Morphological Image Processing

April 29, 2005

Szabolcs Sergyán

sergyan.szabolcs@nik.bmf.hu

BMF NIK

Dilation

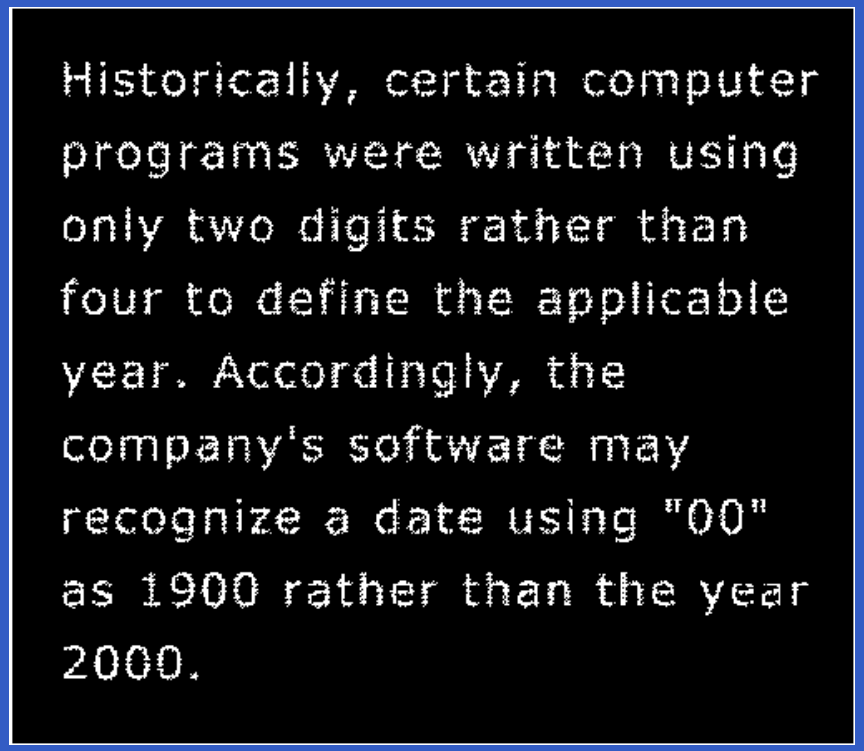
IPT function `imdilate` performs dilation. Its basic calling syntax is

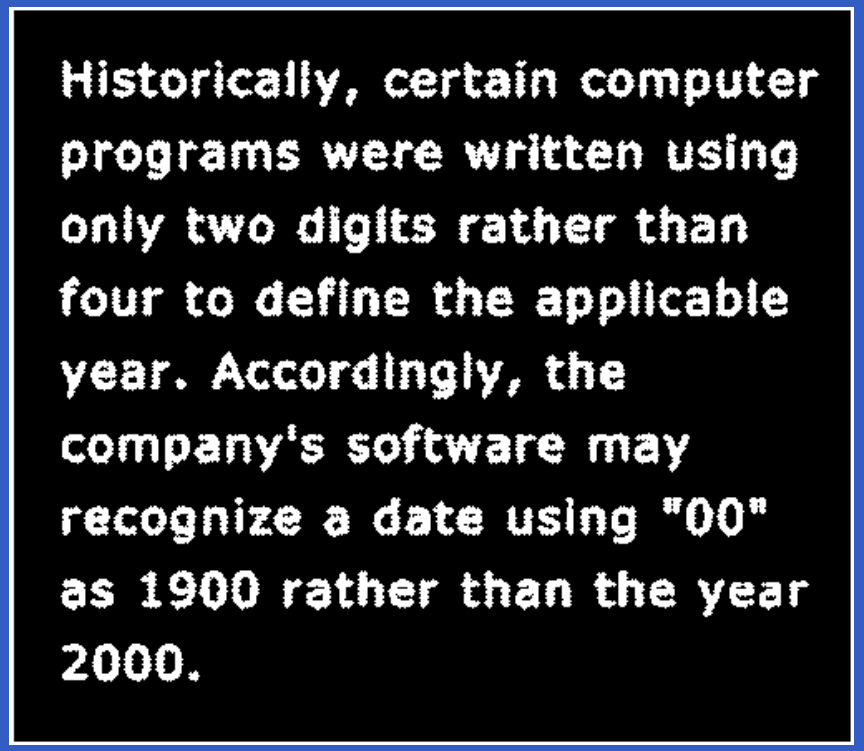
```
A2=imdilate(A,B)
```

where `A` and `A2` are binary images, and `B` is a matrix of 0s and 1s that specifies the structuring element.

Dilation

```
>> A=imread('broken-text.tif');
>> B=[0 1 0;1 1 1;0 1 0];
>> A2=imdilate(A,B);
```





Structuring Element

IPT function `strel` constructs structuring elements with a variety of shapes and sizes. Its basic syntax is

```
se=strel(shape,parameters)
```

where `shape` is a string specifying the desired shape, and `parameters` is a list of parameters that specify information about the shape, such as its size.

Structuring Element

Syntax Forms	Description
`strel('diamond',R)`	Creates a flat, diamond-shaped structuring element, where `R` specifies the distance from the structuring element origin to the extreme points of the diamond.
`strel('disk',R)`	Creates a flat, disk-shaped structuring element with radius `R`.
`strel('line',LEN,DEG)`	Creates a flat, linear structuring element, where `LEN` specifies the length, and `DEG` specifies the angle (in degrees) of the line, as measured in a counterclockwise direction from the horizontal axes.
`strel('octagon',R)`	Creates a flat, octagonal structuring element, where `R` specifies the distance from the structuring element origin to the sides of the octagon, as measured along the horizontal and vertical axes. `R` must be a nonnegative multiple of 3.

Structuring Element

Syntax Forms	Description
`strel('pair',OFFSET)`	Creates a flat structuring element containing two members. One member is located at the origin. The second member's location is specified by the vector `OFFSET`, which must be a two-element vector of integers.
`strel('periodicline',P,V)`	Creates a flat structuring element containing `2*P+1` members. `V` is a two-element vector containing integer-valued row and column offsets. One structuring element member is located at the origin. The other members are located at `1*V`, `-1*V`, `2*V`, `-2*V`, `...`, `P*V`, and `-P*V`.

Structuring Element

Syntax Forms	Description
`strel('rectangle',MN)`	Creates a flat, rectangle-shaped structuring element, where `MN` specifies the size. `MN` must be a two-element vector of nonnegative integers. The first element of `MN` is the number of rows in the structuring element; the second element is the number of columns.
`strel('square',W)`	Creates a square structuring element whose width is `W` pixels. `W` must be a nonnegative integer scalar.
`strel(NHOOD)`	Creates a structuring element of arbitrary shape. `NHOOD` is a matrix of 0s and 1s that specifies the shape.

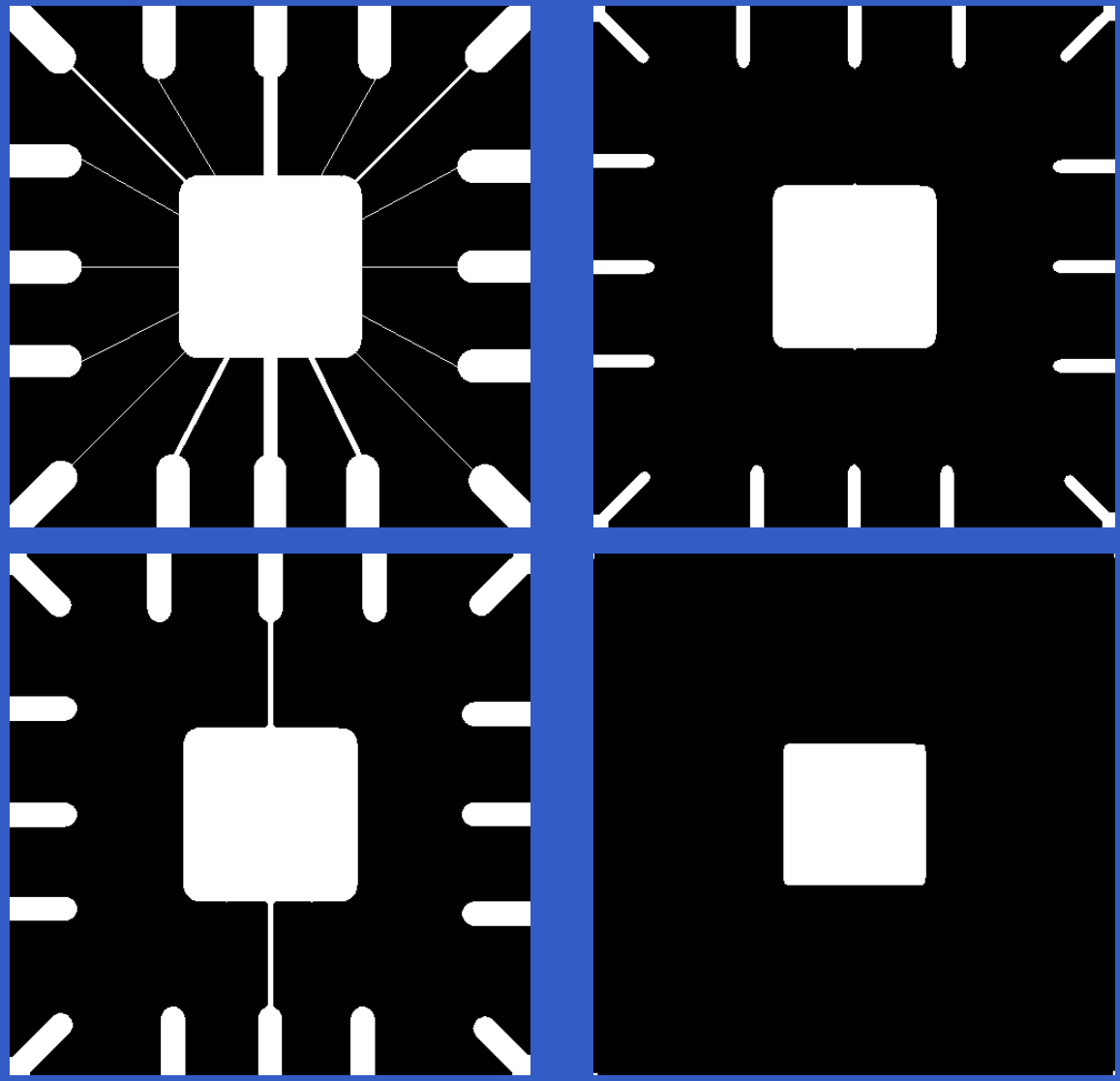
Dilation

```
>> originalI=imread('cameraman.tif');
>> se=strel('disk',2);
>> dilatedI=imdilate(originalI,se);
```

Erosion

```
>> A=imread('wirebond-mask.tif');
>> se=strel('disk',10);
>> A2=imerode(A,se);
>> se=strel('disk',5);
>> A3=imerode(A,se);
>> A4=imerode(A,strel('disk',20));
>> subplot(2,2,1), imshow(A),...
subplot(2,2,2), imshow(A2),...
subplot(2,2,3), imshow(A3),...
subplot(2,2,4), imshow(A4)
```

Erosion



Labeling Connected Components

IPT function `bwlabel` computes all the connected components in a binary image. The calling syntax is

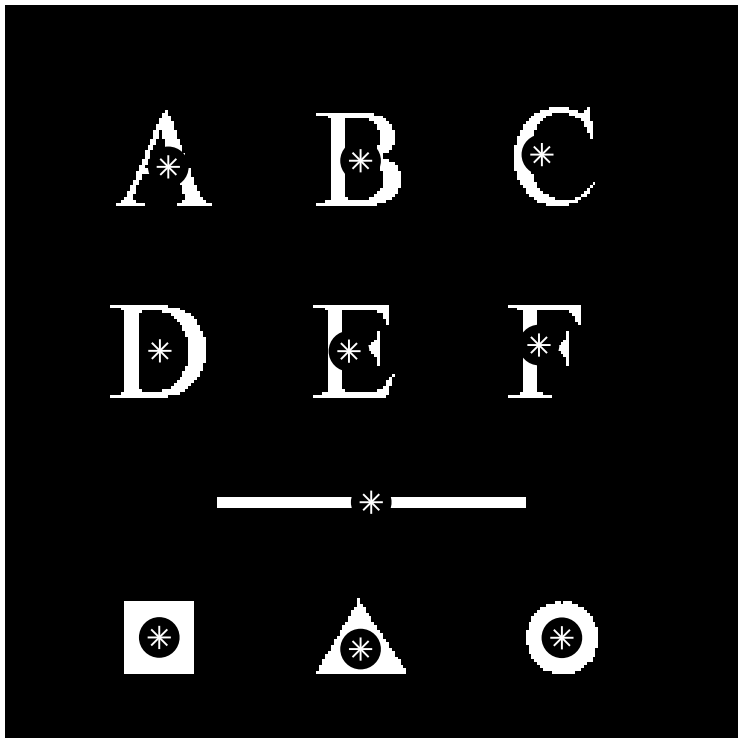
```
[L,num]=bwlabel(f,conn)
```

where `f` is an input binary image and `conn` specifies the desired connectivity (either 4 or 8). Output `L` is called a *label matrix*, and `num` (optional) gives the total number of connected components found. If parameter `conn` is omitted, its value defaults to 8.

Labeling Connected Components

```
>> f=imread('ten-objects.tif');
>> [L,n]=bwlabel(f);
>> [r,c]=find(L==3);
>> rbar=mean(r);
>> cbar=mean(c);
>> imshow(f)
>> hold on
>> for k=1:n
     [r,c]=find(L==k);
     rbar=mean(r);
     cbar=mean(c);
     plot(cbar,rbar,'Marker','o', 'MarkerEdgeColor','k',...
     'MarkerFaceColor','k', 'MarkerSize',10)
     plot(cbar,rbar,'Marker','*', 'MarkerEdgeColor','w')
   end
```

Labeling Connected Components



References

- R. C. Gonzalez, R. E. Woods, S. L. Eddins: Digital Image Processing Using MATLAB. Pearson Prentice Hall, 2004
- R. C. Gonzalez, R. E. Woods: Digital Image Processing. Prentice Hall, 2002
- `http://www.imageprocessingplace.com`