

Course CVCis: Computer Vision with C#

Chapter C5: The Complete Code of the Fourier Recognition Project

Copyright © by V. Miszalok, last update: 13-01-2008

Main Menu after start of VS 2005: File -> New Project... ->
 Visual Studio installed templates: Windows Forms Application
 Name: fourier_recogn1 -> Location: C:\temp -> Create directory for solution: switch
 off -> OK

Delete two superfluous files: Form1.Designer.cs and Program.cs.

Replace the preprogrammed code of Form1.cs by:

```
using System;
using System.Drawing;
using System.Windows.Forms;
using System.Collections;
using System.Text;
using System.IO;

public class Form1 : Form
{
    static void Main() { Application.Run( new Form1() ); }
    const Int32 xSize = 16;
    const Int32 ySize = 16;
    const Int32 no_of_fourier_values = 8;
    Byte[,] i0 = new Byte[ySize ,xSize ];
    Brush[] brush = new Brush[10];
    Brush blackbrush = SystemBrushes.ControlText;
    Brush redbrush = new SolidBrush( Color.Red );
    Brush bluebrush = new SolidBrush( Color.Blue );
    Pen redpen1 = new Pen( Color.Red, 1 );
    Pen redpen5 = new Pen( Color.Red, 5 );
    Pen whitepen;
    Font arial10 = new Font( "Arial", 10 );
    Font arial20 = new Font( "Arial", 20 );
    Int32 i, j, x, y, dx, dy;
    Byte threshold = 1;
    Button[] button = new Button[ySize];
    TextBox textbox;
    Point p0, p1;
    struct crackcode { public Point start; public String cracks; }
    struct prototype { public char name; public float sum_of_deviations; }
    crackcode cc;
    char recogn_result;
    Graphics g;
    ArrayList pointArray = new ArrayList();
    float[] fr, fi;

    public Form1()
    {
        BackColor = Color.White; Text = "Simple Numeral Recognition";
        SetStyle( ControlStyles.ResizeRedraw, true );
        Width = 800; Height = 600;
        for ( i=0; i < 10; i++ )
            brush[i] = new SolidBrush(Color.FromArgb( i*25, i*25, i*25 ) );
        for ( y=0; y < ySize; y++ )
        {
            button[y] = new Button();
            Controls.Add( button[y] );
            button[y].BackColor = Color.Gray; button[y].Text = "nothing";
            button[y].Name = y.ToString(); button[y].Click += new EventHandler( do_it );
        }
        button[0].Name = button[0].Text = "Recognize";
        button[1].Name = button[1].Text = "Teach As";
        button[2].Name = button[2].Text = "Clear";
        button[3].Name = "Threshold";
        button[4].Name = button[4].Text = "Noise";
        button[3].Text = String.Format( "Threshold={0:#}", threshold );
        textbox = new TextBox(); Controls.Add( textbox );
        cc.cracks = "";
    }
}
```

```

}
protected override void OnPaint( PaintEventArgs e )
{ g = e.Graphics;
  String s;
  Rectangle r = ClientRectangle;
  dx = r.Width / ( xSize+2 );
  dy = ( r.Height - 2 * FontHeight ) / ySize;
  for ( y=0; y < ySize; y++ )
  { button[y].Top = y*dy+1;
    button[y].Left = xSize*dx+1;
    if ( y != 1) button[y].Width = r.Width - button[y].Left - 2;
    else      button[y].Width = r.Width - button[y].Left - 25;
    button[y].Height = dy-2;
  }
  textbox.Top    = button[1].Top + ( button[1].Height - textbox.Height ) / 2;
  textbox.Left   = button[1].Left + button[1].Width + 1;
  textbox.Width  = button[0].Width - button[1].Width - 2;
  textbox.TextAlign = HorizontalAlignment.Center;
  for ( y=0; y < ySize; y++ )
    for ( x=0; x < xSize; x++ )
      g.FillRectangle(brush[i0[y,x]], x*dx, y*dy, dx, dy );
  for ( x=0; x < xSize; x++ ) g.DrawLine( redpen1, x*dx, 0, x*dx, ySize*dy );
  for ( y=0; y < ySize; y++ ) g.DrawLine( redpen1, 0, y*dy, xSize*dx, y*dy );
  if ( cc.cracks.Length == 0 )
  { s = "Draw a digit or a letter -> Recognize -> Teach As -> Clear -> Next";
    g.DrawString( s, arial10, redbrush, (xSize*dx)/4, 0 );
    return;
  }
  x = cc.start.X;
  y = cc.start.Y;
  for ( i = 0; i < cc.cracks.Length; i++ )
    switch ( cc.cracks[i] )
    { case 'e': g.DrawLine( redpen5, x*dx, y*dy, (x+1)*dx, y*dy ); x++; break;
      case 's': g.DrawLine( redpen5, x*dx, y*dy, x*dx, (y+1)*dy ); y++; break;
      case 'w': g.DrawLine( redpen5, x*dx, y*dy, (x-1)*dx, y*dy ); x--; break;
      case 'n': g.DrawLine( redpen5, x*dx, y*dy, x*dx, (y-1)*dy ); y--; break;
    }
  g.FillRectangle( bluebrush, x*dx-5, y*dy-5, 11, 11 );
  s = "This is a '" + recogn_result + "' .";
  g.DrawString( s, arial20, redbrush, 0, button[ySize-1].Top + button[ySize-1].Height + 2
);
}

protected override void OnMouseDown( MouseEventArgs e )
{ p0.X = e.X;
  p0.Y = e.Y;
  pointArray.Add(p0);
  whitepen = new Pen( Color.White, dx < dy ? dx : dy );
  whitepen.StartCap = whitepen.EndCap = System.Drawing.Drawing2D.LineCap.Round;
}

protected override void OnMouseMove( MouseEventArgs e )
{ if ( e.Button == MouseButton.None ) return;
  p1.X = e.X;
  p1.Y = e.Y;
  //Insert an additional point if the distance is too far
  if ( Math.Abs( p1.X - p0.X ) >= dx || Math.Abs( p1.Y - p0.Y ) >= dy )
    pointArray.Add( new Point((p1.X + p0.X)/2, (p1.Y + p0.Y)/2) );
  pointArray.Add( p1 );
  g = this.CreateGraphics();
  g.DrawLine( whitepen, p0, p1 );
  p0 = p1;
}
}

```

```

protected void do_it( object sender, System.EventArgs e )
{ switch( ((Button)sender).Name )
  { case "Threshold"://*****
    if ( ++threshold > 9 ) threshold = 1;
    button[3].Text = "Threshold=" + threshold.ToString();
    cc.cracks = ""; recogn_result = '?'; break;
  case "Noise"://*****
    Random random = new Random();
    for ( y=0; y < ySize; y++ )
      for ( x=0; x < xSize; x++ )
        { Int32 noise = random.Next() % 3 - 1;//gives -1 or 0 or +1
          noise += i0[y,x]);//add former gray value
          if (noise < 0) i0[y,x] = 0;
          else if (noise > 9) i0[y,x] = 9;
          else i0[y,x] = (Byte)noise;
        }
    cc.cracks = ""; recogn_result = '?'; break;
  case "Clear"://*****
    for ( y=0; y < ySize; y++ )
      for ( x=0; x < xSize; x++ ) i0[y,x] = 0;
    threshold = 1; button[3].Text = "Threshold=1";
    cc.cracks = ""; pointArray.Clear(); recogn_result = '?'; break;
  case "Recognize"://*****
    //Digitize*****
    for ( i = 0; i < pointArray.Count; i++ )
      { Point vertex = (Point)pointArray[i];
        x = vertex.X / dx;
        y = vertex.Y / dy;
        try { if ( i0[y,x] < 9 ) i0[y,x]++; } catch{};
      }
    //Crack Code 8*****
    //Search for a vertical start crack
    Byte leftpix;
    for ( y=0; y < ySize; y++ )
      for ( x=0; x < xSize; x++ )
        { if ( x > 0 ) leftpix = i0[y,x-1]; else leftpix = 0;
          if ( leftpix < threshold && i0[y,x] >= threshold )
            { cc.start.X = x; cc.start.Y = y; goto start_crack_found; }
        }
    return; //nothing to start with
    start_crack_found: y++;
    ArrayList phil = new ArrayList(); phil.Add( -90 );
    System.Text.StringBuilder cracks = new System.Text.StringBuilder();
    cracks.Append( 's' );
    Char last_crack = 's';
    do
      { switch ( last_crack )
        { case 'e': if ( x == xSize ) { phil.Add(-90); goto n; }
                  if ( y < ySize && i0[y ,x ] >= threshold) { phil.Add(+90); goto s; }
                  if ( i0[y-1,x ] >= threshold) { phil.Add( 0); goto e; }
                  { phil.Add(-90); goto n; }

                  case 's': if ( y == ySize ) { phil.Add(-90); goto e; }
                              if ( x > 0 && i0[y ,x-1] >= threshold) { phil.Add(+90); goto w; }
                              if ( i0[y ,x ] >= threshold) { phil.Add( 0); goto s; }
                              { phil.Add(-90); goto e; }

                  case 'w': if ( x == 0 ) { phil.Add(-90); goto s; }
                              if ( y > 0 && i0[y-1,x-1] >= threshold) { phil.Add(+90); goto n; }
                              if ( i0[y ,x-1] >= threshold) { phil.Add( 0); goto w; }
                              { phil.Add(-90); goto s; }

                  case 'n': if ( y == 0 ) { phil.Add(-90); goto w; }
                              if ( x < xSize && i0[y-1,x ] >= threshold) { phil.Add(+90); goto e; }
                              if ( i0[y-1,x-1] >= threshold) { phil.Add( 0); goto n; }
                              { phil.Add(-90); goto w; }
                }
        e: last_crack = 'e'; cracks.Append( 'e' ); x++; continue;
        s: last_crack = 's'; cracks.Append( 's' ); y++; continue;
        w: last_crack = 'w'; cracks.Append( 'w' ); x--; continue;
        n: last_crack = 'n'; cracks.Append( 'n' ); y--; continue;
      } while ( x != cc.start.X || y != cc.start.Y ); //end of do
    cc.cracks = cracks.ToString();
    Int32 NN, N = cc.cracks.Length;

```

```

//Bad property of function phil: All phil[i] sum up to -360 degrees
float[] phi2 = new float[N];
float circular_angle_per_crack = 360.0f / N;
for ( i=0; i < N; i++ )//Construction of a phi2 with a sum of 0
{ phi2[i] = (Int32)phil[i] + circular_angle_per_crack;
  phi2[i] /= 10.0f;//not important, just to keep the fourier-values low
}
phil.Clear();
//Fourier Transform*****
fr = new float[N/2+1];
fi = new float[N/2+1];
FourierTransform( phi2, fr, fi );
//Compare the Fourier values with the prototypes stored by former teachings
StreamReader instream;
try { instream = new StreamReader( "FourierRecognitionPrototypes.txt" ); }
catch { recogn_result = '?'; break; } //there are no prototypes yet
ArrayList comparisons = new ArrayList();
prototype proto = new prototype();
if ( no_of_fourier_values < fr.Length ) NN = no_of_fourier_values;
else NN = fr.Length;
string a_line;
while ( ( a_line = instream.ReadLine() ) != null )//new line
{ String[] s = a_line.Split( ' ', '/' );//split line into an array of substrings
  if ( s.Length < NN*2 + 1 ) continue;//not enough values in this line
  proto.name = (s[0])[0];//first character of the first substring
  proto.sum_of_deviations = 0;
  for ( i=0, j=1; i < NN; i++, j+=2 )
  { float fr2 = Convert.ToSingle( s[j] );
    float fi2 = Convert.ToSingle( s[j+1] );
    float dr = fr[i] - fr2;
    float di = fi[i] - fi2;
    proto.sum_of_deviations += dr*dr + di*di;
  }
  comparisons.Add(proto);
}
instream.Close();
float minimum = float.MaxValue;
for ( i=0; i < comparisons.Count; i++ )
{ proto = (prototype)comparisons[i];
  if ( proto.sum_of_deviations < minimum )
  { minimum = proto.sum_of_deviations;
    recogn_result = proto.name;
  }
}
comparisons.Clear();
break;
case "Teach As"://*****
Char c; String sfr, sfi;
try { c = textbox.Text[0]; } catch { break; }
if ( Char.IsWhiteSpace( c ) || Char.IsControl( c ) ) break;
MessageBox.Show( "The current shape will be stored as prototype of a " + textbox.Text,
  "Teaching a Shape", MessageBoxButtons.OKCancel );
recogn_result = textbox.Text[0]; // OnPaint(..) will write it in red: "This is a .."
StringBuilder newline = new StringBuilder();
newline.Append( textbox.Text + ": " );
if ( no_of_fourier_values < fr.Length ) NN = no_of_fourier_values;
else NN = fr.Length;
for ( i=0; i < NN; i++ )
{ sfr = String.Format( "{0:F2}", fr[i] ); if ( sfr.Length < 5 ) sfr = "+" + sfr;
  sfi = String.Format( "{0:F2}", fi[i] ); if ( sfi.Length < 5 ) sfi = "+" + sfi;
  newline.Append( sfr + "/" + sfi + " " );
}
StreamWriter outstream = new StreamWriter( "FourierRecognitionPrototypes.txt", true );
outstream.WriteLine( newline );
outstream.Close();
textbox.Text = "";
break;
} // end of switch, end of all cases
Invalidate();
} // end of protected void do_it( object sender, System.EventArgs e )

```

```

private void FourierTransform( float[] a, float[] fr, float[] fi )
{ int Na = a.Length, Nf = fr.Length;
  float sum_r, sum_i; fr[0] = fi[0] = 0;
  for ( i=0; i < Na; i++ ) fr[0] += a[i]; fr[0] /= Na; //fr[0] is the average (always 0 here).
  double dpi_div_N, x1_dpi_div_N, x2_x1_dpi_div_N;
  dpi_div_N = 2 * Math.PI / Na;
  for ( int x1 = 1; x1 < Nf; x1++ )
  { sum_r = sum_i = 0;
    x1_dpi_div_N = (double)x1 * dpi_div_N;
    for ( int x2 = 0; x2 < Na; x2++ )
    { x2_x1_dpi_div_N = (double)x2 * x1_dpi_div_N;
      sum_r += a[x2] * (float)Math.Cos( -x2_x1_dpi_div_N );
      sum_i += a[x2] * (float)Math.Sin( -x2_x1_dpi_div_N );
    }
    fr[x1] = 2 * sum_r / (float)Na;
    fi[x1] = 2 * sum_i / (float)Na;
  }
  fr[Nf-1] /= 2; fi[Nf-1] /= 2;
} // end of FourierTransform
}

```