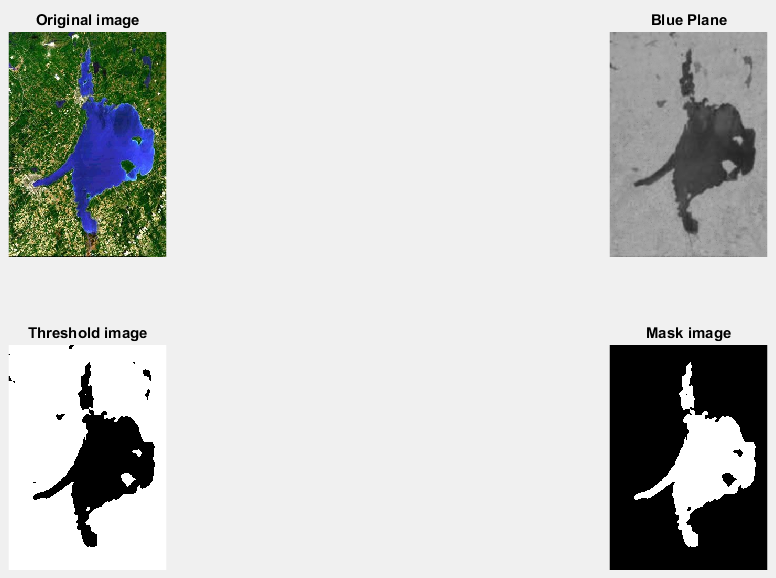
Exercise-1  
Satellite Image Processing.  
Problems Statements  
Download the satellite image of a lake, lake.jpg, located somewhere in the World.  
i- Write an image processing algorithm that measures the surface area of the lake in pixels. Turn in a  
mask image labelling the lake by white pixels and rest of the image by black pixels.  
ii- Given that the height and width of the image correspond to 6.901 Km and 5.258 Km respectively,  
estimate the size of the lake surface in square kilometres. What is its perimeter?

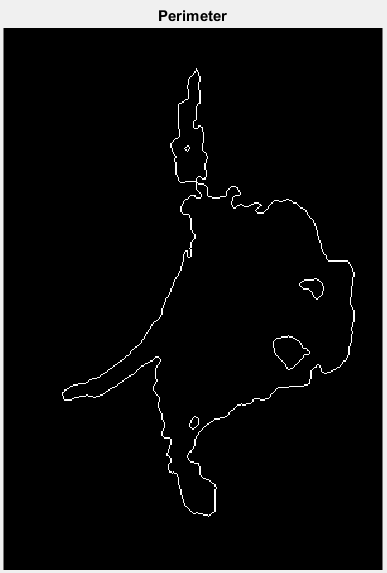
**Solution Approach & Observation**

1. First the image was read with imread. Then the image was converted into LAB format by using makingcform(‘srgb2lab’) and then applied that. Then the image was thresholded using imbinarize and the result was then complemented. After that bwareafilt was used to filter out all the small elements after keeping the biggest one. Then regionprops was used to count the white pixels of the image which denotes the area of the lake.



Converting the image into LAB and showing the blue plane allows to differentiate between blue and other colors. Thresholding turns the image into binary image. And finally masking it allows to distort noise from the image. The pixel number for the area is 25417.

1. From the given height and width area per pixel was calculated. By using bwperim the perimeter of the lake was found. Loops were used to traverse the whole image and count the total number of white pixels which denotes the pixels number of the perimeter. And finally, the perimeter was calculated from the given data.



From the given data value of single axis pixels were found and then that number was multiplied by the pixels count to get the actual value for the perimeter. The value of the perimeter is 20.5743 .

**CODE**

I = imread('Lake.jpg');

c = makecform('srgb2lab');

lb = applycform(I, c);

b = lb(:, :, 3);

bw = imbinarize(b);

com = imcomplement(bw);

bw2 = bwareafilt(com,1,'largest');

per = bwperim(bw2);

res = regionprops(bw2, 'area');

display(res);

[row, column] = size(bw2);

count1= 0;

for i=1:1:row

for j=1:1:column

if(per(i,j)==1)

count1=count1+1;

end

end

end

ph = 6.901/row;

pw = 5.258/column;

Area = ph\*pw\*res.Area;

display(Area);

Perimeter = ph\*count1;

display(Perimeter);

figure,

subplot(2,2,1), imshow(I), title('Original image');

subplot(2,2,2), imshow(b), title('Blue Plane');

subplot(2,2,3), imshow(bw), title('Threshold image');

subplot(2,2,4), imshow(bw2), title('Mask image');

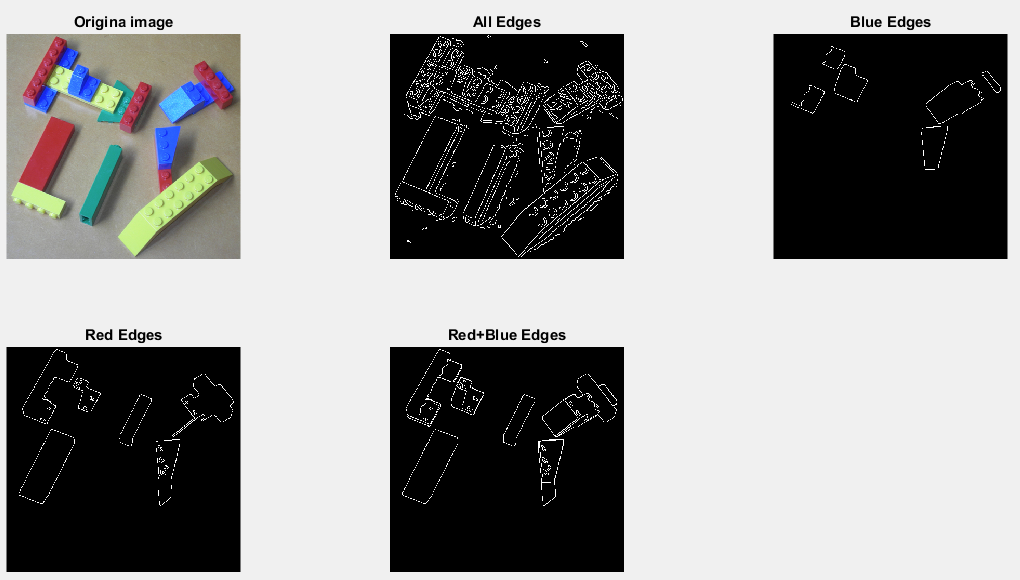
figure,

imshow(per), title('Perimeter');

Exercise-2  
Edge detection  
Problems Statements  
Download the image of toy building blocks, bricks.jpg.  
i- Try to detect all the edges of the image.  
ii- Try to detect firstly the edges for the blue bricks, and then for the red ones.

**Solution Approach & Observation**

The image was read with imread for further processing. The image was then converted to gray using rgb2gray. Then edge function was used to show the edges. Then the image was converted to LAB and blue and red planes were selected. After that the planes were binarized and filtered with bwareafilt. Then the edges for blue plane were dispayed after that the red plane edges were showed. Finally the edges were added.



The edges were detected using Canny method which performs better than other methods. The different color object edge detection became much easier by transforming the image to lab format. Addition of edges gives the final result containing edges for both red and blue planes.

**CODE**

I = imread('bricks.jpg');

gr = rgb2gray(I);

ae = edge(gr, 'Canny');

c = makecform('srgb2lab');

lb = applycform(I, c);

red\_plane = lb(:, :, 2);

blue\_plane = lb(:, :, 3);

red = imbinarize(red\_plane);

blue = imbinarize(blue\_plane);

blue = bwareafilt(blue,1);

red = bwareafilt(red,6);

be = edge(blue, 'Canny');

re = edge(red, 'Canny');

rb = be+re;

figure,

subplot(2,3,1), imshow(I), title('Origina image');

subplot(2,3,2), imshow(ae), title('All Edges');

subplot(2,3,3), imshow(be), title('Blue Edges');

subplot(2,3,4), imshow(re), title('Red Edges');

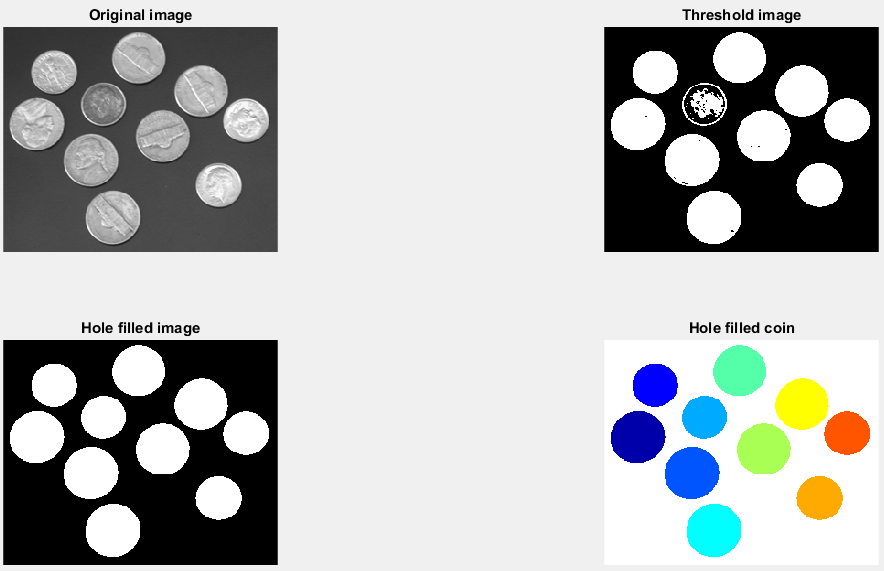
subplot(2,3,5), imshow(rb), title('Red+Blue Edges');

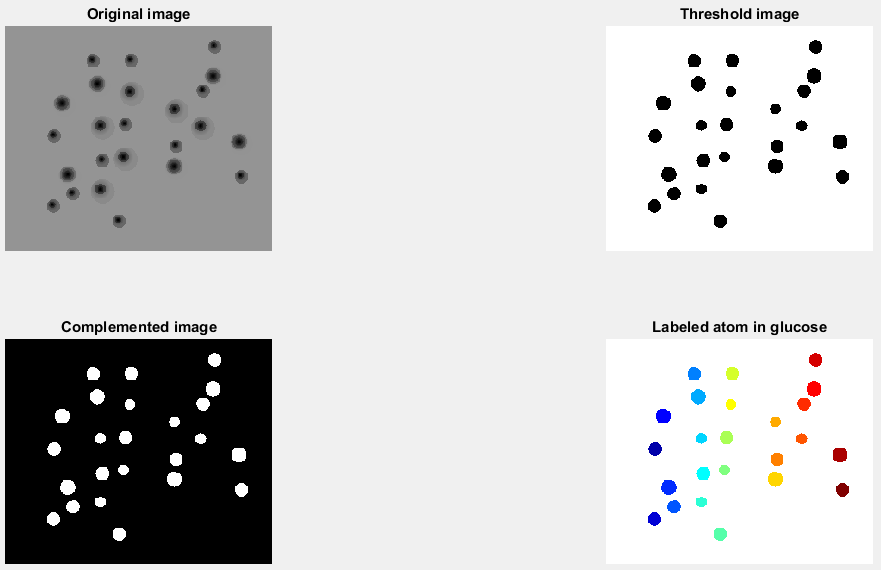
Exercise-3  
Problem Statements  
Use morphological operations and the connected-components Matlab functions to find the number of  
different coins in coins.png and the atoms of the glucose molecule, glucose.tif.

**Solution Approach & Observation**

The same procedure is needed for both the images. At first, the image was read using imread. Then the image was binarized using imbinarized. Then the unwanted holes were filled using imfill. Then bwlabel was used which labels the image and count the objects in the image. Then label2rgb was used to color all the objects differently.

For glucose as the objects were not white so that binazied image was complemented and as there were no holes so imfill was not needed.





The bwlabel works if the objects are white on the black background. So that’s why glucose image was complemented. Holes in objects do not give the same result so for the coins image the holes were needed to be filled. By using label2rgb all the objects are differently.

The number to total coin is 10.

The number of glucose atom is 24.

**CODE**

I = imread('coins.png');

bw = imbinarize(I);

fill = imfill(bw, 'holes');

[labeledImage, numberOfObject] = bwlabel(fill);

fprintf('%i\n', numberOfObject);

figure,

subplot(2,2,1), imshow(I), title('Original image');

subplot(2,2,2), imshow(bw), title('Threshold image');

subplot(2,2,3), imshow(fill), title('Hole filled image');

subplot(2,2,4), imshow(label2rgb(labeledImage)), title('Hole filled coin');

I2 = imread('GLUCOSE.tif');

bw2 = imbinarize(I2);

com = imcomplement(bw2);

[labeledImage2, numberOfObject2] = bwlabel(com);

fprintf('%i\n', numberOfObject2);

figure,

subplot(2,2,1), imshow(I2), title('Original image');

subplot(2,2,2), imshow(bw2), title('Threshold image');

subplot(2,2,3), imshow(com), title('Complemented image');

subplot(2,2,4), imshow(label2rgb(labeledImage2)), title('Labeled atom in glucose');