

Course IPCis: Image Processing with C#

Chapter C3: The Filter Project

Copyright © by V. Miszalok, last update: 10-01-2008

- ↓ [An empty window](#)
- ↓ [Read and display an image, preparations for 6 images](#)
- ↓ [Code for noise](#)
- ↓ [Code for low pass](#)
- ↓ [Code for high passes](#)
- ↓ [Experiments](#)
- ↓ [Sample images](#)
- ↓ [Exercises](#)

An empty window

Guidance for **Visual Studio 2008**:

- 1) Main menu after start of VS 2008: File -> New Project... -> Visual Studio installed templates: Windows Forms Application
Name: filter1 -> Location: C:\temp -> Create directory for solution: switch off -> OK
Form1.cs[Design] appears.
- 2) Two superfluous files must be deleted: Form1.Designer.cs and Program.cs.
You reach these files via the Solution Explorer - filter1-window:
Click the plus-sign in front of branch filter1 and the plus-sign in front of branch Form1.cs.
Right-click the branch Program.cs. A context menu opens. Click Delete. A message box appears:
'Program.cs' will be deleted permanently. Quit with OK.
Right-click the branch Form1.Designer.cs and delete this file too.
- 3) Right-click the gray window Form1. A small context menu opens. Click View Code.
You see now the pre programmed code of Form1.cs. Erase this code completely.
- 4) Write the following three lines into the empty Form1.cs:

```
public class Form1 : System.Windows.Forms.Form
{ static void Main() { System.Windows.Forms.Application.Run( new Form1() ); }
}
```
- 5) Click Debug in the main menu of VS 2008.
A sub menu opens. Click Start Without Debugging Ctrl F5.

Read and display an image

Erase the three lines of code for the empty window in Form1.cs and replace them by:

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

public class Form1 : Form
{ 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", new EventHandler( MenuFileRead ) );
  MenuItem miExit = new MenuItem( "&Exit", new EventHandler( MenuFileExit ) );
  MenuItem miFile = new MenuItem( "&File", new MenuItem[] { miRead, miExit } );
  Menu = new System.Windows.Forms.MainMenu( new MenuItem[] { miFile } );
  Text = "Filter1";
  SetStyle( ControlStyles.ResizeRedraw, true );
  Width = 1024;
  Height = 800;
}

void MenuFileRead( object obj, EventArgs ea )
{ OpenFileDialog dlg = new OpenFileDialog();
  dlg.Filter = "bmp files (*.bmp)|*.bmp|All files (*.*)|*.*" ;
  if ( dlg.ShowDialog() != DialogResult.OK ) return;
  Original = (Bitmap)Image.FromFile( dlg.FileName );
  if ( Original == null ) return;
  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;
      }
  Cursor.Current = Cursors.Arrow;
  Invalidate();
}

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

protected override void OnPaint( PaintEventArgs e )
{ Graphics g = e.Graphics;
  if ( Original == null ) { g.DrawString( "Open an Image File !", Font, bbrush, 0, 0 ); return; }
  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 );
}
}

```

Click Debug -> Start Without Debugging Ctrl F5. Try to read all sorts of images: BMP, ICO, GIF, JPG, PNG, TIFF.

Code for noise

Version2: Finish filter1.

Write the following lines into void MenuFileRead(object obj, EventArgs ea) below the brace of the inner for-loop, but above Cursors.Current = Cursors.Arrow::

```
// 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 ) );
    }
```

Click Debug -> Start Without Debugging Ctrl F5. Add noise to the original image.

Code for low pass

Version3: Finish filter1.

Write the following lines into void MenuFileRead(object obj, EventArgs ea) below the brace of the inner for-loop, but above Cursors.Current = Cursors.Arrow::

```
// 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 ) );
    }
```

Click Debug -> Start Without Debugging Ctrl F5. Try out blurring

Code for high passes

Version3: Finish filter1.

Write the following lines into void MenuFileRead(object obj, EventArgs ea) below the brace of the inner for-loop, but above Cursors.Current = Cursors.Arrow::

```
// 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 ) );
    }
```

Click Debug -> Start Without Debugging Ctrl F5. Try out the high pass filters.

Experiments

- (1) Vary `amplitude` between 10 and 1000.
- (2) Vary `LowpassSize` between 3 and 29. Caution with higher sizes: Computing time rises proportional to `size*size`!
- (3) Vary `MidWeight` between 1 and 1000.
- (4) Use the Noise-image as input for Lowpass (a new `grayarray2` will be necessary).
- (5) Use the Lowpass-image as input for Highpass (a third `grayarray3` will be necessary).
- (6) Simplify `HighGradient` to `HighMaximum = Math.Max(HighpassVertical, HighpassHorizontal)`.

Sample images

The program displays all formats `BMP`, `ICO`, `GIF`, `JPG`, `PNG`, `TIFF`. If You use an old 8-bit graphics board or if You set Your desktop to 256 colors, the colors may look strange.

If You find no sample images on Your hard disk, use the following:

Download: [Butterfly.bmp 217 kB 24Bit-TrueColor-Bild](#)

Download: [Madonna.bmp 18 kB 8Bit-Grauwert-Bild](#)

Download: [Lena256.bmp 66 kB 8Bit-Grauwert-Bild](#)

Download: [Lena512.bmp 258 kB 8Bit-Grauwert-Bild](#)

Download: [Angiography.bmp 66 kB 8Bit-Grauwert-Bild](#)

Exercises

Click `Help` in the main menu of Visual Studio. Click the sub menu `Index`.

Branch to `Filtered by:` and choose `.NET Framework`. Then type to `Look for:` the following key words:

`PixelFormat` enumeration

`Cursor` class, all members

`Math` class, all members

Add more images by displaying several noise images with different `amplitude` and/or several low pass images with different `LowpassSize` and `MidWeight`. Adapt positions and sizes of the images to their number.

Try out different mouse pointers with `Cursors.WaitCursor`.

Try out different high pass filters with 8-neighbourhood pixels (+45 degrees and -45 degrees).

Finish Visual Studio, start the Explorer, delete the complete directory `C:\temp\filter1`

Start Visual Studio again and try to program everything from scratch.

Invent and try out new variants of the program in form of new projects `filter2`, `filter3`.