

Course IPCis: Image Processing with C#

Chapter C3: The Complete Code of the Filter Project

Copyright © by V. Miszalok, last update: 10-05-2010

Copy all this code into an empty Form1.cs of a new Windows Application C#-project filter1 and clear Form1.Designer.cs and Program.cs.

```
using System;
using System.Drawing;
using System.Drawing.Imaging;
using System.Windows.Forms;

public class Form1 : Form
{ [STAThread] static void Main() { Application.Run( new Form1() ); }
  Brush bbrush = SystemBrushes.ControlText;
  Bitmap Original, Noise, Lowpass, HighVertical, HighHorizontal, HighGradient;
  Byte[,] grayarray;
  Int32 x, y, xSize, ySize, gray;

  public Form1()
  { MenuItem miRead = new MenuItem( "&Read a (Small) Image File", new EventHandler( MenuFileRead ) );
    MenuItem miExit = new MenuItem( "&Exit", new EventHandler( MenuFileExit ) );
    MenuItem miFile = new MenuItem( "&Read Another Image Here !", new MenuItem[] { miRead, miExit } );
    Menu = new System.Windows.Forms.MainMenu( new MenuItem[] { miFile } );
    Text = "Filter1";
    SetStyle( ControlStyles.ResizeRedraw, true );
    Width = 1024;
    Height = 800;
    try { //Delete this and the following 6 lines if you have no Internet connection running.
      System.Net.WebRequest webreq =
        System.Net.WebRequest.Create( "http://www.miszalok.de/Images/Madonna.bmp" );
      System.Net.WebResponse webres = webreq.GetResponse();
      System.IO.Stream stream = webres.GetResponseStream();
      Original = (Bitmap)Image.FromStream( stream );
      DoNoiseAndFilters();
    } catch {}
  }

  private void MenuFileRead( object obj, EventArgs ea )
  { OpenFileDialog dlg = new OpenFileDialog();
    if ( dlg.ShowDialog() != DialogResult.OK ) return;
    try { Original = (Bitmap)Image.FromFile( dlg.FileName );
      DoNoiseAndFilters();
    } catch { return; }
  }

  private void DoNoiseAndFilters()
  { Graphics g = CreateGraphics();
    Rectangle cr = ClientRectangle;
    g.DrawImage( Original, 0, 0, cr.Width/3, cr.Height/2 );
    Cursor.Current = Cursors.WaitCursor;
    xSize = Original.Width;
    ySize = Original.Height;
    Noise = new Bitmap( xSize, ySize, PixelFormat.Format16bppRgb555 );
    Lowpass = new Bitmap( xSize, ySize, PixelFormat.Format24bppRgb );
    HighVertical = new Bitmap( xSize, ySize, PixelFormat.Format16bppRgb555 );
    HighHorizontal = new Bitmap( xSize, ySize, PixelFormat.Format16bppRgb555 );
    HighGradient = new Bitmap( xSize, ySize, PixelFormat.Format16bppRgb555 );
    grayarray = new Byte [xSize, ySize];
    for ( y=0; y < ySize; y++ )
      for ( x=0; x < xSize; x++ )
        { Color color = Original.GetPixel( x, y );
          gray = ( color.R + color.G + color.B ) / 3;
          grayarray[x, y] = (Byte)gray;
        }
    // Version 2 Noise
    Int32 amplitude = 256;
    Int32 half_amplitude = amplitude / 2;
    Random random = new Random();
    for ( y=0; y < ySize; y++ )
      for ( x=0; x < xSize; x++ )
        { gray = grayarray[x, y];
          gray += random.Next(amplitude) - half_amplitude;
          if (gray < 0) gray = 0; else if (gray > 255) gray = 255;
          Noise.SetPixel( x, y, Color.FromArgb( gray, gray, gray ) );
        }
    g.DrawImage( Noise, cr.Width/3, 0, cr.Width/3, cr.Height/2 );
  }
}
```

```

// Version 3: Lowpass without border handling
Int32 LowpassSize = 11; //insert an odd size: 3, 5, 7, ..., 29.
Int32 MidWeight = 1; //set a positive mid weight: 1, 2, 3, ..., 1000.
Int32 xx, yy, sum, d = LowpassSize/2;
float divisor = (2*d+1)*(2*d+1) + MidWeight - 1; //MidWeight == 1 means no weight
for ( y=d; y < ySize-d; y++ )
    for ( x=d; x < xSize-d; x++ )
        { sum = (MidWeight-1) * grayarray[x,y]; //extra mid weight
          for ( yy=-d; yy <= d; yy++ )
              for ( xx=-d; xx <= d; xx++ )
                  sum += grayarray[x+xx,y+yy];
          gray = Convert.ToByte( (float)sum / divisor );
          Lowpass.SetPixel( x, y, Color.FromArgb( gray, gray, gray ) );
        }
g.DrawImage( Lowpass, (2*cr.Width)/3, 0, cr.Width/3, cr.Height/2 );

// Version 4 HighHorizontal + HighVertical + Highgradient
Int32 sumleft, sumright, sumtop, sumbottom, diff;
for ( y=1; y < ySize-1; y++ )
    for ( x=1; x < xSize-1; x++ )
        { sumleft = grayarray[x-1,y] + grayarray[x-1,y-1] + grayarray[x-1,y+1];
          sumright = grayarray[x+1,y] + grayarray[x+1,y-1] + grayarray[x+1,y+1];
          sumtop = grayarray[x,y-1] + grayarray[x-1,y-1] + grayarray[x+1,y-1];
          sumbottom = grayarray[x,y+1] + grayarray[x-1,y+1] + grayarray[x+1,y+1];
          diff = Math.Abs( sumleft - sumright );
          if ( diff > 255 ) HighHorizontal.SetPixel( x,y,Color.FromArgb( 255, 255, 255 ) );
          else HighHorizontal.SetPixel( x,y,Color.FromArgb( diff, diff, diff ) );
          diff = sumtop - sumbottom; if ( diff < 0 ) diff *= -1;
          if ( diff > 255 ) HighVertical.SetPixel( x,y,Color.FromArgb( 255, 255, 255 ) );
          else HighVertical.SetPixel( x,y,Color.FromArgb( diff, diff, diff ) );
          Double diff_h = sumleft - sumright;
          Double diff_v = sumtop - sumbottom;
          diff = Convert.ToInt32( Math.Sqrt( diff_h * diff_h + diff_v * diff_v ) );
          if ( diff > 255 ) HighGradient.SetPixel( x,y,Color.FromArgb( 255, 255, 255 ) );
          else HighGradient.SetPixel( x,y,Color.FromArgb( diff, diff, diff ) );
        }
Cursor.Current = Cursors.Arrow;
Invalidate();
}

private void MenuFileExit( object obj, EventArgs ea )
{ Application.Exit(); }

protected override void OnPaint( PaintEventArgs e )
{ Graphics g = e.Graphics;
  Rectangle cr = ClientRectangle;
  g.DrawImage( Original , 0 , 0 , cr.Width/3, cr.Height/2 );
  g.DrawImage( Noise , cr.Width/3 , 0 , cr.Width/3, cr.Height/2 );
  g.DrawImage( Lowpass , (2*cr.Width)/3, 0 , cr.Width/3, cr.Height/2 );
  g.DrawImage( HighHorizontal, 0 , cr.Height/2, cr.Width/3, cr.Height/2 );
  g.DrawImage( HighVertical , cr.Width/3 , cr.Height/2, cr.Width/3, cr.Height/2 );
  g.DrawImage( HighGradient , (2*cr.Width)/3, cr.Height/2, cr.Width/3, cr.Height/2 );
}
}

```