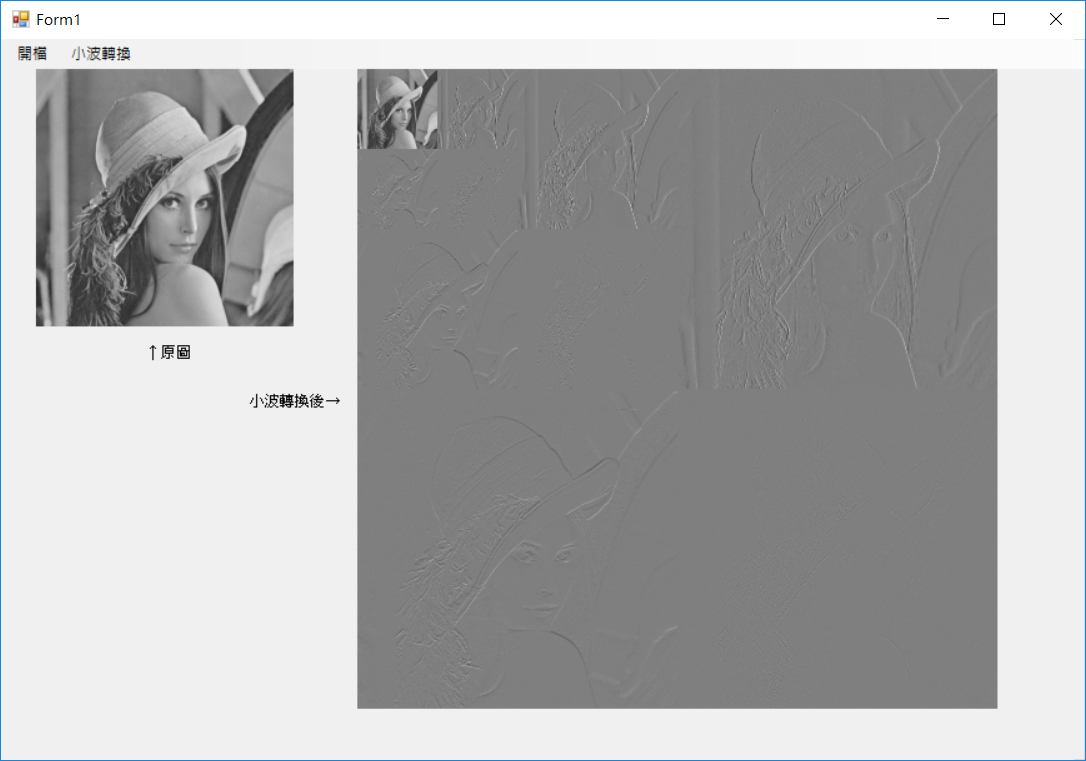
姓名：林佑恩 班級：四子三丙  
學號：1103105336老師：陳昭和老師

結果：



主要程式碼：

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Drawing.Imaging;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace \_20170517\_work04\_WaveletTransform

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

Bitmap ori\_image;

BitmapData ori\_data;

Rectangle imageRect;

byte[] ori\_Values;

int ori\_bytes;

Bitmap wt\_image;

BitmapData wt\_data;

int wt\_bytes;

bool check\_input = false;

private void 開檔ToolStripMenuItem\_Click(object sender, EventArgs e)

{

OpenFileDialog ofd = new OpenFileDialog();

ofd.Filter = "Image Files(\*.BMP;\*.JPG;\*.GIF;\*.PNG)|\*.BMP;\*.JPG;\*.GIF;\*.PNG";

if (ofd.ShowDialog() == DialogResult.OK)

{

pictureBox1.Image = Image.FromFile(ofd.FileName);

}

ori\_image = pictureBox1.Image as Bitmap;

if (ori\_image != null)

{

imageRect = new Rectangle(0, 0, ori\_image.Width, ori\_image.Height);

ori\_data = ori\_image.LockBits(imageRect, ImageLockMode.ReadWrite, ori\_image.PixelFormat);

ori\_bytes = ori\_data.Stride \* ori\_image.Height;

ori\_Values = new byte[ori\_bytes];

System.IntPtr ori\_Ptr = ori\_data.Scan0;

System.Runtime.InteropServices.Marshal.Copy(ori\_Ptr, ori\_Values, 0, ori\_bytes);//複製RGB信息到byte數組

GrayLevel(ori\_image.Height, ori\_image.Width, ori\_data.Stride, ori\_Values);

System.Runtime.InteropServices.Marshal.Copy(ori\_Values, 0, ori\_Ptr, ori\_bytes); //複製byte陣列到RGB

ori\_image.UnlockBits(ori\_data);

check\_input = true;

label1.Visible = true;

}

}

private void 小波轉換ToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (!check\_input) return;

wt\_image = new Bitmap(imageRect.Width, imageRect.Height);

wt\_data = wt\_image.LockBits(imageRect, ImageLockMode.ReadWrite, ori\_image.PixelFormat);

wt\_bytes = wt\_data.Stride \* wt\_image.Height;

var tmp1\_Values = new byte[wt\_bytes];

var tmp2\_Values = new byte[wt\_bytes];

System.IntPtr wt\_Ptr = wt\_data.Scan0;

System.Array.Copy(ori\_Values, 0, tmp1\_Values, 0, ori\_bytes); //把ori\_Values陣列的數值複製到tmp1\_Values陣列

int h = ori\_image.Height;

int w = ori\_image.Width;

WaveletTransform(h, w, 0, ori\_data.Stride, tmp1\_Values, tmp2\_Values); //1-pass

WaveletTransform(h, w, 1, ori\_data.Stride, tmp2\_Values, tmp1\_Values); //2-pass

WaveletTransform(h, w, 2, ori\_data.Stride, tmp1\_Values, tmp2\_Values); //3-pass

System.Runtime.InteropServices.Marshal.Copy(tmp2\_Values, 0, wt\_Ptr, wt\_bytes); //複製byte陣列到RGB

wt\_image.UnlockBits(wt\_data);

pictureBox2.Image = wt\_image;

pictureBox2.Height = wt\_image.Height;

pictureBox2.Width = wt\_image.Width;

label2.Visible = true;

}

public void GrayLevel(int h, int w, int stride, byte[] ori\_bmp)

{

for (int i = 0; i < h; i++) //轉成gray level

{

for (int j = 0; j < w; j++)

{

int p\_Index = i \* stride + j \* 3;

byte pixel = Convert.ToByte((ori\_bmp[p\_Index + 0] + ori\_bmp[p\_Index + 1] + ori\_bmp[p\_Index + 2]) / 3);

ori\_bmp[p\_Index + 0] = pixel; //R

ori\_bmp[p\_Index + 1] = pixel; //G

ori\_bmp[p\_Index + 2] = pixel; //B

}

}

}

public void WaveletTransform(int h, int w,int c, int stride, byte[] bmp1,byte[] bmp2)

{

float value;

int a;

bool check = false;

byte[] tmp = new byte[ori\_bytes];

var count = Math.Pow(2,c);

for (int i = 0; i < h; i++)

{

a = 0;

check = false;

for (int j = 0; j < w; j++)

{

if (j == (w / 2) / count)

{

a = 0;

check = true;

}

if (i < (h / count) && j < (w / count))

{

if (check == false)

value = (float)((bmp1[i \* stride + a \* 3] + bmp1[i \* stride + (a + 1) \* 3]) / 2);

else

value = (float)((bmp1[i \* stride + a \* 3] - bmp1[i \* stride + (a + 1) \* 3]) / 2) + 127;

if (value < 0) value = value - 0.5f;

else value = value + 0.5f;

tmp[i \* stride + j \* 3] = (byte)value;

tmp[i \* stride + j \* 3 + 1] = (byte)value;

tmp[i \* stride + j \* 3 + 2] = (byte)value;

a += 2;

}

else

{

tmp[i \* stride + j \* 3] = bmp1[i \* stride + j \* 3];

tmp[i \* stride + j \* 3 + 1] = bmp1[i \* stride + j \* 3 + 1];

tmp[i \* stride + j \* 3 + 2] = bmp1[i \* stride + j \* 3 + 2];

}

}

}

for (int i = 0; i < w; i++)

{

a = 0;

check = false;

for (int j = 0; j < h; j++)

{

if (j == (h / 2) / count)

{

a = 0;

check = true;

}

if (i < (h / count) && j < (w / count))

{

if (check == false)

value = (float)((tmp[a \* stride + i \* 3] + tmp[(a + 1) \* stride + i \* 3]) / 2);

else

value = (float)((tmp[a \* stride + i \* 3] - tmp[(a + 1) \* stride + i \* 3]) / 2) + 127;

if (value < 0) value = value - 0.5f;

else value = value + 0.5f;

bmp2[j \* stride + i \* 3] = (byte)value;

bmp2[j \* stride + i \* 3 + 1] = (byte)value;

bmp2[j \* stride + i \* 3 + 2] = (byte)value;

a += 2;

}

else

{

bmp2[j \* stride + i \* 3] = tmp[j \* stride + i \* 3];

bmp2[j \* stride + i \* 3 + 1] = tmp[j \* stride + i \* 3 + 1];

bmp2[j \* stride + i \* 3 + 2] = tmp[j \* stride + i \* 3 + 2];

}

}

}

}

}

}