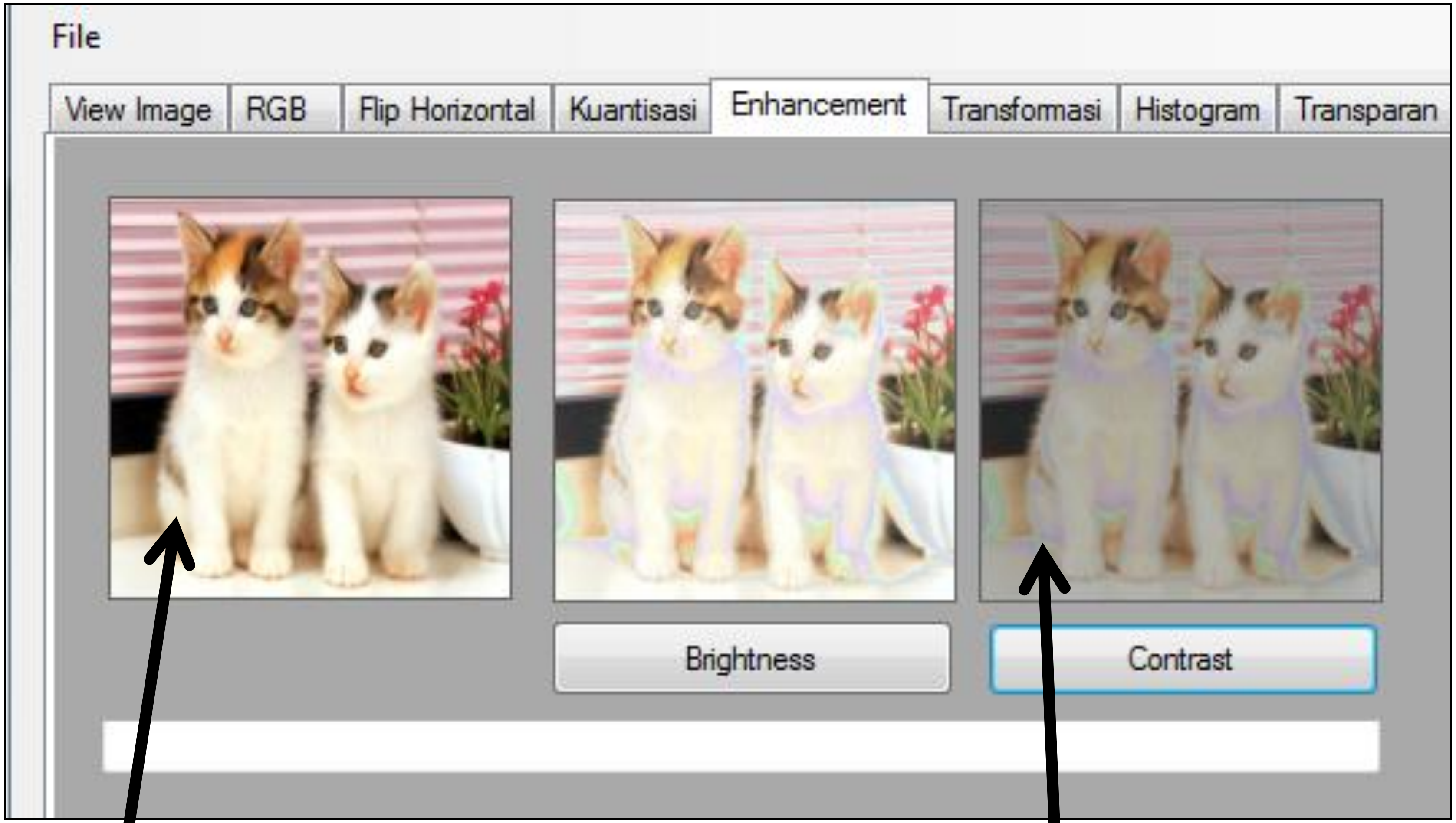


boxEn1

boxEn1

```
Bitmap bmp1 = (Bitmap)boxEn1.Image;
        Color pixelColor;
        int K = 50;
        for (int y = 0; y < bmp1.Height; y++)
        {
            for (int x = 0; x < bmp1.Width; x++)
            {
                pixelColor = bmp1.GetPixel(x, y);
                int red = pixelColor.R;
                int green = pixelColor.G;
                int blue = pixelColor.B;
                if ((red + K) <= 255) { red = red + K; }
                if ((green + K) <= 255) { green = green + K; }
                if ((blue + K) <= 255) { blue = blue + K; }
                bmp1.SetPixel(x, y, Color.FromArgb(red, green, blue));
            }
        }
        boxEn2.Image = new Bitmap(boxEn2.Width, boxEn2.Height);
        boxEn2.SizeMode = PictureBoxSizeMode.StretchImage;
        boxEn2.Image = bmp1;
```

Enhancement



boxEn1

boxEn3

```
Bitmap bmp1 = (Bitmap)boxEn1.Image;
Color pixelColor;
float K = 0.7f;
for (int y = 0; y < bmp1.Height; y++)
{
   for (int x = 0; x < bmp1.Width; x++)
   {
      pixelColor = bmp1.GetPixel(x, y);
      int red = pixelColor.R;
      int green = pixelColor.G;
      int blue = pixelColor.B;
      red   = (int) (K * red);
      green = (int) (K * green);
      blue  = (int) (K * blue);
      if (red > 255)   { red = 255; }
      if (green > 255) { green = 255; }
      if (blue > 255)  { blue = 255; }
      if (red < 0) { red = 0; }
      if (green < 0) { green = 0; }
      if (blue < 0) { blue = 0; }
      bmp1.SetPixel(x, y, Color.FromArgb(red, green, blue));
   }
}
boxEn3.SizeMode = PictureBoxSizeMode.StretchImage;
boxEn3.Image = bmp1;
```

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:

$$xn = 255 - x$$

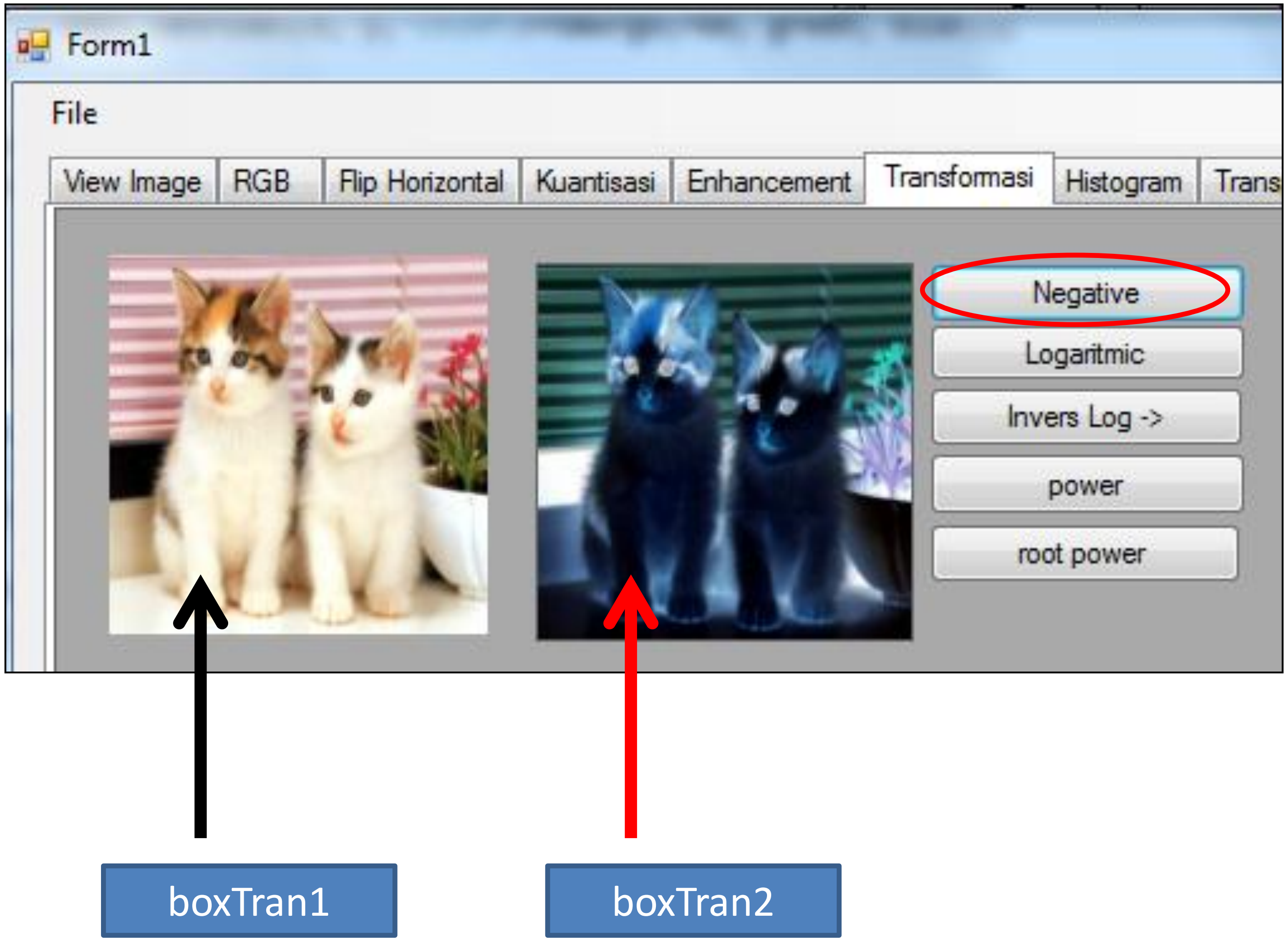
Tranformasi Logaritmik

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

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

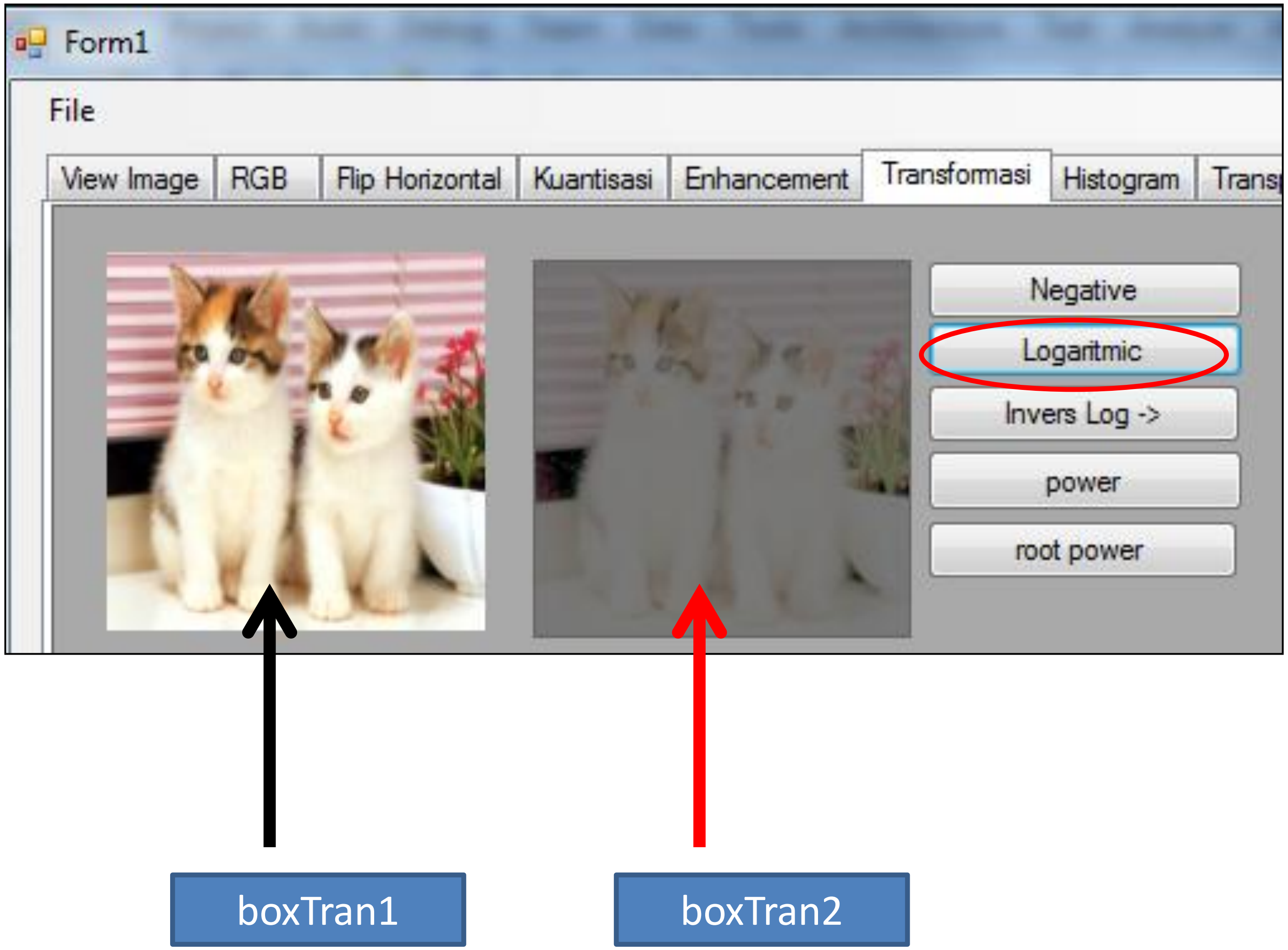
Dimana G adalah citra hasil, F citra asal, c adalah konstanta yang dipasang sebagai efek perubahan kontras

Transformasi



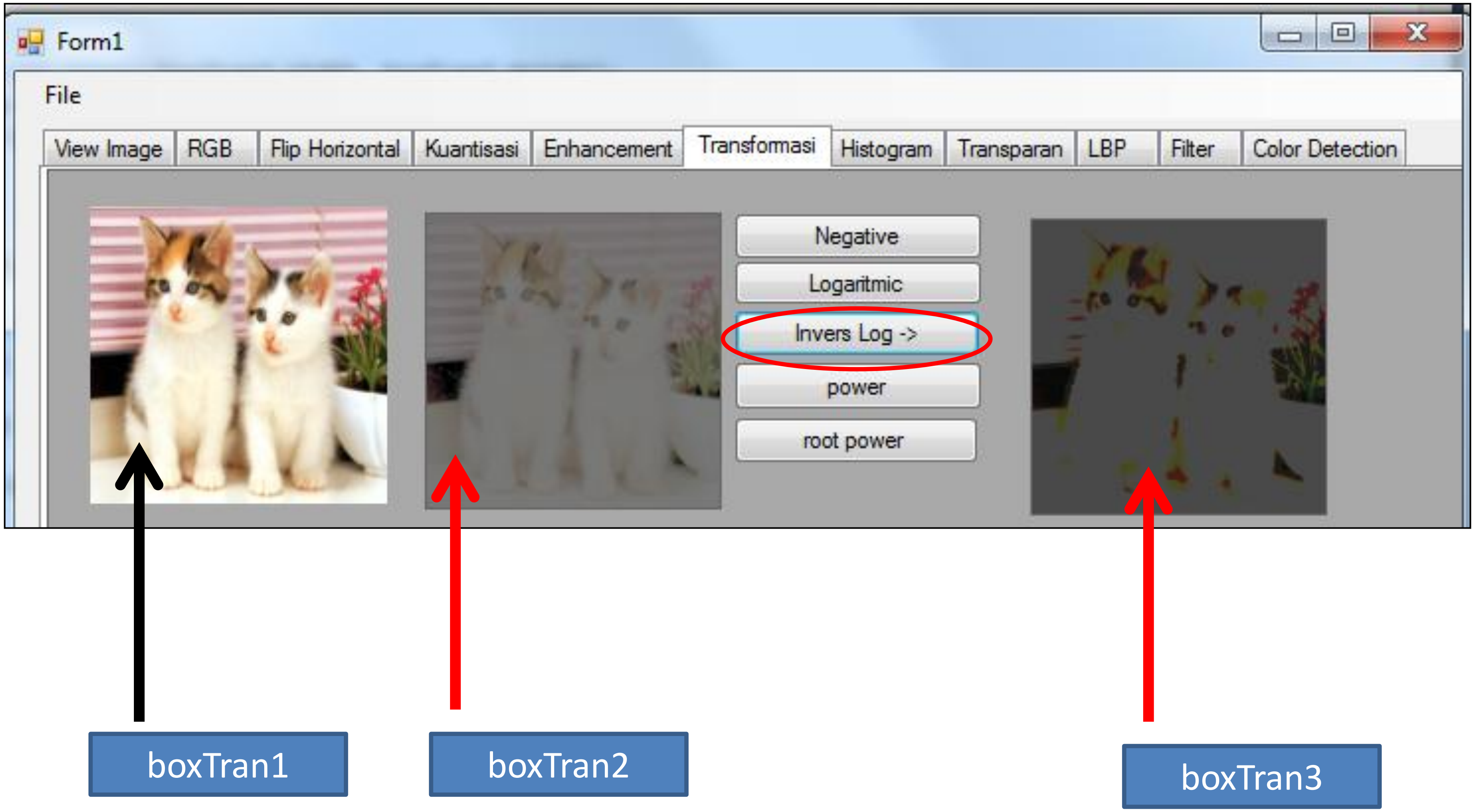
```
Bitmap bmp1 = (Bitmap)boxTran1.Image;
           Color pixelColor;
           for (int y = 0; y < bmp1.Height; y++)
           {
               for (int x = 0; x < bmp1.Width; x++)
               {
                   pixelColor = bmp1.GetPixel(x, y);
                   int red = 255- pixelColor.R;
                   int green = 255 - pixelColor.G;
                   int blue = 255 - pixelColor.B;
                   bmp1.SetPixel(x, y, Color.FromArgb(red, green, blue));
               }
           }
           boxTran2.Image = new Bitmap(boxTran2.Width, boxTran2.Height);
           boxTran2.SizeMode = PictureBoxSizeMode.StretchImage;
           boxTran2.Image = bmp1;
```

Transformasi



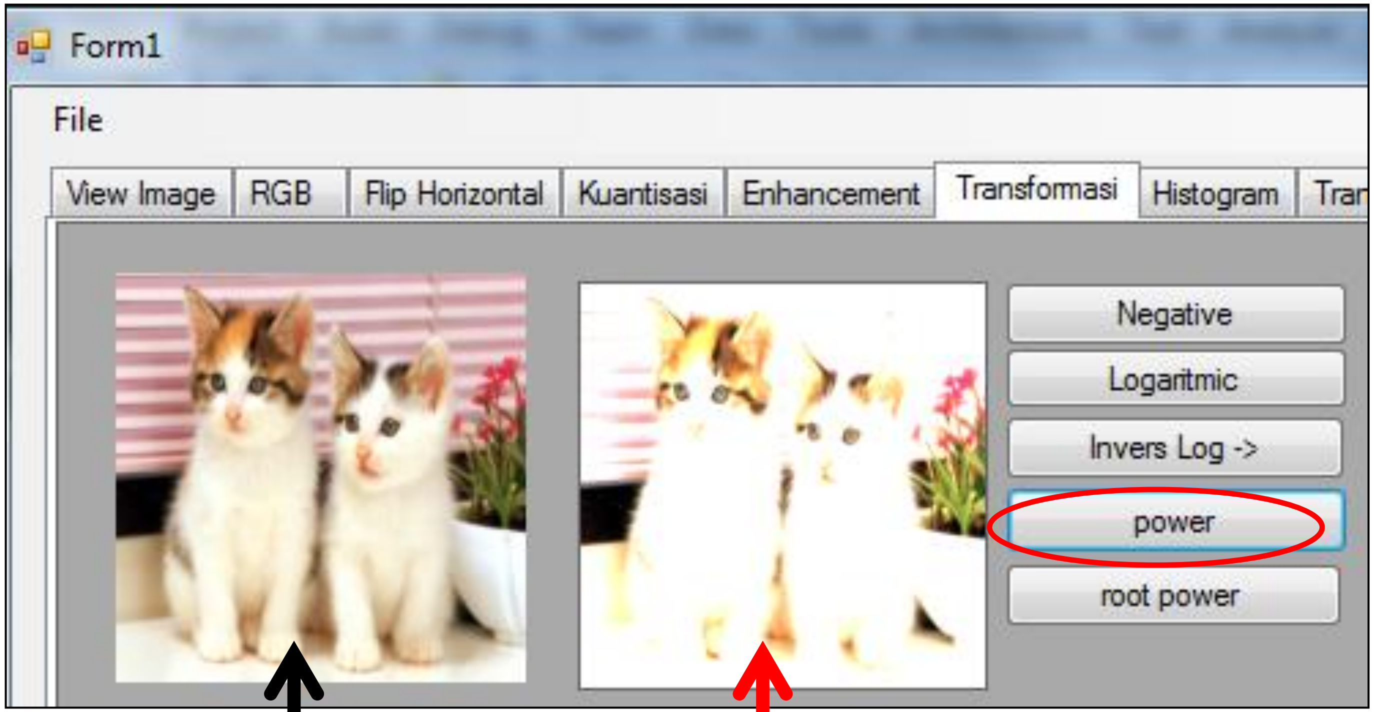
```
Bitmap bmp1 = (Bitmap)boxTran1.Image;
Color pixelColor;
int K = 50;
for (int y = 0; y < bmp1.Height; y++)
{
   for (int x = 0; x < bmp1.Width; x++)
   {
      pixelColor = bmp1.GetPixel(x, y);
      int red = pixelColor.R;
      int green = pixelColor.G;
      int blue =  pixelColor.B;
      red = (int) (K * Math.Log(red,10));
      green = (int)(K * Math.Log(green,10));
      blue = (int)(K * Math.Log(blue,10));
      if (red > 255) { red = 255; }
      if (green > 255) { green = 255; }
      if (blue > 255) { blue = 255; }
      if (red < 0) { red = 0; }
      if (green < 0) { green = 0; }
      if (blue < 0) { blue = 0; }
      bmp1.SetPixel(x, y, Color.FromArgb(red, green, blue));
   }
}
boxTran2.Image = new Bitmap(boxTran2.Width, boxTran2.Height);
boxTran2.SizeMode = PictureBoxSizeMode.StretchImage;
boxTran2.Image = bmp1;
```

Transformasi



```
Bitmap bmp1 = (Bitmap)boxTran2.Image;
Color pixelColor;
int K = 50;
for (int y = 0; y < bmp1.Height; y++)
{
    for (int x = 0; x < bmp1.Width; x++)
    {
        pixelColor = bmp1.GetPixel(x, y);
        int red = pixelColor.R;
        int green = pixelColor.G;
        int blue = pixelColor.B;
        red = (int)(10 * Math.Exp(red / K));
        green = (int)(10 * Math.Exp(green / K));
        blue = (int)(10 * Math.Exp(blue / K));
        if (red > 255) { red = 255; }
        if (green > 255) { green = 255; }
        if (blue > 255) { blue = 255; }
        if (red < 0) { red = 0; }
        if (green < 0) { green = 0; }
        if (blue < 0) { blue = 0; }
        bmp1.SetPixel(x, y, Color.FromArgb(red, green, blue));
    }
}
boxTran3.Image = new Bitmap(boxTran3.Width, boxTran3.Height);
boxTran3.SizeMode = PictureBoxSizeMode.StretchImage;
boxTran3.Image = bmp1;
```

Transformasi

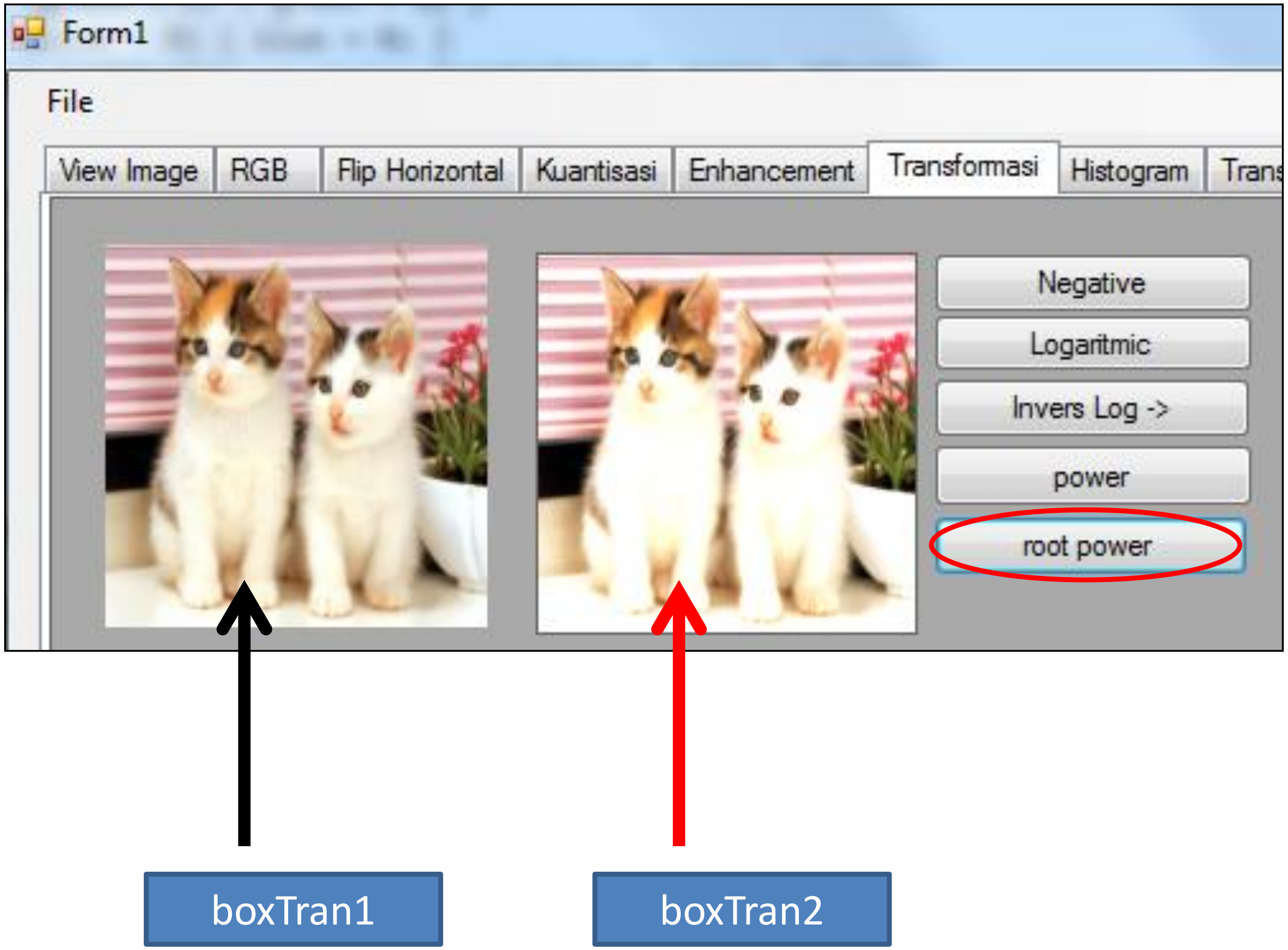


boxTran1

boxTran2

```
Bitmap bmp1 = (Bitmap)boxTran1.Image;
Color pixelColor;
float K = 0.4f;
float K2 = 1.5f;
for (int y = 0; y < bmp1.Height; y++)
{
    for (int x = 0; x < bmp1.Width; x++)
    {
        pixelColor = bmp1.GetPixel(x, y);
        int red = pixelColor.R;
        int green = pixelColor.G;
        int blue = pixelColor.B;
        red = (int)(K * red*Math.Exp(K2));
        green = (int)(K * green*Math.Exp(K2));
        blue = (int)(K * blue*Math.Exp(K2));
        if (red > 255) { red = 255; }
        if (green > 255) { green = 255; }
        if (blue > 255) { blue = 255; }
        if (red < 0) { red = 0; }
        if (green < 0) { green = 0; }
        if (blue < 0) { blue = 0; }
        bmp1.SetPixel(x, y, Color.FromArgb(red, green, blue));
    }
}
boxTran2.Image = new Bitmap(boxTran2.Width, boxTran2.Height);
boxTran2.SizeMode = PictureBoxSizeMode.StretchImage;
boxTran2.Image = bmp1;
```

Transformasi



```
Bitmap bmp1 = (Bitmap)boxTran1.Image;
Color pixelColor;
float K = 0.6f;
float K2 = 1.5f;
for (int y = 0; y < bmp1.Height; y++)
{
   for (int x = 0; x < bmp1.Width; x++)
   {
      pixelColor = bmp1.GetPixel(x, y);
      int red = pixelColor.R;
      int green = pixelColor.G;
      int blue = pixelColor.B;
      red = (int)(K * red * Math.Exp(1/K2));
      green = (int)(K * green * Math.Exp(1/K2));
      blue = (int)(K * blue * Math.Exp(1/K2));
      if (red > 255) { red = 255; }
      if (green > 255) { green = 255; }
      if (blue > 255) { blue = 255; }
      if (red < 0) { red = 0; }
      if (green < 0) { green = 0; }
      if (blue < 0) { blue = 0; }
      bmp1.SetPixel(x, y, Color.FromArgb(red, green, blue));
   }
}
boxTran2.Image = new Bitmap(boxTran2.Width, boxTran2.Height);
boxTran2.SizeMode = PictureBoxSizeMode.StretchImage;
boxTran2.Image = bmp1;
```