Detecting a Cell Using Image Segmentation

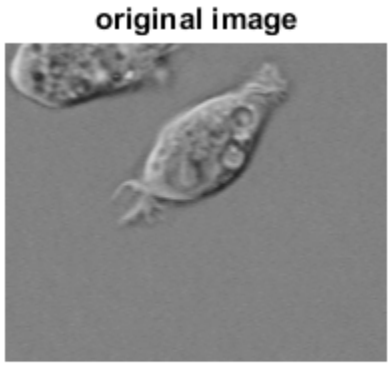
Image segmentation is often an effective approach for identifying objects in an image. Image Processing Toolbox™ offers a variety of techniques for image segmentation.

This example shows how you can detect a cell using edge detection and basic morphology. An object can be easily detected in an image if the object has sufficient contrast from the background. In this example, the cells are prostate cancer cells.

Step 1: Reading the Image

Suppose you have an image of a prostate cancer cell, called `cell.tif`. You start by reading this file into MATLAB®.

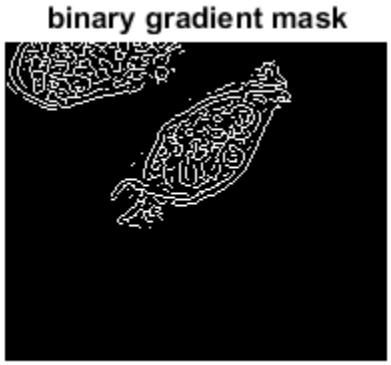
```
I = imread('cell.tif');
figure, imshow(I), title('original image');
text(size(I,2),size(I,1)+15, ...
     'Image courtesy of Alan Partin', ...
     'FontSize',7,'HorizontalAlignment','right');
text(size(I,2),size(I,1)+25, ....
     'Johns Hopkins University', ...
     'FontSize',7,'HorizontalAlignment','right');
```



Step 2: Detecting the Entire Cell

Two cells are present in this image, but only one cell can be seen in its entirety. To detect this cell, you can use *object detection*, or segmentation. The object to be segmented differs greatly in contrast from the background image. Image Processing Toolbox can detect changes in contrast by using operators that calculate the gradient of an image. The gradient image can be calculated, and a threshold can be applied to create a binary mask containing the segmented cell. You can start by using the `edge` function and the Sobel operator to calculate the threshold value. You can then tune the threshold value and use `edge` again to obtain a binary mask that contains the segmented cell.

```
[~, threshold] = edge(I, 'sobel');
fudgeFactor = .5;
BWs = edge(I,'sobel', threshold * fudgeFactor);
figure, imshow(BWs), title('binary gradient mask');
```



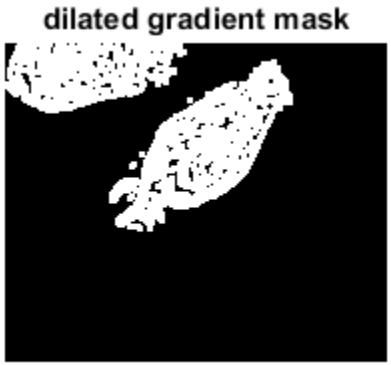
Step 3: Dilating the Image

The resulting binary gradient mask shows lines of high contrast in the image. These lines do not quite delineate the outline of the object of interest. If you compare the mask with the original image, you can see gaps in the lines surrounding the object in the gradient mask. These linear gaps will disappear if the Sobel image is dilated using linear structuring elements, which you can create with the `strel` function.

```
se90 = strel('line', 3, 90);
se0 = strel('line', 3, 0);
```

The binary gradient mask is dilated using the vertical structuring element followed by the horizontal structuring element. You can use the `imdilate` function to dilate the image.

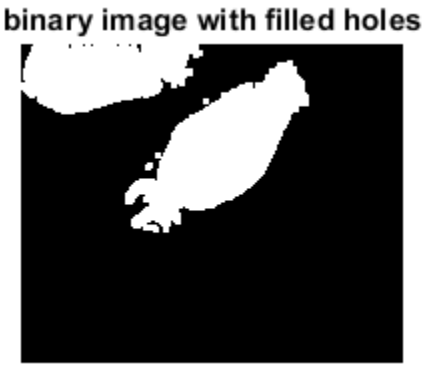
```
BWsdil = imdilate(BWs, [se90 se0]);
figure, imshow(BWsdil), title('dilated gradient mask');
```



Step 4: Filling Interior Gaps

The dilated gradient mask shows the outline of the cell quite nicely, but there are still holes in the interior of the cell. To fill these holes, you can use the `imfill` function.

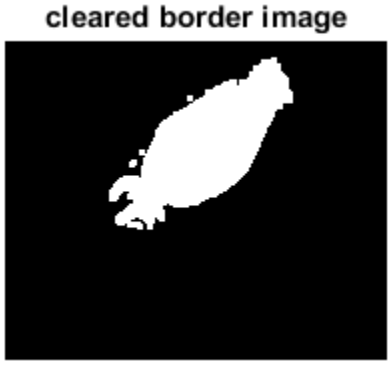
```
BWdfill = imfill(BWsdil, 'holes');
figure, imshow(BWdfill);
title('binary image with filled holes');
```



Step 5: Removing Connected Objects on the Border

The cell of interest has been successfully segmented, but it is not the only object that has been found. You can remove any objects that are connected to the border of the image by using the `imclearborder` function. The connectivity in the `imclearborder` function is set to 4 to remove diagonal connections.

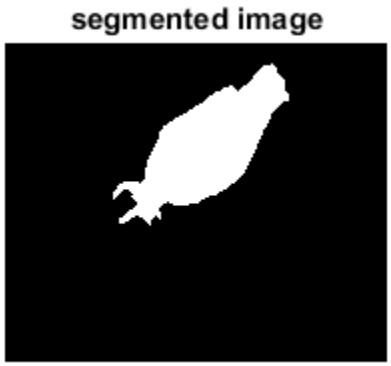
```
BWnobord = imclearborder(BWdfill, 4);
figure, imshow(BWnobord), title('cleared border image');
```



Step 6: Smoothing the Object

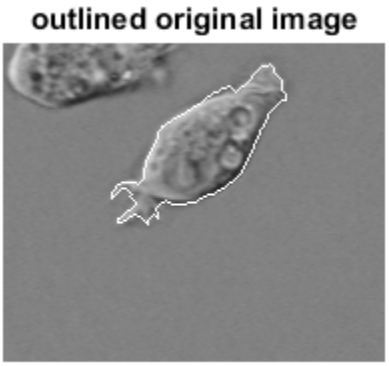
Finally, in order to make the segmented object look natural, you can smooth the object by eroding the image twice with a diamond structuring element. To create the diamond structuring element, you can use the `strel` function.

```
seD = strel('diamond',1);
BWfinal = imerode(BWnobord,seD);
BWfinal = imerode(BWfinal,seD);
figure, imshow(BWfinal), title('segmented image');
```



An alternate method for displaying the segmented object is to place an outline around it. The outline is created by using the `bwperim` function.

```
BWoutline = bwperim(BWfinal);
Segout = I;
Segout(BWoutline) = 255;
figure, imshow(Segout), title('outlined original image');
```



Learn More About Image Processing

- *Segment Images Interactively and Generate MATLAB Code* (download)
- *SegmentTool: An Interactive GUI for Segmenting Images* (download)
- *The Watershed Transform: Strategies for Image Segmentation* (technical article)
- *Tips and Tricks: Solving a Maze with the Watershed Transform* (technical article)
- *Image Processing Toolbox* (product trial)