**Detecting a Cell Using Image Segmentation**

An object can be easily detected in an image if the object has sufficient contrast from the background. We use edge detection and basic morphology tools to detect a prostate cancer cell.

**Contents**

* [Step 1: Read Image](http://www.mathworks.in/products/image/examples.html?file=/products/demos/shipping/images/ipexcell.html#1)
* [Step 2: Detect Entire Cell](http://www.mathworks.in/products/image/examples.html?file=/products/demos/shipping/images/ipexcell.html#2)
* [Step 3: Dilate the Image](http://www.mathworks.in/products/image/examples.html?file=/products/demos/shipping/images/ipexcell.html#3)
* [Step 4: Fill Interior Gaps](http://www.mathworks.in/products/image/examples.html?file=/products/demos/shipping/images/ipexcell.html#5)
* [Step 5: Remove Connected Objects on Border](http://www.mathworks.in/products/image/examples.html?file=/products/demos/shipping/images/ipexcell.html#6)
* [Step 6: Smoothen the Object](http://www.mathworks.in/products/image/examples.html?file=/products/demos/shipping/images/ipexcell.html#7)

**Step 1: Read Image**

Read in the cell.tif image, which is an image of a prostate cancer cell.

I = imread('cell.tif');

figure, imshow(I), title('original image');

text(size(I,2),size(I,1)+15, ...

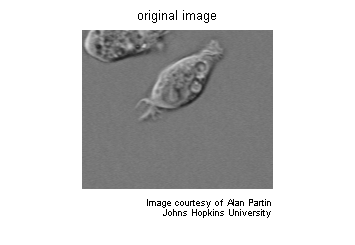
'Image courtesy of Alan Partin', ...

'FontSize',7,'HorizontalAlignment','right');

text(size(I,2),size(I,1)+25, ....

'Johns Hopkins University', ...

'FontSize',7,'HorizontalAlignment','right');



**Step 2: Detect Entire Cell**

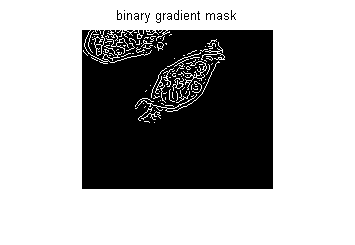
Two cells are present in this image, but only one cell can be seen in its entirety. We will detect this cell. Another word for object detection is segmentation. The object to be segmented differs greatly in contrast from the background image. Changes in contrast can be detected by operators that calculate 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. First, we use edge and the Sobel operator to calculate the threshold value. We then tune the threshold value and use edge again to obtain a binary mask that contains the segmented cell.

[junk threshold] = edge(I, 'sobel');

fudgeFactor = .5;

BWs = edge(I,'sobel', threshold \* fudgeFactor);

figure, imshow(BWs), title('binary gradient mask');



**Step 3: Dilate the Image**

The binary gradient mask shows lines of high contrast in the image. These lines do not quite delineate the outline of the object of interest. Compared to 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 we can create with the strel function.

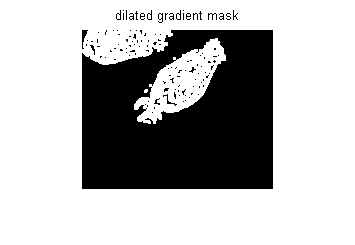
se90 = strel('line', 3, 90);

se0 = strel('line', 3, 0);

The binary gradient mask is dilated using the vertical structuring element followed by the horizontal structuring element. The imdilate function dilates the image.

BWsdil = imdilate(BWs, [se90 se0]);

figure, imshow(BWsdil), title('dilated gradient mask');



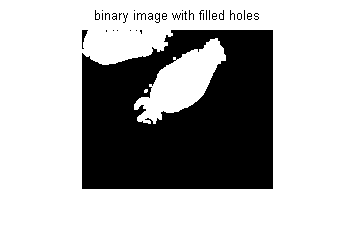
**Step 4: Fill Interior Gaps**

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 we use the imfill function.

BWdfill = imfill(BWsdil, 'holes');

figure, imshow(BWdfill);

title('binary image with filled holes');

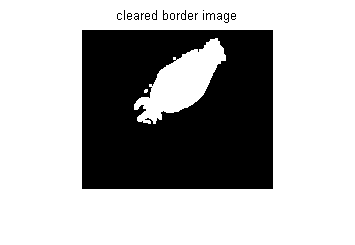


**Step 5: Remove Connected Objects on Border**

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

BWnobord = imclearborder(BWdfill, 4);

figure, imshow(BWnobord), title('cleared border image');



**Step 6: Smoothen the Object**

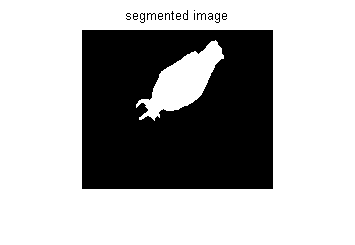
Finally, in order to make the segmented object look natural, we smoothen the object by eroding the image twice with a diamond structuring element. We create the diamond structuring element using the strel function.

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 would be to place an outline around the segmented cell. The outline is created by the bwperim function.

BWoutline = bwperim(BWfinal);

Segout = I;

Segout(BWoutline) = 255;

figure, imshow(Segout), title('outlined original image');

