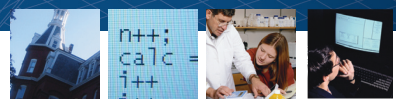


Morphological Reconstruction

From *Digital Image Processing Using MATLAB*, by Rafael C. Gonzalez, Richard E. Woods, and Steven L. Eddins

Products Used

- MATLAB®
- Image Processing Toolbox™

Morphological reconstruction is a useful but little-known method for extracting meaningful information about shapes in an image. The shapes could be just about anything: letters in a scanned text document, fluorescently stained cell nuclei, or galaxies in a far-infrared telescope image. You can use morphological reconstruction to extract marked objects, find bright regions surrounded by dark pixels, detect or remove objects touching the image border, detect or fill in object holes, filter out spurious high or low points, and perform many other operations.

Essentially a generalization of flood-filling, morphological reconstruction processes one image, called the *marker*, based on the characteristics of another image, called the *mask*. The high points, or *peaks*, in the marker image specify where processing begins. The peaks spread out, or dilate, while being forced to fit within the mask image. The spreading processing continues until the image values stop changing.

This excerpt from the second edition of *Digital Image Processing Using MATLAB* teaches what morphological reconstruction means, illustrates some useful manipulations of binary images, and shows how you can use functions in Image Processing Toolbox™ to quickly perform these manipulations.

About the Authors

Rafael C. Gonzalez is a Professor Emeritus of Electrical Engineering and Computer Science at the University of Tennessee, Knoxville.

Richard E. Woods is founder of MedData Interactive and a former professor of Electrical Engineering and Computer Science at the University of Tennessee.

Steven L. Eddins manages the Image Processing and Geospatial Computing development team at MathWorks.

10.5 Morphological Reconstruction

See Sections 11.4.2 and 11.4.3 for additional applications of morphological reconstruction.

Reconstruction is a morphological transformation involving two images and a structuring element (instead of a single image and structuring element). One image, the *marker*, is the starting point for the transformation. The other image, the *mask*, constrains the transformation. The structuring element used defines connectivity. In this section we use 8-connectivity (the default), which implies that B in the following discussion is a 3×3 matrix of 1s, with the center defined at coordinates (2, 2). In this section we deal with binary images; gray-scale reconstruction is discussed in Section 10.6.3.

This definition of reconstruction is based on dilation. It is possible to define a similar operation using erosion. The results are duals of each other with respect to set complementation. These concepts are developed in detail in Gonzalez and Woods [2008].

If G is the mask and F is the marker, the reconstruction of G from F, denoted $R_G(F)$, is defined by the following iterative procedure:

1. Initialize h_1 to be the marker image, F.
2. Create the structuring element: `B = ones(3)`.
3. Repeat:

$$h_{k+1} = (h_k \oplus B) \cap G$$

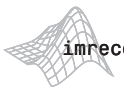
   until $h_{k+1} = h_k$.

4. $R_G(F) = h_{k+1}$.

Marker F must be a subset of G:

$$F \subseteq G$$

Figure 10.21 illustrates the preceding iterative procedure. Although this iterative formulation is useful conceptually, much faster computational algorithms exist. Toolbox function `imreconstruct` uses the "fast hybrid reconstruction" algorithm described in Vincent [1993]. The calling syntax for `imreconstruct` is




```
out = imreconstruct(marker, mask)
```

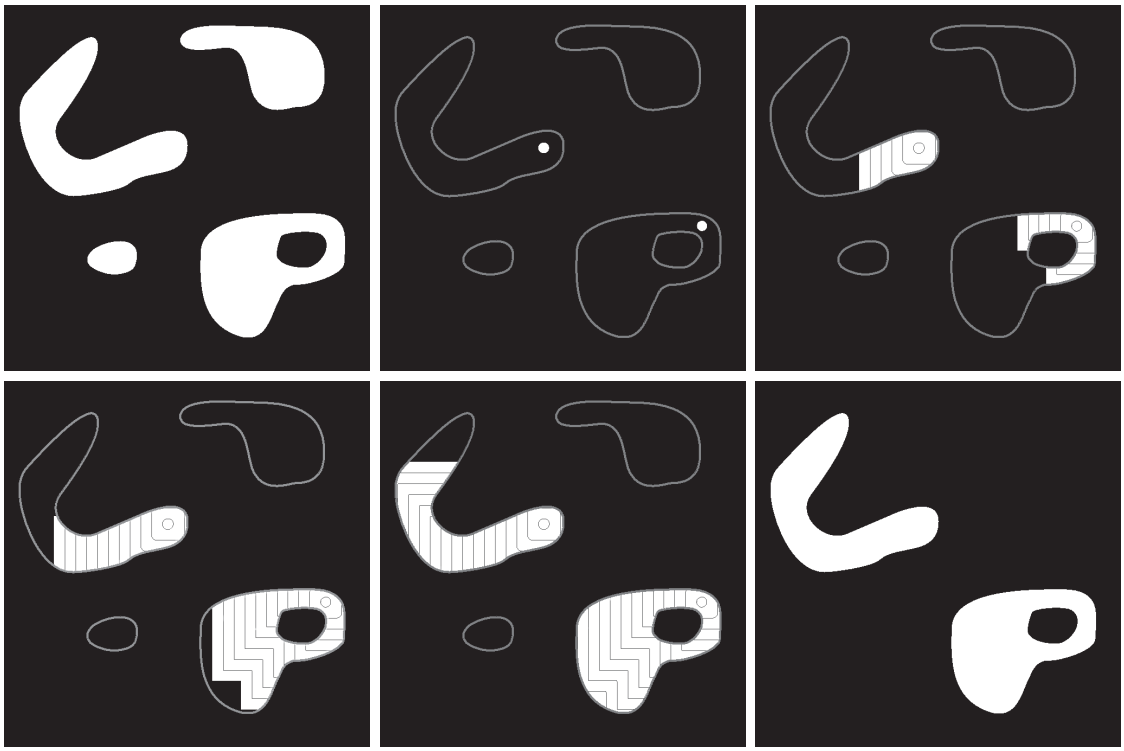
where `marker` and `mask` are as defined at the beginning of this section.

10.5.1 Opening by Reconstruction

In morphological opening, erosion typically removes small objects, and the subsequent dilation tends to restore the shape of the objects that remain. However, the accuracy of this restoration depends on the similarity between the shapes and the structuring element. The method discussed in this section, *opening by reconstruction*, restores the original shapes of the objects that remain after erosion. The opening by reconstruction of an image G using structuring element B, is defined as $R_G(G \ominus B)$.

EXAMPLE 10.8: Opening by reconstruction.

■ A comparison between opening and opening by reconstruction for an image containing text is shown in Fig. 10.22. In this example, we are interested in extracting from Fig. 10.22(a) the characters that contain long vertical strokes.



a b c
d e f

FIGURE 10.21 Morphological reconstruction. (a) Original image (the mask). (b) Marker image. (c)–(e) Intermediate result after 100, 200, and 300 iterations, respectively. (f) Final result. (The outlines of the objects in the mask image are superimposed on (b)–(e) as visual references.)

Because both opening and opening by reconstruction have erosion in common, we perform that step first, using a thin, vertical structuring element of length proportional to the height of the characters:

```
>> f = imread('book_text_bw.tif');
>> fe = imerode(f, ones(51, 1));
```

Figure 10.22(b) shows the result. The opening, shown in Fig. 10.22(c), is computed using `imopen`:

```
>> fo = imopen(f,ones(51, 1));
```

Note that the vertical strokes were restored, but not the rest of the characters containing the strokes. Finally, we obtain the reconstruction:

```
>> fobr = imreconstruct(fe, f);
```

a b
c d
e f
g

FIGURE 10.22 Morphological reconstruction: (a) Original image. (b) Image eroded with vertical line; (c) opened with a vertical line; and (d) opened by reconstruction with a vertical line. (e) Holes filled. (f) Characters touching the border (see right border). (g) Border characters removed.

The result in Fig. 10.22(d) shows that characters containing long vertical strokes were restored exactly; all other characters were removed. The remaining parts of Fig. 10.22 are explained in the following two sections. ■

10.5.2 Filling Holes

Morphological reconstruction has a broad spectrum of practical applications, each characterized by the selection of the marker and mask images. For example, let I denote a binary image and suppose that we choose the marker image, F, to be 0 everywhere except on the image border, where it is set to $1 - I$:

$$F(x, y) = \begin{cases} 1 - I(x, y) & \text{if } (x, y) \text{ is on the border of } I \\ 0 & \text{otherwise} \end{cases}$$

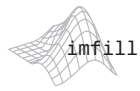
Then,

$$H = \left[R_{I^c}(F) \right]^c$$

is a binary image equal to I with all holes filled, as illustrated in Fig. 10.22(e).

Toolbox function `imfill` performs this computation automatically when the optional argument `'holes'` is used:

```
g = imfill(f, 'holes')
```



This function is discussed in more detail in Section 12.1.2.

10.5.3 Clearing Border Objects

Another useful application of reconstruction is removing objects that touch the border of an image. Again, the key task is to select the appropriate marker to achieve the desired effect. Suppose we define the marker image, F, as

$$F(x,y) = \begin{cases} I(x,y) & \text{if } (x,y) \text{ is on the border of } I \\ 0 & \text{otherwise} \end{cases}$$

where I is the original image. Then, using I as the mask image, the reconstruction

$$H = R_I(F)$$

yields and image, H, that contains only the objects touching the border, as Fig. 10.22(f) shows. The difference, $1 - H$, shown in Fig. 10.22(g), contains only the objects from the original image that do not touch the border. Toolbox function `imclearborder` performs this entire procedure automatically. Its syntax is

```
g = imclearborder(f, conn)
```

where `f` is the input image and `g` is the result. The value of `conn` can be either 4 or 8 (the default). This function suppresses structures that are lighter than their surroundings and that are connected to the image border.

For More Information

- *Digital Image Processing Using MATLAB*, 2e
  www.mathworks.com/img-processing-matlab
- Image Overlay Using Transparency
  www.mathworks.com/img-transparency
- Steve on Image Processing
  http://blogs.mathworks.com/steve

Resources

VISIT
www.mathworks.com/academia

TECHNICAL SUPPORT
www.mathworks.com/support

ONLINE USER COMMUNITY
www.mathworks.com/matlabcentral

DEMOS
www.mathworks.com/demos

TRAINING SERVICES
www.mathworks.com/training

THIRD-PARTY PRODUCTS AND SERVICES
www.mathworks.com/connections

Worldwide CONTACTS
www.mathworks.com/contact

E-MAIL
info@mathworks.com

91822v00 04/10