

Course CVCis: Computer Vision with C#

Chapter C4: The Complete Code of the Simple 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: simple_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;

public class Form1 : Form
{ static void Main() { Application.Run( new Form1() ); }
  const Int32 xSize = 11;
  const Int32 ySize = 12;
  Byte[,] i0      = new Byte[ySize ,xSize  ];
  Byte[,] clv     = new Byte[ySize ,xSize+1];
  Brush[] brush   = new Brush[10];
  Brush blackbrush = SystemBrushes.ControlText;
  Brush redbrush  = new SolidBrush( Color.Red );
  Brush bluebrush = new SolidBrush( Color.Blue );
  Pen redpen      = new Pen( Color.Red, 5 );
  Pen whitepen;
  Font arial10    = new Font( "Arial", 10 );
  Font arial20    = new Font( "Arial", 20 );

  Int32 i, x, y, dx, dy;
  Byte threshold = 1;
  Button[] button = new Button[ySize];
  Point p0, p1, start = new Point();
  ArrayList all_crackcodes = new ArrayList();
  struct crackcode
  { public Byte   no;
    public Point start;
    public String cracks;
    public Int32 perimeter, area, xmin, xmax, ymin, ymax;
  }
  Int32 recogn_result; Graphics g;
  ArrayList pointArray = new ArrayList();

  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 = "Digitize";
    button[1].Name = button[1].Text = "Crack Code 8";
    button[2].Name = button[2].Text = "Recognize";
    button[3].Name =                    "Threshold";
    button[4].Name = button[4].Text = "Noise";
    button[5].Name = button[5].Text = "Clear";
    button[3].Text = String.Format( "Threshold={0:#}", threshold );
  }
}
```

```

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;
    button[y].Width = r.Width - button[y].Left - 2;
    button[y].Height = dy-2;
  }

  for ( y=0; y < ySize; y++ )
    for ( x=0; x < xSize; x++ )
      g.FillRectangle( brush[i0[y,x]], x*dx, y*dy, dx, dy );
  if ( all_crackcodes.Count == 0 )
  { s = "Draw a 0,1,2,..9 -> Digitize -> Crack Code 8 -> Recognize !";
    g.DrawString( s, arial10, redbrush, (xSize*dx)/4, 0 );
    return;
  }
  for ( i=0; i < all_crackcodes.Count; i++ )
  { crackcode c = (crackcode)all_crackcodes[i];
    x = c.start.X;
    y = c.start.Y;
    for ( Int32 ii = 0; ii < c.perimeter; ii++ )
      switch ( c.cracks[ii] )
      { case 'e': g.DrawLine( redpen, x*dx, y*dy, (x+1)*dx, y*dy ); x++; break;
        case 's': g.DrawLine( redpen, x*dx, y*dy, x*dx, (y+1)*dy ); y++; break;
        case 'w': g.DrawLine( redpen, x*dx, y*dy, (x-1)*dx, y*dy ); x--; break;
        case 'n': g.DrawLine( redpen, x*dx, y*dy, x*dx, (y-1)*dy ); y--; break;
      }
    g.FillRectangle( bluebrush, x*dx-5, y*dy-5, 11, 11 );
    if ( recogn_result >= 0 ) s = "This is a " + recogn_result.ToString() + ".";
    else s = "I don't know what it is.";
    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, EventArgs e )
{ switch( ((Button)sender).Name )
  { case "Digitize"://*****
    i0 = new Byte[ySize,xSize];
    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{};
    }
    all_crackcodes.Clear(); recogn_result = -1; break;
  }
}

```

```

case "Threshold"://*****
    if ( ++threshold > 9 ) threshold = 1;
    button[3].Text = "Threshold=" + threshold.ToString();
    all_crackcodes.Clear(); recogn_result = -1; 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;
            }
    all_crackcodes.Clear(); recogn_result = -1; 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";
    all_crackcodes.Clear(); pointArray.Clear(); recogn_result = -1; break;
case "Crack Code 8": crackcode_finder(); break;
case "Recognize":
    if ( all_crackcodes.Count == 0 ) { recogn_result = -1; Invalidate(); break; }
    ArrayList shape_crackcodes = new ArrayList();
    Int32 area = 0, index_biggest_cc = 0;
    for ( i=0; i < all_crackcodes.Count; i++ )
        { crackcode c = (crackcode)all_crackcodes[i];
          if ( c.area > area ) { area = c.area; index_biggest_cc = i; }
        }
    shape_crackcodes.Add(all_crackcodes[index_biggest_cc]);
    for ( i=0; i < all_crackcodes.Count; i++ )
        { crackcode c = (crackcode)all_crackcodes[i];
          if ( c.area < 0 )
              { Int32 count = 0;
                for ( x=c.start.X+1; x < xSize; x++ )
                    if ( clv[c.start.Y,x] == index_biggest_cc+1 ) count++;
                if ( count%2 == 1 ) shape_crackcodes.Add( all_crackcodes[i] );
              }
        }
    all_crackcodes.Clear();
    for ( i=0; i < shape_crackcodes.Count; i++ )
        all_crackcodes.Add( shape_crackcodes[i] );
    shape_crackcodes.Clear();
    if ( all_crackcodes.Count > 3 ) { recogn_result = -1; break; }
    if ( all_crackcodes.Count == 3 ) { recogn_result = 8; break; }
    crackcode cc = (crackcode)all_crackcodes[0];
    if ( all_crackcodes.Count == 2 )
        { crackcode ci = (crackcode)all_crackcodes[1];
          Single sdx = (Single)(cc.xmax+cc.xmin)/2.0f - (Single)(ci.xmax+ci.xmin)/2.0f;
          Single sdy = (Single)(cc.ymax+cc.ymin)/2.0f - (Single)(ci.ymax+ci.ymin)/2.0f;
          Single distance = (Single)Math.Sqrt( sdx*sdx - sdy*sdy );
          if ( distance < (cc.ymax-cc.ymin)/5.0f ) { recogn_result = 0; break; }
          if ( sdy < 0.0f ) recogn_result = 6; else recogn_result = 9; break;
        }
    //one crack code only
    if ( (cc.ymax-cc.ymin) > 3*(cc.xmax-cc.xmin) ) { recogn_result = 1; break; }
    Single thickness_top=0, thickness_mid=0, thickness_bottom=0;
    Int32 one_third = (3*cc.ymin + cc.ymax-cc.ymin)/3;
    Int32 two_third = (3*one_third + cc.ymax-cc.ymin)/3;
    Int32 left=0, right=0, n_top=0, n_mid=0, n_bottom=0;
    for ( y=cc.ymin; y < cc.ymax; y++ )
        { for ( x=cc.xmin; x < cc.xmax; x++ )
            if ( clv[y,x] == cc.no ) { left = x; break; }
          for ( x=cc.xmax; x >= cc.xmin; x-- )
            if ( clv[y,x] == cc.no ) { right = x; break; }
          if ( y < one_third ) { thickness_top += right-left; n_top++; } else
          if ( y <= two_third ) { thickness_mid += right-left; n_mid++; } else
          { thickness_bottom += right-left; n_bottom++; }
        }
}

```

```

if ( n_top <= 0 || n_mid <= 0 || n_bottom <= 0 ) { recogn_result = -1; break; }
thickness_top    /= n_top;
thickness_mid    /= n_mid;
thickness_bottom /= n_bottom;
if ( thickness_bottom < 2f )
    if ( thickness_top > thickness_mid ) { recogn_result = 7; break; }
    else { recogn_result = 4; break; }
if ( thickness_mid < 2f ) { recogn_result = 2; break; }
//The following decision between 3 and 5 is based on vector graphics !
//When both start & end points are at the left side -> 3 otherwise -> 5
Int32 startx = ((Point)pointArray[0]).X;
Int32 endx   = ((Point)pointArray[pointArray.Count-1]).X;
Int32 minx = startx, maxx = startx;
for ( i=1; i < pointArray.Count; i++ )
{ if ( ((Point)pointArray[i]).X < minx ) minx = ((Point)pointArray[i]).X;
  if ( ((Point)pointArray[i]).X > maxx ) maxx = ((Point)pointArray[i]).X;
}
if ( Math.Abs(endx-startx) < (maxx-minx)/3 ) { recogn_result = 3; break; }
else { recogn_result = 5; break; }
} // end of cases, end of switch
Invalidate();
} // end of protected void do_it( object sender, System.EventArgs e )

private void crackcode_finder()
{ all_crackcodes.Clear();
  for ( y=0; y < ySize; y++ )
    for ( x=0; x <= xSize; x++ ) clv[y,x] = 0;
  start.X = start.Y = 0;
  for ( Byte cc_no = 1; start_crack_search(); cc_no++ )
  { crackcode cc = new crackcode();
    System.Text.StringBuilder cracks = new System.Text.StringBuilder();
    cc.start = start; cracks.Append('s'); cc.area = 0; cc.no = cc_no;
    x = cc.xmin = cc.xmax = start.X;
    cc.ymin = start.Y; y = cc.ymax = start.Y+1; Char last_crack = 's';
    do
    { switch ( last_crack )
      { case 'e': if ( x == xSize ) goto n;
                if ( y < ySize && i0[y ,x ] >= threshold) goto s;
                if ( i0[y-1,x ] >= threshold) goto e; goto n;
      case 's': if ( y == ySize ) goto e;
                if ( x > 0 && i0[y ,x-1] >= threshold) goto w;
                if ( i0[y ,x ] >= threshold) goto s; goto e;
      case 'w': if ( x == 0 ) goto s;
                if ( y > 0 && i0[y-1,x-1] >= threshold) goto n;
                if ( i0[y ,x-1] >= threshold) goto w; goto s;
      case 'n': if ( y == 0 ) goto w;
                if ( x < xSize && i0[y-1,x ] >= threshold) goto e;
                if ( i0[y-1,x-1] >= threshold) goto n; goto w;
      }
    e: last_crack = 'e'; cracks.Append( 'e' ); x++; cc.area += y;
      if ( x > cc.xmax) cc.xmax = x; continue;
    s: last_crack = 's'; cracks.Append( 's' ); y++; clv[y-1,x] = cc_no;
      if ( y > cc.ymax) cc.ymax = y; continue;
    w: last_crack = 'w'; cracks.Append( 'w' ); x--; cc.area -= y;
      if ( x < cc.xmin ) cc.xmin = x; continue;
    n: last_crack = 'n'; cracks.Append( 'n' ); y--; clv[y ,x] = cc_no;
      if ( y < cc.ymin ) cc.ymin = y; continue;
    } while ( x != start.X || y != start.Y ); //end of do
    cc.cracks = cracks.ToString(); cc.perimeter = cc.cracks.Length;
    all_crackcodes.Add( cc );
  } // end of for
}

private Boolean start_crack_search()
{ Byte left;
  for ( y=start.Y; y < ySize; y++ )
    for ( x=0; x < xSize; x++ )
    { if ( x > 0 ) left = i0[y,x-1]; else left = 0;
      if ( left < threshold && i0[y,x] >= threshold && clv[y,x] == 0 )
        { start.X = x; start.Y = y; clv[y,x] = 1; return true; }
    }
  return false;
}
}

```