CO543 - Image Processing
Lab 02

1. Image thresholding

Image thresholding is a way of partitioning an image into a foreground and background.

Lab Task 01 : Write a function to perform image thresholding using point processing taking the image file and the threshold value from the user.

2. Image arithmetic operations

Basic arithmetic operations like addition, subtraction, division, multiplication can be performed quickly and easily on image pixels for a variety of effects and applications. These operations are carried out between corresponding pixel pairs of two images of the same size.

Lab Task 02 : Read two images and perform addition and subtraction.

I=I1+I2; # Addition of two

I=I1-I2; # Subtraction of two images

Now, use inbuilt functions

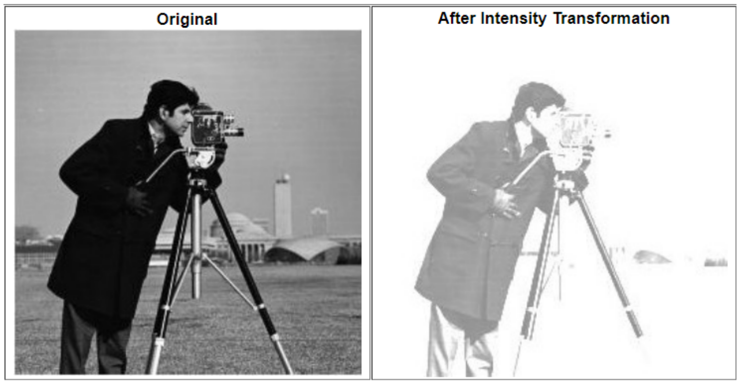
- OpenCV function, **cv2.add()** or simply by numpy operation, **res = img1 + img2**. Both images should be of same depth and type, or second image can just be a scalar value.

- OpenCV function, **cv2.subtract()** or simply by numpy operation, **res = img1 - img2**.

3. Spatial Processing

• Intensity Transformation Functions

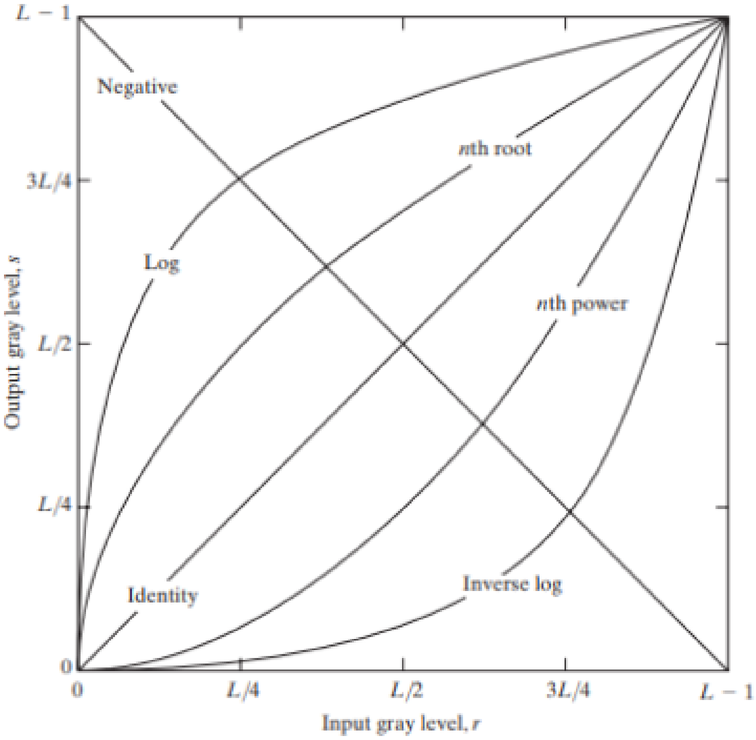
When working with gray-scale images, sometimes there would be a requirement to modify the intensity values. For instance, you may want to reverse black and the white intensities or you may want to make the darks darker and the lights lighter.



Original

After Intensity Transformation

An application of intensity transformations is to increase the contrast between certain intensity values so that you can pick out things in an image. Intensity transformations are applied on images for contrast manipulation or image thresholding.

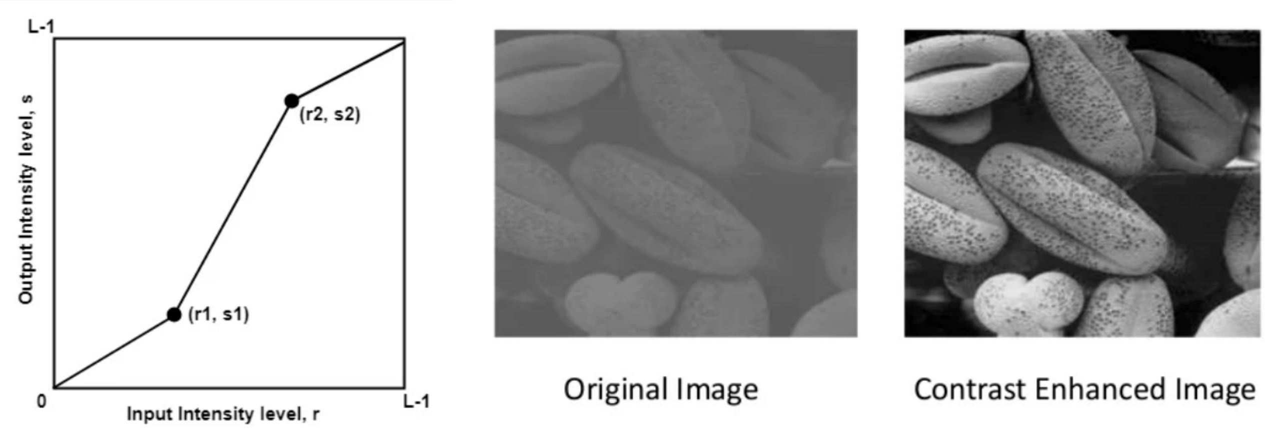


The following are commonly used intensity transformations:

- Image Negatives (Linear)
- Log Transformations
- Power-Law (Gamma) Transformations
- Piecewise-Linear Transformation Functions

Contrast Stretching

Contrast stretching (often called normalization) is a simple image enhancement technique that attempts to improve the contrast in an image by `stretching' the range of intensity values it contains to span a desired range of values.It increases the contrast between the darks and the lights.

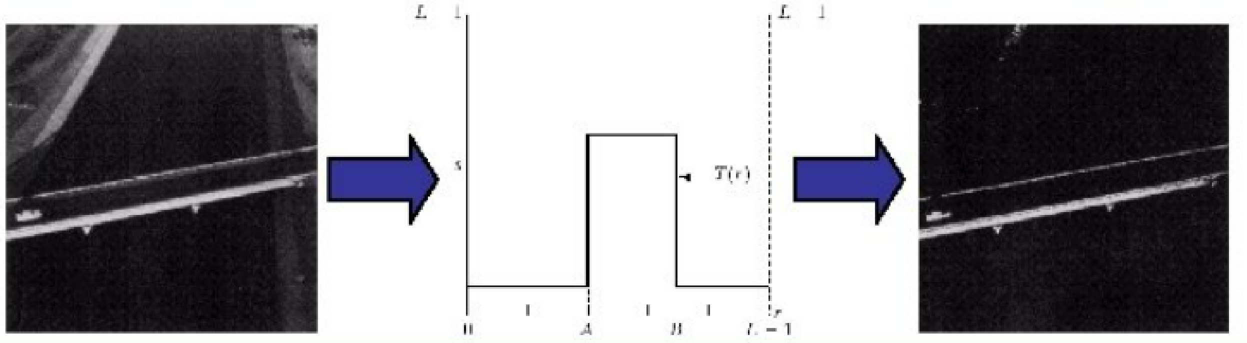


Original Image

Contrast Enhanced Image

Intensity / Gray level Slicing

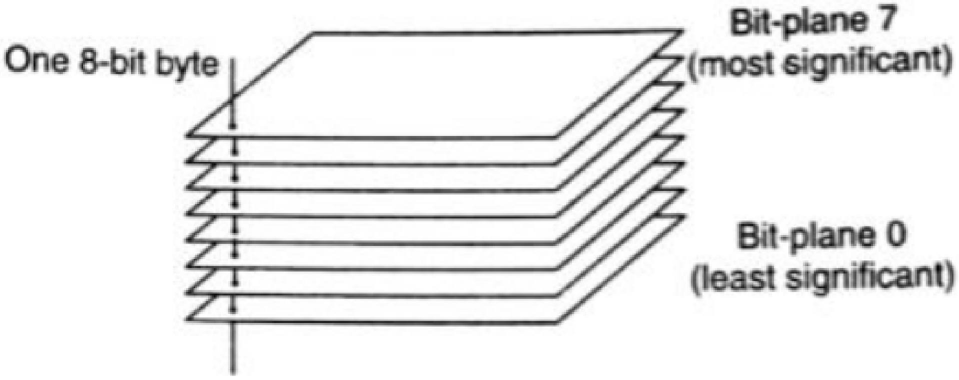
Gray level Slicing manipulates groups of intensity levels in an image up to specific range by diminishing rest or by leaving them alone. This method is useful for highlighting the features of an image.

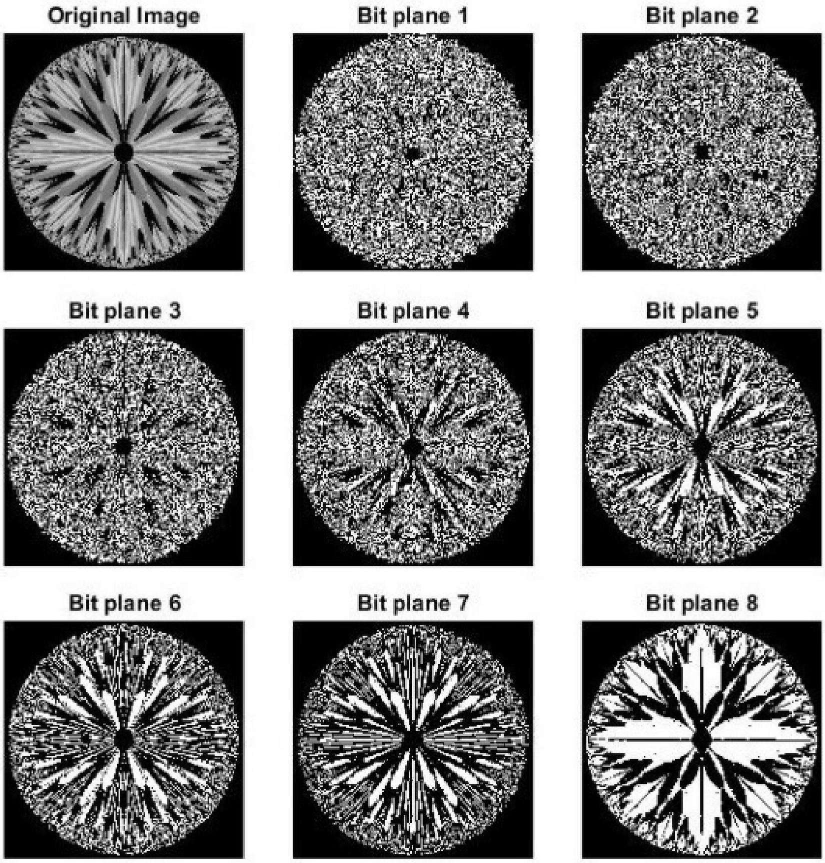


- **Bit Plane Slicing**

Instead of highlighting gray level images, highlighting the contribution made to total image appearance by specific bits might be desired. Suppose that each pixel in an image is represented by 8 bits. Imagine the image is composed of 8, 1-bit planes ranging from bit plane 1-0 (LSB)to bit plane 7 (MSB).

In terms of 8-bits bytes, plane 0 contains all lowest order bits in the bytes comprising the pixels in the image and plane 7 contains all high order bits.




Original Image

Bit plane 1

Bit plane 2

Bit plane 3

Bit plane 4

Bit plane 5

Bit plane 6

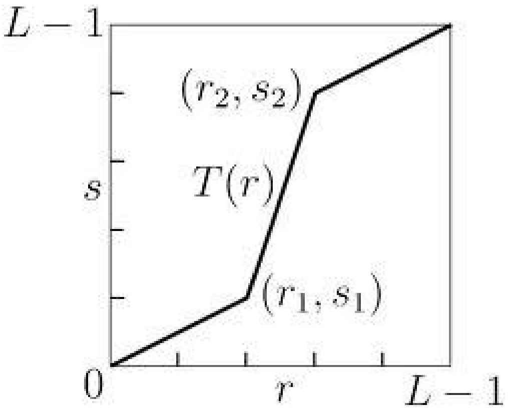
Bit plane 7

Bit plane 8

Lab Task 03 :

1. Write simple programs to demonstrate the following. Show the original and resultant images in the same figure to compare them easily.
    a. Log transformation
    b. Power transformation
    c. Contrast Stretching
    d. Gray level slicing
    e. Bit plane slicing

2. Consider the graph for a typical transformation function used for Contrast Stretching in the given figure and determine the behavior of the function with respect to given changes.

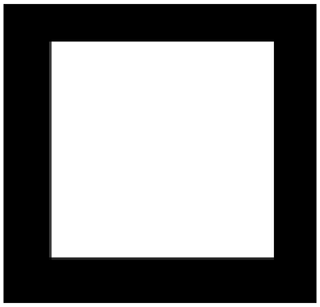


    a. When r1 =s1 and r2=s2
    b. When r1=r2, s1=0 and s2=L-1

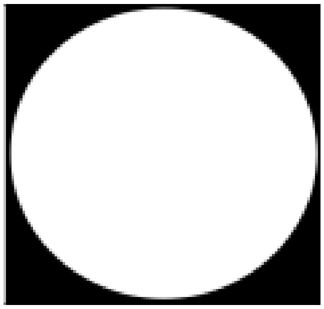
---

4. Masking

Lab Task 04 : Write a program to read any image, resize it to 256x256. Apply the masks shown in following figures so that only the middle part of the image is visible.



128×128



5. Brightness

Lab Task 05 : Write your own Python OpenCV function addbrightness() and use it to increase brightness of a given image.(Hint: Use Image arithmetic operations)

---

6. Histogram Processing

- ## Histogram

In an image processing context, the histogram of an image normally refers to a histogram of the pixel intensity values. This histogram is a graph showing the number of pixels in an image at each different intensity value found in that image.
It shows how many times a particular gray level(intensity) appears in an image.

Why Histograms?

- It is the basis for numerous spatial domain processing techniques
- Manipulating a histogram can use effectively for image enhancement
- Provides useful image statistics
- Information derived from a histogram can be used in other image processing techniques like compression, segmentation.

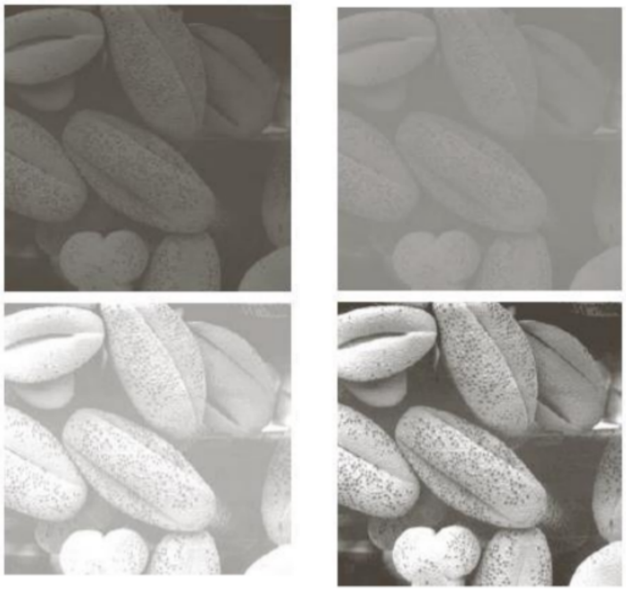
- ## Histogram Equalization

Histogram Equalization is a computer image processing technique used to improve contrast in images . It accomplishes this by effectively spreading out the most frequent intensity values, i.e. stretching out the intensity range of the image.

Lab Task 06 :

1. **Histogram Calculation in OpenCV**
   Use inbuilt OpenCV cv2.calcHist() function to display the histogram of a given image.
2. **Histogram Calculation in Numpy**
   Use inbuilt numpy np.histogram() function to display the histogram of a given image.
3. Then write your own histogram functions for the following scenarios
   a. Show a histogram plot for a grayscale image.
   b. Show three histograms for a given RGB image.

4. Consider the four images given in the resources folder. Plot the histogram for each image. Perform Histogram Equalization on each image and plot the histograms of the resultant images. Comment on the results you have obtained.



Submission

- You need to submit **a report** (Lab02_E17XXX.pdf) displaying Screenshots of the codes, **results** from your code (your **input and output images** under each section after performing the required functions) for each lab task
- You need to submit **all python files** containing the relevant functions named according to the relevant question names or as indicated in the lab sheet, along with a **main function** to run them and display your outputs. Make sure to include the **images** you used to run the codes as well.

Submit a folder named Lab02_E17XXX.zip whare XXX is your E number including all ,

· **Python source codes**
· **All Input images**
· **Report**