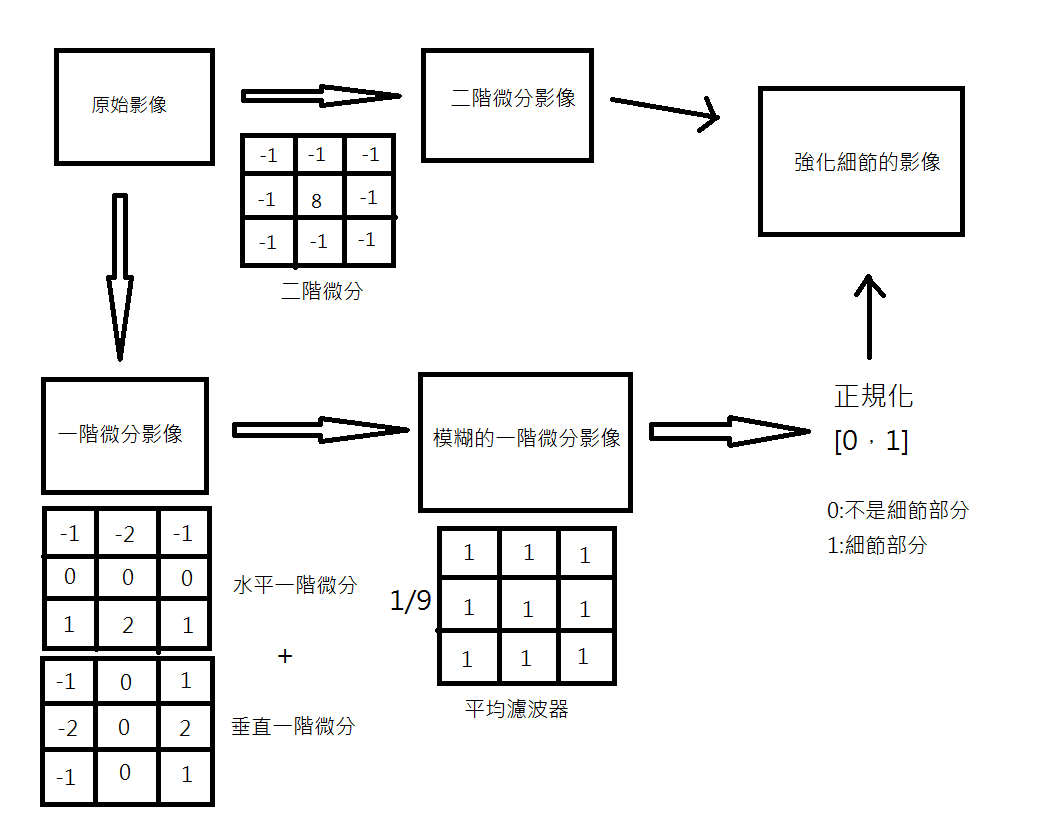
7108056022 陳冠霖

作法:



第一步:

讀入原始影像。

程式碼:

Image<Bgr, byte> dest1 = null;

Size size1;

/\*按下button1(import)載入原始影像(麗娜(316\*316))\*/

private void Button1\_Click(object sender, EventArgs e)

{

var dialog = new OpenFileDialog();

dialog.Filter = "影像(\*.jpg/\*.png/\*.gif/\*.bmp)|\*.jpg;\*.png;\*.gif;\*.bmp";

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

{

var filename = dialog.FileName;

IntPtr image = CvInvoke.cvLoadImage(filename, Emgu.CV.CvEnum.LOAD\_IMAGE\_TYPE.CV\_LOAD\_IMAGE\_ANYCOLOR);

size1 = CvInvoke.cvGetSize(image);

dest1 = new Image<Bgr, byte>(size1);

CvInvoke.cvCopy(image, dest1, IntPtr.Zero);

pictureBox1.Image = dest1.ToBitmap();

}

}

第二步:

將原始影像變成二階微分影像，原始影像的每個像素的RGB值乘上二階微分遮罩(-1,-1, -1,-1,8,-1,-1,-1,-1)，來得到二階微分影像。

程式碼:

}

/\*按下button2(process2)將picturebox1的原始影像乘上二階微分遮罩(-1, -1, -1, -1, 8, -1, -1, -1, -1)，得到二階微分影像(銳化影像)\*/

private void Button2\_Click(object sender, EventArgs e)

{

/\*將picturebox1的原始影像宣告為Bitmap點陣圖的格式\*/

Bitmap bm1 = new Bitmap(pictureBox1.Image);

/\*宣告二階微分遮罩(shaper)數值為(-1, -1, -1, -1, 8, -1, -1, -1, -1)\*/

int[] shaper = new int[] { -1, -1, -1, -1, 8, -1, -1, -1, -1 };

int sizewidth2 = pictureBox2.Size.Width;

int sizeheight2 = pictureBox2.Size.Height;

/\*宣告三個變數newrr, newgg, newbb，用來放最終的RGB值\*/

int newrr, newgg, newbb;

int count = 0;

/\*對所有的影像像素做處理(316\*316)，轉灰階\*/

for (int i = 0; i < sizewidth2 - 1; i++)

{

for (int j = 0; j < sizeheight2 - 1; j++)

{

Color c1 = bm1.GetPixel(i, j);

int r1 = c1.R;

int g1 = c1.G;

int b1 = c1.B;

int avg1 = (r1 + g1 + b1) / 3;

bm1.SetPixel(i, j, Color.FromArgb(avg1, avg1, avg1));

}

}

/\*對所有的影像像素做處理(316\*316)\*/

for (int i = 1; i < sizewidth2 - 1; i++)

{

for (int j = 1; j < sizeheight2 - 1; j++)

{

count = 0;

newrr = 0;

newgg = 0;

newbb = 0;

/\*二階微分遮罩為3\*3， 所以每次處理的像素要和周圍附近８個像素一起處理，乘上遮罩，來得到經過遮罩改變後的影像\*/

for (int x = -1; x <= 1; x++)

{

for (int y = -1; y <= 1; y++)

{

/\*利用Color屬性，抓取原始影像的RGB值\*/

Color c2 = bm1.GetPixel(i + x, j + y);

int rrr = c2.R;

int ggg = c2.G;

int bbb = c2.B;

/\*將原始影像的RGB值乘上二階微分遮罩(shaper)，得到新的二階微分影像的RGB值，放入前面宣告的newrr, newgg, newbb\*/

newrr += rrr \* shaper[count];

newgg += ggg \* shaper[count];

newbb += bbb \* shaper[count];

count++;

}

}

/\*

newrr = newrr > 255 ? 255 : newrr;

newrr = newrr < 0 ? 0 : newrr;

newgg = newgg > 255 ? 255 : newgg;

newgg = newgg < 0 ? 0 : newgg;

newbb = newbb > 255 ? 255 : newbb;

newbb = newbb < 0 ? 0 : newbb;

\*/

/\*如果像素新的RGB值，超過255或小於0，超過255就讓它等於255，小於0就讓它等於0\*/

if (newrr > 255)

{

newrr = 255;

}

else if (newrr < 0)

{

newrr = 0;

}

if (newgg > 255)

{

newgg = 255;

}

else if (newgg < 0)

{

newgg = 0;

}

if (newbb > 255)

{

newbb = 255;

}

else if (newbb < 0)

{

newbb = 0;

}

/\*將經過遮罩改變新的RGB值，存在點陣圖Bitmap(bm1)的格式裡。\*/

bm1.SetPixel(i - 1, j - 1, Color.FromArgb(newrr, newgg, newbb));

}

}

/\*將存在點陣圖Bitmap(bm1)中的RGB值，貼到pictureBox2裡的image\*/

pictureBox2.Image = bm1;

}

第三步:

將原始影像變成一階微分影像，一階微分遮罩分為垂直(-1, 0, 1, -2, 0, 2, -1, 0, 1)和水平(-1, -2, -1, 0, 0, 0, 1, 2, 1)兩種遮罩，所以將每個像素的RGB值分別乘上一階微分的水平和垂直遮罩，接著講處理完的兩種像素的RGB值平方後相加開根號，就能得到一階微分的影像了。

程式碼:

/\*按下button3(process3)將picturebox1的原始影像乘上一階微分遮罩，一階微分遮罩分為水平和垂直，各別對原始影像乘上水平和垂直遮罩，接著將兩個改變後的影像結果平方相加開根號，得對一階微分影像\*/

private void Button3\_Click(object sender, EventArgs e)

{

/\*將picturebox1的原始影像宣告為Bitmap點陣圖的格式\*/

Bitmap bm2 = new Bitmap(pictureBox1.Image);

/\*水平遮罩(-1, -2, -1, 0, 0, 0, 1, 2, 1 )\*/

int[] xh = new int[] { -1, -2, -1, 0, 0, 0, 1, 2, 1 };

/\*垂直遮罩(-1, 0, 1, -2, 0, 2, -1, 0, 1)\*/

int[] yh = new int[] { -1, 0, 1, -2, 0, 2, -1, 0, 1 };

int sizewidth3 = pictureBox3.Size.Width;

int sizeheight3 = pictureBox3.Size.Height;

/\*宣告三個變數newrrx, newggx, newbbx，用來放乘上水平遮罩的RGB值\*/

int newrrx, newggx, newbbx;

/\*宣告三個變數newrry, newggy, newbby，用來放乘上垂直遮罩的RGB值\*/

int newrry, newggy, newbby;

/\*宣告三個變數finalr, finalg, finalb，用來放最終的RGB值(水平遮罩影像平方加垂直遮罩影像平方，然後開根號)\*/

double finalr, finalg, finalb;

int count1 = 0, count2 = 0;

/\*對所有的影像像素做處理(316\*316)，轉灰階\*/

for (int i = 0; i < sizewidth3 - 1; i++)

{

for (int j = 0; j < sizeheight3 - 1; j++)

{

Color c1 = bm2.GetPixel(i, j);

int r1 = c1.R;

int g1 = c1.G;

int b1 = c1.B;

int avg1 = (r1 + g1 + b1) / 3;

bm2.SetPixel(i, j, Color.FromArgb(avg1, avg1, avg1));

}

}

/\*對所有的影像像素做處理(316\*316\*/

for (int i = 1; i < sizewidth3 - 1; i++)

{

for (int j = 1; j < sizeheight3 - 1; j++)

{

finalr = 0;

finalg = 0;

finalb = 0;

count1 = 0;

newrrx = 0;

newggx = 0;

newbbx = 0;

/\*水平遮罩為3\*3， 所以每次處理的像素要和周圍附近８個像素一起處理，乘上遮罩，來得到經過遮罩改變後的影像\*/

for (int x = -1; x <= 1; x++)

{

for (int y = -1; y <= 1; y++)

{

/\*利用Color屬性，抓取原始影像的RGB值\*/

Color c1 = bm2.GetPixel(i + x, j + y);

int rrr = c1.R;

int ggg = c1.G;

int bbb = c1.B;

/\*將原始影像的RGB值乘上一階微分水平遮罩(xh)，得到新的一階微分水平影像的RGB值，放入前面宣告的newrrx, newggx, newbbx\*/

newrrx += rrr \* xh[count1];

newggx += ggg \* xh[count1];

newbbx += bbb \* xh[count1];

count1++;

}

}

newrry = 0;

newggy = 0;

newbby = 0;

count2 = 0;

/\*垂直遮罩為3\*3， 所以每次處理的像素要和周圍附近８個像素一起處理，乘上遮罩，來得到經過遮罩改變後的影像\*/

for (int x = -1; x <= 1; x++)

{

for (int y = -1; y <= 1; y++)

{

/\*利用Color屬性，抓取原始影像的RGB值\*/

Color c2 = bm2.GetPixel(i + x, j + y);

int rrr = c2.R;

int ggg = c2.G;

int bbb = c2.B;

/\*將原始影像的RGB值乘上一階微分垂直遮罩(yh)，得到新的一階微分水平影像的RGB值，放入前面宣告的newrry, newggy, newbby\*/

newrry += rrr \* yh[count2];

newggy += ggg \* yh[count2];

newbby += bbb \* yh[count2];

count2++;

}

}

/\*將經過遮罩處理產生出的一階微分垂直和水平影像的RGB三個值，分別平方相加開根號起來，放入前面宣告的finalr, finalg, finalb，得到最後的一階微分影像\*/

finalr = Math.Sqrt((newrrx \* newrrx) + (newrry \* newrry));

finalg = Math.Sqrt((newggx \* newggx) + (newggy \* newggy));

finalb = Math.Sqrt((newbbx \* newbbx) + (newbby \* newbby));

/\*如果像素新的RGB值，超過255或小於0，超過255就讓它等於255，小於0就讓它等於0\*/

if (finalr > 255)

{

finalr = 255;

}

else if (finalr < 0)

{

finalr = 0;

}

if (finalg > 255)

{

finalg = 255;

}

else if (finalg < 0)

{

finalg = 0;

}

if (finalb > 255)

{

finalb = 255;

}

else if (finalb < 0)

{

finalb = 0;

}

/\*將經過遮罩改變新的RGB值，存在點陣圖Bitmap(bm2)的格式裡\*/

bm2.SetPixel(i - 1, j - 1, Color.FromArgb((int)finalr, (int)finalg, (int)finalb));

}

}

/\*將存在點陣圖Bitmap(bm2)中的RGB值，貼到pictureBox3裡的image\*/

pictureBox3.Image = bm2;

}

第四步:

將上一步得到的一階微分影像做模糊化，使用平均濾波器的方式，將每個像素的RGB值乘上遮罩(1,1,1,1,1,1,1,1,1)，後將的得到新像素的RGB值除以9達到平均的效果，就能得到模糊過的一階微分影像。

程式碼:

/\*按下button4(process3)將picturebox3的一階微分影像，做模糊化，利用平均濾波器的方式，將一階微分影像乘上遮罩(1, 1, 1, 1, 1, 1, 1, 1, 1)，最後全部得到的RGB值除以9，就能得到模糊的一階微分影像\*/

private void Button4\_Click(object sender, EventArgs e)

{

/\*將picturebox3的一階微分影像宣告為Bitmap點陣圖的格式\*/

Bitmap bm3 = new Bitmap(pictureBox3.Image);

/\*平均濾波器(模糊遮罩)(1, 1, 1, 1, 1, 1, 1, 1, 1)\*/

int[] low = new int[] { 1, 1, 1, 1, 1, 1, 1, 1, 1 };

int sizewidth4 = pictureBox4.Size.Width;

int sizeheight4 = pictureBox4.Size.Height;

/\*宣告三個變數newr, newg, newb，用來放最終的RGB值\*/

int newr, newg, newb;

int count = 0;

/\*對所有的影像像素做處理(316\*316)\*/

for (int i = 1; i < sizewidth4 - 1; i++)

{

for (int j = 1; j < sizeheight4 - 1; j++)

{

count = 0;

newr = 0;

newg = 0;

newb = 0;

/\*平均濾波器(模糊遮罩)為3\*3， 所以每次處理的像素要和周圍附近８個像素一起處理，乘上遮罩，來得到經過遮罩改變後的影像\*/

for (int x = -1; x <= 1; x++)

{

for (int y = -1; y <= 1; y++)

{

/\*利用Color屬性，抓取一階微分影像的RGB值\*/

Color c1 = bm3.GetPixel(i + x, j + y);

int rrr = c1.R;

int ggg = c1.G;

int bbb = c1.B;

/\*將原始影像的RGB值乘上模糊遮罩(low)，得到新的一階微分影像(經過平均濾波器處理)的RGB值，放入前面宣告的newr, newg, newb\*/

newr += rrr \* low[count];

newg += ggg \* low[count];

newb += bbb \* low[count];

count++;

}

}

/\*接著再將得到的新的RGB值(經過遮罩處理的newr, newg, newb)除以9來得到模糊化的一階微分影像的RGB值\*/

newr = newr / 9;

newg = newg / 9;

newb = newb / 9;

/\*將經過改變新的RGB值，存在點陣圖Bitmap(bm3)的格式裡\*/

bm3.SetPixel(i, j, Color.FromArgb(newr, newg, newb));

}

}

/\*將存在點陣圖Bitmap(bm3)中的RGB值，貼到pictureBox4裡的image\*/

pictureBox4.Image = bm3;

}

第五步:

將模糊過後的一階微分影像進行正規化，正規化利用平均數和標準差的方式，將細節部分的RGB值設為1，平坦部分的RGB值設為0，所以必須找到一個適合的標準來判斷，什麼是細節，什麼是平坦，正規畫出來的值小於0.2的地方視為平坦值為0，大於0.2的地方為細節值為1。

程式碼:

/\*按下button5(process4)將picturebox4的模糊一階微分影像，做正規化，細節部分為1，平坦部分0\*/

private void Button5\_Click(object sender, EventArgs e)

{

/\*將picturebox4的模糊一階微分影像宣告為Bitmap點陣圖的格式\*/

Bitmap bm4 = new Bitmap(pictureBox4.Image);

int sizewidth5 = pictureBox5.Size.Width;

int sizeheight5 = pictureBox5.Size.Height;

/\*宣告三個變數newr, newg, newb，用來放正規化的RGB值\*/

double newr, newg, newb;

/\*平均數\*/

int avgr = 0, avgg = 0, avgb = 0;

/\*標準差\*/

int abr = 0, abg = 0, abb = 0;

/\*放正規化的值\*/

double abrr = 0, abgg = 0, abbb = 0;

/\*平均數計算\*/

for (int i = 0; i < sizewidth5 - 1; i++)

{

for (int j = 0; j < sizeheight5 - 1; j++)

{

/\*利用Color屬性，抓取模糊一階微分影像的RGB值\*/

Color c1 = bm4.GetPixel(i, j);

int rrr = c1.R;

int ggg = c1.G;

int bbb = c1.B;

avgr += rrr;

avgg += ggg;

avgb += bbb;

}

}

avgr /= 99856;

avgg /= 99856;

avgb /= 99856;

/\*標準差計算\*/

for (int i = 0; i < sizewidth5 - 1; i++)

{

for (int j = 0; j < sizeheight5 - 1; j++)

{

/\*利用Color屬性，抓取模糊一階微分影像的RGB值\*/

Color c1 = bm4.GetPixel(i, j);

int rrr = c1.R;

int ggg = c1.G;

int bbb = c1.B;

abr += ((rrr - avgr) \* (rrr - avgr));

abg += ((ggg - avgg) \* (ggg - avgg));

abb += ((bbb - avgb) \* (bbb - avgb));

}

}

abr /= 99856;

abg /= 99856;

abb /= 99856;

/\*標準差得值\*/

abrr = Math.Sqrt(abr);

abgg = Math.Sqrt(abr);

abbb = Math.Sqrt(abr);

/\*正規化計算\*/

for (int i = 0; i < sizewidth5 - 1; i++)

{

for (int j = 0; j < sizeheight5 - 1; j++)

{

/\*利用Color屬性，抓取模糊一階微分影像的RGB值\*/

Color c1 = bm4.GetPixel(i, j);

int rrr = c1.R;

int ggg = c1.G;

int bbb = c1.B;

newr = (rrr - avgr) / abrr;

newg = (ggg - avgg) / abgg;

newb = (bbb - avgb) / abbb;

/\*小於0.2為平坦，大於0.2為細節\*/

if (newr <= 0.2)

{

newr = 0;

}

else

{

newr = 1;

}

if (newg <= 0.2)

{

newg = 0;

}

else

{

newg = 1;

}

if (newb <= 0.2)

{

newb = 0;

}

else

{

newb = 1;

}

/\*

if(newr == 1 && newg == 1 && newb == 1)

{

newr = 255;

newg = 255;

newb = 255;

}

\*/

bm4.SetPixel(i, j, Color.FromArgb((int)newr, (int)newg, (int)newb));

}

}

/\*將存在點陣圖Bitmap(bm4)中的RGB值，貼到pictureBox5裡的image\*/

pictureBox5.Image = bm4;

}

第六步:

上一步藉著正規化模糊一階影像，我們可以找到細節和平坦部分，接著我們將二階微分影像與正規化後的模糊一階影像的每個RGB值進行相乘，保留了細節部分，把平坦的部份去掉，將去掉的部分也就是RGB值為0的地方，用原始影像呈現，這樣我們就能得到強化細節的原始影像。

程式碼:

/\*按下button6(process5)由picturebox5的正規化的影像與picturebox2的二階微分影像，兩個的像素RGB值互相相乘，將細節部分保留，去除平坦的部分，用來強化原始影像的細節\*/

private void Button6\_Click(object sender, EventArgs e)

{

/\*將picturebox2的二階微分影像宣告為Bitmap點陣圖的格式\*/

Bitmap bm5 = new Bitmap(pictureBox2.Image);

/\*將picturebox5的正規化的模糊一階微分影像宣告為Bitmap點陣圖的格式\*/

Bitmap bm6 = new Bitmap(pictureBox5.Image);

/\*將picturebo1的原始影像宣告為Bitmap點陣圖的格式\*/

Bitmap bm7 = new Bitmap(pictureBox1.Image);

int sizewidth6 = pictureBox6.Size.Width;

int sizeheight6 = pictureBox6.Size.Height;

/\*宣告三個變數newr, newg, newb，用來放最終的RGB值\*/

int newr, newg, newb;

/\*對所有的影像像素做處理(316\*316)\*/

for (int i = 0; i < sizewidth6 - 1; i++)

{

for (int j = 0; j < sizeheight6 - 1; j++)

{

newr = 0;

newg = 0;

newb = 0;

/\*利用Color屬性，抓取二階微分影像的RGB值\*/

Color c1 = bm5.GetPixel(i, j);

int r1 = c1.R;

int g1 = c1.G;

int b1 = c1.B;

/\*利用Color屬性，抓取正規化的模糊一階微分影像的RGB值\*/

Color c2 = bm6.GetPixel(i, j);

int r2 = c2.R;

int g2 = c2.G;

int b2 = c2.B;

/\*利用Color屬性，抓取原始影像的RGB值\*/

Color c3 = bm7.GetPixel(i, j);

int r3 = c3.R;

int g3 = c3.G;

int b3 = c3.B;

/\*將二階微分影像的RGB值乘上正規化的模糊一階微分影像的RGB值，來得到新的RGB值(保留細節部分，捨去平坦部分)，放入前面宣告的newr, newg, newb\*/

newr = r1 \* r2;

newg = g1 \* g2;

newb = b1 \* b2;

/\*如果像素新的RGB值，超過255或小於0，超過255就讓它等於255，小於0的話為平坦部分，使用原始影像的像素RGB值就可以了\*/

if(newr > 1)

{

newr = 255;

}

if(newg > 1)

{

newg = 255;

}

if(newb > 1)

{

newb = 255;

}

if (newr == 0 && newg == 0 && newb == 0)

{

newr = r3;

newg = g3;

newb = b3;

}

/\*將經過改變新的RGB值，存在點陣圖Bitmap(bm5)的格式裡\*/

bm5.SetPixel(i, j, Color.FromArgb(newr, newg, newb));

}

}

/\*將存在點陣圖Bitmap(bm5)中的RGB值，貼到pictureBox6裡的image\*/

pictureBox6.Image = bm5;

}

結果圖:

