Aplikasi Geometry Process Menggunakan Visual Studio

Fajar Syakhfari

Fajar_060@yahoo.com

http://syakhfarizonedevils.blogspot.com

Pendahuluan

Pengolahan citra merupakan proses pengolahan dan analisis citra yang banyak melibatkan persepsi visual. Proses ini mempunyai ciri data masukan dan informasi keluaran yang berbentuk citra. Dalam definisi yang lebih luas, pengolahan citra digital juga mencakup semua data dua dimensi. Pengolahan citra bertujuan untuk memperbaiki kualitas citra agar mudah diinterpretasi oleh manusia atau mesin (dalam hal ini komputer).

Format data citra digital berhubungan erat dengan warna. Pada kebanyakan kasus, terutama untuk keperluan penampilan secara visual, nilai data digital merepresentasikan warna dari citra yang diolah. Adapun contoh aplikasi Pengolahan Citra Digital ini salah satu nya adalah Geometry Process yang akan saya jelaskan pada kesempatan ini.

Pada kesempatan ini penulis mengaplikasikan Pengolahan Citra Digital ini ke dalam Bahasa Pemrograman C# menggunakan Microsoft Visual Studio.

Isi

Apa itu Geometry Process ?

Geometry Process adalah memodifikasi susunan piksel berdasarkan pada beberapa transformasi geometri. Artinya adalah koordinat piksel berubah akibat transformasi, sedangkan intensitasnya tetap. Sedangkan dalam definisi bahasa pemrograman, Geometry process yaitu algoritma untuk memanipulasi lokasi atau koordinat pixel pada citra. Dalam pengolahan citra geometry process dibagi menjadi beberapa bagian, antara lain : Scalling (Penskalaan Citra), Rotasi, translasi, dan Flip / Mirroring (Pencerminan Citra).

Pengubahan geometry dari citra f(x,y) menjadi citra baru f ’(x ,y) dapat ditulis sebagai :

$$\mathbf{f'(x' , y') = f(g_1(x , y), g_2(x , y))}$$

dalam hal ini, $g_1(x , y)$ dan $g_2(x , y)$ adalah fungsi transformasi geometrik. Dengan kata lain,

$$\mathbf{x' = g_1(x , y)}$$
$$\mathbf{y' = g_2(x , y)}$$

1. **Scalling (Penskalaan Citra)**

Penskalaan citra, disebut juga *image zooming*, yaitu proses untuk mengubah ukuran citra asli (**zoom in** / memperbesar ukuran citra asli atau **zoom out** / memperkecil ukuran citra asli).

Hal – hal yang perlu diperhatikan pada proses penskalaan :

1. Resolusi citra tidak bisa ditingkatkan
2. Jumlah piksel yang dimiliki tidak lebih daripada jumlah yang ada pada citra asli.
3. Selalu ada degradasi citra :
   - Dalam memperbesar, holes harus diisi dengan nilai piksel tertentu melalui interpolasi ataupun *educated guest*.
   - Dalam memperkecil, ada piksel – piksel yang dihilangkan melalui cara perataan (*averaging*).

Rumus penskalaan dapat ditulis sebagai :

x' = Sx . x

y' = Sy . y

Keterangan : **Sx = faktor skala horizontal**

Sy = faktor skala vertikal

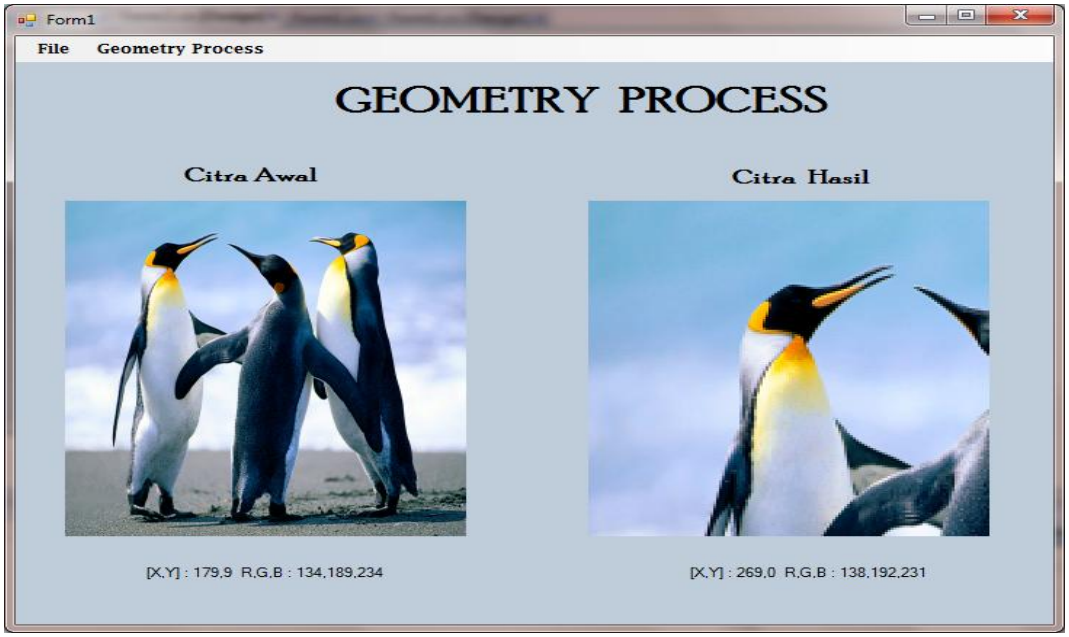
- **Zoom in (Pembesaran citra)**

Operasi zoom in dengan faktor skala = 2 (Sx = Sy = 2) diimplementasikan dengan menyalin setiap piksel sebanyak 4 kali. Jadi, citra 2 x 2 piksel akan menjadi 4 x 4 piksel.

Algoritma Zoom in (Pembesaran Citra) adalah sebagai berikut :

```
private void zoomInToolStripMenuItem_Click(object sender, EventArgs e)
{
  //Deklarasi citra baru berdasarkan 2 kali ukuran citra awal
    img1 = new Bitmap(img.Width * 2, img.Height * 2);
  //Struktur bersarang untuk meload semua pixel berdasarkan citra awal
    for (x = 0; x < img.Width; x++)
    {
       for (y = 0; y < img.Height; y++)
       {
           pixelColor = img.GetPixel(x, y);
         //Menyimpan pixel baru untuk citra yang baru
           img1.SetPixel(2 * x, 2 * y, pixelColor);
           img1.SetPixel((2 * x) + 1, 2 * y, pixelColor);
           img1.SetPixel(2 * x, (2 * y) + 1, pixelColor);
           img1.SetPixel((2 * x) + 1, (2 * y) + 1, pixelColor);
       }
     }
       //Menyimpan citra baru pada pictureBox2
         pictureBox2.Image = img1;
 }
```

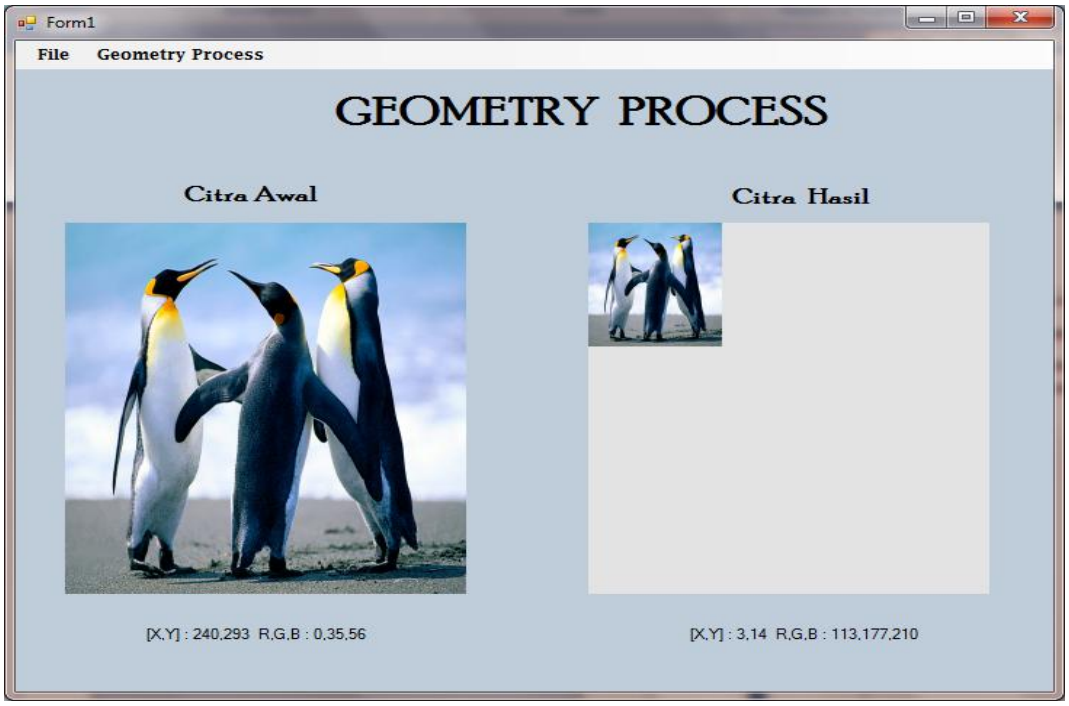
Hasil operasi zoom in pada citra adalah sebagai berikut :



- **Zoom out (Pengecilan citra)**

Operasi zoom out dengan faktor skala = $\frac{1}{2}$ dilakukan dengan mengambil rata-rata dari piksel yang bertetangga menjadi 1 piksel.

Hasil operasi zoom out pada citra adalah sebagai berikut :



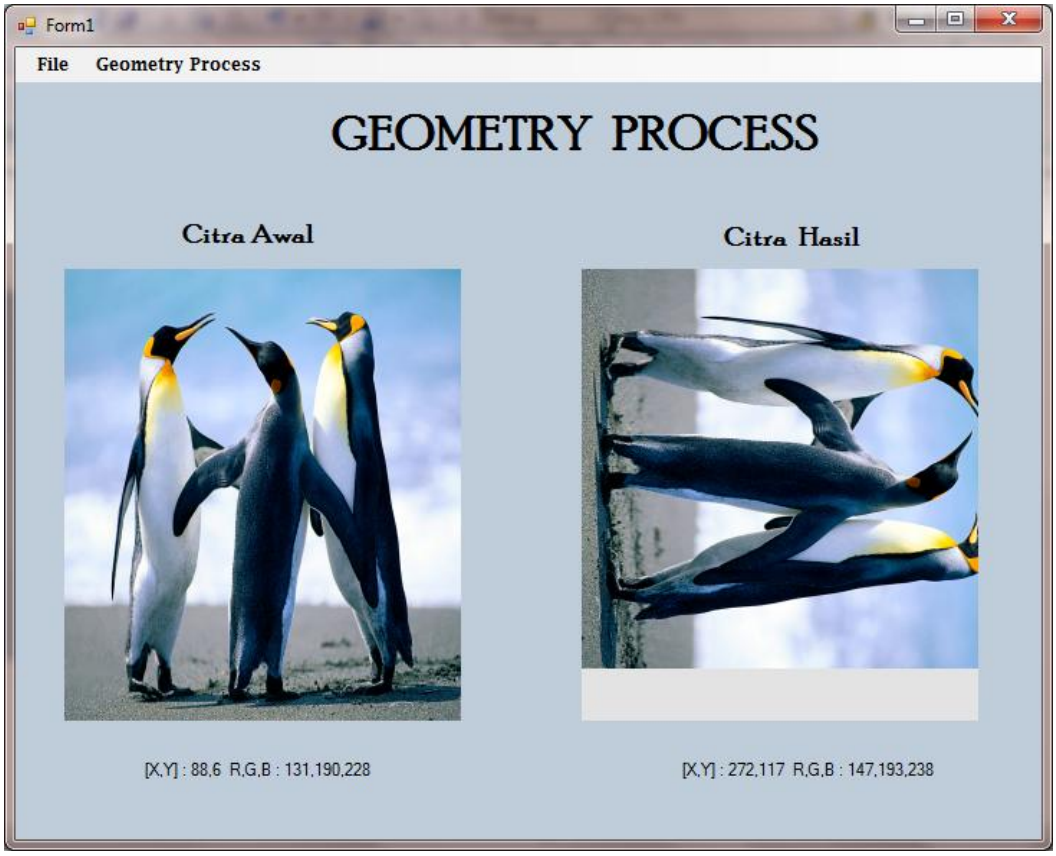
2. Rotasi (Pemutaran)

Rotasi adalah perputaran citra sesuai dengan arah perputaran dan besar sudut yang diinginkan. Jika sudut kelipatan 90^o, maka rotasi dapat dilakukan lebih sederhana yaitu dengan cara transposisi tanpa perlu floating point. Sedangkan rotasi 180^0, diimplementasikan dengan melakukan rotasi 90^o dua kali.

Algoritma Rotasi 90^o adalah sebagai berikut :

```
private void toolStripMenuItem2_Click(object sender, EventArgs e)
{
   //Deklarasi citra baru berdasarkan  ukuran citra awal
     img1 = new Bitmap(img.Height, img.Width);
   //Struktur bersarang untuk meload semua pixel berdasarkan citra awal
     for (int x = 0; x < img.Width; x++)
     {
       for (int y = 0; y < img.Height; y++)
       {
          pixelColor = img.GetPixel(x, y);
        //Menyimpan pixel baru untuk citra yang baru
          img1.SetPixel(img1.Width - 1 - y, x, pixelColor);
       }
     }
     pictureBox2.Image = img1;   //Menyimpan citra baru di pictureBox2
}
```

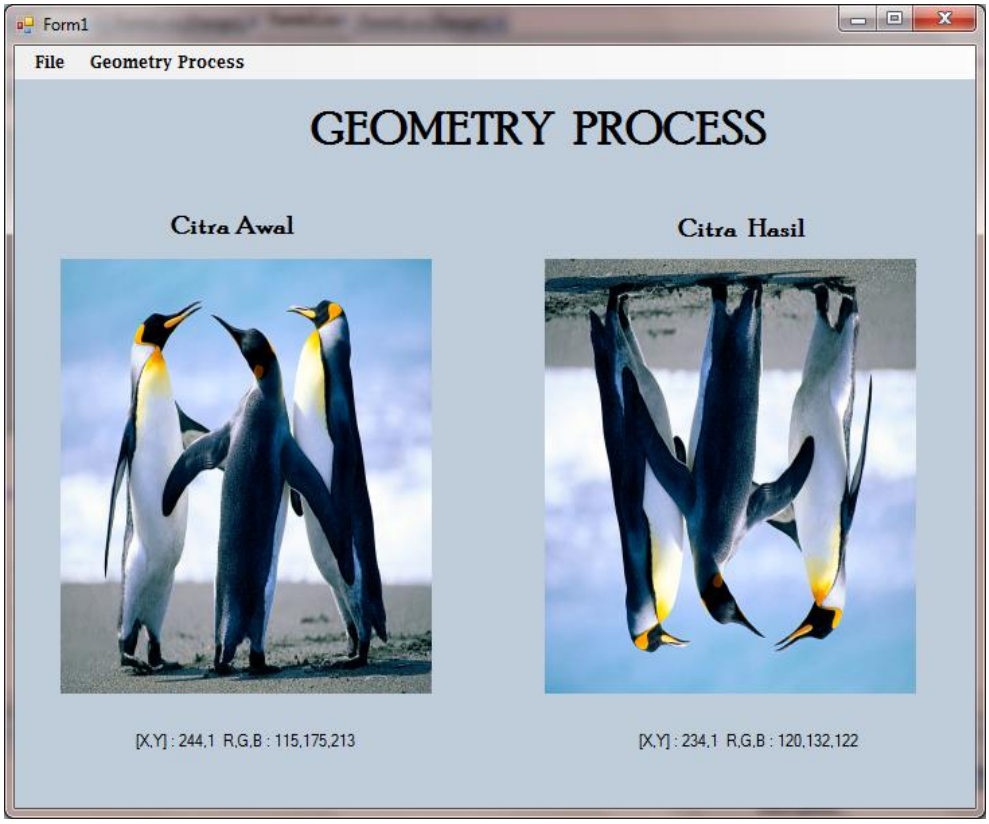
Hasil operasi Rotasi 90^o pada citra adalah sebagai berikut :



Algoritma Rotasi 180^o adalah sebagai berikut :

```
private void toolStripMenuItem3_Click(object sender, EventArgs e)
{
    //Deklarasi citra baru berdasarkan  ukuran citra awal
    img1 = new Bitmap(img.Width, img.Height);
    //Struktur bersarang untuk meload semua pixel berdasarkan citra awal
    for (int x = 0; x < img.Width; x++)
    {
        for (int y = 0; y < img.Height; y++)
        {
            pixelColor = img.GetPixel(x, y);
            //Menyimpan pixel baru untuk citra yang baru
            img1.SetPixel(img1.Width - 1 - x, img1.Height - 1 - y, pixelColor);
        }
    }
    pictureBox2.Image = img1; ;  //Menyimpan citra baru di pictureBox2
}
```

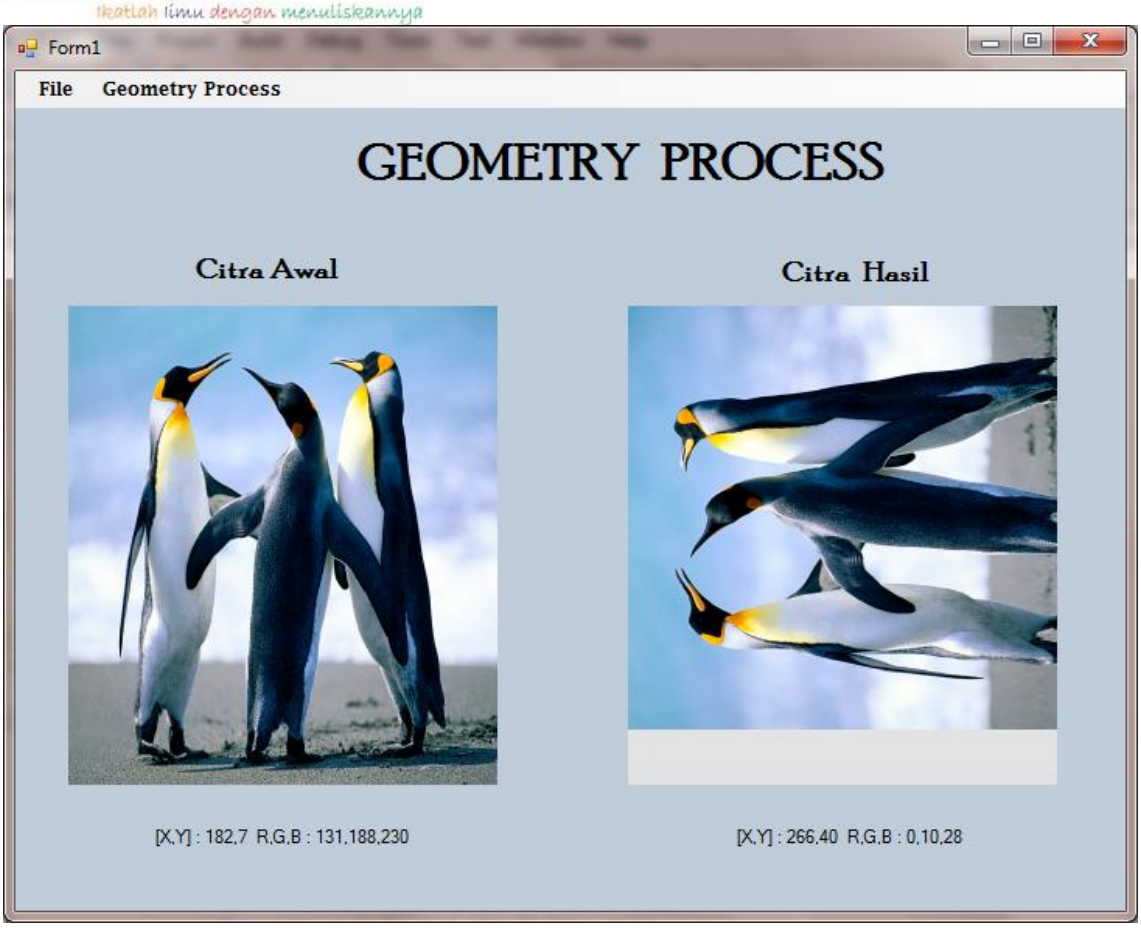
Hasil operasi Rotasi 180^{o} pada citra adalah sebagai berikut :



Algoritma Rotasi 270^{o} adalah sebagai berikut :

```
private void toolStripMenuItem4_Click(object sender, EventArgs e)
{
    //Deklarasi citra baru berdasarkan  ukuran citra awal
    img1 = new Bitmap(img.Height, img.Width);
    //Struktur bersarang untuk meload semua pixel berdasarkan citra awal
    for (int x = 0; x < img.Width; x++)
    {
        for (int y = 0; y < img.Height; y++)
        {
            pixelColor = img.GetPixel(x, y);
            //Menyimpan pixel baru untuk citra yang baru
            img1.SetPixel(y, img1.Height - 1 - x, pixelColor);
        }
    }
    pictureBox2.Image = img1;  //Menyimpan citra baru di pictureBox2
}
```

Hasil operasi Rotasi 270^{o} pada citra adalah sebagai berikut :



3. Translasi

Operasi Translasi merupakan operasi untuk mengubah posisi gambar melakukan penambahan atau pengurangan baik pada koordinat x dan atau koordinat y suatu citra.

Rumus Translasi dapat ditulis sebagai :

$nX = x + dX$

$nY = y + dY$

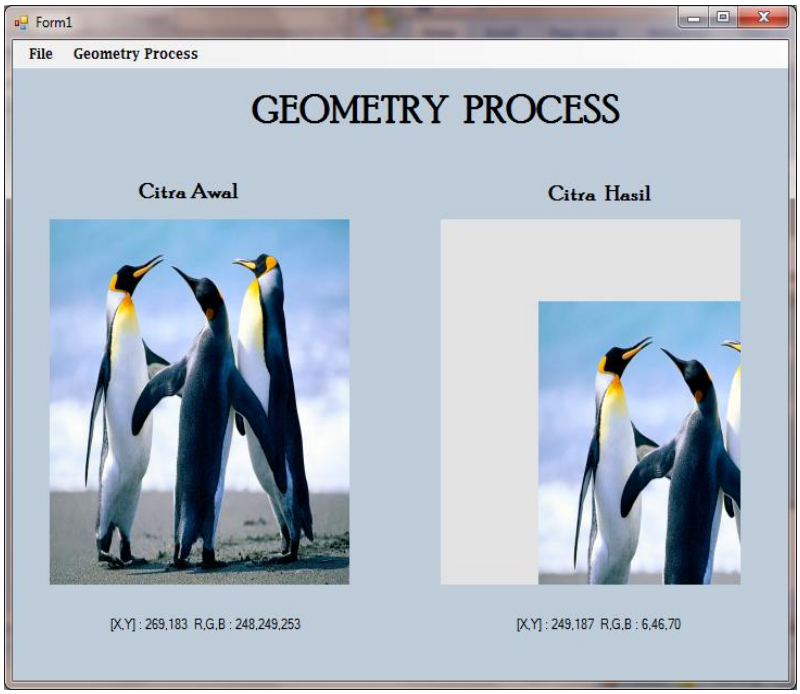
Keterangan : dX = besar pergeseran dalam arah x

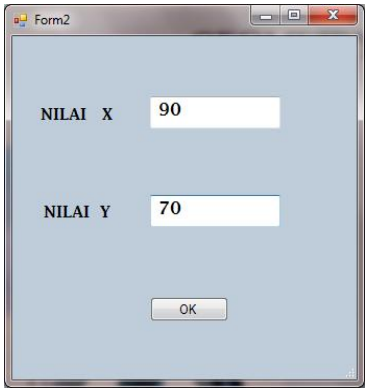
dY = besar pergeseran dalam arah y

Algoritma Translasi citra adalah sebagai berikut :

```
private void translasiToolStripMenuItem_Click(object sender, EventArgs e)
{
    Form2 f2 = new Form2(); // Deklarasi form2 dengan f2
    //Deklarasi citra baru berdasarkan  ukuran citra awal
    img1 = new Bitmap(img.Width, img.Height);
    if (f2.ShowDialog() == DialogResult.OK)
    {
        dX = Convert.ToInt32(f2.textBox1.Text);
        dY = Convert.ToInt32(f2.textBox2.Text);
        //Struktur bersarang untuk meload semua pixel berdasarkan citra awal
        for (int x = 0; x < img.Width; x++)
        {
            for (int y = 0; y < img.Height; y++)
            {
                if (((x + dX) >= 0) && ((x + dX) < img.Width) && ((y + dY) >= 0) && ((y + dY) < img.Height))
                {
                    pixelColor = img.GetPixel(x, y);
                    nX = x + dX;
                    nY = y + dY;
                    //Menyimpan pixel baru untuk citra yang baru
                    img1.SetPixel(nX, nY, pixelColor);
                }
            }
        }
        pictureBox2.Image = img1; //Menyimpan citra baru di pictureBox2
    }
}
```

Hasil operasi Translasi pada citra adalah sebagai berikut :





4. Flip / Mirroring (Pencerminan)

Flip / Mirroring adalah proses pencerminan citra berdasarkan cermin vertikal atau horizontal. Operasi pencerminan merupakan salah satu operasi geometri yang paling sederhana.

Berikut beberapa kriteria pada efek pencerminan :

- Horizontal mirroring → pencerminan pada sumbu y
- Vertikal mirroring → pencerminan pada sumbu x
- Gabungan → pencerminan pada sumbu x atau sumbu y.

Untuk mencerminkan gambar secara horizontal terhadap garis vertical di tengah gambar, maka digunakan rumus :

x' = w - 1 - x

Keterangan : w = lebar gambar

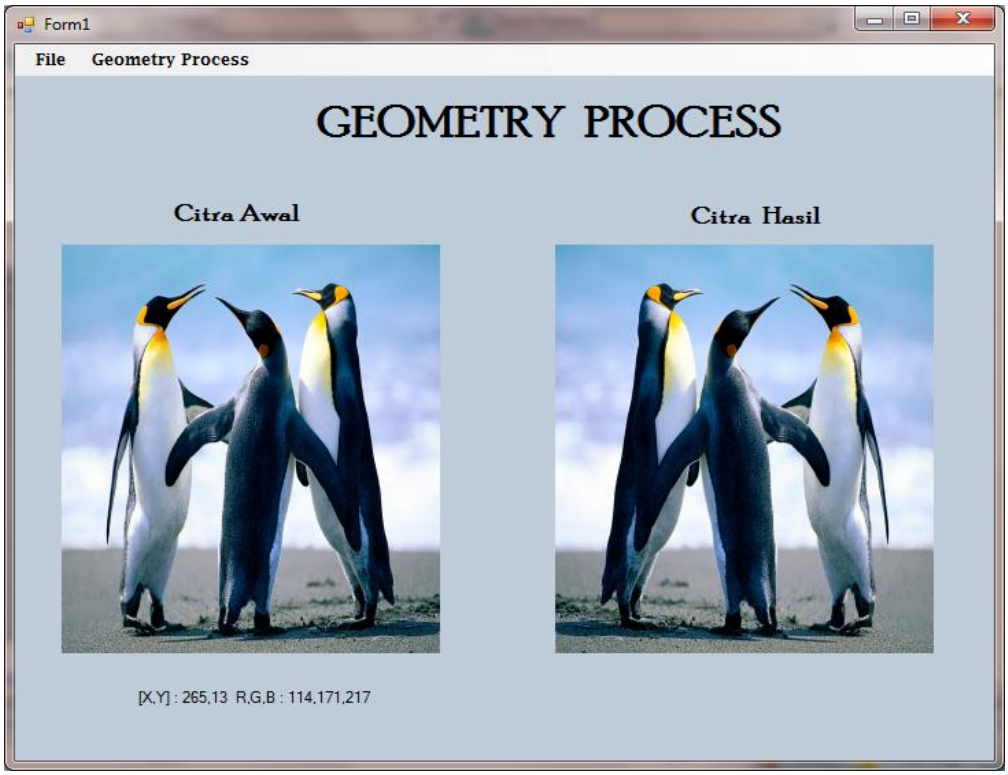
x = posisi awal koordinat x

x'= posisi hasil koordinat x

Algoritma Flip Horizontal citra adalah sebagai berikut :

```
private void horizontalToolStripMenuItem_Click(object sender, EventArgs e)
{
   //Deklarasi citra baru berdasarkan  ukuran citra awal
    img1 = new Bitmap(img.Width, img.Height);
   //Struktur bersarang untuk meload semua pixel berdasarkan citra awal
    for (x = 0; x < img.Width; x++)
    {
        for (y = 0; y < img.Height; y++)
        {
            //Mendapatkan data pixel pada area form
            pixelColor = img.GetPixel(x, y);
            //Menyimpan pixel baru untuk citra yang baru
            img1.SetPixel(img.Width - 1 - x, y, pixelColor);
        }
    }
    pictureBox2.Image = img1; //Menyimpan citra baru di pictureBox2
}
```

Hasil operasi Flip Horizontal pada citra adalah sebagai berikut :



Sedangkan untuk mencerminkan gambar secara vertikal terhadap garis horizontal di tengah gambar, digunakan rumus :

y' = h – 1 – y

Keterangan : h = tinggi gambar

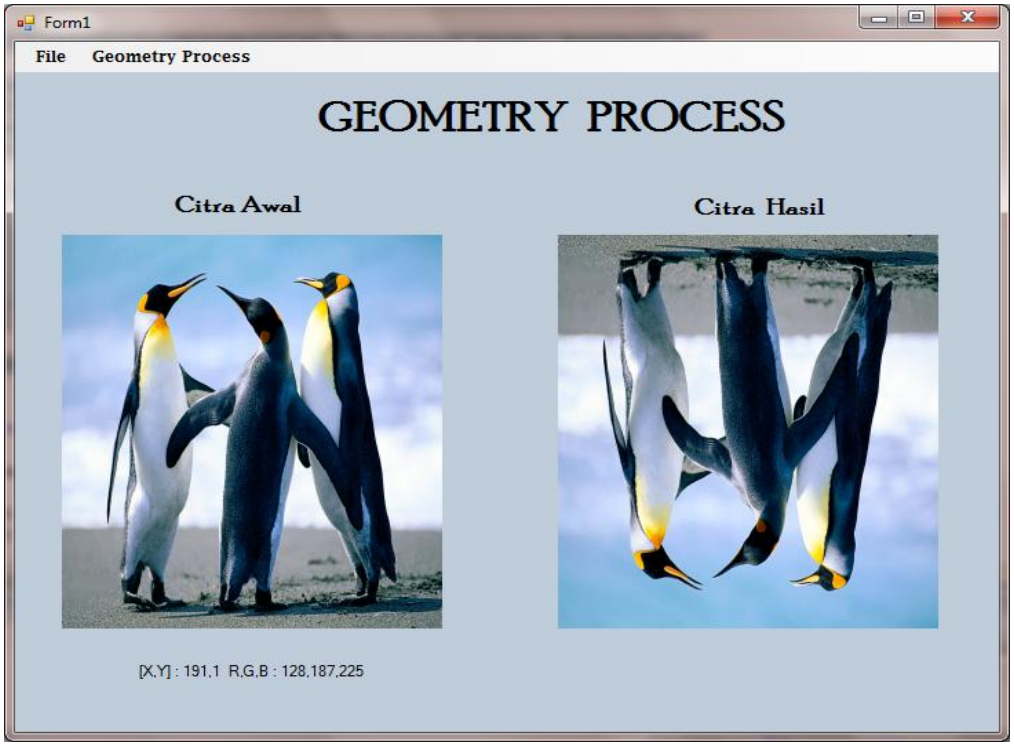
y = posisi awal koordinat y

y'= posisi hasil koordinat y

Algoritma Flip Vertikal citra adalah sebagai berikut :

```
private void vertikalToolStripMenuItem_Click(object sender, EventArgs e)
{
    //Deklarasi citra baru berdasarkan  ukuran citra awal
    img1 = new Bitmap(img.Width, img.Height);
    //Struktur bersarang untuk meload semua pixel berdasarkan citra awal
    for (x = 0; x < img.Width; x++)
    {
        for (y = 0; y < img.Height; y++)
        {
            //Mendapatkan data pixel pada area form
            pixelColor = img.GetPixel(x, y);
            //Menyimpan pixel baru untuk citra yang baru
            img1.SetPixel(x, img.Height - 1 - y, pixelColor);
        }
    }
    pictureBox2.Image = img1;  //Menyimpan citra baru di pictureBox2
}
```

Hasil operasi Flip Vertikal pada citra adalah sebagai berikut :



Penutup

Mudah-mudahan artikel ini sangat berguna bagi Anda. Source code lengkapnya terdapat pada file lampiran.

Keterangan : - Untuk menjalankan .exe program, dapat di buka di dalam folder geometryProcess → bin → Debug → geometryProcess.exe

- Untuk melihat listing program dapat di buka di dalam folder geometryProcess, pilih form1.cs lalu buka editor teks yang kita inginkan.

Referensi : - http://setiawanhadi.googlepages.com/geometri.pdf

Biografi Penulis

Fajar Syakhfari. Lahir di Bandung, 28 Oktober 1990. Menyelesaikan pendidikan menengah atas di SMAN 1 Cileunyi Bandung jurusan IPA tahun 2005-2008 dan saat ini sedang menempuh kuliah D3 jurusan Teknik Informatika di Universitas Padjadjaran Bandung angkatan 2008. Saat ini sedang fokus kepada bahasa pemrograman PHP.

Informasi lebih lanjut tentang penulis :
Email : fajar_060@yahoo.com
Blog : http://syakhfarizonedevils.blogspot.com