Digital Image Processing: Lab 1 Exercises

Getting started with Matlab, Reading and visualizing Imaging

Experiment 1: Download the RGB image of Crab Nebulae
https://www.google.com/search?q=crab+nebulae&source=lnms&tbm=isch&sa=X&ved=0ahUKEwiUp_ad78fjAhWBEXwKHVxADoYQ_AUIESgB&biw=883&bih=913&dpr=1.33#imgrc=YVSL1DJmEwDK5M:

Let's try some basic MATLAB functions for loading, saving, displaying and manipulating digital images.

```
%reading the RGB image
im = imread('images/crab_nebula.png');
imshow(im);
s = size(im);

%convert to gray image
im_gray = rgb2gray(im);
s_gray = size(im_gray);

%convert to binary image
im_bw = im2bw(im);
You can find out the luminance threshold, specified as a number in the range
[0,1]
```

Luminance threshold, specified as a number in the range [0, 1]

```
T = graythresh(im)
```

You can also specify the luminance level

```
level = 0.7;
im_bw = im2bw(im, level);
s_bw = size(s_bw)

Visualizing the images
subplot(1,3,1); imshow(im);
subplot(1,3,2); imshow(im_gray);
subplot(1,3,3); imshow(im_bw)

Plotting specific rows and columns
row8 = im(8,:);
row8;
plot(row8);title('Row8'); axis tight;
```

Try plotting a different row and columns.

Experiment 2: Now let us adjust image intensity values or colormap.

`imadjust(I,[low_in high_in],[low_out high_out],gamma)` maps the values in intensity image `I` to new values in `J` such that values between `low_in` and `high_in`map to values between `low_out` and `high_out`. Values below `low_in` and above `high_in` are clipped; that is, values below `low_in` map to `low_out`, and those above `high_in` map to `high_out`. You can use an empty matrix (`[]`) for [`low_in high_in`] or for [`low_out high_out`] to specify the default of `[0 1]`

`gamma` specifies the shape of the curve describing the relationship between the values in `I` and `J`. If `gamma` is less than 1, the mapping is weighted toward higher (brighter) output values. If `gamma` is greater than 1, the mapping is weighted toward lower (darker) output values. If you omit the argument, `gamma` defaults to 1 (linear mapping).

```
im = imread('images/crab_nebula.png');
J = imadjust(im, [], []);
imshow(J);
```

Change different ranges of gamma and see how the image changes.
Produce the complement image

Experiment 3: Enhance contrast Histogram Equalization

Histogram equalization is a method in image processing of contrast adjustment using the image's histogram. Histogram equalization often produces unrealistic effects in photographs; however it is very useful for scientific images like thermal or x-ray images, often the same class of images to which one would apply false color. Also histogram equalization can produce undesirable effects when applied to images with low color depth. For example, if applied to 8-bit image displayed with 8 bit gray scale it will further reduce color depth (number of unique shades of gray) of the image. Histogram equalization will work the best when applied to images with much higher depth than palette size, like continuous data or 16-bit gray-scale images.

```
im = imread('images/crab_nebula.png');
J = histeq(im);
subplot(1,2,1);imshow(im)
subplot(1,2,2); imshow(J)
```

Display histogram

```
J = histeq(im);
subplot(1,2,1);imhist(im);title('Original')
subplot(1,2,2); imhist(J);('Processed')
```

Task 1:

Write a MATLAB code that will do the following

1. Read any gray scale image. **(2 pts)**
2. Display that image with scale. **(2 pts)**
3. Again display the image such that the pixels having intensity values below than 50 will display as black and pixels having intensity values above than 150 will display as white. And the pixels between these will display as it is. **(6 pts)**

Task 2:

Write a MATLAB code that reads a gray scale image and generates the flipped image of original image. **(5 pts)**