

Re: dilation

Source: <http://sci.tech-archive.net/Archive/sci.image.processing/2006-07/msg00076.html>

- *From:* "vonschwartzwalder" <vonschwartzwalder@xxxxxxx>
 - *Date:* 12 Jul 2006 07:42:28 -0700
-

Your English is much better than my Italian!

I took your code and converted it to Java (easiest for me right now), leaving your comments and using your variable names.

It works. I did make a couple of subtle changes:

- In that final inner loop I changed the indexing to remove the '-k' from the indices
- In that final loop I changed the indexing to be inside the temping

The above shouldn't have caused you to get results that were identical to the input. For me, they gave incorrect output (result was placed one pixel off, and I had array index out of bound errors).

If you are actually getting the same output as input I suspect that you are not working on the image data you think you are. You may want to check your pointer, and perhaps print them out to see what you are using exactly. I could do that if I had your library and used c++ and

From what I can see, your algorithm works and produces correct results.

It appears to be your implementation that is at fault. I've included my code below so you can compare them and look for differences.

I hope that helps,

duane

```
=====
/**
 *
 */
package mil.nawcwg.isp;

import java.awt.Dimension;

import ij.ImagePlus;
```

Re: dilation

```
import ij.io.OpenDialog;
import ij.process.ByteProcessor;
import ij.process.ImageProcessor;

/**
 * @author dvs
 */
public class Dilator {

/**
 * @param args
 */
public static void main(String[] args) {

String imageName = "";

// -----
// get input image
// -----
if (args.length != 0) {
imageName = args[0];
}
else {
OpenDialog od = new OpenDialog("Select an image", "");
if (od.getFileName().length() != 0) {
imageName = od.getDirectory() + od.getFileName();
}
else {
System.err.println("no image selected");
return;
}
}

// open image
ImagePlus imp = new ImagePlus(imageName);
ImageProcessor ip = imp.getProcessor();

// make image gray
ip = ip.convertToByte(false);

imp.show();

// -----
// c++ converted to Java
// -----
ImageProcessor m_inputImage = ip;
int foreground = 255;
int background = 0;
```

Re: dilation

Re: dilation

```
//kernel dimension 3x3
int k = 1;
int[] kernel = new int[(2*k+1)*(2*k+1)];

//output image dimensions set to: (h-2k)x(w-2k)
Dimension size = new Dimension();
int h, w;
h = m_inputImage.getHeight();
w = m_inputImage.getWidth();
size.height = h-2*k;
size.width = w-2*k;
ImageProcessor m_outputImage = new ByteProcessor(size.width,
size.height);

/*
1 1 1
kernel = [ 1 1 1 ]
1 1 1

*/
for(int i = 0; i < 2*k+1; i++) {
for(int j = 0; j < 2*k+1; j++) {
kernel[i*(2*k+1)+j] = 255;
}
}

//create binary image from the input image (m_inputImage)
ImageProcessor tempimg = m_inputImage.duplicate();
tempimg.threshold(128); // no autothresholding for now

ImagePlus tempimp = new ImagePlus("test", tempimg);
tempimp.show();

//it copies the input image's binarization to the output image
//now all pixels values 1s or 0s
for(int r = k; r < size.height; r++) {
for(int c = k; c < size.width; c++) {
m_outputImage.set(c, r, tempimg.get(c, r));
}
}

//it cycles on height n width of tempimg
int[] vicinato;
for(int r = k+1; r < tempimg.getHeight()-k-1; r++) {
for(int c = k+1; c < tempimg.getWidth()-k-1; c++) {

//if the (r,c) pixel belongs to the background, then i'm gonna
search
//for its 8 connected pixel. If one of these pixel belongs to
the
//foreground, i set it to foreground value and exit for cycle.
```

Re: dilation

Re: dilation

```
if(tempimg.get(c, r) == background) {
vicinato = neighD8(tempimg, r, c);
for(int z = 0; z < 8; z++) {
if (vicinato[z] == foreground) {
m_outputImage.set(c, r, foreground);
break;
}
}
}
}
}

ImagePlus outimp = new ImagePlus("output", m_outputImage);
outimp.show();

}

public static int[] neighD8(ImageProcessor ip, int y, int x) {
int[] result = new int[8];

int index = 0;
for (int i = -1; i <= 1; i++) {
for (int j = -1; j <= 1; j++) {
if (i != j) {
result[index] = ip.get(x+i, y+j);
index++;
}
}
}
return result;
}

}
}
```

=====
h4mm3r` wrote:

vonschwartzwalder wrote:

Hi back,

Hi,

first of all thanks for having answered to my post. I just wanted to say that i cut comments off 'cause they were in italian and i hadn't time to comment it in english.

When you say it doesn't work, what does that mean? What does the output look like?

Re: dilation

Re: dilation

It looks exact the same of the input image (temping)

You do things differently than I do, but that's just a personal thing.

inputdata is created and then never used.

Ya i know, i've to refactor the code :(

But, the thing I found that is probably giving you the bad results is in the inner loop in the final loops: you are adding the foreground pixel to the current pixel for each foreground pixel in the kernel. You likely wanted to just set the pixel to the foreground and the break out of that inner loop.

I fixed some points but it still doesn't work as it should

```
void CElabImageDlg::dilation(void) {
//kernel dimension 3x3
int k = 1;
int *kernel = new int[(2*k+1)*(2*k+1)];

if(m_inputImage->nChannels!=1) {
AfxMessageBox("immagine RGB!");
return;
}

//output image dimensions set to: (h-2k)x(w-2k)
CvSize size;
int h,w;
h = m_inputImage->height;
w = m_inputImage->width;
size.height = h-2*k;
size.width = w-2*k;
m_outputImage = cvCreateImage(size,IPL_DEPTH_8U,1);

/*
1 1 1
kernel = [ 1 1 1 ]
1 1 1

*/
for(int i = 0; i < 2*k+1; i++) {
```

Re: dilation

Re: dilation